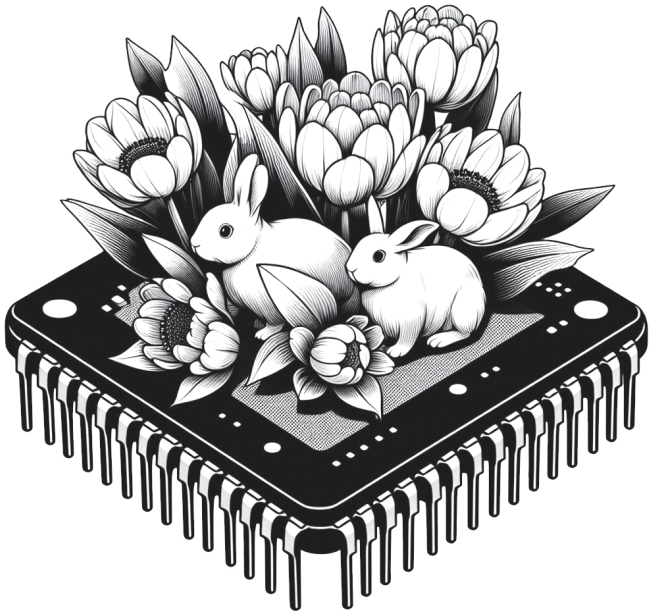
Werkcollege-opgaven behorende bij het college Natuur en Berekening

Cursusjaar 2025-26, versie 30 maart 2026



dr. G.A.W. Vreeswijk

Universiteit Utrecht

Inhoud

Versies

– 03 feb. Bijgewerkt naar 2026.

– 18 feb. Langton's mier: Een typo in de nummering van de vakjes gecorrigeerd $[(-1,1) \to (-1,0)]$.

– 17 mrt. Typos, waar nodig variabelen hernoemen naar unieke namen, een 4e bonus-som toegevoegd aan Hoofdstuk 17. (Iets voor volgend jaar.)

– 30 mrt. Een hoofdstuk "Herhaalde spelen" toegevoegd. Een verkennende som over de logistieke afbeelding toegevoegd.

Voorwoord

Dit zijn werkcollege-opgaven die horen bij het college Natuur en Berekening, editie 2026.

Het aantal opgaven, en de omvang van sommige, is in eerste instantie misschien wat overweldigend. Misschien helpt het om aan te geven hoe deze collectie tot stand is gekomen. Een aantal opgaven is speciaal geschreven voor een vorige editie van het vak, genaamd Inl. Adaptieve Systemen (IAS). In de jaren daarna zijn er opgaven bijgeschreven, en zijn er nog een handvol wat grotere inlever-opgaven omgewerkt tot werkcollege-opgaven. De inlever-opgaven zijn meestal bewerkelijker dan de oorspronkelijke werkcollege-opgaven. Soms kost het maken van één zo'n omgewerkte inleveropgave al gauw 30-60 minuten. Houd hier rekening mee. In de periode 2020-2024 zijn telkens aan de voorkant van elke sectie kleinere opgaven ingevoegd. Een kleinere opgave is bedoeld als inleiding en kan vaak in enkele minuten worden gemaakt.

Graag wil ik iedereen die met deze bundel hebben gewerkt, bedanken voor hun vragen, opmerkingen, suggesties, en correcties.

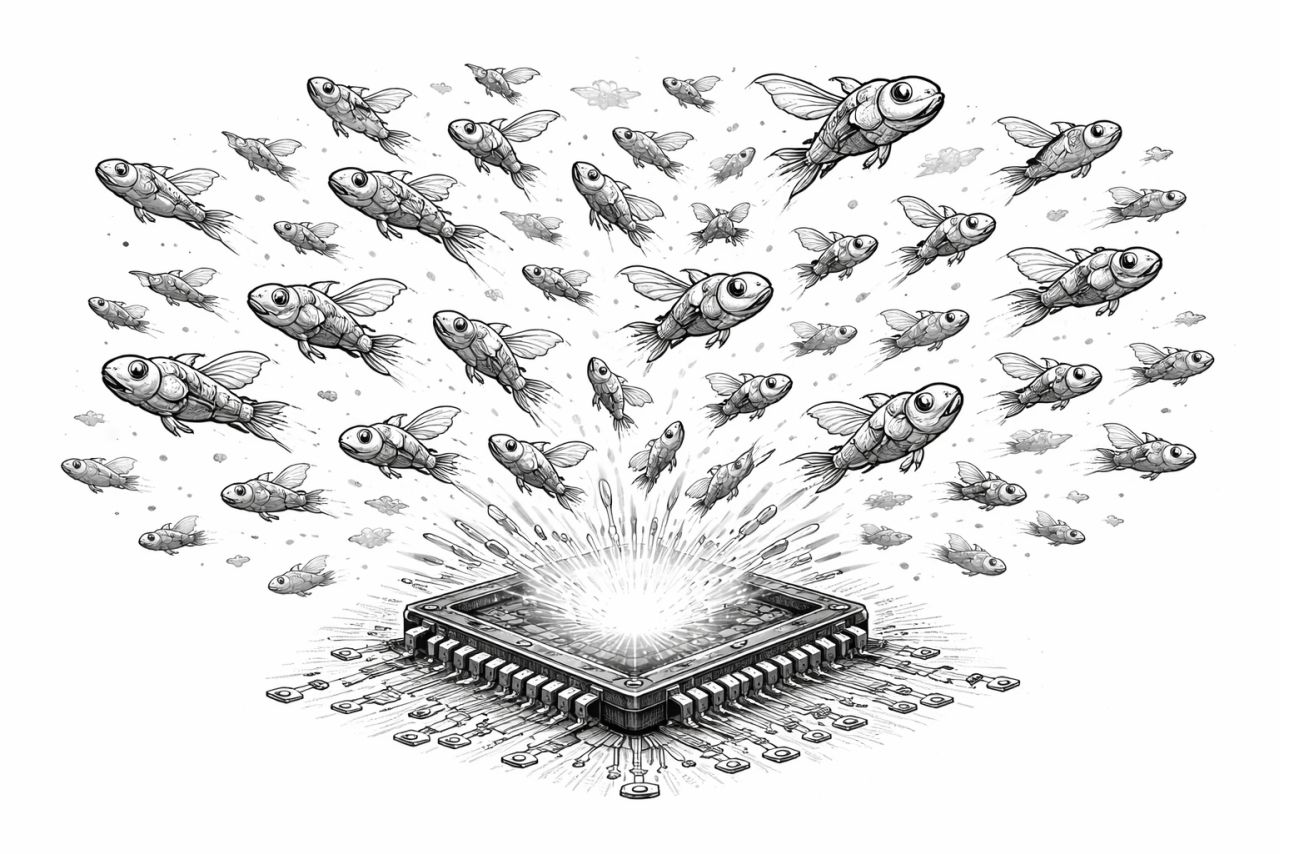
GV, Utrecht, 30 maart 2026.

Deel I. Inleiding

1 Inleiding

Onderstaande opgaven horen bij het voorwoord ("Preface") en Hoofdstuk 1: "Introduction" uit TCBoN: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation, Gary W. Flake, MIT Press, 2000.

Gary Flake stelt in zijn voorwoord dat de verbluffende complexiteit van de natuur niet voortkomt uit ingewikkelde blauwdrukken, maar uit de herhaling van simpele, lokale regels. Hij stelt dan ook voor de computer in te zetten als het ultieme laboratorium om deze "berekening van de natuur" te simuleren en te begrijpen.



Een variatie op de kaft-illustratie van Flake's TCBoN (Bron: Gemini 3).

Flake daagt zowel het reductionisme (bottom-up denken) als het dirigisme (top-down denken) uit. Volgens hem begrijp je een systeem niet door je blind te staren op de afzonderlijke onderdelen (reductionisme), maar ook niet door het te zien als een centraal gestuurde organisatie (dirigisme). Hij stelt dat je beter kunt kijken naar de patronen die ontstaan dankzij de *interactie* tussen die onderdelen. Daarmee pleit hij voor een vorm van holisme: een visie waarbij een systeem gedefinieerd wordt door de interactie tussen haar onderdelen. Hierdoor ontstaat een geheel dat meer is dan de som der delen. Dat laatste noemen we *synergie*. Schoonheid, die zich manifesteert in ruimtelijke of temporele patronen, is in deze visie simpelweg een bijproduct van de interactie tussen onderdelen die elk een eenvoudig algoritme (een vast stelsel van regels) volgen.

Enfin. Lees de tekst eerst door, en maak waar nodig aantekeningen. Beantwoord daarna de onderstaande vragen. Je zult misschien weer in de tekst moeten zoeken.

1. Op blz. *xiii* wordt Henri Poincaré aangehaald:

   > *"The scientist does not study nature because it is useful; (. . .) a pure intelligence can grasp."* (Science et méthode, 1908, Part I. Ch. 1, p. 22.)

   Op welke twee typen van schoonheid doelt Poincaré?

2. Flake opent met

   > *"Engineers study interesting real-world problems but fudge their results. Mathematicians get exact results but study only toy problems. But computer scientists, being neither engineers nor mathematicians, study toy problems and fudge their results."*

   Daarna legt hij uit wat hij met deze grap wil illustreren. Wat wil Flake met deze grap uitleggen? Zoek eerst uit wat de volgende frases betekenen:

   - A toy problem.
   - To make someone the butt of a joke.
   - To straddle the fence.

3. (p. xiv). Waarom kunnen volgens Flake met name eenvoudige simulaties een dieper inzicht geven in complexe (natuurkundige, biologische, economische en sociale) processen?

4. (p. xiv). Flake bespreekt verschillen de manier om TCBoN te lezen. Welke metafoor gebruikt Flake hiervoor? In termen van de metafoor, welke drie manier zijn dat?

5. (p. vi). Voor wie heeft Flake dit boek geschreven?

6. (p. 2) Wat probeert Flake te illustreren met het eend ("Duck") voorbeeld?

7. (p. 2) Wat bedoeld Flake met het volgende?

   > *"Specifically, reductionism fails when we try to use it in a reverse direction."*

8. (p. 2) Volgens Flake zijn er drie manieren om te begrijpen "hoe dingen werken". Welke manieren zijn dat?

9. (p. 3) Flake bespreekt emergentie. In feite definieert hij dit begrip in één zin. Geef Flake's definitie zoals die verwoord is op p. 3. Geef commentaar op deze definitie. Vind je hem goed, of juist niet? Vind je er iets aan ontbreken?

10. (p. 3) Eerder gaf Flake aan dat er drie manieren zijn om te begrijpen "hoe dingen werken". Welke twee paradigma's beschouwd Flake als uitersten van deze drie manieren

11. (p. 3) "Nature is frugal." Wat bedoeld Flake hier mee? (Let op: er wordt hier meer gevraagd dan simpel een vertaling van de betreffende zin.) Wat heeft de quote van Einstein bovenaan Hoofdstuk 1 hier mee te maken? Google ook eens op "Ockhams scheermes".

12. (p. 4) Flake geeft enkele voorbeelden van parallellisme in de natuur. Bedenk zelf nog een voorbeeld.

13. (p. 4) Geef een voorbeeld van feedback in de natuur.

14. (p. 2-4) Flake spreekt over drie verschillende attributen die de interactie tussen agenten zo interessant maakt. Welke attributen zijn dat?

15. (p. 5) Op pagina 5 wordt het doel van het boek verwoord! Wat is dit doel?

16. Op de website van het boek vermeldt Flake dat drie sleutel-ideeën herhaaldelijk terugkomen in het boek. Welke drie ideeën zijn dat? *("Basically, there are three key ideas that keep reappearing throughout the book.").* Belangrijker: wat wordt er mee bedoeld?

17. (p. 5-6) Hoe ziet Flake de rol van informatica in de moderne wetenschap?

18. Geef zoveel mogelijk sleutelwoorden uit Hoofdstuk 1, met hun betekenis. Je begrippenlijst wordt afgeturfd tegen onze begrippenlijst. Voorgift:

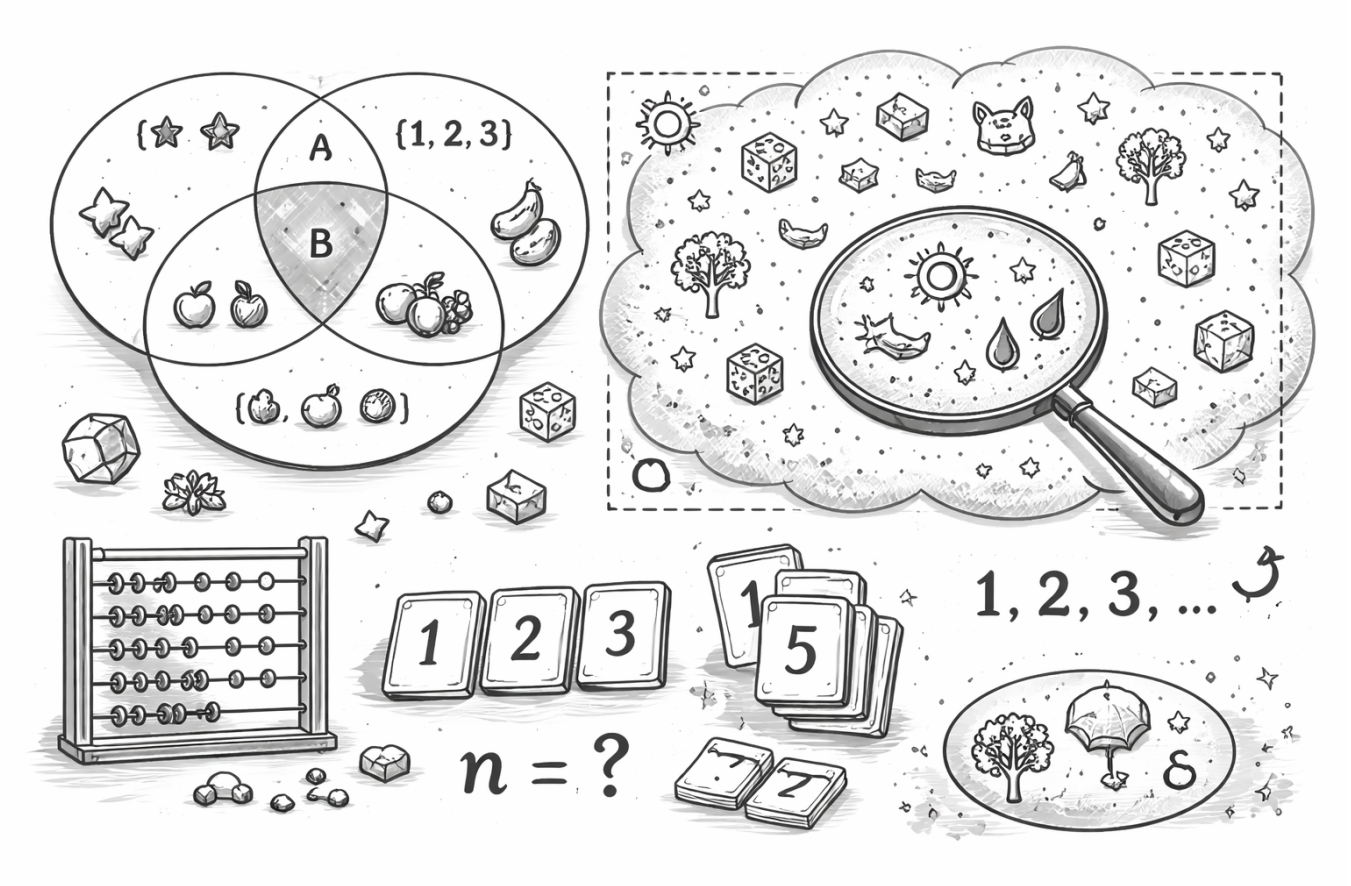
    Reductionisme De benadering, of het geloof, dat de werking van complexe systemen alleen begrepen kan worden door deze systemen en hun werking ervan te reduceren naar hun onderdelen, de werking van hun onderdelen, en de interacties tussen hun onderdelen. Een meer extremere versie van het reductionisme gaat er van uit dat complexe systemen niets meer of minder zijn dan de som van hun onderdelen, de werking van hun onderdelen, en de interacties tussen hun onderdelen.

Deel II. Kardinaliteit en maat

2 Kardinaliteit

Het woord “kardinaliteit’ staat intuïtief voor “het aantal elementen in een verzameling”; of: de ‘grootte” van een verzameling. Als een verzameling eindig is, dan is kardinaliteit een triviaal begrip: we tellen het aantal elementen totdat ze allemaal zijn gepasseerd. Het getal waar we dan mee eindigen is dan de kardinaliteit.

Ook oneindige verzamelingen kennen een kardinaliteit. Sommige oneindige verzamelingen kennen een verschillende kardinaliteit (bijvoorbeeld $\mathbb{N}$ en $\mathbb{R}$). Ook in de informatica zijn er verzamelingen met een verschillende kardinaliteit: het bewijst dat er problemen bestaan die een computer simpelweg nooit kan oplossen. De extra “exotische” elementen in de grotere verzameling laten zien waar de grenzen van berekenbaarheid liggen. Zo helpt het tellen van oneindigheid ons te begrijpen wat technisch onmogelijk is.



(Bron: Gemini 3, prompt: “Generate a 300x120 black and white impression of counting, cardinality, sets, and sample space. No text in image.”)

Deze opgaven horen bij Hoofdstuk 2: “Number systems and infinity” uit TCBoN: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation, Gary W. Flake, MIT Press, 2000.

1. Produceer een Nederlandse samenvatting van Hoofdstuk 2 (Number Systems and Infinity) van *The computational Beauty of Nature* (Flake, 1998). Zorg er voor dat de samenvatting kort, compleet, juist, en samenhangend is. Het is raadzaam deze opgave te combineren met Opgave 2 op pagina 44. (Antwoord achter in het dictaat.)

2. Geef zelf definities van de volgende begrippen.

   (a) Eindig. Hint: elke eindige verzameling is de vereniging van een kleinere eindige verzameling en een 'nieuw" element. Bijvoorbeeld, $\{11, 41, 85, 91, 93\} = \{11, 41, 85, 91\} \cup \{93\}$ (of $\{41, 85, 91, 93\} \cup \{11\}$, dat maakt verder niet uit—verzamelingen kennen geen volgorde). Op deze manier kan de notie "eindige verzameling" inductief worden gedefinieerd.

   (b) Oneindig.

   (c) Aftelbaar.

   (d) Aftelbaar oneindig.

   (e) Overaftelbaar oneindig.

3. Hier de eerste kardinaalgetallen (met wat suggestie door puntjes)

$$\begin{array}{c} 0\,, 1, 2, \ldots, 100, \ldots, \aleph_0\,, \aleph_1, \aleph_2, \ldots, \aleph_{100}, \ldots, \\ |\emptyset|, \;,\;,\quad,\quad,\quad, |\mathbb{N}|, \quad,\quad,\quad,\quad, \end{array}$$

   (a) Geef het kardinaalgetal, of de kardinaalgetallen, die behoren bij de notie "overaftelbaar oneindig".

   (b) Welk van de begrippen genoemd bij (2) corresponderen met precies één kardinaalgetal?

   (c) De continuümhypothese zegt dat, na het kardinaalgetal van $\mathbb{N}$, het eerstvolgende kardinaalgetal gelijk is aan het kardinaalgetal van $\mathbb{R}$. Onder welk kardinaalgetal zou in dat geval $|\mathbb{R}|$ moeten worden gezet?

4. Geef de kardinaliteit (de grootte) van de volgende verzamelingen.

   (a) Alle pixels op een 4K scherm.

   (b) Alle zandkorrels op aarde.

   (c) $\mathbb{R}$.

   (d) Het interval $(0, 1)$.

   (e) Het platte vlak $\mathbb{R} \times \mathbb{R}$.
   Gebruik de truc dat decimale representaties van elementen uit het eenheidsvierkant $(0, 1)^2$ kunnen worden vervlochten tot een element uit het eenheidsinterval $(0, 1)$. Bijvoorbeeld $(1/2, \sqrt{2}/2)$ wordt afgebeeld op $0.5700070100060708010108 0 \ldots$. (Ga na dat deze constructie inderdaad een bi-jectie oplevert.)

5. Geef de kardinaliteit (de grootte) van de volgende verzamelingen.

   (a) Alle open intervallen in $\mathbb{R}$. Voorbeelden: $(-1, 1)$, $(2, 4)$, $(\sqrt{2}, 100)$, en $(-99, \pi)$.

   (b) $\mathbb{R} \times \{1, 2, 3, 4\}$.

   (c) Alle intervallen uit $\mathbb{R}$. Voorbeelden: $[-1, 1]$, $[2, 4)$, $(\sqrt{2}, 100)$, en $(-99, \pi]$.

   (d) Alle deelverzamelingen van $\mathbb{R}$: $2^{\mathbb{R}}$.

(e) (Wat moeilijker.) De *verzameling* van alle open intervallen in $\mathbb{R}$. Hint: deze verzameling correspondeert bijna 1-1 met de verzameling van paren van reële getallen.

6. Laat zien dat de verzameling gehele getallen, $\mathbb{Z}$, aftelbaar is.

   (a) Eerst informeel door alle elementen van $\mathbb{Z}$ op een (oneindige) rij te zetten.

   (b) Dan formeel, door een surjectie te geven van $\mathbb{N} = \{1, 2, 3, \dots\}$ naar $\mathbb{Z}$.[1] (Of, omgekeerd, een injectie $\mathbb{Z}$ naar $\mathbb{N}$.) Laat wel even zien dat je inderdaad een surjectie (danwel injectie) hebt geconstrueerd.

7. Laat zien dat de verzameling van derde-machten
   $T = \{\ t \in \mathbb{Z} \mid \text{er is een } z \in \mathbb{Z} \text{ zodanig dat } z^3 = t\ \}$ aftelbaar is. Eerst weer schetsmatig, dan weer formeel. Geef de bijbehorende surjectie (of injectie).

8. (a) Laat zien: $\mathbb{N} \cup \{0\}$ is aftelbaar.

   (b) Laat zien: $\mathbb{N} \cup \{0, -1, -2, \dots, -m\}$ is aftelbaar.

9. Laat (informeel) in een rooster zien:

   (a) $\mathbb{N} \times \{0, 1\}$ is aftelbaar.

   (b) $\mathbb{N} \times \mathbb{N}$ is aftelbaar.

   (c) $\mathbb{Z} \times \mathbb{Z}$ is aftelbaar.

   (d) Waarom volgt uit 9c dat $\mathbb{Q}$ aftelbaar is?

10. Iemand beweert dat er eigenlijk twee vormen van aftelbaarheid bestaan: "gewone aftelbaarheid," met één open einde

    $$a_1, a_2, a_3, \dots,$$

    en "dubbele aftelbaarheid," met twee open einden

    $$\dots, a_{-3}, a_{-2}, a_{-1}, a_0, a_1, a_2, a_3, \dots.$$

    De verzameling $\mathbb{N} = \{1, 2, 3, \dots\}$ zou dan een typisch voorbeeld zijn van "gewone aftelbaarheid," en de verzameling $\mathbb{Z} = \{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ zou dan een typisch voorbeeld zijn van "dubbele aftelbaarheid". Wat zou je hier op zeggen?

11. Welke van de volgende termen zijn kardinaliteiten? 4, n ($n \in \mathbb{N} \cup \{0\}$), eindig, aftelbaar, $\mathbb{N}$, $|\mathbb{N}|$, aftelbaar oneindig, x ($x \in \mathbb{R}, x \geq 0$), $\mathbb{R}$, $|\mathbb{R}|$, overaftelbaar oneindig, ontelbaar.

12. Zij A en B aftelbaar. Bewijs de aftelbaarheid van de volgende verzamelingen.

    (a) Een willekeurige verzameling X, met $X \subseteq A$.

    (b) $A \cup B$, als B eindig is.

    (c) $A \cup B$.

    (d) $A \cap B$.

    (e) $A \times B$.

[1] Helaas is er geen overeenstemming over het antwoord op de vraag of $0 \in \mathbb{N}$. In de wiskunde is het gebruikelijk dat men vanaf 1 telt, dus daar $0 \notin \mathbb{N}$. In de informatica is het gebruikelijk dat men vanaf nul telt. (Denk aan het indexeren van arrays in programmeertalen.) In deze tekst wordt de wiskundige conventie gevolgd.

(f) $A \backslash B$.

13. Zij $A_1, \ldots, A_n$ aftelbaar. Bewijs:

    (a) $\bigcup_{i=1}^{n} A_i = A_1 \cup \cdots \cup A_n$ is aftelbaar.

    (b) $\bigcap_{i=1}^{n} A_i = A_1 \cap \cdots \cap A_n$ is aftelbaar.

    (c) $\prod_{i=1}^{n} A_i = A_1 \times \cdots \times A_n$ is aftelbaar.

    (d) Alle eindige en niet-complementaire verzamelingstheoretische combinaties van $A_1, \ldots, A_n$ zijn aftelbaar, dat wil zeggen: alle eindige verzamelingstheoretische combinaties van $A_1, \ldots, A_n$ zijn aftelbaar, zolang er gecombineerd wordt met operatoren uit $\{\cup, \cap, \times\}$.

14. Zij $A_1, \ldots, A_n, \ldots$ een aftelbare rij van eindige verzamelingen. Bewijs:

    (a) $\bigcup_{i=1}^{\infty} A_i = A_1 \cup \cdots \cup A_n \cup \ldots$ is aftelbaar.

    (b) $\bigcap_{i=1}^{\infty} A_i = A_1 \cap \cdots \cap A_n \cap \ldots$ is aftelbaar.

15. Zij $A_1, \ldots, A_n, \ldots$ een aftelbare rij van aftelbare (mogelijk oneindige) verzamelingen. Bewijs:

    (a) $\bigcup_{i=1}^{\infty} A_i = A_1 \cup \cdots \cup A_n \cup \ldots$ is aftelbaar.

    (b) $\bigcap_{i=1}^{\infty} A_i = A_1 \cap \cdots \cap A_n \cap \ldots$ is aftelbaar.

16. Zij $A_1, \ldots, A_n, \ldots$ een aftelbare rij van eindige verzamelingen.

    (a) Laat met een tegenvoorbeeld zien dat

    $$\prod_{i=1}^{\infty} A_i = A_1 \times \cdots \times A_n \times \ldots$$

    in het algemeen niet meer aftelbaar is, zelfs niet als alle A_i eindig zijn. Om overaftelbaarheid van het product aan te tonen kan (weer) een diagonaalargument worden gebruikt.
    Hint: denk eerst eens aan (eindige en oneindige) producten van getallen, zoals $1 \times 1 \times 1 \times 1$, en

    $$1 \times 1 \times 1 \times 1 \times \ldots,$$

    en $2 \times 2 \times 2 \times 2$, en

    $$2 \times 2 \times 2 \times 2 \times \ldots$$

    (b) Onder welke voorwaarden is het oneindige Cartesisch product wél aftelbaar?
    Hint: bekijk eens het product

    $$1 \times 2 \times 1 \times 1 \times 7 \times 1 \times 1 \times \ldots$$

    met eindig veel factoren ongelijk 1. Wat zou daar de uitkomst van zijn? Als je een antwoord gevonden hebt, probeer je voorwaarden dan wat op te rekken door (sommige?) A_i groter te maken. Hoe ver kun je daar in gaan?

17. (a) Laat zien dat dat de verzameling van alle functies van $\mathbb{N}$ naar $\{0, 1\}$, genoteerd als $\{0, 1\}^{\mathbb{N}}$ overaftelbaar is. (Hint: elk zo'n functie kan worden gerepresenteerd door een oneindige bitstring. Pas nu Cantor's diagonaalmethode toe.)

(b) Waarom volgt hieruit dat de machtsverzameling van $\mathbb{N}$, i.e., de verzameling van alle deelverzamelingen van $\mathbb{N}$, overaftelbaar is?

(c) Laat op dezelfde manier zien dat $\mathbb{N}^{\mathbb{N}}$ overaftelbaar is.

18. (a) Laat zien dat de verzameling van alle deelverzamelingen van $\mathbb{N}$ overaftelbaar is. (Hint: $\{0,1\}^{\mathbb{N}} \sim 2^{\mathbb{N}}$.)

(b) Laat zien dat de verzameling van alle eindige deelverzamelingen van $\mathbb{N}$ daarentegen wèl aftelbaar is.

19. Een *co*-finiete deelverzameling van $\mathbb{N}$ is een deelverzameling A van $\mathbb{N}$ met de eigenschap dat $\mathbb{N} - A$ eindig is (dus: een verzameling met een eindig complement).

Laat zien dat de verzameling van alle *co-finiete* deelverzamelingen van $\mathbb{N}$ aftelbaar is.

20. We noteren $A \sim B$ dan en slechts dan als A en B dezelfde kardinaliteit bezitten. (Dus $\mathbb{N} \sim \mathbb{Z} \sim \mathbb{Q} \sim \mathbb{A} \nsim \mathbb{R}$.) De letter $\mathbb{A}$ representeert de verzameling van algebraïsche getallen (oplossingen van veeltermen). Bijvoorbeeld, $1/(7-\sqrt{3}) \in \mathbb{A}$.

Laat, door te redeneren met bi-jecties, zien dat het hebben van dezelfde kardinaliteit een equivalentie-relatie (reflexief, symmetrisch, transitief) is op verzamelingen.

21. Bewijs of beredeneer de volgende beweringen door het geven van een bi-jectie.

(a) $[0,2] \sim [0,1]$.

(b) $[0,2) \sim [0,1)$.

(c) $[0,2) \sim (0,1]$.

(d) $(0,1) \sim (-1,1)$.

(e) $\mathbb{R}^+ = \{x \in \mathbb{R} \mid x > 0\} \sim (0,1)$.

(f) $\mathbb{R} \sim (-1,1)$.

(g) $\mathbb{R}_0^+ = \{x \in \mathbb{R} \mid x \geq 0\} \sim (0,1]$.

(h) $\mathbb{R}_0^+ \sim [0,1)$.

(i) $\mathbb{R}^+ \cup \{\omega\} \sim (0,1]$. De letter omega staat voor het eerste oneindige ordinaalgetal, maar dat is niet zo belangrijk. Je hoeft slechts te gebruiken dat je er naast $\mathbb{R}^+$ nog een singleton set bijkrijgt die het voor jou makkelijker maakt een bijectie met $(0,1]$ te construeren.

(j) $\mathbb{R}_0^+ \cup \{\omega\} \sim [0,1]$.

22. Bewijs of beredeneer de volgende beweringen. Maak, indien nodig, gebruik van de stelling van Schröder-Bernstein.

(a) $[0,1) \sim [0,1]$.

(b) $[-1,2) \sim [3,400]$.

(c) $[0,1] \sim \mathbb{R}$. Hint: om $\mathbb{R}$ te comprimeren tot een klein beeld binnen $[0,1]$ kan bv. arctan gebruikt worden.

(d) Eenheidsschijf $\sim$ eenheidsvierkant.

- De eenheidsschijf is de gesloten gevulde cirkel met middelpunt $(0,0)$ en straal 1. De rand hoort er ook bij, daar slaat het woord "gesloten" op.
- Het eenheidsvierkant is gelijk aan $[0,1]^2$.

(e) Doughnut (= gevulde zwemband) $\sim$ snaar van een gitaar.

23. (Moeilijker.) Laat, door direct een bi-jectie geven, dus zonder gebruikmaking van de Stelling van Schroöder-Bernstein, zien dat $[0,1] \sim [0,1)$. Hint: begin met

$$f : [0,1] \to [0,1) : x \mapsto x.$$

Dit is eigenlijk geen functievoorschrift, want het beeld van 1 valt buiten het co-domein $[0,1)$ van f. Ga deze fout repareren door 1 op een getal dat wel binnen $[0,1)$ ligt, zeg $1/2$, af te beelden. Het probleem is nu dat $1/2$ in het domein niet meer kan worden afgebeeld op $1/2$ in het co-domein, immers $1/2$ in het co-domein wordt al geclaimd door 1: $f(1) = 1/2$. Bij de reparatie hebben we dus een nieuwe fout geïntroduceerd. Repareer deze laatste fout en ga door. Laat vervolgens zien dat de gevonden functie inderdaad een bi-jectie is.

24. We zeggen dat verzameling A kleiner of gelijk is aan machtigheid dan verzameling B, als er een verzameling $A' \subseteq B$ bestaat, zo dat $B \sim A'$. In dat geval noteren we $A \leq_1 B$. We zeggen dat A een strict kleinere machtigheid bezit dan B, notatie $A <_1 B$, als $A \leq_1 B$ maar $A \nsim B$.

Reproduceer het bewijs dat elke verzameling een strict kleinere machtigheid bezit dan haar machtsverzameling. (Zie slides.)

25. Laat J een computertaal zijn, bijvoorbeeld Java. We identificeren J voor het gemak met alle (syntactisch correcte) programma's in de taal J. Dus $j \in J \Leftrightarrow j$ is een (syntactisch correct) programma in de taal J. Als programma j ariteit k heeft (k input-waarden nodig heeft), dan noteren we dat met j/k.

(a) Bewijs dat de verzameling J aftelbaar is.

(b) Stel dat er op dit moment N verschillende programmeertalen zijn: $J_1, \ldots, J_N$, neem N maar lekker groot. Beschouw de verzameling van alle programma's die met deze talen te maken zijn.

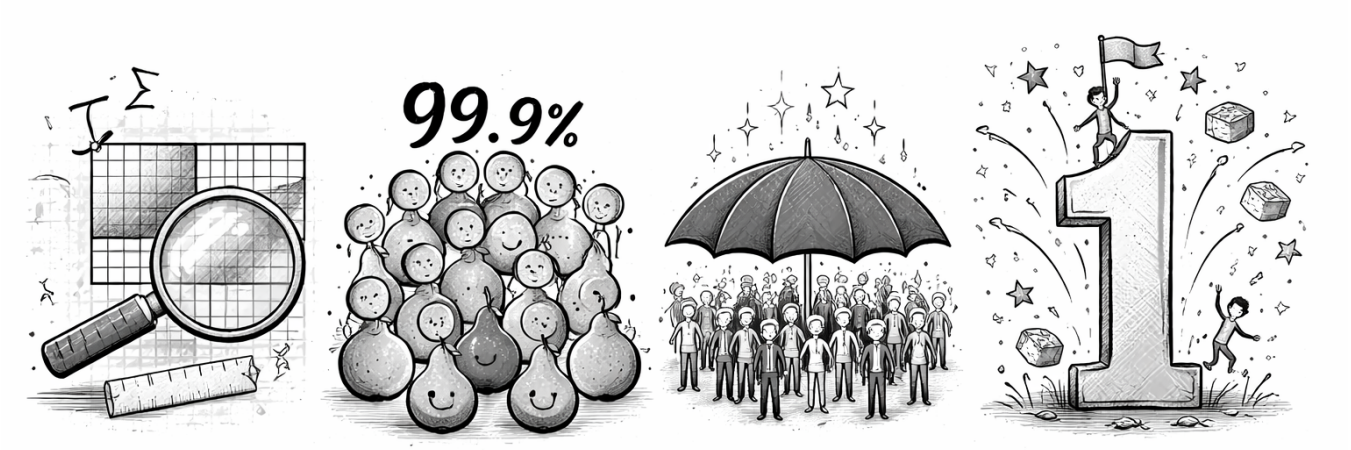
Laat zien dat deze verzameling ook aftelbaar is. (Je kunt een direct bewijs geven of minzaam verwijzen naar één van de resultaten van de opgaven op blz. 9. In dat laatste geval moet je wel nauwkeurig vermelden om welke opgave het gaat, en waarom het resultaat van die opgave van toepassing is.)

(c) Tja, eigenlijk is het moeilijk schatten hoeveel programmeertalen er zijn, laat staan dat je kunt schatten hoeveel talen er ooit nog bij zullen komen. Laten we niet kinderachtig doen: laat $J_1, \ldots, J_N, \ldots$ een aftelbaar oneindige rij zijn van oude talen (Algol, PLI) bestaande talen (C, Java, C#, Ruby, ...) en talen die ooit nog zullen komen (C+++, C#+, ...).

Is deze verzameling aftelbaar? Waarom (niet)?

3 Maat

In de wiskunde en informatica helpt het begrip maat om de omvang van verzamelingen te bepalen. Soms is een groep uitzonderingen zo klein dat deze de algemene regel niet beïnvloedt. We noemen dit een verwaarloosbare verzameling met maat nul. Met de term "bijna alle" kunnen we kort en bondig zeggen dat een eigenschap overal geldt, behalve voor die onbelangrijke uitzonderingen. Dit maakt bewijzen en berekeningen veel efficiënter en overzichtelijker.[2]



(Bron: Gemini 3, prompt: "Generate a playful 300x110 black and white impression of measure, almost all, almost surely, and probability one. No text in image."—jammer dat Gemini er niet op kwam om er een dart board in te zetten.)

> In de de context van een aftelbaar oneindige verzameling betekent "bijna alle": "alle, op eindig veel uitzonderingen na".

Ten opzichte van een aftelbaar oneindige verzameling heeft een eindige verzameling namelijk maat[3] nul—tenminste, als alle elementen even zwaar wegen. We komen hier later op terug.

1. Welke uitspraken zijn waar?

   (a) Bijna alle natuurlijke getallen zijn groter dan 10.

   (b) Bijna alle natuurlijke getallen zijn even.

   (c) Bijna alle priemgetallen zijn oneven.

   (d) Bijna alle vereenvoudigde breuken hebben een teller en een noemer met een absolute waarde groter dan 100.

[2] In TCBoN duikt het begrip "almost all" voor het eerst op in de context van chaos-theorie: "The resulting sequence of heads and tails, for almost all initial starting conditions, will be indistinguishable from true randomness" (Sec. 14.1, p. 224, "Chaos and Randomness"). Flake legt helaas verder niet uit wat dit precies betekent, maar dat gaan we hier dus wel doen.

[3] Algemeen wiskundig woord voor gewicht, of omvang.

(e) Bijna alle breuken hebben, als ze vereenvoudigd zijn, teller en noemer met een absolute waarde groter dan 100.

(f) Bijna alle mensen hebben drie oren.

(g) Bijna alle natuurlijke getallen zijn trouw.

Een getal heet *trouw* als het de som is van drie ongelijke, elkaar delende getallen.[4]

Opzoeken op internet volstaat, je hoeft je antwoord verder niet te bewijzen. (Vraag 2, INMO 2011. Het Engelse woord voor "trouw" is "faithful".)

Waarom werd deze vraag gesteld? Om eens te werken met een verzameling waarvan niet meteen op het eerste gezicht duidelijk is of deze eindig danwel oneindig is.

In de de context van een overaftelbaar oneindige verzameling betekent "bijna alle": "alle, op aftelbaar veel uitzonderingen na".

Ten opzichte van een overaftelbaar oneindige verzameling heeft een aftelbare verzameling namelijk maat nul—tenminste, als alle elementen even zwaar wegen. Dit hoef je nu alleen nog maar als gegeven aan te nemen, je hoeft het nog niet te begrijpen.

2. Welke uitspraken zijn waar?

(a) Bijna alle reële getallen zijn irrationaal (i.e., geen breuk).

(b) Bijna alle reële getallen zijn transcendent (i.e., niet algebraïsch, i.e., geen oplossing van één of andere veelterm).

(c) Bijna alle functies van $\mathbb{N}$ naar $\mathbb{N}$ bezitten geen functievoorschrift. (Een functievoorschrift is een formule die beschrijft wat die functie "doet".)

Hint: een functievoorschrift is een string. Wat is de kardinaliteit van de verzameling van mogelijke strings (uit een vast alfabet)?

(d) Bijna alle punten in $\mathbb{R}^2$ zijn geen roosterpunten.

(e) Bijna geen enkel punt in $\mathbb{R}^2$ is een roosterpunt.

(f) Bijna alle punten in $\mathbb{R}^2$ liggen buiten een roosterlijn.

(g) Bijna alle punten in $\mathbb{R}^2$ liggen buiten de schijf in de oorsprong met straal 10.

Als elementen niet even zwaar wegen, voldoet bovenstaande definitie dus niet meer. We moeten dan gebruik maken van een zg. *kansmaat*. De volgende opgaven zijn om te oefenen met kans-experimenten en hun mogelijk uitkomsten.

3. Beschouw het kans-experiment waarin herhaaldelijk, zonder te stoppen, een munt wordt opgeworpen. Geef de kardinaliteit van de uitkomstenruimte. Motiveer.

4. Beschouw het kans-experiment waarin tien keer een munt wordt opgeworpen. Bij kop noteren we 1, anders 0. Een *realisatie* is een uitkomst van zo'n experiment. Een voorbeeld van een realisatie in dit experiment is 0100010101. Ga na dat dit experiment eindig veel realisaties bezit: 0000000000, 0000000001, 0000000010, ..., 1111111111.

[4] Een natuurlijk getal n heet *trouw* als er natuurlijke getallen $a \mid b \mid c$ ("a deelt b, b deelt c") bestaan, zodanig dat $a + b + c = n$.

(a) Hoeveel?

(b) De verzameling van alle realisaties in een kans-experiment noemen we de *uitkomstenruimte*. Beschouw het kans-experiment waarin herhaaldelijk een munt wordt opgeworpen, totdat een kop verschijnt. Geef de kardinaliteit (of machtigheid, dat is hetzelfde) van de uitkomstenruimte. (I.e., geef aan dat de uitkomstenruimte eindig is, danwel aftelbaar oneindig, danwel gelijkmachtig is met $\mathbb{R}$, en laat zien waarom dat zo is.)

5. Geef (of, althans, beschrijf) de uitkomstenruimte, en de kardinaliteit van de uitkomstenruimte, van de volgende experimenten.

   (a) Tien keer een muntstuk werpen.

   (b) n keer een muntstuk werpen, voor een vaste n, waarbij $n \in \mathbb{N} \cup \{0\}$. Beargumenteer waarom je antwoord voor $n = 0$ juist is.

   (c) Een onbepaald eindig aantal keer een muntstuk werpen. ("Onbepaald" = maakt niet uit hoeveel.)

   (d) Herhaaldelijk een muntstuk werpen, totdat kop verschijnt.

   (e) Herhaaldelijk, zonder te stoppen, een muntstuk werpen.

6. Geef de uitkomstenruimte, en de kardinaliteit van de uitkomstenruimte, van de volgende experimenten.

   (a) Tien keer een dobbelsteen werpen.

   (b) n keer een dobbelsteen werpen, voor een vaste n, waarbij $n \in \mathbb{N} \cup \{0\}$. Beargumenteer waarom je antwoord voor $n = 0$ juist is.

   (c) Een onbepaald eindig aantal keer een dobbelsteen werpen. ("Onbepaald" = maakt niet uit hoeveel.)

   (d) Herhaaldelijk een dobbelsteen werpen, totdat een zes verschijnt.

   (e) Herhaaldelijk, zonder te stoppen, een dobbelsteen werpen.

Als niet alle elementen van een verzameling even zwaar wegen, betekent "bijna alle": "de (kans)maat van de verzameling van uitzonderingen is nul".

7. Bekijk het volgende experiment: net zo lang een dobbelsteen werpen totdat een zes verschijnt. Als een zes verschijnt beschouwen we dat als een succes. De realisatie 34126 is een voorbeeld van een succes, de realisatie 52424143354522 . . . (de stippeltjes suggereren verder gooien zonder ooit op een zes uit te komen) is een voorbeeld van een mislukking.

   Waar of niet waar? "Bijna alle realisaties van dit experiment zijn een succes."

   (a) Als "bijna alle" betekent: "alle, op aftelbaar veel na".

   (b) Als "bijna alle" betekent: "de kans op een mislukking is nul".

8. In vergelijking met Opgave 2 op de pagina hiervoor hanteren we nu het kanstheoretische begrip van 'bijna alle" (de laatste box).

   Waar of onwaar, en waarom?

(a) Bijna alle punten in $\mathbb{R}^2$ zijn geen roosterpunten.

(b) Bijna alle punten in $\mathbb{R}^2$ liggen buiten een roosterlijn.

(c) Bijna alle punten in $\mathbb{R}^2$ liggen buiten de eenheidsschijf. (De eenheidsschijf is de gesloten gevulde cirkel met middelpunt $(0, 0)$ en straal 1. De rand hoort er ook bij, daar slaat het woord "gesloten" op. Als de rand er niet bij hoort wordt de schijf "open" genoemd.)

9. Er is een resultaat dat zegt dat het aandeel priemgetallen in $\{1, \ldots, n\}$ toegroeit naar $(\ln n)/n$ voor stijgende n.[5] Waar of niet waar? "Bijna geen enkel natuurlijk getal is priem."

10. Als een gebeurtenis met kans 1 plaatsvindt, dan zeggen we dat die gebeurtenis *bijna zeker* is, of dat die gebeurtenis *bijna zeker* plaatsvindt (Eng.: *almost surely*, of *a.s.*). Welke van de volgende gebeurtenissen zijn zeker? En bijna zeker?

    (a) Het gooien van een zes, bij herhaaldelijk werpen van een dobbelsteen.

    (b) Bij het willekeurig stoppen van 11 pennen in 10 (oorspronkelijk lege) etuis, is er na afloop een etui met 2 of meer pennen.

    Hint: stop (denkbeeldig) pennen in etuis, en probeer te voorkomen dat in één etui meerdere pennen terechtkomen. Beschrijf wat er gebeurt.

    (c) In een groepje van 15 zijn er twee in dezelfde maand jarig.

    (d) In een groepje van 6 mensen zijn er 3 die elkaar kennen, of 3 die onbekenden voor elkaar zijn. (Elkaar kennen is hier een symmetrische, niet-reflexieve relatie.)

    Je mag het antwoord opzoeken! (Sleutelwoorden: Ramsey, friends, strangers.)

    (e) Het missen van de roos in een dartspel.

    (f) We gaan het dartspel abstraheren.

    Het missen van de oorsprong, bij het aselect kiezen van een punt in de eenheidsschijf. (In vraag 8c wordt uitgelegd wat "eenheidsschijf" betekent.)

    (g) Een onsterfelijke aap ramt op een toetsenbord. En blijft er op rammen.

    Op een gegeven moment verschijnt de regel "Het leven is een feestje, je moet alleen zelf de slingers ophangen."

Samenvatting

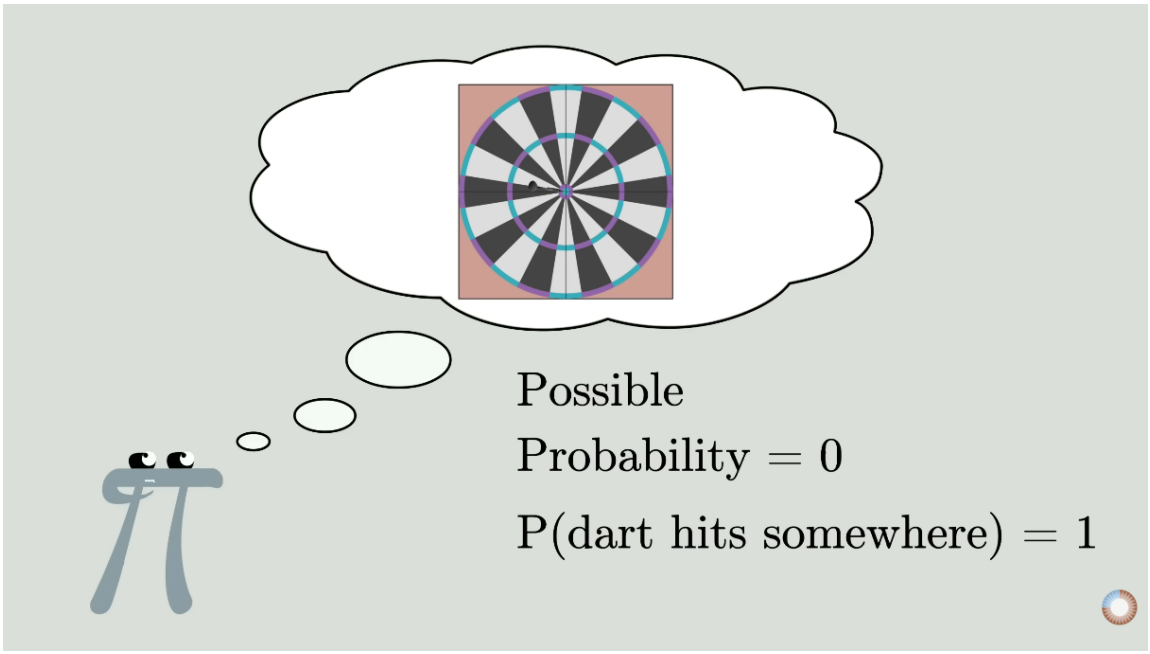
- In de context van een *eindige verzameling* wordt de notie "bijna alle" niet gebruikt.
- In de context van een *aftelbaar oneindige verzameling* (bv. $\mathbb{N}$) betekent "bijna alle": alle, op eindig veel na.
- In de context van een *overaftelbare verzameling* (bv. $\mathbb{R}$) betekent "bijna alle" een van de volgende.

    1. alle, op aftelbaar veel na.
    2. alle, op een verzameling van (kans-) maat nul na.

    (1) impliceert (2), maar niet omgekeerd. Een voorbeeld van het laatste is het antwoord op Opg. 8b

[5] Priemgetallenstelling (Hadamard / de la Vallée Poussin, 1896).

Bekijk ook eens 3Blue1Brown's Youtube video over kans 0 vs. onmogelijk:



Deel III. Berekenbaarheid

4 Turing machines

Een Turing-machine is een theoretisch model van een computer die symbolen leest en schrijft op een oneindige band. In zijn historische paper uit 1936 bewees Alan Turing dat zijn machine, exact evenveel kan als de lambda-calculus van Alonzo Church. Kort daarna werd aangetoond dat Turing's machine evenveel kan als de verzameling recursieve functies voorgesteld door Kurt Gödel en Jacques Herbrand. Men werd het er over eens dat deze drie equivalentie noties (Turing-machine, λ-calculus en recursieve functies) samen een goede vertegenwoordiger waren voor het begrip "berekenbaarheid".

Een bewerking van een Alamy Stock Photo (Origineel in het publieke domein).

Problemen die een Turing-machine niet kan oplossen (zoals het Halting Problem), kunnen door geen enkele computer ter wereld worden opgelost, hoe krachtig ze ook worden. Er kan namelijk worden aangetoond dat elk computerprogramma, in welke taal dan ook, kan worden vertaald naar een programma voor de Turing-machine. Dit geldt ook voor talen die parallellisme en concurrente processen ondersteunen.

1. Een transitie wordt gewoonlijk genoteerd als "A1 → B0<", met de gebruikelijke betekenis: "als in toestand A een 1 gelezen wordt, neem dan toestand B aan, schrijf een 0, en schuif tenslotte de tape één cel naar links". Bijgevolg zal de kop dan één cel naar rechts verschuiven over de tape.

   Om onderscheid te maken tussen toestand-symbolen en tape-symbolen, en eigenlijk voor het gemak, worden voor toestand-symbolen letters gebruikt, en voor print-symbolen cijfers.

   (a) Vaak worden afkortingen gebruikt. Bijvoorbeeld, de transitie "A1 → B1<" kan worden afgekort tot "A1 → B<". Immers, de 1 onder de lees/schrijf-kop blijft ongemoeid. Geef afkortingen voor de volgende transities.

i. A1 → A0<.
ii. K0 → B0>.
iii. Z0 → Z0>.
iv. H1 → H1.

(b) Leg het verschil uit tussen "A1 → B0<" en de volgende transities (zo er een verschil is). Wat is je algemene conclusie?

i. A1 → B0>.
ii. 1A → B0<.
iii. A1 → 0B<.
iv. A1 → B<0.
v. A1 → 0<B.
vi. A1 → <B0.

2. Een 3 wordt in unaire notatie geschreven als 111. Hier is een transitie-tabel voor een Turing-machine welke getallen in unaire notatie omzet in het getal nul (een string van nul 1-en). Deze machine implementeert dus de functie $Z : \mathbb{N} \to \mathbb{N} : n \mapsto 0$.

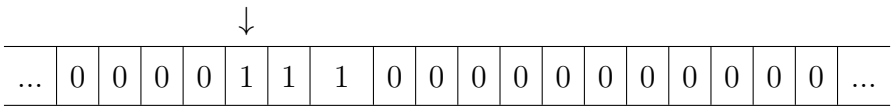
Transitie-tabel:

A1	<
A0	B>
B1	0>
B0	H<

Executie-log voor invoer $N = 3$:

State/see: A/1; do: <
State/see: A/1; do: <
State/see: A/1; do: <
State/see: A/0; do: B>
State/see: B/1; do: 0>
State/see: B/1; do: 0>
State/see: B/1; do: 0>
State/see: B/0; do: H<

lees/schrijf-kop



Uitleg:

- Zolang de machine in toestand A is, en 1-en leest, zal deze de tape naar links bewegen (vandaar het symbool "<"), net zo lang totdat een 0 gelezen wordt. In dit proces beweegt de kop dus naar rechts over de tape. Als een 0 gelezen wordt, zet de machine zichzelf in toestand B, en schuift deze haar tape één vakje naar rechts. De lees/schrijf-kop staat dan over de laatste 1.
- Zolang de machine in toestand B is en 1-en leest, zal deze de 1-en vervangen door 0-en, en de tape naar rechts bewegen, net zo lang totdat de machine op een 0 stuit. (Dit is dan noodzakelijkerwijs de 0 onmiddellijk voorafgaand aan de string 1-en die nu inmiddels vervangen is door allemaal 0-en.) Als de machine op een 0 stuit, beweegt deze haar tape nog één vakje naar links (de kop naar rechts), en zet zichzelf in toestand H (van "halt"). De lees/schrijf-kop is dan teruggekeerd op haar start-cel, zoals dat per conventie geëist wordt.

(a) Schrijf, of omschrijf, de executie-log voor $N = 2$.

(b) Geeft het programma (de transitie-tabel) het correcte antwoord voor $N = 0$?

3. (a) Geef een transitie-tabel voor een Turing-machine die 1 optelt bij een getal.

(b) Geef de executie-log voor $N = 3$.

(c) Geeft je programma (de transitie-tabel) het correcte antwoord voor $N = 0$?

4. Dezelfde drie vragen als bij Vraag 3a, maar nu voor de functie $N \mapsto N + 3$ toegepast op het getal $N = 2$.

Een kantoor-klerk (Eng.: desk clerk; ook wel: computor) die, op basis van een vaste lijst met instructies (rood stapeltje), berekeningen uitvoert op gegeven data (petrol-blauwe berg), en de resultaten daarvan weer teruggeeft aan de opdrachtgever (groene uitdraai). [Een bewerking van een Alamy Stock Photo (Origineel in het publieke domein).]

5. Zoals bekend beschikt elke Turing-machine over een potentieel oneindige tape. Als de machine er aan het rechter-eind af dreigt te lopen, wordt er rechts een extra stuk tape aangeplakt. Analoog voor de situatie waarin de machine er aan het linker-eind dreigt af te lopen.

   Welk soort machine krijgen we als we de Turing-machine beperken tot een eindig stuk tape? Zo maar wat voorbeelden van soorten machines: de eindige automaat, de push-down automaat, de register-machine, de lineair begrensde automaat, Petri-netten, en cellulaire automaten.

   Hint: elke configuratie van de Turing-machine en haar tape (dus de toestand van de Turing-machine op dat moment, waar de lees/schrijf-kop staat, en wat er dat moment allemaal op de tape staat) is ook een toestand. Een “super-toestand” of “snap shot”, zo je wil. Hoe groot is het aantal mogelijke super-toestanden? Eindig of oneindig? Zie super-toestanden in gedachten als cirkels, en zie transities tussen super-toestanden als pijlen. Nu moet je het antwoord wel kunnen geven.

6. We zeggen dat een Turing-machine de functie $f : \mathbb{N} \to \mathbb{N}$ *berekent* als de machine elke unaire representatie van een getal omzet naar haar f-beeld, ook in unaire notatie, en de machine daarna stopt in toestand H op de start-cel. Als voor een getal x de functie ongedefinieerd is, dus als $f(x) = \perp$, zal de machine niet stoppen in toestand H op de start-cel. (In dat geval zal de machine dus op de start-cel stoppen in een andere toestand, of stoppen op een andere cel dan de start-cel, of helemaal niet stoppen.) Kijk bijvoorbeeld naar de functie

$$f : \mathbb{N} \to \mathbb{N} : x \mapsto \frac{3x}{x \,\%\, 2}.$$

De bijbehorende functiewaarden-tabel is daarmee

x	0	1	2	3	4	5	6	7	...
$f(x)$	$\perp$	3	$\perp$	9	$\perp$	15	$\perp$	21	...

Dus als S de start-toestand, en H de halt-toestand is, dan eindigt de configuratie

					S													
					↓													
...	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	...

in

					H													
					↓													
...	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	...

En de configuratie

					S													
					↓													
...	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	...

bijvoorbeeld in

										D								
										↓								
...	1	1	0	0	1	0	1	0	0	0	0	1	0	0	0	1	0	1	...

waarbij D een willekeurige toestand is. Of de berekening eindigt niet, dat kan ook.

Zoals bekend beschikt elke Turing-machine per definitie over slechts eindig veel toestanden. Het idee daarbij is dat het instructieboek van een kantoorklerk slechts eindig veel instructies kan bevatten. Analoog bestaat een computerprogramma ook maar uit eindig veel instructies.

Laat zien dat, als Turing-machines mogen beschikken over oneindig veel toestanden, bijvoorbeeld aftelbaar oneindig veel toestanden, ze elke functie $\mathbb{N} \to \mathbb{N}$ kunnen uitrekenen, waarmee het begrip "berekenbare functie" z'n betekenis verliest. Alles functies van $\mathbb{N}$ naar $\mathbb{N}$ zouden dan ineens berekenbaar zijn, en dan is r niets meer om over te praten.

Hint: introduceer voor elke $x \in \mathbb{N}$ een toestand R_x, met de betekenis dat getal x gelezen is. Introduceer verder voor elke $y \in \mathbb{N}$ een toestand W_y, met de betekenis dat getal y gaat worden geschreven. Overtuig jezelf er vervolgens van dat deze operaties ook werkelijk op een Turing-machine kunnen worden geïmplementeerd. Welke transities tussen toestanden van type R en toestanden van type W moeten verder nu nog worden geïntroduceerd?

7. Beargumenteer, of laat zien, dat, als een Turing-machine mag beschikken over oneindig veel toestanden, Turing's stop-probleem beslisbaar wordt—voor zowel de klasse van machines met eindig veel toestanden, als de klasse van machines met oneindig veel toestanden.

   Hint: gebruik het feit dat, bijvoorbeeld middels Gödelnummering, elke transitie-tabel, en daarmee elke Turing-machine, eenduidig te coderen is als een unair getal x. Dit is intuïtief in te zien doordat gewone computers informatie óók binair opslaan. Op deze manier kan een transitie-tabel als invoer worden aangeboden aan welke Turing-machine dan ook.

8. Het is mogelijk elke transitie-tabel, en daarmee elke Turing-machine, eenduidig (i.e., ondubbelzinnig) te coderen als een unair getal x, zie de uitleg in Vraag 7.

   Een *universele Turing-machine* is een Turing-machine die elke andere Turing-machine kan emuleren, door als invoer (i) een codering van de transitie-tabel van de te emuleren Turing-machine, en (ii) de invoer van de te emuleren Turing-machine op haar tape te zetten, gescheiden door een marker.

   Gebruik voor de volgende vragen Wikipedia, een LLM, of een andere bron.[6]

   (a) Wie stelde het begrip universele Turing-machine voor?

   (b) Wie gaf het eerste voorbeeld van een universele Turing-machine, en wanneer?

   (c) Vervolgens werd gezocht naar minimale universele Turing-machines, dat wil zeggen, universele Turing-machines met zo min mogelijk toestanden en tape-symbolen. Een Turing-machine met m toestanden en n tape-symbolen wordt een (m, n)-Turing-machine genoemd. Welke (m, n) Turing-machine's werden zoal ontdekt, en wanneer?

   (d) Wat is de betekenis en het nut van de notie *universele Turing-machine*?

[6] Mijn generatie is nog geleerd niet te citeren uit Wikipedia. Inmiddels wordt hier wat soepeler mee omgegaan. Als het er echt van afhangt, moet informatie afkomstig van Wikipedia of een LLM natuurlijk worden gedubbelcheckt.

5 Busy Beavers

Een stoppende Turing-machine met N mogelijke toestanden, $N \geq 1$, wordt een *busy beaver* genoemd als deze op een lege tape (een tape met allemaal nullen) na starten minstens zo veel stappen uitvoert als iedere andere stoppende Turing-machine met N toestanden.

Ook wordt een busy beaver gedefinieerd als een Turing-machine die een zo groot mogelijk getal (in unaire notatie) op een lege tape afdrukt als iedere andere Turing-machine met net zo veel mogelijke toestanden. De laatste heet dan een Σ-busy beaver. De eerste definitie (officieel: een S-busy beaver) wordt niettemin vaker gebruikt, en doet misschien ook meer recht aan de naam "busy beaver".

Fig. 1 op de volgende pagina toont enkele bekende waarden.[7]

[7] Op 2 Juli 2024: "We are extremely happy to announce that, after two years of intense collaboration, the goal

N	$S(N)$	$\Sigma(N)$	wanneer ontdekt	
1	1	1		
2	6	4	1963	
3	21	6	1963	
4	107	13	1983	
5	$47,176,870$	$4,098$	2024	[5]
6	$> \underbrace{10^{10^{\cdot^{\cdot^{10}}}}}_{15}$	$> \underbrace{10^{10^{\cdot^{\cdot^{10}}}}}_{15}$	2010	

Fig. 1: Enkele bekende waarden voor twee busy beaver functies.

1. Beargumenteer dat er voor vaste N altijd meerdere busy beavers bestaan.

2. Download het Netlogo Turing-machine model van de cursus-site en voer de daarin opgenomen busy beaver-machine van orde $S = 4$ uit. Na hoeveel stappen zou deze machine officieel moeten stoppen? We zijn niet zo flauw om te vragen dit te controleren.

3. De busy beaver-functie is gegeven door

   $$\mathrm{BB}_S : \mathbb{N} \to \mathbb{N} : N \mapsto \text{het aantal stappen dat een busy beaver van orde } N \text{ uitvoert.}$$

   Beargumenteer dat de busy beaver-functie ook echt een functie is, dat wil zeggen dat voor vaste $N \in \mathbb{N}$ er slechts één getal M is welke staat voor het maximaal aantal stappen van een stoppende automaat van orde N.

   Hint: laat zien dat, voor elke $N \geq 1$, de verzameling van Turing-machines met N toestanden die stoppen, niet leeg, en eindig is. Bedenk verder dat het maximum van een niet-lege en eindige verzameling getallen wel-gedefinieerd is.

4. Een functie heet *totaal* als deze overal gedefinieerd is. Een functie heet *partieel* als deze mogelijk niet overal gedefinieerd is. (Dus de collectie partiële functies omvat de collectie totale functies.) Een functie $B : X \to Y$ heet *strict monotoon stijgend* als voor alle $x_1, x_2 \in X$ geldt dat $B(x_1) < B(x_2)$ zodra $x_1 < x_2$.

   Laat zien dat de busy beaver functie totaal is, en strict monotoon stijgt. (Beide feiten komen uiteraard niet als een verrassing, maar we moeten het toch op enig moment voor eens en voor altijd hebben vastgesteld.)

5. Bewijs dat de busy beaver-functie onberekenbaar is.

   Hint: stel dat de busy beaver-functie wél berekenbaar is, en laat vervolgens zien dat hiermee dan Turing's originele stop-probleem kan worden beslist (wat dus niet kan, zodat de oorspronkelijke aanname noodgedwongen onjuist moet zijn).

6. Een *axiomatische theorie* is, populair gezegd, een formalisme waarbinnen netjes kan worden geredeneerd. Een axiomatische theorie is altijd toegesneden op een domein. Een voorbeeld is de axiomatische theorie van de propositielogica. Er bestaan verschillende axiomatiseringen van de propositielogica. Mendelson's axiomatisering, bijvoorbeeld, bestaat uit drie axioma's

---

of the Busy Beaver Challenge has been reached: the conjecture BB(5) 472 = 47,176,870 is proved. Exceeding our expectations, the proof has been formally written in Coq by collaborator mxdys whose work improves on and/or uses the contributions of more than 10 other bbchallenge contributors." Coq en busy beaver. Go figure ...

(a) $A \to (B \to A)$.

(b) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$.

(c) $(\neg A \to \neg B) \to ((\neg A \to B) \to A)$.

en de afleidingsregel *modus ponens*: $A, A \to B \vdash B$. Hiermee kunnen alle geldige beweringen uit de propositielogica worden afgeleid. Analoog zijn er axiomatiseringen van de predikatenlogica, de rekenkunde, de verzamelingenleer, de theorie van programmacorrectheid, de meetkunde, de kansrekening, de sociale keuzetheorie, speltheorie, onderhandelingstheorie, en nog veel meer.

Alle axiomatische theorieën bezitten de eigenschap dat alle ware beweringen in die theorie systematisch één voor één kunnen worden opgesomd. Dit is niet zo moeilijk voor te stellen: je hoeft er alleen maar voor te zorgen dat de opsomming garandeert dat elke gegenereerde stelling zelf op enig moment ook weer aan de beurt komt om, samen met andere gegenereerde stellingen, nog meer stellingen te genereren. Kijk naar propositielogica. Daar kunnen alle geldige formules worden gegenereerd door simpelweg alle formules systematisch te enumereren (i.e., één voor één op te sommen), om ze meteen bijvoorbeeld met waarheidstafels of semantische tableaus op waarheid te checken. Toon dan vervolgens alleen die formules die waar zijn. Zo worden alle ware formules uit de propositielogica opgesomd.

Beargumenteer dat er geen axiomatische theorie bestaat zodanig dat, voor alle $N \geq 1$, de ware formule $\mathrm{BB}(N) = k$ een stelling (i.e., een afleidbare bewering) is. Hint: uit het ongerijmde. Dan zou je namelijk een algoritme hebben om voor elke $N \geq 1$, de waarde van $\mathrm{BB}(N)$ te bepalen, wat de conclusie van Opgave 5 op de pagina hiervoor zou ondergraven.

Bespreek de intuïtieve betekenis van hetgeen net bewezen is. I.e., wat betekent het nu eigenlijk? Raadpleeg hiervoor eventueel bronnen als [1] en [23].

7. Leg uit, dat uit het vorige onderdeel meteen volgt dat er dus nooit een inductieve formule kan bestaan voor de busy beaver functie á la

$$\mathrm{BB}(N) = [\text{ een formule met } \mathrm{BB}(N-1) \text{ }].$$

8. Laat f en g twee functies $\mathbb{N} \to \mathbb{N}$ zijn. We zeggen dat g *harder stijgt dan* f, als er een punt is waarop g vanaf dan groter is dan f. In wiskunde-taal betekent dit dat er een $N \in \mathbb{N}$ is, zo dat voor alle $N \leq x$ geldt dat $f(x) < g(x)$.

   Een bekend feit is dat alle busy beaver functies, waaronder dus ook BB_S en BB_Σ, harder stijgen dan welke berekenbare functie dan ook. Dit is niet zo makkelijk te bewijzen. Wat we wel kunnen doen is een iets bescheidener resultaat bespreken, in stapjes.

   (a) Laat zien dat geen enkele berekenbare functie harder stijgt dan de functie BB_Σ.[8] Hint: laat zien dat, als dat wel zo is, Turing's stop-probleem beslisbaar is.

   (b) In het vorige onderdeel werd aangetoond dat geen enkele berekenbare functie harder stijgt dan BB_Σ. Moeilijker te bewijzen is de sterkere bewering dat BB_Σ harder stijgt dan elke berekenbare functie. Een iets zwakkere variant kan wel makkelijk worden bewezen.

   Definieer $\mathrm{BB}'_\Sigma(x)$ als het grootste aantal stappen dat een stoppende Turing-machine met x toestanden kan uitvoeren, als op het begin van de tape een getal (in unaire notatie) staat ten hoogste x.

[8] In Opgave 6 op pagina 21 wordt uitgelegd wat het betekent als gezegd wordt dat een Turing-machine M functie f uitrekent.

(c) Beredeneer dat $\mathrm{BB}_\Sigma(x) \leq \mathrm{BB}'_\Sigma(x)$, voor alle $x \in \mathbb{N}$.

(d) Beredeneer dat BB'_Σ harder groeit dan elke berekenbare functie.

9. We gaan nu bewijzen dat BB_Σ harder groeit dan elke berekenbare functie.

(a) Elk getal bezit een unaire representatie en een binaire representatie. Bijvoorbeeld, 6 in binaire notatie is 110, en 6 in unaire notatie is 111111. Ga na (door op internet te zoeken) dat er Turing-machines bestaan die een getal in binaire notatie omzetten naar een getal in unaire notatie.

(b) Beargumenteer dat er een constante K is, zó dat voor elke $N \geq 0$ een Turing-machine M_N bestaat met

$$K + \lceil \log_2 N \rceil$$

toestanden, die op een blanke tape het getal N schrijft, in unaire notatie.

(c) Beargumenteer dat er een constante C bestaat waarvoor geldt dat

$$\mathrm{BB}_\Sigma(C + \lceil \log_2 n \rceil) \geq f(n),$$

voor alle $n \geq 0$, en voor alle berekenbare f.

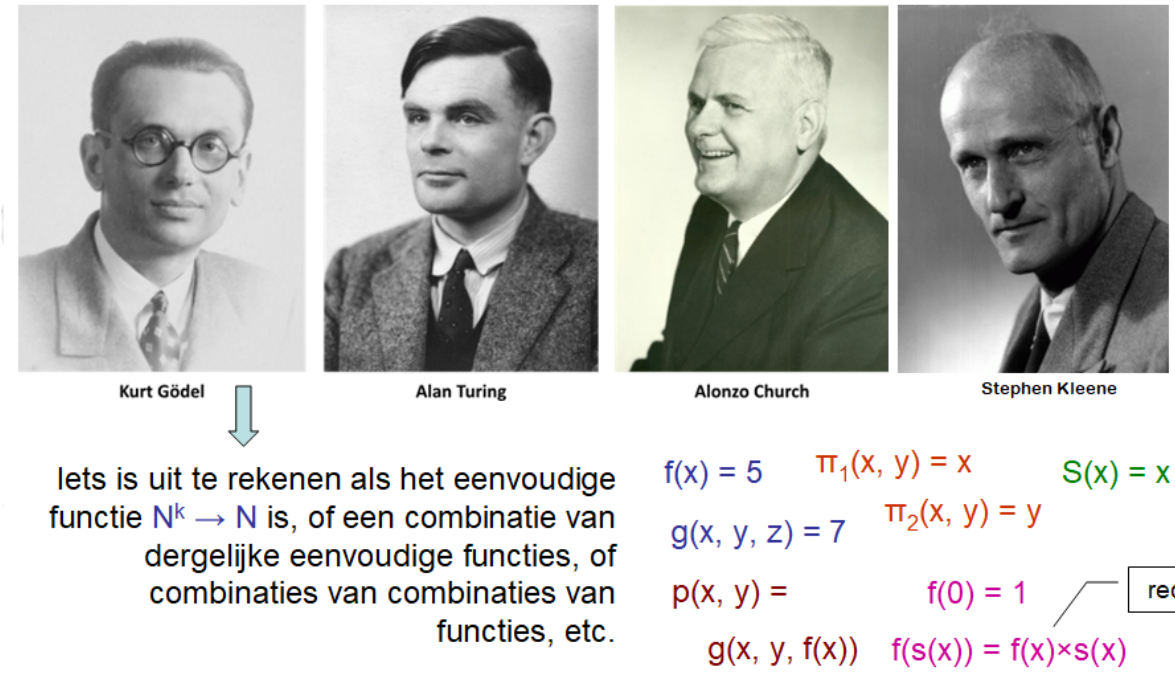
(d) Zij f willekeurig en berekenbaar. Beargumenteer dat, voor voldoend grote n,

$$\mathrm{BB}_\Sigma(n) > f(n).$$

Hint: voor voldoend grote n geldt $n > C + \lceil \log_2 n \rceil$. Gebruik verder het feit dat BB_Σ een strict monotoon stijgende functie is (Opgave 4 op pagina 24).

6 Recursieve functies

Recursieve functies zijn essentieel voor kunstmatige intelligentie omdat ze complexe taken opdelen in kleinere, herhaalbare stappen. Omdat recursieve functies gebaseerd zijn op bekende wiskundige regels (ehm ... functies), dienen ze als betrouwbare bouwstenen voor algoritmen. Hierdoor kan een computer patronen herkennen en zelfstandig problemen oplossen. Het stelt AI in staat om met logische precisie te leren en berekeningen eindeloos te verfijnen.



Gödel, Turing, Church, en Kleene (spreek uit: STEE-ven KLAY-nee, zoals "rainy") legden samen de basis voor de informatica; via logica en algoritmes bewezen ze wat computers wel én absoluut níet kunnen. (Bron: slides "Natuur en Berekening".)

Opgaven:

1. Van de algemeen recursieve functies zijn er twee basis-functies, en één klasse van basis-functies. Benoem deze drie.

2. Van de algemeen recursieve functies zijn er drie constructie-methoden. Benoem deze drie.

Elementaire bouwstenen

Laten we alles, behalve recursie en minimalisatie, "elementaire bouwstenen" noemen.

3. Gebruik de notatie π_k^n voor de n-plaatsige projectie-functie $(x_1, \ldots, x_n) \mapsto x_k$. Waar komt de functie $(x, y, z) \mapsto y$ mee overeen? Is deze functie een elementaire bouwsteen?

4. Laat zien dat de identiteits-functie $I : x \mapsto x$ gelijk is aan één van de projectie-functies. Hiermee heb je laten zien dat de identiteits-functie kan worden geconstrueerd uit elementaire bouwstenen.

5. Laat zien hoe de functie $f(x) = x + 2$ kan worden samengesteld uit elementaire bouwstenen.

6. Dezelfde vraag voor $g(x) = x + 3$.

7. Officieel is er alleen het getal 0. De andere getallen worden gemaakt middels de successor-functie, S, en functie-compositie. Getallen $S(0)$, $S(S(0))$, $S(S(S(0)))$, ... worden dan afgekort tot $S^1(0)$, $S^2(0)$, $S^3(0)$, ..., en zo tot 1, 2, 3, ... Laat zien dat de functie $e(x) = 3$ kan worden samengesteld uit elementaire bouwstenen.

8. Gebruik de notatie c_k^n voor de n-plaatsige constante functie die haar n argumenten altijd op het getal k afbeeldt. Voor alle getallen $x_1, \ldots, x_n$ geldt dan dus $c_k^n(x_1, \ldots, x_n) = k$. Laat zien dat de klasse van constante functies kan worden samengesteld uit elementaire bouwstenen. Hint: gebruik de functies

$$\begin{array}{rrll} \text{projectie op de 1e coördinaat:} & \pi_1^n : & (x_1, \ldots, x_n) & \mapsto x_1 \\ \text{de nul-functie:} & Z : & x & \mapsto 0, \\ +k\text{-functie:} & S^k : & x & \mapsto x + k. \end{array}$$

Primitief recursieve functies

Functie-compositie alleen volstaat niet om alle (naar wat later blijkt: berekenbare) functies $\mathbb{N} \to \mathbb{N}$ te bouwen. Er is recursie nodig. De faculteitsfunctie $f(x) = x!$ is een voorbeeld van een recursieve functie:

$$f(n) = \begin{cases} 0 & \text{als } n = 0, \\ f(n-1) \times n & \text{anders.} \end{cases} \qquad \text{Of:} \qquad \begin{cases} f(0) & = 0 \\ f(n) & = f(n-1) \times n, \ n > 0. \end{cases}$$

Recursie in de theorie van recursive functies heet “primitieve recursie”.

9. We laten zien dat optelling $p(x, y) = x + y$ middels primitieve recursieve kan worden geconstrueerd. Daartoe schrijven we eerst

$$\begin{cases} p(x, 0) & = x \\ p(x, y + 1) & = p(x, y) + 1. \end{cases}$$

   (a) Neem dit format over, en vul op de juiste plaatsen de functies I en S in (identiteitsfunctie, respectievelijk successor-functie). Je hebt nu laten zien dat optelling middles primitieve recursie kan worden gedefinieerd. Daarmee behoort optelling nu officieel tot de klasse van primitief recursieve functies.

   (b) We kunnen nu afleiden dat $1 + 1 = 2$, met alleen de elementaire functies 0, S en p:

$$\begin{array}{rll} 1 + 1 = p(S(0), S(0)) & & \text{definitie van } + \text{ en de getallen 1,} \\ = S(p(S(0), 0)) & & \text{definitie van } p\text{, 2e clausule,} \\ = S(S(0)) & & \text{definitie van } p\text{, 1e clausule,} \\ = 2 & & \text{terugschrijven in gebruikelijke notatie.} \end{array}$$

   Doe hetzelfde voor $2 + 2 = 4$.

10. (a) Laat nu, volgens dezelfde methode als in de vorige opgave, zien dat vermenigvuldiging $m : (x, y) \mapsto xy$ primitief recursief is. Gebruik daarbij het feit dat optelling primitief recursief is. Met andere worden, dankzij het antwoord op de vorige opgave mag je optelling nu gewoon inzetten als bouwsteen. Hint: $x(y + 1) = xy + x$.

    (b) Laat, net zoals in Opgave 9b zien dat $3 \times 2 = 6$, met alleen de elementaire bouwstenen 0, S, p en m. Om het een en ander overzichtelijk te houden mag je 3 schrijven voor $S(S(S(0)))$, en dit ook doen voor andere getallen. Je mag ook $+1$ schrijven voor S. Begin dus met $3 \times 2 = m(3, 2)$. Werk daarna je bewijs om naar successor-notatie, dat wil zeggen, met x' als afkorting voor $S(x)$. Dus dan begin je met $0''' \times 0'' = m(0''', 0'')$.

11. Laat zien dat de faculteitsfunctie primitief recursief is, nu bekend is dat optelling en vermenigvuldiging middels primitieve recursieve kunnen worden geconstrueerd.

12. Laat zien dat machtsverheffing primitief recursief is.

13. We hebben het gehad over primitieve recursie, zonder eigenlijk het algemene schema daarvan te geven. Geef dat nu, voor éénplaatsige functies, zo nodig met hulp van Wikipedia, een LLM, of een andere bron.

14. Geef het algemene schema van primitieve recursie voor meerplaatsige functies.

De Ackermann-functie

Het blijkt dat ontzettend veel functies uit $\mathbb{N}^k \to \mathbb{N}$ middels primitieve recursie kunnen worden geconstrueerd. We noemen: optellen, aftrekken, vermenigvuldigen, delen met rest, machtsverheffen, combinaties daarvan, de grootste gemene deler, de fibonacci-functie, en ga zo maar door. Maar ook de functies "$<$", "$\leq$", "$>$", "$\geq$", en "$=$" zijn primitief recursief als deze worden opgevat als functies van $\mathbb{N}^2$ naar $\{0, 1\}$. De getallen"1" en "0" staat dan respectievelijk voor 'waar' en "niet waar". (We gaan dit later zien met de deler-functie "$a \mid b$" in Opgave 20j op pagina 32.)

Pas in 1928 werd een functie gevonden die níet primitief recursief is. De naam van de persoon die deze functie vond, was Wilhelm Ackermann. De gevonden functie heet dan ook toepasselijk *de Ackermann-functie*. Helaas is de Ackermann-functie zélf niet de meest eenvoudige. Later werden iets eenvoudiger functies gevonden die óók niet primitief recursief zijn. Hier is er zo één, gevonden door de Hongaarse wiskundige Rósza Péter:

$$\begin{cases} A(m, \quad 0 \quad) = m + 1, \\ A(0, \quad n + 1) = A(1, n), \\ A(m + 1, n + 1) = A(\, A(m, n + 1)\, , n). \end{cases} \tag{1}$$

De dubbele recursie in de laatste regel zorgt er voor dat Péter's functie sneller groeit dan alle primitief recursieve functies. (Dit laatste wordt genoemd als feit en hoef je niet in te zien.)

15. Welke rekenkundige operatie(s) gebruikt Péter's functie? (Afgezien van dubbele recursie, wat trouwens geen rekenkundige operatie is.)

16. Vul de volgende tabel met waarden van $A(m, n)$.

$m\rightarrow$ / $n\downarrow$	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0														
1														
2														
3														
4														

Hint: doe dit rij voor rij.

- Begin eerst met rij $n = 0$. Immers, $A(m, 0) = m + 1$ [de 1e clausule uit (1)].
- Nu de rij $n = 1$. Gebruik de relatie $A(0, n + 1) = A(1, n)$ (de 2e clausule) om het eerste element van deze rij uit te rekenen.
  Nu moet rij $n = 1$ verder worden gevuld. Gebruik daarvoor de relatie

  $$A(m + 1, 1) = A(A(m, 1), 0)$$

  [de 3e clausule uit (1), met $n = 1$]. Bijvoorbeeld,

  $$A(1, 1) = A(A(0, 1), 0) = A(2, 0) = 3.$$
  $$A(2, 1) = A(A(1, 1), 0) = A(3, 0) = 4.$$
  $$A(3, 1) = A(A(2, 1), 0) = \ldots$$

  Probeer al doende regelmaat te ontdekken, om zo sneller de rest van de rij te kunnen invullen.
- Nu de rij $n = 2$. Gebruik weer de 2e clausule uit (1) om het eerste element van deze rij uit te rekenen.
  Gebruik nu de relatie

  $$A(m + 1, 2) = A(A(m, 2), 1)$$

  (de 3e clausule uit (1), met $n = 2$). Bijvoorbeeld,

  $$A(1, 2) = A(A(0, 2), 1) = A(3, 1) = 5.$$
  $$A(2, 2) = A(A(1, 2), 1) = A(5, 1) = \ldots$$
  $$A(3, 2) = A(A(2, 2), 1) = A(7, 1) = \ldots$$

  Probeer weer regelmaat te ontdekken.
- Nu de rij $n = 3$. Gebruik weer de 2e clausule uit (1) om het eerste element uit te rekenen. Gebruik vervolgens de relatie

  $$A(m + 1, 3) = A(A(m, 3), 2)$$

  om $A(1, 3)$ en $A(2, 3)$ uit te rekenen. Bijvoorbeeld:

  $$A(1, 3) = A(A(0, 3), 2) = A(5, 2) = 13.$$
  $$A(2, 3) = A(A(1, 3), 2) = \ldots$$

  Je ziet dat je tabel, die maar loopt tot en met kolom $m = 13$, niet groot genoeg is om $A(3, 3)$ uit te rekenen.

- Rekenen tenslotte $A(0,4)$ uit.
- Op welke problemen stuit je als je $A(1,4)$ wil uitrekenen?

17. Het volgende kan worden bewezen (hoef je niet te doen):

$n\downarrow$ \ $m\rightarrow$	0	1	2	...	m	...
0					$m+1$	
1					$2+(m+3)-3$	
2					$2\times(m+3)-3$	
3					$2^{(m+3)}-3$	
4					$\underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{m+3}-3$	

18. Bereken $A(0,0)$, $A(1,1)$, $A(2,2)$, en $A(3,3)$.

19. Bereken $A(4,4)$.

Samenvatting: veel (wat later blijkt: berekenbare) functies kunnen worden verkregen middels primitieve recursie, maar sommige niet, waaronder de Ackermann-functie.

Alle recursieve functies

Om, buiten de primitief recursieve functies, *alle* berekenbare functies, inclusief de Ackermann-functie, te kunnen uitdrukken als recursieve functie, is μ-minimalisatie (kortweg minimalisatie) nodig. Hier is de definitie voor 2-plaatsige functies.[9]

$$\mu y f(x,y) = \begin{cases} \text{de minimale waarde van } y \text{ waarvoor } f(x,y)=0 & \text{als zo'n } y \text{ bestaat,} \\ \bot & \text{anders.} \end{cases}$$

Opmerkingen.

- Het symbool "$\bot$" ("bottom") staat voor "ongedefinieerd".
- $\mu y f(x,y)$ is een 1-plaatsige functie $\mathbb{N}\to\mathbb{N} : x \mapsto \mu y f(x,y)$. We kunnen dus schrijven $g(x) = \mu y f(x,y)$.
- Voor een minimale waarde $y*$ moet gelden dat $f(x,y)$ óók gedefinieerd is voor alle $y < y*$. Deze eis wordt gesteld omdat μ-minimalisatie intuïtief correspondeert met het itereren van de rij $f(x,0)$, $f(x,1)$, $f(x,2)$, ..., net zo lang totdat $f(x,y*)=0$. Tijdens het genereren van die rij moeten alle tussentijdse berekeningen een resultaat teruggeven. Voor geen enkele $y < y*$ mag de berekening "hangen" of "crashen".

  μ-Minimalisatie kan worden gezien het introduceren van de while-loop in de theorie van recursive functies. Bekend is dat programmeertalen niet Turing-compleet zijn zonder een while-loop of vergelijkbare constructie, en zo is dat ook met de theorie van recursive functies en μ-minimalisatie. (Dit hoef je zelf verder niet aan te tonen.)

[9] Er bestaat ook een definitie voor n-plaatsige functies, $n > 2$. Deze verloopt analoog, maar hebben we hier verder niet nodig.

Voorbeelden:

- Als $f(x,y) = |y - 2x|$, dan is $\mu y f(x,y) = 2x$. Immers, $y = 2x$ is de laagste waarde van $y \geq 0$ zodanig dat $|y - 2x| = 0$.
- Als $k(x,y) = |x - y^2|$, dan

$$\mu y k(x,y) = \begin{cases} \sqrt{x} & \text{als } x \text{ een kwadraat is,} \\ \perp & \text{anders.} \end{cases}$$

  Immers, als x geen kwadraat is, zal $|x - y^2| \neq 0$ voor alle $y \in \mathbb{N}$. Omdat k op sommige waarden ongedefinieerd is, wordt k een *partieel recursieve functie* genoemd. In feite worden alle recursieve functies partieel recursieve functies genoemd, omdat in de constructie mogelijk μ-minimalisatie is gebruikt.

20. Geef van de volgende functies $g(x) = \mu y f(x,y)$.

    (a) $f(x,y) = y$.

    (b) $f(x,y) = x$. Hint: maak onderscheid tussen de gevallen $x = 0$ en $x > 0$.

    (c) $f(x,y) = x + y$. Gebruik ook weer gevals-onderscheid.

    (d) De functie $\mathbb{N} \times \mathbb{N} : (x,y) \mapsto x \,\%\, y$ wordt de *modulo-functie* genoemd. Het geeft de rest bij deling. Voorbeelden:

$$\begin{array}{lllllll} 12 \,\%\, 0 = \perp, & 12 \,\%\, 1 = 0, & 12 \,\%\, 2 = 0 & 12 \,\%\, 3 = 0, & 12 \,\%\, 4 = 0, & 12 \,\%\, 5 = 2, & 12 \,\%\, 6 = 0, \\ 12 \,\%\, 7 = 5, & 12 \,\%\, 8 = 4, & 12 \,\%\, 9 = 3, & 12 \,\%\, 10 = 2, & 12 \,\%\, 11 = 1, & 12 \,\%\, 12 = 0, & 12 \,\%\, 13 = 12 \end{array}$$

    Immers, 13 past nul keer in 12 en dan hou je 12 over. Verder geldt $n \,\%\, 0 = \perp$ en $0 \,\%\, n = 0$, voor alle $n \geq 0$.

    $f(x,y) = x \,\%\, 3$.

    (e) $f(x,y) = y \,\%\, 3$.

    (f) $f(x,y) = 3 \,\%\, x$.

    (g) $f(x,y) = 3 \,\%\, y$. Let op, dit is een listige. Herlees zo nodig de 3e opmerking hierboven.

    (h) $f(x,y) = x \,\%\, y$.

    (i) $f(x,y) = y \,\%\, x$.

    (j) Laat

$$a \mid b =_{Def} \begin{cases} 1 & \text{er is een } x \in \mathbb{N} : ax = b, \\ 0 & \text{anders.} \end{cases}$$

    Als $a \mid b = 1$ zeggen we: "a deelt b of: "a is een deler van b". Met deze definitie is $a \mid b$ gedefinieerd voor alle $a, b \in \mathbb{N}$. Voorbeelden: $7 \mid 14$, $8 \nmid 14$, $4 \mid 12$, en $5 \nmid 12$. Verder is elk getal een deler van 0, en is 0 is alleen een deler van zichzelf: $7 \mid 0$, $0 \nmid 7$, en $0 \mid 0$.

    $f(x,y) = x \mid y$.

    (k) $f(x,y) = y \mid x$.

    (l) $f(x,y) = (y+1) \mid x$. Dit is een moeilijker opgave; de vorige twee opgaven waren in feite opwarmertjes voor deze opgave. Er kan alvast verklapt worden dat $\mu y f(x,y)$ overal gedefinieerd is.

We laten het hierbij.

Samengevat zijn er primitief recursieve functies, en partiëel recursieve functies.

- Primitief recursieve functies worden opgebouwd met functie-compositie (kortweg: compositie) en primitieve recursie (kortweg: recursie). Maar μ-minimalisatie (kortweg: minimalisatie) is niet toegestaan. Bijgevolg zijn primitief recursieve functies altijd op al hun argumenten gedefinieerd. Ze zijn, zoals dat heet, *totaal*.
- Partiëel recursieve functies zijn mogelijk ook opgebouwd met minimalisatie. Bijgevolg kunnen partiëel recursieve functies op sommige van hun argumenten ongedefinieerd zijn: ze kunnen *partieel* gedefinieerd zijn.

De verzameling van *alle* recursieve functies (ook wel: algemeen recursieve functies) is gelijk aan de verzameling van functies die met alle mogelijke constructie-methoden kunnen worden gebouwd, en is dus gelijk aan de verzameling van partiëel recursieve functies. Kortweg: recursive functies = algemeen recursieve functies = partiëel recursieve functies.

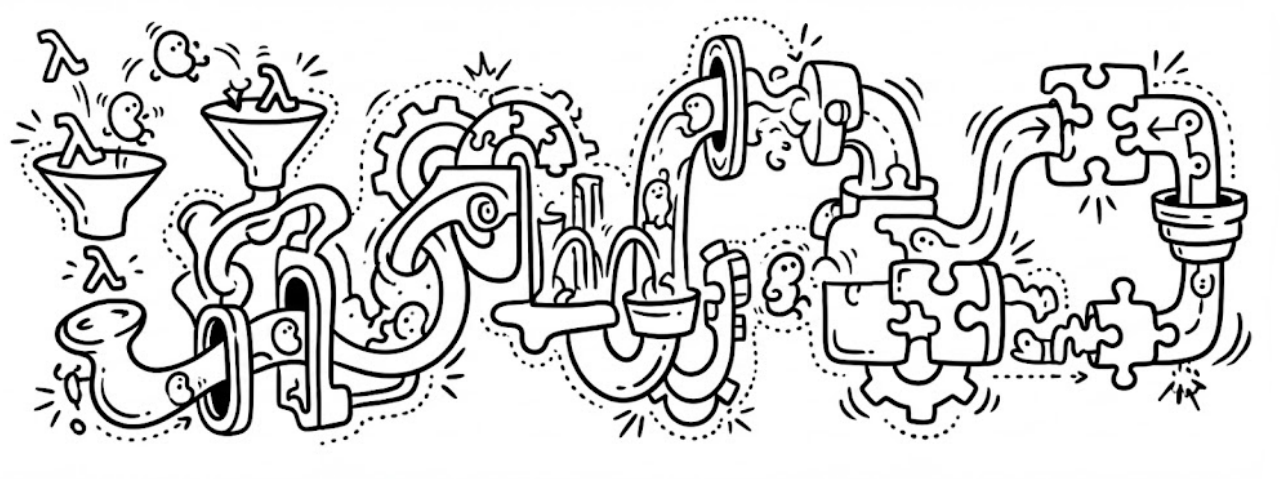
Kurt Gödel's schokkende (jawel) onvolledigheidsstellingen (1931) legde de basis voor de ontwikkeling van de recursieve functie-theorie. Deze theorie werd verder ontwikkeld door Stephen Kleene, Emil Post, en Alan Turing.

Al gauw bleek dat recursieve functies Turing-compleet zijn. Dit kan worden bewezen door een Turing-machine te emuleren middels recursieve functies, maar in echt werd dit bewezen door een al ander Turing-compleet mechanisme, de zogenaamde μ-*calculus*, te emuleren met recursieve functies. De μ-*calculus* is een soort propositielogica, uitgebreid met programmeer-achtige constructies. Deze wordt hier verder niet besproken.

Een eenvoudiger calculus, welke ten grondslag ligt aan de theorie van de berekenbaarheid is de zogenaamde λ-calculus: spreek uit "lambda-calculus". Deze is zo belangrijk dat hier een apart hoofdstuk aan gewijd is.

7 De lambda-calculus

De lambda-calculus, uitgevonden door Alonzo Church, is de basis van het functioneel programmeren (LISP, Haskell, Erlang, OCaml, Scala, F#, Clojure, Elixir, ML, Scheme). In plaats van stap-voor-stap instructies, draait alles om functies die andere functies aanroepen. Voor de informatica bewees het dat simpele logica complexe berekeningen kan oplossen. In de AI helpt dit systeem om abstracte concepten en computertalen te begrijpen, wat essentieel is voor het bouwen van slimme algoritmen.



Een associatieve impressie van de lambda-calculus (Bron: Gemini 3).

1. Evalueer de volgende lambda-expressies.
   (a) $(\lambda x.x^2)3$.
   (b) $(\lambda x.x^2)7$.
   (c) $(\lambda w.u + w)4$.
   (d) $(\lambda x.x + y^2)5$.
   (e) $(\lambda y.x + y^2)3$.
   (f) $(\lambda u.u + w^2)9$.
   (g) $(\lambda x.x^2)[(\lambda y.y + 1)4]$. (Hint: van binnenuit.)
   (h) $(\lambda y.y + 1)[(\lambda x.x^2)7]$.
   (i) $(\lambda y.y + z)[(\lambda x.x^2)7]$.
   (j) $(\lambda w.w)(\lambda x.x)$.
2. Evalueer de volgende lambda-expressies.
   (a) $(\lambda x.(\lambda y.x))8$.
   (b) $(\lambda x.(\lambda y.x^y))3$.

(c) $(\lambda y.(\lambda x.x^y))3$.

(d) $\lambda x.((\lambda y.x^y)2)$. (Hint: van binnenuit.)

(e) $(\lambda x.((\lambda y.x^y)2))3$. Er zijn twee manieren: van binnenuit evalueren, en van buitenaf evalueren. Probeer beiden eens, en ga dan na dat je op hetzelfde antwoord uitkomt.

Hoe expressies in de λ-calculus ook geëvalueerd worden, altijd komt er hetzelfde uit. Officieel wordt gezegd dat de ongetypeerde lambda-calculus *confluent* is, of de *Church-Rosser* eigenschap bezit.

(f) $(\lambda x.(\lambda y.x^y))2$.

(g) $((\lambda x.(\lambda y.x^y))2)3$.

(h) De expressie $\lambda xyz.(\dots)$ is een afkorting voor $(\lambda x.(\lambda y.(\lambda z.(\dots))))$.
Evalueer nu $(((\lambda xy.x^y)2)3)$.

3. In λ-calculus kan propositie-logica worden bedreven. Laat

$$\begin{aligned} T &=_{Def} \lambda x.(\lambda y.x), \\ F &=_{Def} \lambda x.(\lambda y.y), \\ \neg &=_{Def} \lambda x.((xF)T). \end{aligned}$$

Bewijs nu: $\neg T = F$. Tip: om λ-expressies beter uit elkaar te kunnen houden, is het handig om elke expressie haar eigen variabelen te geven. Laat in dit geval bijvoorbeeld $T = \lambda u.(\lambda v.u)$.

4. Bewijs ook $\neg F = T$. Geef elke expressie weer haar eigen variabelen.

5. Laat

$$\text{of} =_{Def} \lambda x.(\lambda y.(xT)y).$$

Bewijs: of $TF = T$. Haakjes associëren naar links, dus of $TF = (\text{of}\ T)F$. Hint: een tussenresultaat is $(TT)F$. Schrijf vervolgens het eerste voorkomen van T voluit.

6. Bewijs: of $FF = F$.

7. In de λ-calculus kan ook worden geprogrammeerd. Laat

$$\text{if } a \text{ then } b \text{ else } c =_{Def} abc$$

Bewijs: "if T then a else $b = a$". Hint: expandeer op enig moment T. Bewijs ook "if F then a else $b = b$".

8. In de λ-calculus kan ook worden gerekend. Daartoe moeten eerst getallen worden gedefinieerd.[10] Church, de bedenker van de λ-calculus, stelde het volgende voor:

$$\begin{aligned} 0 &=_{Def} \lambda f.(\lambda x.x) \\ 1 &=_{Def} \lambda f.(\lambda x.f(x)) \\ 2 &=_{Def} \lambda f.(\lambda x.f(f(x))) \\ 3 &=_{Def} \lambda f.(\lambda x.f(f(f(x)))) \\ \vdots &\qquad \vdots \end{aligned}$$

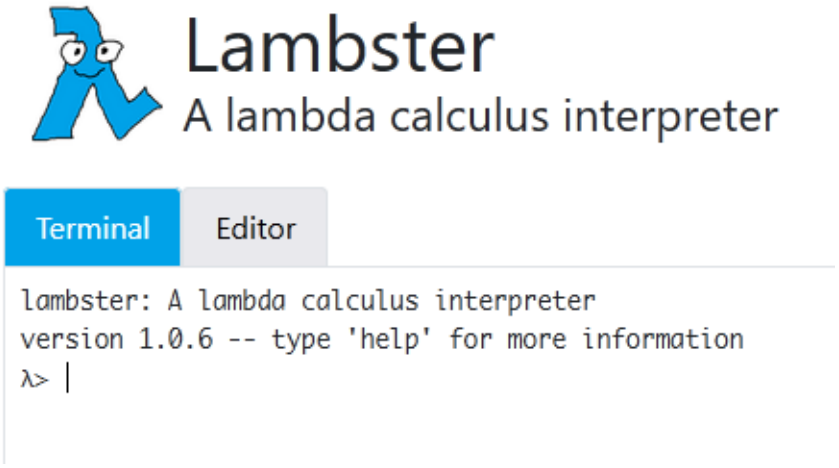
[10] De oplettende lezer zal opmerken dat in de eerste opgave al werd gewerkt met getallen terwijl deze nog niet officieel waren gedefinieerd. Dat is waar. Maar om in een eerste opgave al meteen te beginnen met Church numerals is ook zo wat.

Bijvoorbeeld, het getal 3 staat voor de handeling waarbij een bepaalde functie drie keer op een waarde wordt toegepast. Als getallen zo worden gedefinieerd worden ze *Church numerals* genoemd.

Laat

$$M + N =_{Def} \lambda x.(\lambda y.(Mx)((Nx)y))$$

Bewijs: $1 + 1 = 2$. Gebruik daarvoor een online λ-calculus interpretator, bijvoorbeeld Lambster.



8 Turing's stop-probleem

Het stop-probleem is de vraag of er een programma bestaat dat voor elk willekeurig ander programma (met bijbehorende input) kan voorspellen of het ooit stopt of eeuwig blijft draaien. Alan Turing bewees dat zo'n algemeen programma onmogelijk is.

Om dit te visualiseren gebruiken we de zogenaamde *halting-tabel*. In deze tabel staat elk mogelijk programma in een rij en elke mogelijke input in een kolom. De cel op het kruispunt zou dan "stopt" (1) of "Stopt niet" (0) moeten bevatten.

h	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	…
q	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	…
j_1	1	1	1	1	0	1	1	0	1	…
j_2	1	1	1	0	0	1	0	1	0	…
j_3	1	0	0	1	1	1	1	1	1	…
j_4	1	1	1	1	1	1	1	1	1	…
j_5	1	1								
j_6	1	1								
j_7	0	0								
j_8	0	1								
j_9	1	1								
…	…	…								

- Rij *q/1* verschilt van alle rijen j_i */1*.
- Dus de *q/1* is geen van j_i, dus niet programmeerbaar.
- Maar met hulp van *h/2* is makkelijk code te schrijven voor *q/1*. (Ga na!)

  Dus *q/1* is wel programmeerbaar--tegenspraak.
- De aanname dat *h/2* bestaat is dus onhoudbaar.

Turing bewees met een slimme logische truc (diagonaal-bewijs) dat er altijd situaties zijn waar de tabel zichzelf tegenspreekt. Hierdoor weten we dat er een harde grens zit aan wat computers kunnen begrijpen: ze kunnen niet over hun eigen gedrag in de toekomst beslissen zonder het simpelweg uit te voeren. (Bron: slides college "Natuur en Berekening".)

Onderstaande opgaven horen bij Hoofdstuk 3: "Computability and Incomputability" uit The Computational Beauty of Nature, Gary W. Flake, MIT Press, 2000.

1. Formuleer in eigen woorden de Church-Turing these. Beantwoord zo nodig voor jezelf eerst de volgende vragen.

   - *De "universele gereedschapskist".* Denk aan het idee dat het niet uitmaakt of je een probleem probeert op te lossen met een Turing-machine, de lambda-calculus of een moderne taal als Python. Wat zegt dit over de grens van wat er überhaupt uit te rekenen valt?
   - *Van intuïtie naar definitie.* We hebben allemaal een gevoel bij wat "stap voor stap uitrekenbaar" betekent (intuïtie). Hoe probeert deze these dat vage gevoel om te zetten in een harde, wiskundige grens?

- *De "gelijke kracht".* Waarom denk je dat we nooit een programmeertaal hebben uitgevonden die méér kan dan de simpele Turing-machine uit 1936? Focus op de ontdekking dat al deze verschillende systemen uiteindelijk tegen dezelfde muur aanlopen.

2. In de volgende opgaven wordt steeds gesproken over "Java-programma's", eigenlijk alleen maar om een concrete programmeertaal in gedachten te nemen. Maar er is niets bijzonders aan Java. Dankzij de Church-Turing these kan deze worden vervangen door elke andere programmeertaal, bijvoorbeeld C, C#, Python, of Netlogo. In de praktijk blijken alle programmeertalen even krachtig als Java. (Misschien niet allemaal even handig of efficiënt, maar dat is wat anders.)

   Wat is de officiële term voor "even krachtig als" in de context van berekeningsmechanismen? Hint: de naam "Turing" komt er in voor.

3. Een Java-programma j welke precies n argumenten nodig heeft om te kunnen werken bezit plaatsigheid, of *ariteit* n, notatie j/n. We spreken ook wel over een n-Java programma. Een Java-programma dat bijvoorbeeld gemeenschappelijke klinkers uit twee woorden haalt, zal waarschijnlijk 2-plaatsig zijn. Immers, de invoer zal typisch bestaan uit twee woorden.

   Welke ariteit zal een programma p hebben dat alle priemgetallen één voor één afdrukt? Hoe noteer je dat met een schuine streep?

4. In deze en volgende opgaven oefenen we met reductie-argumenten, dit hebben we later nodig. Een reductie-argument is een redenering van het type: "We hebben een programma p dat over instanties (i.e., gevallen) b van probleem B kan beslissen. Daarnaast hebben we een programma t (een "vertalingsprogramma") dat instanties a van probleem A automatisch kan vertalen naar instanties b van probleem B. Dus hebben we een programma $p \circ t$ (spreek uit: "p na t") dat over instanties a van probleem A kan beslissen:

   $$p \circ t : A \longrightarrow_t B \longrightarrow_p \{\text{ja}, \text{nee}\}."$$

   Ga voor jezelf na dat programma $p \circ t$ nu inderdaad over instanties van A kan beslissen.

   (Reductieopgave.) Stel, het vermenigvuldigen van twee getallen is niet triviaal (we weten niet hoe dit moet), en stel dat iemand een algoritme gevonden heeft om twee getallen te vermenigvuldigen—"Eureka!" Beargumenteer dat er dan ook een algoritme bestaat voor het uitrekenen van ne machten, voor willekeurige n.[11]

   Geldt het omgekeerde ook? Dus zou een algoritme voor machtsverheffen een algoritme voor vermenigvuldigen kunnen opleveren? Waarom (niet)?

5. (Reductieopgave.) SAT is het probleem om een model te vinden voor een CNF in de propositielogica. (Raadpleeg zo nodig de cursus "Inleiding Logica".) 3SAT is het schijnbaar eenvoudiger probleem om een model te vinden voor een 3CNF. Een 3CNF is een CNF waarbij elke term uit ten hoogste drie literals bestaat. Een literal is een propositieletter of de negatie van een propositieletter.

   Laat zien dat 3SAT desalniettemin minstens zo moeilijk is als SAT.

   Hint 1: laat zien dat instanties van SAT eenvoudig automatisch vertaald kunnen worden naar instanties van 3SAT. Hint 2: bekijk de term

   $$x_1 \vee \neg x_2 \vee \neg x_3 \vee x_4 \vee x_8 \tag{2}$$

[11] Bij het woord algoritme ligt de klemtoon op de i. Dus: algorítme.

en de 3CNF

$$(x_1 \vee y_1) \wedge (\neg y_1 \vee \neg x_2 \vee y_2) \wedge (\neg y_2 \vee \neg x_3 \vee y_3) \wedge (\neg y_3 \vee x_4 \vee y_4) \wedge (\neg y_4 \vee x_8).$$

Welke modellen maken (2) waar? Welke modellen maken (5) waar?

6. (Reductieopgave.) Neem een ongericht netwerk in gedachten, bijvoorbeeld een spoorwegnetwerk. Een *overdekking* is een verzamelingen punten in dat netwerk, zodanig dat elke verbinding tussen twee punten grenst aan tenminste één ander punt. Een verzamelingen punten noemen we *onafhankelijk* als geen enkel paar uit die verzameling verbonden is—denk aan snackbars op treinstations: voor de volledigheid zouden we willen dat aan elke lijn een snackbar ligt (overdekking), en uit economische overwegingen liever niet meer dan één snackbar (onafhankelijk).

   Stel, iemand heeft een algoritme gevonden dat, voor willekeurige n, kan bepalen of een netwerk een overdekking van n punten bezit. Beargumenteer dat er dan ook een algoritme bestaat dat, voor willekeurige k, kan bepalen of een netwerk een onafhankelijke verzameling ter grootte k bezit. Hint: gebruik het feit dat, als S een netwerk overdekt, het complement van S (alle punten buiten S) onafhankelijk is. Het omgekeerde geldt trouwens ook: als S onafhankelijk is, dan is S^c een overdekking.

7. (Onbeslisbaarheid van het invoerloze stop-probleem.) Het stop-probleem is het vraagstuk of er een computerprogramma $h/2$ bestaat dat voor alle programma/input-combinaties (j, i) kan beslissen of programma $j/1$ met input i stopt. Turing liet met een diagonaalargument zien dat zo'n computerprogramma h (h verwijst naar "halting") niet kan bestaan. Het stop-probleem is daarmee onbeslisbaar.

   Toon met een reductie-argument aan dat er zelfs geen programma (of mechanische procedure of algoritme) $h/1$ kan bestaan dat voor alle invoer-loze programma's kan beslissen of deze stoppen. (Het antwoord is te vinden in de slides. Belangrijk is hier dat je nu het antwoord in eigen woorden kan geven.)

8. Beargumenteer de onbeslisbaarheid van de volgende problemen. Aanwijzing: gebruik voor elk van die problemen een reductieargument. I.e., neem aan dat er toch een beslis-algoritme zou bestaan voor dit probleem, en laat dan zien dat er dan ook een algoritme zou bestaan voor een probleem waarvan al bekend is dat het onbeslisbaar is (bijvoorbeeld het invoerloze stop-probleem).

   (a) (Het print-probleem.) Bepaal voor een willekeurig programma j en een willekeurig symbool a of j ooit a zal afdrukken.

   (b) (Het uniforme stop-probleem.) Bepaal voor een willekeurig programma $j/1$ of het stopt op alle invoer.

   (c) (Het programma-equivalentieprobleem.) Bepaal voor twee willekeurige $j/0$ en $j'/0$ of deze zich hetzelfde gedragen, i.e., beiden hetzelfde afdrukken en beiden stoppen/niet stoppen.[12]

9. Een perfecte virusscanner is een programma, $s/1$, die voor elke input v kan bepalen of v een virus is. Beargumenteer, ook weer met een reductie-argument, dat er geen perfecte virusscanner kan bestaan. (Het antwoord is te vinden in de slides. Reproduceer dit nu in eigen woorden.)

10. (Beslisbaarheid van bewijsbaarheid.)

[12] Jammer. Wat zou het fijn zijn als informaticadocenten nooit meer programmeerhuiswerk hoeven na te kijken.

(a) Is bewijsbaarheid in de propositielogica beslisbaar? Zo ja hoe? Zo nee, waarom niet?

(b) Dankzij de cursus Inleiding Logica weet je dat bewijsbaarheid in predikatenlogica onbeslisbaar is. Formuleer in eigen woorden wat dat betekent.

(c) Bewijsbaarheid in de predikatenlogica is overigens wel semi-beslisbaar. Wat betekent dat? En hoe bruikbaar is dat? I.e., kun je er nog wat mee?

(d) Hoe zou onbeslisbaarheid van bewijsbaarheid bewezen kunnen worden? Je hoeft geen bewijs te geven, alleen een idee hoe dit eventueel zou kunnen. Hint: reductie.

11. Beargumenteer dat voor programma's met een *eindige* (ook wel: begrensde) lees- en schrijfruimte, het stop-probleem beslisbaar is.

    Hints:

    (a) Gebruik de notie "toestand van een systeem". De toestand van een systeem is een complete momentopname (Engels: "snapshot") van dat systeem. Vergelijk het met een save-game of een hibernate-bestand: in dergelijke bestanden staat voldoende informatie om te weten hoe verder te gaan na inlezen.

    (b) Gebruik het gegeven dat, als een systeem (game, programma, machine, ...) slechts eindig veel toestanden kan aannemen, en het systeem maar blijft doorlopen, dan het dan op den duur een toestand zal moeten herbezoeken.[13]

12. Waarom is het antwoord op Vraag 11 niet in strijd met de onbeslisbaarheid van het stop-probleem? In Turing's diagonalisatie-bewijs wordt namelijk nergens geëist of gebruikt dat programma's onbeperkt veel schrijfruimte mogen gebruiken ... ?

13. Laat $n \in \mathbb{N}$. Nagaan of een programma meer dan nMB aan lees- en schrijfruimte gebruikt, is beslisbaar. Hint: bekijk de redeneringen en het antwoord bij Vraag 11.

14. Het vermoeden van Collatz zegt dat, voor elk positief getal n, herhaalde toepassing van

$$f(n) = \begin{cases} n/2 & \text{als } n \text{ even,} \\ 3n+1 & \text{anders.} \end{cases}$$

    op positieve gehele getallen altijd naar 1 leidt. Bijvoorbeeld, voor 3 hebben we $3 \to 10 \to 5 \to 16 \to 8 \to 4 \to 2 \to 1$. (Inderdaad: voor 10, 5, 16, 8, 4 en 2 is het vermoeden van Collatz hiermee nu ook getest.) Tot nu toe is het vermoeden van Collatz bewezen noch weerlegd.

    (a) Leg uit dat, als het uniforme stop-probleem beslisbaar is, we een computer uitsluitsel kunnen laten geven over het vermoeden van Collatz.

[13] Als je iets van schaken afweet, kun je een analogie trekken met de "3× dezelfde stelling"-regel, die zegt dat bij drie keer dezelfde stelling (dezelfde stukken op dezelfde velden, met ook dezelfde speler aan de beurt, dezelfde zetmogelijkheden, en niet noodzakelijk opeenvolgend), beide spelers het recht hebben remise te claimen. Omdat er "slechts" eindig veel stellingen zijn, werkt dit al in de hand dat schaakpartijen op den duur eindigen (ga na!). Maar het is geen garantie. Er moet namelijk een moment zijn dat tenminste één speler inderdaad zo'n herhaling herkent én remise claimt. Als dat laatste niet gebeurt is er altijd nog de "5× dezelfde stelling"-regel, die zegt dat remise automatisch optreedt bij vijf herhalingen. De "5× dezelfde stelling"-regel werd overigens pas in 2014 ingevoerd. Dus pas sinds 2014 kan echt worden gegarandeerd dat schaakpartijen altijd eindigen!—mits iets of iemand bijhoudt welke stellingen op het bord verschijnen en een seintje geeft als voor de vijfde maal eenzelfde stelling op het bord verschijnt. Bij een gebrekkige administratie valt dus niet uit te sluiten dat deelnemers bezig zijn aan een partij waarvan de uitslag allang vaststaat.

(b) Stel, we mogen alleen gebruiken dat Turing's originele stop-probleem beslisbaar is. (Dat is het niet, maar stel.) Probeer weer uitsluitsel te geven over het vermoeden van Collatz. Mocht dit niet lukken, probeer dan aan te geven waarom dit niet lukt.

15. Amy en Brian bediscussiëren het stop-probleem voor willekeurige programma/input-combinaties bestaande uit hoogstens 1000 karakters. Dus programma's en (invoer-)strings zijn ieder hoogstens 1000 karakters lang. Amy en Brian korten dit probleem af met de letter P. Amy beweert dat P beslisbaar is, en heeft daar een argument voor. Brian beweert dat P onbeslisbaar is en heeft daar ook een argument voor.

Over het volgende zijn beide partijen het eens: als een eindig uitgangsalfabet wordt gekozen, dan bestaan er slechts eindig veel strings met een lengte van hoogstens 1000 karakters. Deze strings kunnen in een rij worden gezet: $i_1, i_2, \ldots, i_N$. N is natuurlijk ontzettend groot, maar dat doet er niet toe, de rij is eindig. Precies dezelfde rij strings kan ook worden gezien als een rij Java-programma's: $j_1, j_2, \ldots, j_N$. De meeste programma/input-combinaties (j, i) zullen stoppen, bijvoorbeeld omdat j niet compileert, of omdat programma j wanneer het wordt uitgevoerd met input i netjes stopt. Andere programma/input-combinaties (j, i) zullen niet stoppen. Het is het één of het ander: de combinatie (j, i) stopt, of de combinatie (j, i) stopt niet. Er bestaat dus een tabel van die het gedrag van al deze programma/input-combinaties weergeeft, met een 1 als de combinatie stopt, en een 0 anders:

	i_1	i_2	i_3	i_4	$\ldots$	i_N
j_1	1	1	1	1	$\ldots$	1
j_2	1	1	0	1	$\ldots$	1
j_3	1	1	0	1	$\ldots$	1
j_4	1	1	1	1	$\ldots$	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
j_N	0	0	0	1	$\ldots$	0

Waar precies nullen en enen staan hoeven we niet te weten, maar zeker is dat de tabel bestaat. Je kunt het ook zo zien: er zijn $2^{N \times N}$ mogelijke tabellen (kies voor elke entry een 0 of een 1). Eén van deze tabellen moet "bij toeval" het stop-gedrag van alle $N \times N$ programma/input-combinaties correct weergeven. Amy en Brian noemen deze tabel T.

Amy: P is beslisbaar, want je kunt een controleprogramma, zeg H, schrijven door de hele tabel T hard te coderen (i.e., letterlijk op te nemen) in H. Als H een combinatie (j, i) aangeboden krijgt, zal H de 0/1-waarde van (j, i) opzoeken in T en afdrukken.

Brian: P is onbeslisbaar want op T kan diagonalisatie worden toegepast:

	i_1	i_2	i_3	i_4	$\ldots$	i_N
j_1	$\underline{\mathbf{1}}$	1	1	1	$\ldots$	1
j_2	1	$\underline{\mathbf{1}}$	0	1	$\ldots$	1
j_3	1	1	$\underline{\mathbf{0}}$	1	$\ldots$	1
j_4	1	1	1	$\underline{\mathbf{1}}$	$\ldots$	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
j_N	0	0	0	1	$\ldots$	$\underline{\mathbf{0}}$
q	0	0	1	0	$\ldots$	1

en

$$q : x \longrightarrow \text{ als } H(x, x) = 1 \text{ ga dan in een lus, anders stop} \tag{3}$$

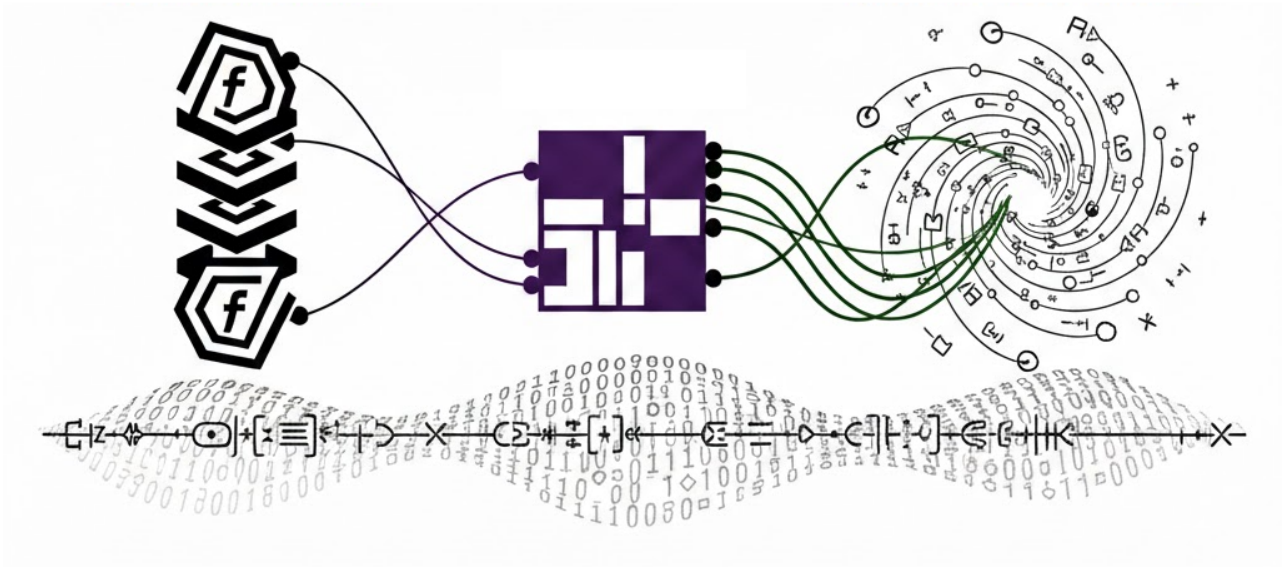
is kort, in ieder geval korter dan 1000 karakters.

(a) Neem aan dat programma's self-contained zijn. In het bijzonder mag worden aangenomen dat de programma's geen gebruik mogen maken van externe packages, libraries, of subroutines. Wie heeft gelijk? Waarom is het argument van de ander ondeugdelijk?

(b) Laat de aanname uit het vorige onderdeel varen. Wie heeft er nu gelijk?

9 Berekenbaarheid

De berekenbaarheidstheorie werkt met berekenbare functies, beslisbare verzamelingen, semi-beslisbare verzamelingen, opsombare verzamelingen, en complementair-opsombare verzamelingen. Op deze manier verkrijgen we een instrumentarium waarmee belangrijke definitieve uitspraken kunnen worden gedaan over de onberekenbaarheid van sommige functies, en de onbeslisbaarheid van sommige verzamelingen.



Een associatieve impressie van berekenbare functies en beslisbare verzamelingen (Bron: GPT Image 1.5).

Zo is bijvoorbeeld de verzameling van geldige formules uit de predikatenlogica onbeslisbaar: er kan principieel geen universeel algoritme bestaan dat voor elke formule uit de predikaatlogica bepaalt of deze waar is.

Met dit instrumentarium kunnen ook functies worden geconstrueerd die sneller groeien dan welk computerprogramma dan ook kan bijhouden, wat aantoont dat menselijke intelligentie grenzen blootlegt die geen enkele machine kan doorbreken. Voorbeelden van dergelijke idioot hard groeiende functies zijn de Ackermann-functie, de Goodstein-functie, en de Busy-Beaverfunctie (Hoofdstuk 5 op pagina 23).

Onderstaande opgaven horen bij Hoofdstuk 3: “Computability and Incomputability” uit The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation, Gary W. Flake, MIT Press, 2000.

1. Schrijf een Nederlandse samenvatting van Hoofdstuk 3 (Computability and Incomputability) van *The computational Beauty of Nature* (Flake, 1998). Zorg er voor dat de samenvatting kort, compleet, juist, en samenhangend is. Het is raadzaam deze opgave te combineren met Opgave 2. (Antwoord achter in het dictaat.)

2. Leg een begrippenlijst aan voor hoofdstukken 2 en 3 van *The computational Beauty of Nature* (Flake, 1998). (Antwoord achterin.)

3. In Opgave 9 op pagina 8 werd gevraagd informeel aan te tonen dat $\mathbb{N} \times \mathbb{N} \sim \mathbb{N}$. Dat deed je dan waarschijnlijk door een zig-zaglijn door $\mathbb{N} \times \mathbb{N}$ te trekken.

   Wat als iemand je vraagt om een concrete 1-1 correspondentie tussen $\mathbb{N} \times \mathbb{N}$ en $\mathbb{N}$? Dan zal blijken dat de zig-zaglijn moeilijk te vangen is in een expliciete formule. Bovendien zal blijken dat het erg bewerkelijk is aan te tonen dat deze formule injectief, surjectief, en berekenbaar is, en een berekenbare inverse bezit.

   Als alternatief bekijken we de volgende afbeelding

   $$f : \mathbb{N} \times \mathbb{N} \to \mathbb{N} : (x, y) \mapsto 2^{x-1}(2y - 1). \tag{4}$$

   waarbij $\mathbb{N} = \{1, 2, 3, \dots\}$. Wat we hier doen is niet meteen uit te leggen maar omgekeerd wel: we ontbinden elk getal in een maximale 2-macht maal een restfactor.

   (a) Teken in een $\{1, 2, 3, 4, 5\} \times \{1, 2, 3, 4, 5\}$ assenstelsel de beelden van f. Dat gaat 't makkelijkst als je eerst de beelden invult van paren met y-coördinaat 1 (1, 2, 4, 8, ...), dan de beelden invult van paren met x-coördinaat 1 (1, 3, 5, 7, ...), dan de rij met y-coördinaat 2, dan de rij met y-coördinaat 3, etc.

   Toon het volgende aan

   (b) f is berekenbaar. (Dit aantonen kan superkort!)

   (c) f is surjectief.

   (d) f is injectief.

   (e) f heeft een inverse f^{-1}

   (f) f^{-1} is berekenbaar.

4. (Gödelnummering.) Het is mogelijk elk programma $j \in J$ mechanisch te associëren met een uniek getal uit $\mathbb{N}$, het zg. *Gödel getal* van j. Gödel getallen zorgen er voor dat programma's opsombaar, i.e., mechanisch aftelbaar, zijn. Gödel getallen zorgen er dus voor dat de aftelling kan worden uitgevoerd door een computerprogramma. Een Gödel-nummering kan op verschillende manieren worden aangelegd. (Zie verder CBoN.)

   Gebruik in de volgende opgaven de ASCII-nummering van karakters. Geef van de volgende strings het Gödel getal.

   (a) a.

   (b) abba.

   (c) a c.

5. Herleid de volgende (Gödel-) getallen naar een string.

   (a) $2^{97}3^{98}$.

   (b) $2^{104}3^{111}5^{105}$.

   (c) $2^{104}3^{97}5^{108}7^{108}11^{111}$.

6. Een verzameling $X \subseteq \mathbb{N}$ heet *opsombaar*[14] als er een nul-plaatsig computer-programma $j/0$ bestaat dat precies alle getallen in X afdrukt.

   Voor een opsom-algoritme van X geldt het volgende.[15]

   – Het heeft geen input nodig.
   – Het hoeft de elementen van X niet in een bepaalde volgorde af te drukken.
   – Het mag elementen van X herhaald afdrukken.
   – Er kunnen willekeurige lange en onvoorspelbare pauzes zitten tussen het afdrukken van getallen.

   Stel programma j somt een voor ons onbekende verzameling X op. In de eerste twee seconden worden 34 getallen afgedrukt. Toevalligerwijs (?) zijn dit de eerste 34 kwadraten. Het programma blijft lopen. Na tien minuten is er nog geen volgend getal afgedrukt. Wat kan over X gezegd worden? Over j?

7. Beargumenteer dat elke eindige verzameling $X \subseteq \mathbb{N}$ opsombaar is.

8. Zij $X, Y \subseteq \mathbb{N}$ opsombaar. Beargumenteer:

   (a) $X \cup Y$ is opsombaar. Hint: er zijn dus programma's j en j' die X en Y afdrukken.
   (b) $X \cap Y$ is opsombaar. Hint: vang print-statements van j en j' op, en druk pas af als er aan een voorwaarde is voldaan.
   (c) Zij X en Y opsombaar. Dan is $X \times Y$ opsombaar.

9. Zij $X_1, \ldots, X_n$ opsombaar. Beargumenteer.

   (a) De vereniging $\bigcup_{i=1}^{n} X_i$, de doorsnede $\bigcap_{i=1}^{n} X_i$ en het Cartesisch product $\prod_{i=1}^{n} X_i$ zijn ook opsombaar. (Hint: gebruik het resultaat uit Onderdeel 8.)
   (b) (Eindige) verzamelingstheoretische combinaties van $X_1, \ldots, X_n$ zijn opsombaar, zolang er gecombineerd wordt met operatoren uit $\{\cup, \cap, \times\}$.

10. Beargumenteer het volgende:

    (a) De verzameling van alle mogelijke Java-programma's is aftelbaar.
    (b) Het aantal verschillende deelverzamelingen van $\mathbb{N}$ dat afgedrukt kan worden door inputloze Java-programma's is aftelbaar.
    (c) Er bestaan hoogstens aftelbaar oneindig veel verschillende opsombare deelverzamelingen van $\mathbb{N}$.

11. Bij een aftelbare verzameling wordt met "bijna alle" bedoeld: "alle, op eindig veel na". Bij een overaftelbare verzameling wordt met "bijna alle" bedoeld: "alle, op aftelbaar veel na".

    (a) Beargumenteer dat bijna alle deelverzamelingen van de natuurlijke getallen niet opsombaar zijn.

[14] Nederlandse equivalenten: enumereerbaar, semi-beslisbaar, semi-recursief. Engelse equivalenten: enumerable, recursively enumerable, r.e., computably enumerable, c.e., semi-recursive, semi-decidable, semi-computable, Turing-recognisable.

[15] Bij het woord algoritme ligt de klemtoon op de i. Dus: algorítme. Wil je meer weten over de herkomst van het woord algoritme, speel dan Assassin's Creed Mirage.

(b) Beschouw het kans-experiment waarin herhaaldelijk, zonder te stoppen, een munt wordt opgeworpen (zie ook Opgave 4 op pagina 13). Beargumenteer dat bijna alle realisaties van zo'n experiment niet door een computer kunnen worden gegenereerd.

12. Beargumenteer het volgende:

(a) Er bestaat tenminste één deelverzameling van $\mathbb{N}$ die niet opsombaar is. (Hint: herinner je het resultaat van Cantor's diagonalisatiestelling.)

(b) $\mathbb{N}$ bevat meer niet-opsombare deelverzamelingen dan opsombare deelverzamelingen!

13. We zeggen dat een programma $j/1 \in J$ kan *beslissen* over $X \subseteq \mathbb{N}$ als het voor elke $n \in \mathbb{N}$ kan uitmaken of $n \in X$. Een verzameling X wordt *beslisbaar*[16] genoemd als er een programma $j \in J$ bestaat dat over X kan beslissen.

Zonder beperking der algemeenheid kan eigenlijk worden aangenomen dat j in dergelijke gevallen zó geschreven is dat het een 1 afdrukt als $n \in X$ en anders een 0. (Als j niet zo geschreven is, dan is het makkelijk voor te stellen hoe j kan worden omgeschreven naar een programma j' dat wel aan deze eisen voldoet.)

Laat zien:

(a) Elke eindige verzameling $X \subseteq N$ is beslisbaar.

(b) Als $X, Y \subseteq N$ beslisbaar zijn, dan zijn $X \cap Y$, $X \cup Y$, $X \backslash Y$, en $X \times Y$ het ook.

14. Zij X een beslisbare verzameling. Beargumenteer dat $Y = \{\, 2x \mid x \in X \}$ ook beslisbaar is. (Let op in welke richting je converteert.)

15. Een verzameling $X \subseteq \mathbb{N}$ heet *complementair opsombaar* als het complement opsombaar is, zie ook TCBoN pp. 46-48. Alternatieve termen: co-opsombaar, co-enumereerbaar, co-recursief enumereerbaar, co-recursively enumerable, co-r.e.

Als j een algoritme is dat het complement van Y opsomt en een getal x afdrukt, dan kunnen we alleen maar concluderen dat $x \notin Y$. Het doet denken aan het tekenen van een plaatje door stipjes te zetten op de achtergrond. Omdat we geen stipjes mogen zetten op de plek van het plaatje zelf, kunnen we alleen maar conclusies trekken over de vorm van het plaatje door naar de stipjes in de achtergrond te kijken. Met een beetje verbeeldingskracht kunnen complementair opsombare verzamelingen wel worden gezien als de "zwarte gaten" van de berekenbaarheidstheorie.

Beredeneer:

(a) $\mathbb{N}$ bevat evenveel opsombare als complementair opsombare verzamelingen.

(b) $\mathbb{N}$ bevat aftelbaar veel complementair opsombare verzamelingen. (Hint: gebruik het vorige onderdeel en Opgave 10.)

(c) $\mathbb{N}$ bevat overaftelbaar veel verzamelingen die opsombaar noch complementair opsombaar zijn.

16. De Pools-Amerikaans wiskundige Emil Post (1897-1954) was, naast Alan Turing en Alonzo Church, één van de meest belangrijke grondleggers van de informatica. Post werd geboren in Polen en emigreerde als kind naar Amerika.

[16] Nederlands equivalent: recursief. Engelse equivalenten: decidable, recursive, recognisable by a Turing machine that always halts.

Stelling (Emil Post). *$X \subseteq \mathbb{N}$ is beslisbaar $\Leftrightarrow$ X is zowel opsombaar als complementair opsombaar.*

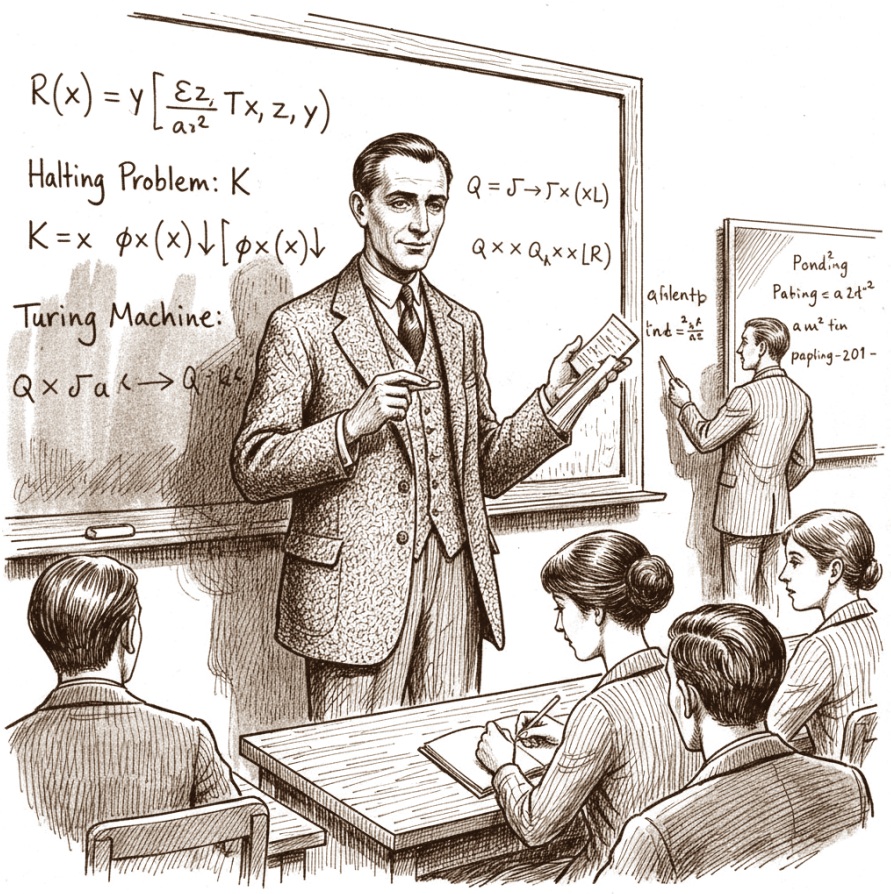
De stelling van Post legt dus een verband tussen wat uitgerekend en wat opgesomd kan worden.

Veel belangrijke stellingen zijn moeilijk te formuleren en moeilijk te bewijzen. Andere belangrijke stellingen zijn makkelijk te formuleren maar moeilijk te bewijzen (denk aan de grote stelling van Fermat). Deze belangrijke stelling is makkelijk te formuleren én makkelijk te bewijzen. Je kunt dus zelf makkelijk een bewijs geven. Probeer het eens.

Tip: van beslisbaar naar r.e. & co-r.e. is eenvoudig. Omgekeerd is een klein beetje lastiger, omdat je nu twee programma's hebt: eentje die X opsomt, en eentje die $\mathbb{N}\backslash X$ opsomt. Laat dan zien dat die twee programma's aanleiding geven tot een derde programma dat over X beslist. Het idee is dat dit derde programma de executie van de eerste twee programma's vervlecht. Vervlechten betekent: om beurten rekentijd geven.

Let op: "vervlechten" betekent NIET dat het derde programma de twee eerste programma's vraagt afwisselend elementen van "hun" verzameling te laten afdrukken. Waarom is dit geen goede oplossing?

Hier nog een anekdote.



Post's classes were tautly organised affairs. Each period would begin with student recitations covering problems and proofs of theorems from the day's assignment. These were handed out apparently at random and had to be put on the blackboard without the aid of textbooks or notes. Woe betide the hapless student who was unprepared. He (or rarely she) would have to face Post's "more in sorrow than anger

> *look". In turn, the students would recite on their work. Afterwards, Post would get out his 3-by-5 cards and explain various fine points. The class would be a success if he completed his last card just as the bell rang. Questions from the class were discouraged: there was no time. Surprisingly, these inelastic pedagogic methods were extremely successful, and Post was a very popular teacher.* (M. Davis, *Emil L. Post: his life and work*, herdruk 1994, afbeelding gegeneerd door Gemini 3, en nabewerkt.)

Zou anno 2026 een docent daar nog mee wegkomen? De vraag stellen is hem beantwoorden. In schril contrast met Post's ijzeren tucht, is de hedendaagse collegezaal een oase van vrijblijvendheid. Met telefoons binnen handbereik en een informele sfeer, zou zo'n zwijgzame beurt aan het bord nu prompt leiden tot een collectieve burn-outverklaring.

Opmerkingen:

- Op de (door Gemini gegeneerde) tekening zien we een student die op de muur schrijft. Misschien is dit zijn passief-agressief verzet tegen de militaire discipline van zijn docent.
- "3-by-5 cards" zijn gelinieerde kartonnen index-kaarten, met een grootte van 3x5 inch. Ze zijn nog steeds te koop.

Een pakje "3-by-5 cards".

Deel IV. Turmites en cellulaire automaten

10 Langton's mier

Langton's mier ("Langton's ant") wordt onder meer beschreven in TCBoN (Flake), Netlogo en Wikipedia [21]. Voor de goede orde wordt uitgegaan van de volgende aannamen.

1. De mier loopt van veld (cel, vakje, patch) naar veld op een oneindig rooster.
2. Velden bezitten coördinaten. De mier begin op veld $(0, 0)$.

					2,2	
	-2,2			1,1		
		-1,0	0,0	1,0		
					2,-1	

3. Bij aanvang zijn alle velden wit, en wijst de mier wijst naar het Noorden.
4. Per tijdsmoment (ook wel: tik) is de volgorde van handelingen als volgt:
   (a) Is het huidige veld wit, draai dan 90^0 naar rechts; draai anders 90^0 naar links.
   (b) Wissel het huidige veld van kleur. (Dus wit wordt zwart, en zwart wordt wit.)
   (c) Doe een stapje vooruit naar het volgende veld.

   Deze volgorde van handelingen staat bekend als het *turn-flip-move* regime, en is de standaard.

In Langton's oorspronkelijke artikel uit 1986 loopt de mier (die toen nog "vant"—virtual ant—werd genoemd) volgens een *move-turn-flip* regime [21]. In TCBoN p. 264 (Flake) wandelt de mier volgens een *move-flip-turn'* regime, waarbij *turn'* staat voor 90^0 naar links—in plaats van naar rechts—draaien op witte velden. Later hanteerde nagenoeg iedereen, inclusief Langton, de nu geldende standaard.

1. Leg uit waarom het *move-flip-turn* regime identiek is aan het *move-turn-flip* regime.
2. (a) Voer de eerste vijf stappen uit. Welke cellen, uitgedrukt in coördinaten, worden door de mier achtereenvolgens bezocht?

(b) Beschrijf het pad van de mier op de lange termijn. (Het is niet aan te bevelen zelf het pad verder te ontwikkelen. Beter is om het antwoord op internet te zoeken.)

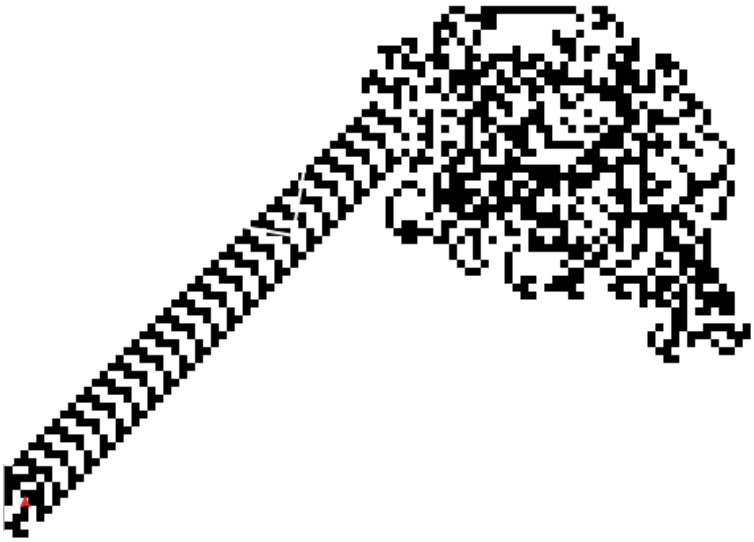


Fig. 2: Het spoor van Langton's mier.

(c) Zijn er velden waar de mier regelmatig naar zal terugkeren? Of zal elk veld slechts een eindig aantal malen worden bezocht?

(d) Zal de mier ooit het veld $(1000, 1000)$ bezoeken?

3. Langton's ant begint op een willekeurig ingekleurd oneindig rooster. Vakjes zijn zwart of wit.

(a) Laat zien dat, als in een initiële globale toestand de vakjes gunstig gekleurd zijn, de mier in een zig-zag patroon (rechts-links-rechts-links- ...) naar rechtsboven (het Noord-Oosten) kan lopen. (Hint: forceer de mier in een zig-zag.)

(b) Op welk vakje staat de mier na acht stappen? Na negen stappen? Na N stappen? (Hint: maak onderscheid tussen even en oneven N.)

(c) Laat zien dat de mier na N stappen hemelsbreed nooit verder dan

$$\sqrt{\frac{N^2+1}{2}}$$

van vakje 0, 0 af kan liggen.

4. Zoals aan het begin van deze sectie is uitgelegd, beweegt de mier volgens het *turn-flip-move* regime.

(a) In principe zijn er dus nog 5 andere regimes mogelijk. Waarom 5 en welke zijn dat? (Hint: kort de operaties af naar de letters T, F en M. Hoeveel permutaties bezit een woord bestaande uit 3 letters?)

(b) Hoeveel regimes zijn er in principe mogelijk als er twee versies van *turn* bestaan, te weten één die bij een wit veld naar rechts draait (de originele *turn*, T), en één die bij een wit veld naar links draait (de tegendraai, ook wel genoteerd als *turn'*, of C)?

(c) Welke van de in onderdeel 4b gevonden regimes geven identieke patronen? (Hint: genereer voor elk regime een patroon ter lengte één. Kijk vervolgens welke patronen identiek zijn. Controleer vervolgens van identieke patronen of deze *toevallig* na één stap danwel onvermijdelijk na *elk aantal* stappen identiek zijn. Doe dit laatste door te redeneren—niet door nog meer patronen te genereren.)

(d) Welke van de in onderdeel 4b gevonden regimes geven identieke patronen, modulo spiegeling, rotatie en translatie? Dus hoeveel essentieel verschillende regimes bestaan er? (Hint: kijk wat er gebeurt als de letters F en M naast elkaar staan, en verwisseld worden. Doe dit ook met andere mogelijke paren.)

5. Tijd-reversibiliteit.

   (a) Beargumenteer dat elke wandeling van Langton's mier tijd-reversibel is, i.e., dat elke wandeling eenduidig kan worden teruggelopen.

   (b) Leg uit hoe Langton's mier, op basis van een schijnbaar willekeurig patroon, de voorpagina van een krant kan schrijven.

   (c) Leg uit hoe Langton's mier kan worden gebruikt voor het versleutelen van afbeeldingen.

6. Het is bekend dat Langton's mier, startende op een volledig blank veld, na een groot aantal stappen van het vlak afwandelt middels een zogenaamde snelweg.

   Een *kiem* is een minimale start-configuratie van waaruit de mier een snelweg produceert. (Een start-configuratie is minimaal als het zo weinig mogelijk zwarte vakjes bevat.)

   (a) Verzin een manier (of methode zo je wil) om een goede kiem te vinden. Hint: Langton's mier is tijd-reversibel.

   (b) (Programmeer-oefening.) Welke minimale configuratie heb jij gevonden?

7. Langton's ant loopt over een $M \times N$ grid, waarvan de randen geïdentificeerd zijn tot een torus. Dus als de mier er rechts afloopt, verschijnt deze links (en omgekeerd), en als de mier er boven afloopt verschijnt deze aan de onderkant (en omgekeerd).

   (a) Hoeveel verschillende toestanden kan het grid aannemen? I.e., hoeveel kleuringen van het grid zijn er mogelijk?

   (b) Hoeveel verschillende posities kan de mier aannemen op het grid? (Hint: iets met het aantal vakjes en het aantal mogelijke verschillende oriëntaties van de mier.)

   (c) Hoeveel verschillende globale toestanden zijn er mogelijk als de toestanden van het grid en de mier worden gecombineerd? Geef een zo eenvoudig mogelijke expressie.

   (d) Beredeneer dat de mier, waar deze ook begint, uiteindelijk periodiek gedrag zal vertonen.

   (e) Geef een bovengrens voor een periode.

8. Beschouw het volgende "bewijs" van Langton's ant highway conjecture. We definiëren de omgeving van een mier als de 5×5 omgeving rond de mier, inclusief oriëntatie. Dus de omgeving beweegt met die mier mee. Bijvoorbeeld: stel de mier wijst naar het Noorden, en de bovenste vijf cellen zijn zwart. Dit is dan hetzelfde als de mier naar het Westen wijst, en de vijf linker-cellen zijn zwart—voor de mier komt dit op hetzelfde neer.

Nu zijn er slechts eindig veel van dergelijke omgevingen, dus op een gegeven moment moet de mier weer belanden in dezelfde omgeving. Vanaf dat moment gaat de mier zijn gedrag herhalen. (Wij weten inmiddels dat, op een leeg (i.e., wit) rooster, die herhaling een periode $N = 104$ heeft.) Dus het gedrag zal dan zoiets zijn als

$$\underbrace{\text{NWZW}\ldots\text{NOZW}}_{\text{aanloopfase}}(\underbrace{\text{ZONW}\ldots\text{NWZW}}_{\text{cyclus ter lengte 104}})^*$$

Vervolgens gebruiken we de stelling van Kong en Cohen om te concluderen dat dit gedrag semi-periodiek is. Immers, periodiek kan het niet zijn, want periodiciteit impliceert een begrensd spoor, terwijl de stelling van Kong en Cohen nu juist zegt dat het spoor van Langton's mier onbegrensd is. Ziehier het "bewijs". Hint: De fout is te vinden in het gedeelte vóór de voorbeeldstring NWZW...

9. Beschouw de volgende "weerlegging" van Langton's ant highway conjecture. Gajardo *et al.* bewezen in 2002 dat Langton's ant Turing-compleet is [13]. Dus elke Turing-machine kan worden gesimuleerd door Langton's ant, in het bijzonder een Turing-machine $\sqrt{2}$ afdrukt (en niets meer doet dan dat). Dan kan de corresponderende emulatie middels Langton's nooit een highway-bouwen, omdat een highway periodiek is, en de (binaire of decimale) ontwikkeling van een irrationaal getal dat niet is.

10. (Programmeer-oefening.) Langton's ant is is een discreet systeem, dat wil zeggen dat bewegingen per tijdseenheid ("tik") en per vakje plaatsvinden. Om nu de dynamiek van Langton's ant beter te begrijpen, is het inzichtelijk een deel van dit systeem *continu* te maken. Niet alles, alleen de volgende zaken.

   i) In plaats van dat vakjes zwart of wit zijn, staan we toe dat vakjes grijswaarden $v \in [0, 1]$ aannemen, waarbij 0 dan "wit", en 1 dan "zwart" representeert. Dus vakjes nemen nu grijswaarden aan.[17]

   ii) Verder nemen we aan dat de draairichting van de mier een conservatieve uitbreiding is van de draairichting in het discrete model. Eerst was het zo dat de mier 90^0 naar rechts draaide op witte velden, en 90^0 naar links draaide op zwarte velden. Nu laten we de mier op een vakje met kleur v

   $$180v - 90 \text{ graden}$$

   (naar links dus—dat is de conventie) draaien. Op deze manier draait de mier nog steeds 90^0 naar rechts op witte velden, en 90^0 naar links op zwarte velden.

   iii) Als een mier aankomt op een veld wordt de kleur nog steeds omgedraaid, maar nu volgens de formule

   $$v_{\text{nieuw}} = 1 - v.$$

   Op deze manier is het nog steeds zo dat witte velden zwart worden gekleurd, en zwarte velden wit. Maar daarnaast is het nu zo dat lichtgrijze velden donkergrijs worden gekleurd, en omgekeerd.

   iv) Na elke tik verspreidt de kleur zwart met een zogenaamde diffusie-factor α naar alle acht buur-cellen. Dus elke tik zal elke cel αv aan haar acht buren geven (elk van de buren ontvangt dus $\alpha v/8$), en $(1 - \alpha)v$ zelf houden. Uiteraard zal elke cel ook inkomende waarden van v ontvangen, tenzij de cel omringd is door witte cellen.

[17] E.L. James (echte naam Erika Mitchell) zou dit zeker waarderen.

Vragen.

(a) Controleer dat met deze aanpak de mier nog steeds 90^0 naar rechts draait op witte velden, en 90^0 naar links op zwarte velden.

(b) Controleer dat $\alpha = 0$ samenvalt met de originele versie van Langton's mier.

(c) Programmeer dit model met diffusie-factor $\alpha = 10^{-4}$. Laat de mier lopen vanaf een blank veld. Er ontstaat een grijs gebied. Hoe gedraagt de mier zich in het inwendige van dit gebied? En aan de rand?

(d) Zal de continue mier ook in staat zijn een snelweg te bouwen?

(e) Bespreek de betekenis van de continue mier in relatie tot het snelweg-vermoeden van de discrete mier (i.e., Langton's ant). Je kunt deze vraag misschien het best aanpakken door de overeenkomsten en verschillen te bespreken met de originele—discrete—versie van Langton's mier. Geef in het bijzonder aandacht aan de volgende punten.

i. Het gedrag van de continue en discrete mier op, of dichtbij, de rand.

ii. Het effect van verschillende waarden van α, inclusief de waarde $\alpha = 0$.

iii. De looprichting van de continue mier in het inwendige, en de betekenis daarvan voor de discrete mier.

iv. Het al dan niet ontstaan van een snelweg. Wat zouden nodige en/of voldoende voorwaarden kunnen zijn voor het ontstaan van een snelweg?

11. Langton's mier met meer dan twee mogelijke celtoestanden.

(a) Gegeven is Langton's mier met toestand-string RL, geplaatst op $(0, 0)$, met oriëntatie Noord. Cellen kunnen dus 2 toestanden aannemen, laten we zeggen respectievelijk wit en rood. Beschrijf het gedrag in de eerste 5 stappen. Op welke cel bevindt de mier zich tenslotte?

(b) Dezelfde vraag, met instructie-string LR.

(c) Dezelfde vraag, met instructie-string LLR, en 10 stappen. Cellen kunnen dus 3 verschillende toestanden aannemen, laten we zeggen respectievelijk wit, rood en groen.

(d) Dezelfde vraag, met instructie-string $RRRL$, en 10 stappen. Cellen kunnen dus 4 verschillende toestanden aannemen, laten we zeggen respectievelijk wit, rood, groen en blauw.

(e) Dezelfde vraag, met instructie-string $LRLRLR$, en 20 stappen. Kleuren respectievelijk wit, rood, groen, blauw, geel en sky.

12. Langton's mier met meer dan twee mogelijke celtoestanden.

(a) Beredeneer dat RL, $RLRL$, $RLRLRL$, en in het algemeen $(RL)^n$, $n \geq 1$, hetzelfde gedrag vertonen als de originele versie van Langton's ant.

(b) Beredeneer dat, voor elke eindige string $x \in \{L, R\}^+$, het gedrag van x^n overeenkomt met het gedrag van x, voor alle $n \geq 1$.

13. Langton's originele mier, dus de mier met instructie-string RL. In plaats van de celtoestand na een eerste bezoek op te hogen, kan er voor worden gekozen de celtoestand pas na, zeg, om de N, bezoeken op te hogen. Laten we dan spreken over een "aarzelende mier" (Eng.: *hesitant ant*).

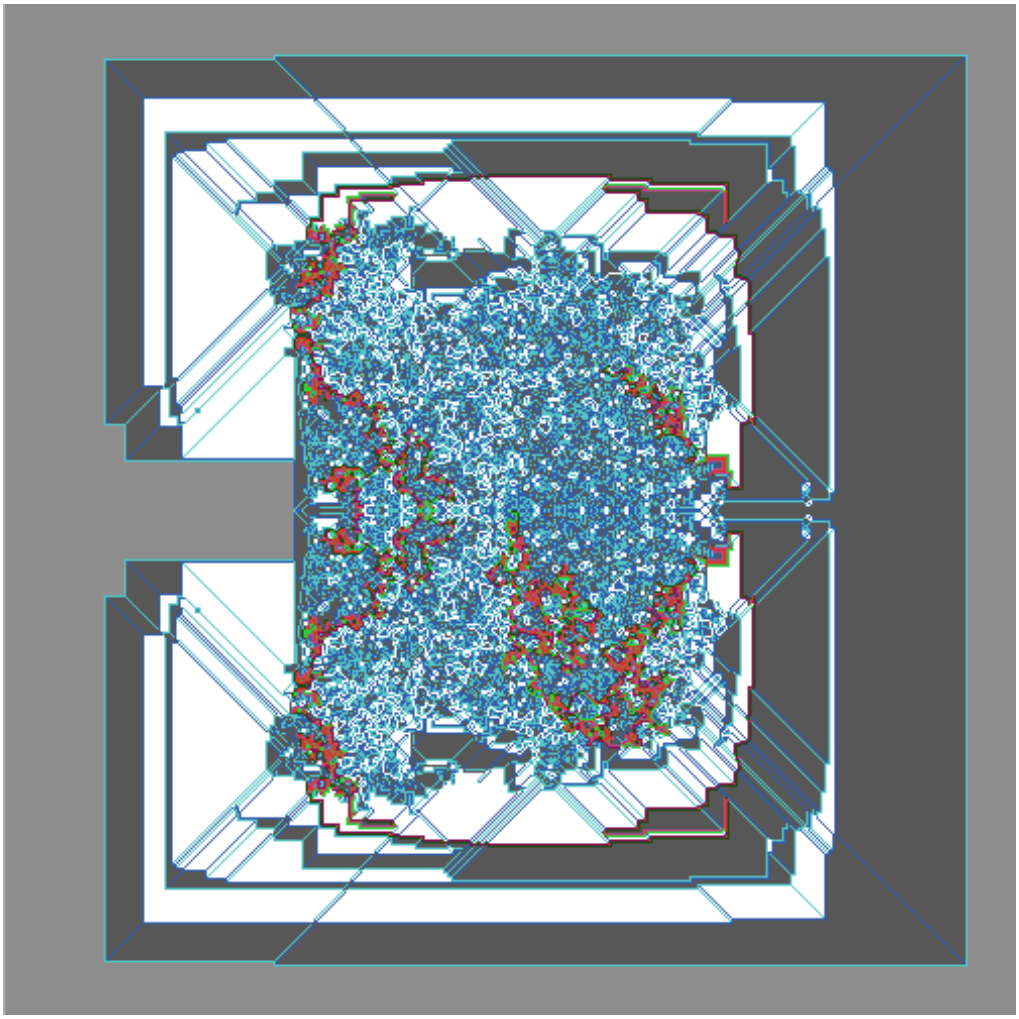


Fig. 3: "Professor's brein, verbonden aan een chip." (Instructie-string `RRRLLLLLLRRR`.)

(a) Beredeneer dat de aarzelende mier van orde N hetzelfde gedrag vertoont als de mier met instructie-string

$$R^N L \;=\; \underbrace{R \ldots R}_{N} L$$

(b) Probeer het gedrag van een aarzelende mier te voorspellen als N erg groot wordt.

14. (a) Beredeneer dat Langton's mier uit elk vooraf gedefinieerd gebied ontsnapt. Zie ook TCBoN en de slides. Of zoek op: Cohen-Kong theorem, a.k.a. Cohen-Kung theorem.

(b) Gaat dit ook op bij de veralgemeniseerde versie van Langton's mier, dat wil zeggen, met meer dan twee mogelijke cel-toestanden? Waarom (niet)? Zo ja, geldt het voor alle elementen uit $\{L, R\}^+$? Zo nee, waarom niet?

15. (Programmeer-oefening.) Twee of meer mieren kunnen met elkaars sporen gaan interacteren. We bestuderen twee experimenten, elk met twee RL-mieren. Deze twee mieren nemen beurtelings een stapje: eerst Mier 1, dan Mier 2, dan weer Mier 1, enzovoort. Beschrijf wat er gebeurt in de volgende situaties. In beide experimenten start

(a) Mier 1 op cel $(-6, -6)$ in oriëntatie Noord; Mier 2 start op cel $(5, 8)$ in oriëntatie Zuid.

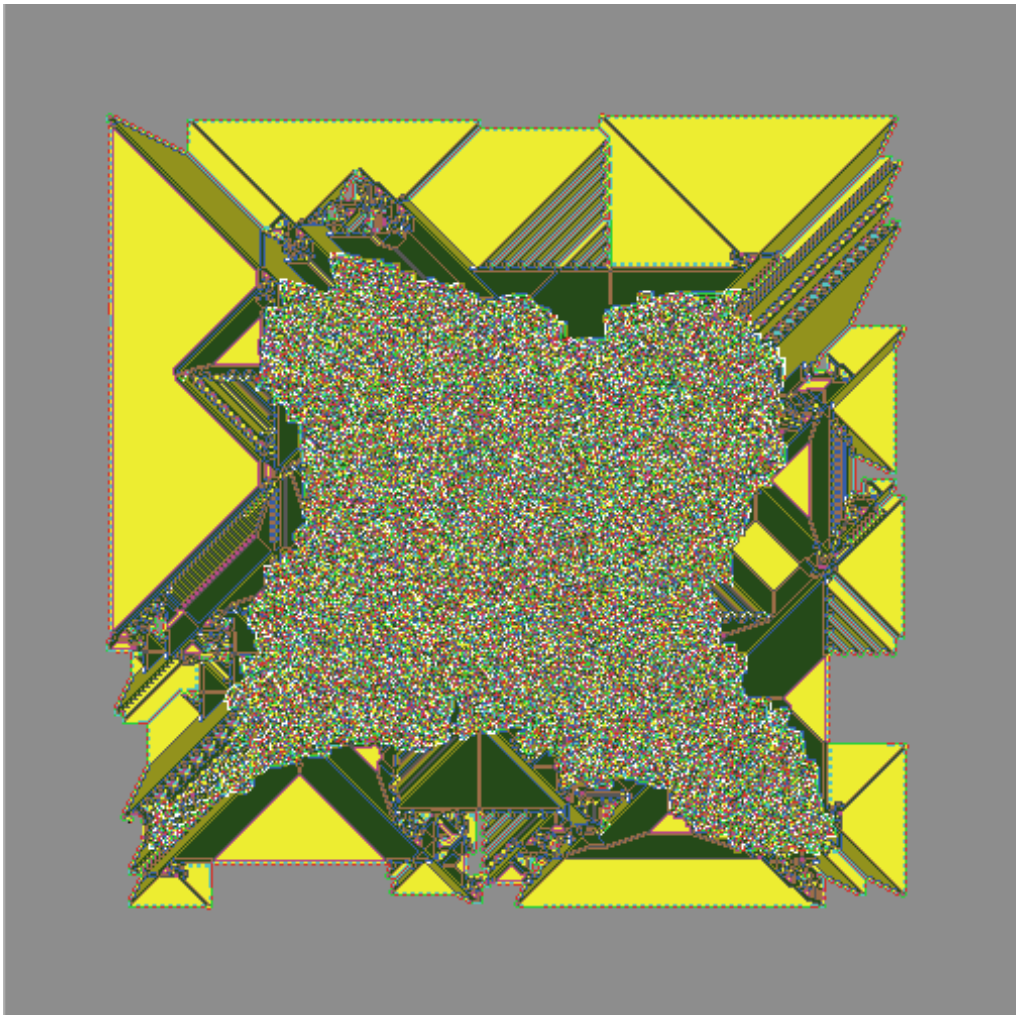


Fig. 4: "Het uitbrei(d)end vierkant." (Instructie-string `LLRLLLRRRRR`.)

(b) Mier 1 start weer op dezelfde plek, in dezelfde oriëntatie. Mier 2 start op cel $(23, 19)$ in oriëntatie West.

16. In 2000 bewezen Gajardo, Moreira en Goles [13] dat (de originele versie van) Langton's mier elk systeem van logische schakelingen (Engels: Boolean circuit) kan simuleren, door de mier te laten starten op een geschikte beginconfiguratie. Een beginconfiguratie is een vulling van het rooster waarbij een bepaald en eindig aantal cellen al de toestand 1 (zwart) bezit. De rest bezit toestand 0 (wit).

    Beredeneer dat uit Gajardo *et al.*'s stelling volgt dat Langton's mier Turing-volledig is, en dat problemen gerelateerd aan Langton's mier ("verlaat de mier het vlak?", "bezoekt de mier ooit de cel met coördinaten $(1000, 1000)$?", ...) onbeslisbaar kunnen zijn.

17. Langton's mier is (dus) Turing-volledig. We bekijken de consequenties hiervan voor het snelweg-vermoeden. Voor de duidelijkheid worden hier nog twee definities herhaald:

    > *i*) Het snelweg-probleem is het probleem om te bepalen of er een systematisch methode (een algoritme dus) bestaat, die voor elke eindige begin-configuratie kan bepalen of deze een snelweg genereert.[a]

ii) Het snelweg-vermoeden is het vermoeden dat elke eindige begin-configuratie een snelweg genereert.

[a] Een eindige begin-configuratie is een oneindig blank grid, met eindig veel zwarte cellen.

De eerste vragen zijn opwarmertjes.

(a) Bekijk het probleem of Langton's mier op een leeg grid een snelweg genereert. Is dit probleem opgelost? Is het beslisbaar?

(b) Wanneer kan het snelweg-vermoeden worden beschouwd als opgelost?

(c) Is het snelweg-vermoeden beslisbaar? Leg uit, en duid je antwoord, i.e., leg uit wat de draagwijdte, het gewicht, of de impact van jouw gegeven antwoord is.

(d) Wanneer kan het snelweg-probleem worden beschouwd als opgelost?

(e) In de literatuur wordt algemeen gesteld dat het snelweg-probleem waarschijnlijk onbeslisbaar is. Hoe zou dit kunnen worden aangetoond? Hint: reductie-argument.

(f) (Voortbordurend op het vorige onderdeel.) In de literatuur zijn de details nooit uitgewerkt, zodat de onbeslisbaarheid van het snelweg-probleem officieel nog steeds open is. Inventariseer middels eigen inzicht, en zo nodig met behulp van Gen-AI, mogelijke hordes en bottlenecks bij het opstellen van een reductie-algoritme?

(g) Stel dat het snelweg-vermoeden juist blijkt. Is hiermee dan het snelweg-probleem opgelost?

(h) Stel, er is aangetoond dat het snelweg-probleem beslisbaar is, *zonder* dat een concreet beslis-algoritme wordt gegeven. In zo'n geval zeggen we dat er "slechts" een existentieel bewijs bestaat voor de beslisbaarheid van het snelweg-probleem. Bespreek de hieruit volgende consequenties voor het snelweg-vermoeden.

(i) Stel, er is een concreet beslis-algoritme. Bespreek de consequenties.

(j) Beargumenteer dat, als het snelweg-probleem beslisbaar is, het snelweg-vermoeden semi-beslisbaar is. Dus geef, gebruikmakend van de aanname dat het snelweg-probleem beslisbaar is, een semi-beslisalgoritme voor het snelweg-vermoeden:

18. Van mier RLR (dus drie cel-toestanden) is onbekend of deze een snelweg genereert op een blank grid. Meer bepaald: als men de mier idioot lang laat lopen, ontstaat er geen snelweg—niet na 1000 tikken, niet na 10,000 tikken, en niet na 10 miljard tikken. Zoiets betekent uiteraard niet dat er nooit een snelweg ontstaat. Is het vraagstuk of een RLR mier op een blank grid ooit een snelweg produceert beslisbaar? Motiveer je antwoord.

19. Turmites.

(a) Formuleer het verschil tussen Langton's ant en turmites (afkorting voor: Turing termites).

(b) Hoe kan het dat een Turing-machine, met slechts één inwendige toestand, níet Turing-compleet is, en Langton's ant—een turmite met óók slechts één inwendige toestand—dat wél is?

20. (a) Leg waarom Langton's ant elke afbeelding—zeg elke 4K bitmap met 10-bit kleurdiepte—kan genereren.

(b) Leg hoe elke afbeelding kan worden gegenereerd door een turmite. Hoeveel toestanden zijn daar maximaal voor nodig?

(c) Dezelfde vragen, maar dan voor het genereren van een (ongecomprimeerde) 10-bit 4K speelfilm.

11 Cellulaire automaten

1. Bepaal de ontwikkeling van de volgende drie patronen in Conway's Game of Life. Bepaal in welke categorie deze patronen vallen: dekpunt, periodiek, semi-periodiek, transient dekpunt (instabiel, maar eindigend in dekpunt), transient periodiek (instabiel, maar eindigend in cyclus), of transient semi-periodiek. Het is toegestaan een simulator zoals Golly te gebruiken.

   (a) Twee levende cellen op een rij.

   (b) Drie levende cellen op een rij.

   (c) Vier levende cellen op een rij.

   (d) Vijf levende cellen op een rij.

   (e) Vijf levende cellen in de vorm van een plus-teken.

   (f) Glider. (Drie horizontale cellen, met boven de derde cel een vierde cel, en linksboven de vierde cel de vijfde cel.)

   (g) Gosper's glider gun. (Periodieke figuur die—ook periodiek—gliders afsch[eidt/iet].)

2. De regels voor Conway's game of life kunnen als volgt worden geformuleerd:

   (a) Een lege cel wordt bewoond als er precies drie bewoonde buren zijn (geboorte).

   (b) Een bewoonde cel wordt verlaten als er minder dan twee bewoonde buren zijn (eenzaamheid), of er meer dan drie bewoonde buren zijn (overbevolking).

   Misschien zijn er kortere formuleringen, misschien ook formuleringen die niet conditioneren op de toestand van de centrumcel. Dit laatste is dan makkelijker te implementeren. Probeer zo'n formulering te bedenken.

3. (a) Geef de definitie van een totalistische cellulaire automaat.

   (b) Is Conway's game of life een totalistische automaat?

   (c) Een alternatieve definitie van totalisme is, dat de volgende toestand van een cel wordt bepaald door het aantal levende cellen in de *omgeving* van de betreffende cel, dus inclusief de centrumcel zelf. Is Conway's game of life volgens deze definitie nog steeds totalistisch?

   (d) Zoek op internet de bevestiging, én de ontkenning op van GoL's totalisme. Vind je hits voor "game of life is not totalistic" (zoekstring binnen quotes)?

   (e) Welke definitie is het meest gangbaar?

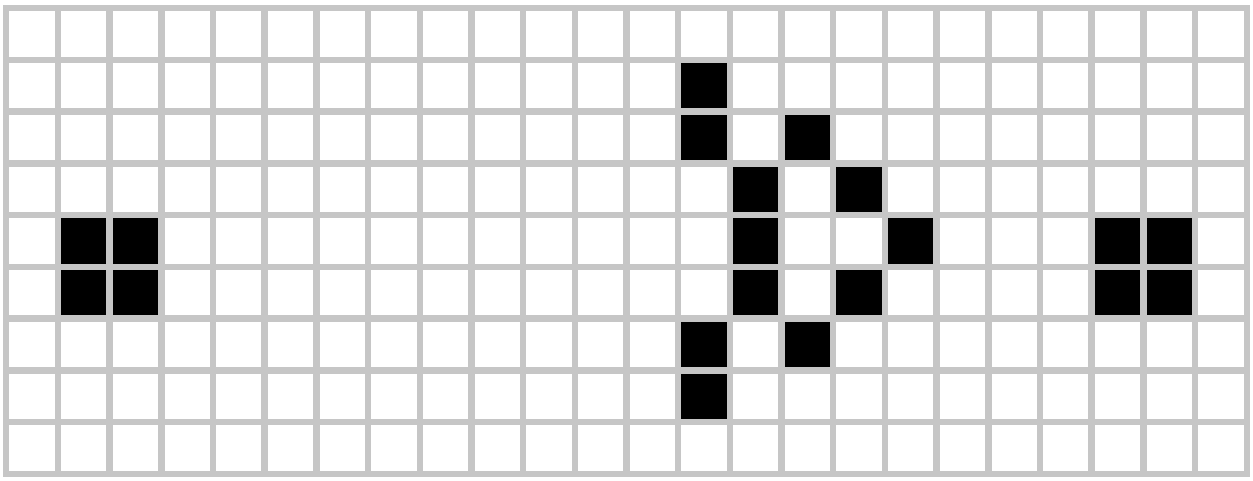
4. Conway's game of life. Bewijs of weerleg: er bestaan configuraties die uitgroeien tot configuraties van willekeurige grootte.

5. Conway's game of life. We bekijken een torus met 6×15 cellen waarvan exact 50 cellen leven. Welke van de volgende beweringen zijn waar.

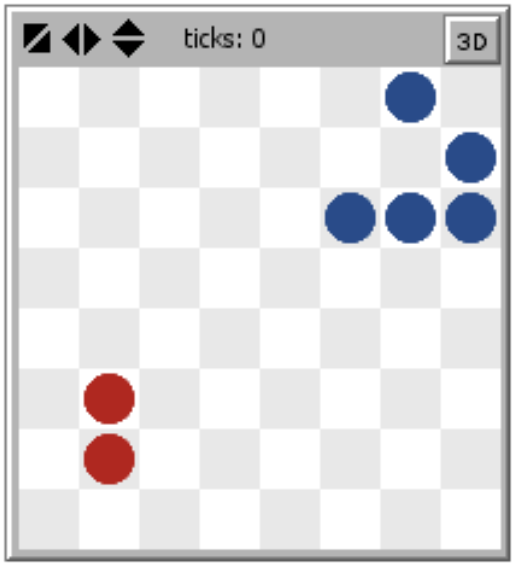
   (a) De configuratie zal nooit uitsterven.

(b) De configuratie zal uiteindelijk uitsterven.

6. Beschouw de cellulaire automaat met regels volgens Conway's game of life op een 32×32 rooster. De roosterranden zijn op de gebruikelijke manier met elkaar geïdentificeerd (torus). Beschrijf het gedrag van de volgende configuratie. Gebruik zo nodig een GOL calculator.



7. Beschouw de cellulaire automaat met regels volgens Conway's game of life. De randen van het rooster zijn in deze cellulaire automaat met elkaar geïdentificeerd volgens de fles van Klein, waarbij in dit geval de linker- en rechterrand van het rooster in tegengestelde oriëntatie met elkaar zijn geïdentificeerd. Rode cellen bevinden zich aan de voorkant, blauwe cellen bevinden zich aan de achterkant.

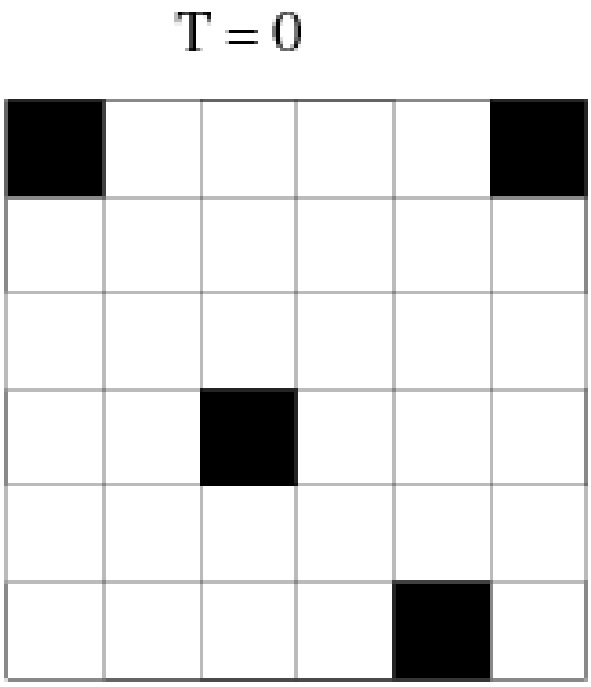


Het canvas heeft een voor- en een achterkant. (Heeft het niet, maar stel.) Als het helpt: de voorkant is zwart en de achterkant is wit. We pakken denkbeeldig de rechterkant van het canvas vast, draaien dit 180 graden in de diepte (3e dimensie) en plakken dit vervolgens aan de linkerrand. Nu is een Möbiusband ontstaan waarbij de voor- en achterkant van het canvas, dus de zwarte en witte kant, met elkaar verbonden zijn. (Ga na, en maak, als dat helpt, desnoods zelf een Möbiusband van papier.) Er is nu ook geen voor- en achterkant

meer, immers, het witte vlak loopt over in het zwarte vlak en omgekeerd. De CA heeft dus ineens ook 2x zo veel cellen. (Ga na!) Vervolgens plakken we de boven- en onderkant van het canvas op de "gewone"' manier aan elkaar: het oppervlak is nu de fles van Klein. Ook dit oppervlak bestaat natuurlijk uit een zwart en een wit gedeelte. (Om een fles van Klein te maken kun je ook eerst een cilinder maken, en dan de onder- en bovenrand van de cilinder omgekeerd aan elkaar plakken.) Laat twee gliders starten, één op het zwarte gedeelte, vroeger de voorkant van het canvas (rood) en één op het witte gedeelte, vroeger de achterkant van het canvas (blauw). Gliders veranderen van kleur als ze bewegen van het zwarte naar het witte gedeelte, en andersom. Zie ook de implementatie in Netlogo.

Geef de configuratie van deze automaat na acht iteraties.

8. De volgende afbeelding representeert de globale toestand van een twee-dimensionale cellulaire automaat:



Deze automaat werkt als volgt. Elke cel heeft vier buren, gevormd door links, rechts, boven, onder. (Dit wordt een Von Neumann omgeving genoemd. De randen lopen in elkaar over (Eng.: periodic boundary conditions). De transitie-regels zijn als volgt: als één of drie buren actief waren, dan wordt (of blijft) de cel actief. Anders wordt (of blijft) de cel inactief.

Bepaal de globale toestand in de eerste drie generaties.

9. Is Langton's mier een cellulaire automaat? Waarom wel / niet? Hint: kijk even naar Opgave 19 op pagina 64.

10. Geef een 1-dimensionale cellulaire automaat met

$$K = 7,\ R = 2$$

(a) Die, voor alle startconfiguraties, uitmondt in stabiel gedrag.

(b) Die, voor sommige startconfiguraties, uitmondt in periodiek gedrag. (Id.)

(c) Die, voor alle startconfiguraties, uitmondt in periodiek gedrag. (Id.)

(d) Die, voor alle startconfiguraties, uitmondt in periodiek gedrag, met vooraf gekozen periode n, waarbij $1 < n < 6$. (Id.)

Beargumenteer voor alle gevallen waarom je oplossing correct is.

11. Geef de codering van een 1-dimensionale totalistische cellulaire automaat met

$$K = 7,\ R = 2$$

(a) Die, voor alle startconfiguraties, uitmondt in stabiel gedrag, terwijl $\lambda = 1$. (Meerdere antwoorden mogelijk.)

(b) Die, voor alle startconfiguraties, uitdooft, terwijl $\lambda \neq 0$.

Beargumenteer voor alle gevallen waarom je oplossing correct is.

12. Bekijk de één-dimensionale cellulaire automaat die ontstaat door cellen te nemen die zijn geïndexeerd door Z (je krijgt een rij cellen die links en rechts oneindig doorloopt). Elke cel kan in twee mogelijke toestanden verkeren, te weten 0 en 1. Als overgangsfunctie wordt het volgende voorschrift gebruikt:

$$s_{\text{nieuw}} = (s_{\text{oud}} + s_{\text{oud}}(\text{rechterbuur})) \bmod 2$$

Optellen modulo 2 zorgt ervoor dat $1 + 1 = 0$ en dus een bit blijft.

(a) Een *pre-image* is een mogelijke vorige globale configuratie. Beargumenteer dat elke globale toestand precies twee pre-images bezit.

(b) Een globale toestand heet *eindig* als bijna alle cellen uit staan. (Met "uit" bedoelen we toestand 0, en met "bijna alle" bedoelen we "alle, op eindig veel na".) Laat zien, of beargumenteer, dat twee verschillende eindige globale toestanden ook de volgende generatie weer gegarandeerd van elkaar verschillen.

13. Beargumenteer dat het in principe mogelijk is een CA te definiëren die je favoriete speelfilm op een 4K TV afspeelt, full color en 60fps.

14. (Verdichtingspunt in $\mathbb{R}$.) Een verdichtingspunt (ook wel: limietpunt, clusterpunt, of ophopingspunt) x van een verzameling A is een punt dat willekeurig dicht kan worden benaderd door A, in die zin dat elke omgeving van x, hoe klein ook, wel een element $a \in A$ bevat, ongelijk aan x.

In $\mathbb{R}$ betekent omgeving: een open intervalletje om x. Dus $(-1, 10)$ is bijvoorbeeld één van de vele omgevingen van 0. In $\mathbb{R}^n$ betekent omgeving: een open bolletje om x. (Open = zonder schil.)

(a) Teken de situatie van een limietpunt. Give it your best. Hint: teken één A, één x dicht tegen A aan, en veel omgevingen rond x.

(b) Bepaal de verdichtingspunten van de rijen $\{1/n\}_n$ en $\{(-1)^n(1 + 1/n)\}_n$.

(c) Beredeneer dat elke omgeving van een verdichtingspunt van A oneindig veel punten uit A bevat.

(d) Bepaal de verdichtingspunten van singleton $\{2\}$ (bekeken als deelverzameling van $\mathbb{R}$).

(e) Bepaal de verdichtingspunten van $\mathbb{Z}$ (ook bekeken als deelverzameling van $\mathbb{R}$).

(f) Bepaal de verdichtingspunten van $\mathbb{Q}$ (ook bekeken als deelverzameling van $\mathbb{R}$). Hint: getallen als $\sqrt{2}$ en π kunnen willekeurig dicht benaderd worden door breuken.

(g) Van de verzameling

$$\begin{aligned} A &= \{\frac{1}{n} + \frac{1}{m}\}_{n,m} \\ &= \{\frac{1}{n} + \frac{1}{m} \mid n, m\} \\ &= \{\frac{1}{n} + \frac{1}{m} \mid n, m \in \mathbb{N}\}. \end{aligned}$$

Tip: deelverzamelingen van A kunnen worden gerepresenteerd door paren van (eindige of oneindige rijen) uit $\mathbb{N}$. Bijvoorbeeld

$$(1, 4, 5, 6, 10, 17, \ldots \text{ (loopt door)};$$
$$3, 5, 7, 100, 117 \text{ (stopt))} \quad (5)$$

is zo'n element. (Omdat volgorde bij verzamelingen er niet toe doet, worden hierbij geen deelverzamelingen uitgesloten.) Verdeel deze paren in vier klassen, afhangende of n en m, of beiden, doorlopen. Het paar (5) hoort bijvoorbeeld tot de klasse (loopt door, stopt). Als je elementen per klasse bekijkt is het makkelijk alle limietpunten te identificeren en er geen over het hoofd te zien.

15. (Verdichtingspunt in toestandruimten.)

(a) Een *toestand* van een CA is een momentopname (Eng.: snapshot) van de automaat. Beredeneer dat voor de beschrijving van een toestand, een beschrijving van de inhoud van alle cellen voldoende is, dat wil zeggen, voldoende om het gedrag van die automaat te bepalen als de klok weer wordt gestart.

Om die reden is het geen probleem een toestand te identificeren met een grid-configuratie (kort: configuratie).

(b) Hoe groot is de toestandruimte van Conway's game of life 1) op een 8×5 rechthoek; 2) op een 8×5 torus; 3) op een oneindig grid.

(c) De afstand tussen twee configuraties kan worden gedefinieerd als het aantal cellen waarin ze verschillen. Bekijk de configuratie met een glider op $(0, 0)$ [maakt even niet uit in welke oriëntatie]. Laten we dit toestand 1 noemen. Toestand 2 is dan de configuratie direct na toestand 1, toestand 3 is de configuratie direct na toestand 2, enzovoort. Laat toestand 0 de lege configuratie zijn. Wat is de afstand tussen de verschillende configuraties? Bv. de afstand tussen C0 en C1, tussen C0 en C2, tussen C0 en C3, etc? Maar ook tussen bijvoorbeeld C12 en C13, tussen C12 en C14, tussen C12 en C15, etc.? Is er ooit een afstand 1 tussen twee configuraties? En een afstand 3? Let op: modulo translatie, spiegeling en rotatie kan een glider nog steeds twee verschillende vormen aannemen. Tip: wijzig "life" in Netlogo met kleuren om bij te kunnen houden welke cellen veranderen.

(d) Leg uit waarom Jarkko Kari's stelling onwaar is met deze metriek en dit scenario,. i.e, het scenario van één glider in een verder leeg grid.

(e) Een andere notie van afstand is het verschil van elk paar cellen (1 bij een verschil, en 0 anders), te delen door de afstand naar de oorsprong. Dat dit inderdaad een legale notie van afstand is bewijzen we hier niet. Zijn willekeurig kleine afstanden nu mogelijk? Zijn clusterpunten van toestanden nu mogelijk? Welke afstanden zijn überhaupt nu mogelijk?

(f) Beargumenteer dat het scenario van één glider in een verder leeg grid nu geen tegenvoorbeeld is van Jarkko Kari's stelling.

16. Jarkko Kari's stelling.

(a) König's lemma zegt: "een oneindige, maar overal eindig-vertakkende, boom bezit een oneindige tak". Beargumenteer waarom dit zo is.

(b) Zij C een cellulaire automaat op Z^3 met S de (eindige en niet lege) verzameling van mogelijke toestanden die elke individuele cel kan aannemen. (In Conway's life is S bijvoorbeeld $S = \{\text{levend}, \text{dood}\}$.) Beargumenteer dat elke ontwikkeling ("run") een verdichtingspunt bezit.

17. A simulation of a forest fire is designed on a grid of cells. (If it helps: Netlogo's canvas.) There are green, red, and black cells, representing forest, fire, and bare ground, respectively. At the start of the simulation all cells are green. The dynamics is given by the following rules. All probabilities are fixed during a simulation.

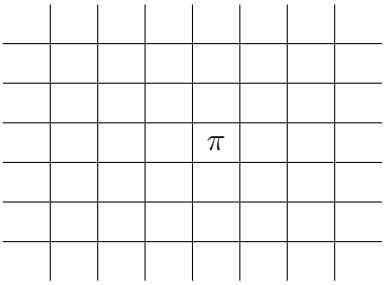
> At each tick
>
> - **Ignite**. If a cell is green, it will become red with probability i.
> - **Burn**. If a cell is red.
>   - **Spread**. Each green neighbour may become red with probability s.
>   - **Extinguish**. The cell becomes black with probability e.
> - **Vegetate** If a cell is black, it will become green with probability v.

The current system of rules does not constitute a cellular automaton. Why? (Hint: it's not the probabilities, and the exact shape of the neighborhoods doesn't matter either.)

Can the rule set be converted to a system of rules that *does* constitute a CA? Why?

18. Beschouw een $N \times N$ grid, met $N \in \mathbb{N}, N > 1$. Een zogenaamde *node count learner* π loopt dit grid als volgt af: π staat op vakjes (dus niet op roosterpunten) en neemt één stapje per tijdseenheid. Er zijn vier mogelijke stapjes: N, E, S, en W (noord, oost, zuid, west). Op elke tegel die π bezoekt schrijft de robot hoe vaak deze daar is geweest. Op elk moment stapt π naar de tot dan toe minst vaak bezochte tegel.

Als het helpt kun je denken aan een stofzuigrobot die altijd naar het tot dan toe minst bezochte buurveld gaat.



(a) Bepaal of dit scenario kan worden gemodelleerd met behulp een cellulaire automaat.

Hint: Merk op dat deze opgave ongespecificeerd laat wat de robot doet als meerdere buur-tegels een minimale bezoek-frequentie bezitten. I.e., deze opgave stelt geen eisen aan het gedrag van de robot in dergelijke gevallen.

Deze onder-specificatie maakt de opdracht niet moeilijker, maar houdt deze realistisch en geeft jou de vrijheid om zelf te bepalen hoe je bepaalde ambigue manoeuvreer-situaties oplost. De opdracht staat bijvoorbeeld toe dat er een regel is die voorschrijft naar welke buur de robot π bij meerdere minimaal bezochte buren verplicht is toe te stappen, maar de opdracht staat ook toe dat de robot bij meerdere keuze-alternatieven willekeurig kiest tussen minimaal bezochte buren.

(b) Kan dit scenario ook worden gemodelleerd met twee node count learners op één grid? Met meer dan twee?

19. Formuleer een cellulaire automaat die hetzelfde gedrag vertoont als Langton's mier door de volgende deelvragen te beantwoorden.

(a) In hoeveel mogelijke toestanden kan een cel verkeren? Welke namen heb je die toestanden gegeven? Welke gedachte zit er achter je naamgeving?

(b) Gebruikt je CA een Moore-omgeving of een Von-Neumann omgeving?

(c) Geef alle regels voor je CA. Heb je voor een Moore-omgeving gekozen, dan bezit elke regel de volgende vorm:

$$(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9) \to x_5'.$$

Daarbij is

x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9

de omgeving van x_5, en x_5' de nieuwe toestand van x_5. Heb je voor een Von-Neumann gekozen, dan bezit elke regel de volgende vorm:

$$(x_1, x_2, x_3, x_4, x_5) \to x_3'$$

waarbij

	x_1	
x_2	x_3	x_4
	x_5	

de omgeving van x_3 is en x_3' de nieuwe toestand van de centrumcel.

Je mag, indien gewenst, gebruikmaken van de volgende afkortingsmechanismen:

— Aangeven dat alleen regels waarbij de centrumcel veranderd, opgeschreven worden.

— Aangeven dat regels modulo rotatie-4 gelden. Bijvoorbeeld, de volgende regels zijn gelijk modulo rotatie-4:

$$(1,1,0,0,0,0,0,0,0) \to 0$$
$$(0,0,1,0,0,1,0,0,0) \to 0$$
$$(0,0,0,0,0,0,0,1,1) \to 0$$
$$(0,0,0,1,0,0,1,0,0) \to 0$$

Deze kunnen samen gespecificeerd worden als

$$(1, 1, 0, 0, 0, 0, 0, 0, 0) \to 0 \pmod{\text{rot-4}}$$

— Aangeven dat regels modulo rotatie-1 gelden.
— Sterretjes op coördinaten waarvan de waarde onbelangrijk is ("don't care's").

(d) Beargumenteer dat de geconstrueerde CA tijd-reversibel is.

(e) Hoe configuratie te vinden waarbij na 1000 stappen alles blank is, op turtle na?

20. Drossel en Schwabl introduceerden in 1992 een belangrijk eerste model voor het modelleren van bosbranden [10]. Later werd dit bekend als het DS-FF model (D&S forest fire). Dit model is gebaseerd op een zogenaamde *stochastische cellulaire automaat*, dat wil zeggen een cellulaire automaat waar de volgende toestand van een cel probabilistisch afhangt van de toestand van de cellen in de omgeving.[18]

- Simulatie vindt plaats op een rechthoekig grid met vierkante cellen, typisch een torus-oppervlak bestaande uit 64×64 cellen.
- De omgeving van een cel word gevormd door de acht omringende cellen.
- De toestand-verandering van cellen vindt, net zoals bij Conway's game of life, synchroon plaats.
- Elke cel (lap grond) kan zich in drie mogelijke toestanden bevinden: zwart (leeg), groen (begroeid), en rood (brandend).
- In de bepaling van een volgende toestand spelen drie kans-parameters $f, g, p \in [0, 1]$ een rol. Deze staan resp. voor spontane ontbranding, immuniteit, en groei.

  R1 Met kans g zal een begroeid stuk grond niet vatbaar zijn voor brand, bijvoorbeeld vanwege weersomstandigheden (regen), terreincondities (te veel onderbrekingen tussen begroeiing) of anderszins.

  R2 Een voor brand vatbaar stuk grond zal met een kleine kans f spontaan (bijvoorbeeld door blikseminslag) ontbranden.

  R3 Een voor brand vatbaar stuk grond zal met zekerheid ontbranden als een naburig stuk grond brandt.

  R4 Een brandend stuk grond zal de volgende generatie zwart zijn.

  R5 Met een kleine kans p zal een leeg stuk grond de volgende generatie begroeid zijn.

  De kans dat een begroeid en door brand begrensd stuk grond zal ontbranden is dus $1 - g$, en de kans dat een begroeid stuk grond spontaan zal ontbranden is dus $f(1 - g)$.

(a) Beschrijf het emergente gedrag bij $g = 0.5$, $p = 0.002$ en $f = 0.00001$. (Random start.) We verwachten 3-5 regels beschrijving.

(b) Beschrijf (in 3-5 regels) het emergente gedrag bij $g = 0$, $p = 1$ en $f = 0$.

(c) Beschrijf het emergente gedrag bij $g = 0.8$, $p = 1$ en $f = 0$. (Random start met enkele vuurhaarden.)

(d) In geval (20c) lijkt de globale toestand te convergeren naar een globale limiet. Beschrijf deze limiet.

[18] Een eenvoudige Netlogo-implementatie is te vinden op `https://ics.uu.nl/docs/vakken/ias/stuff/forest-fire.zip`.

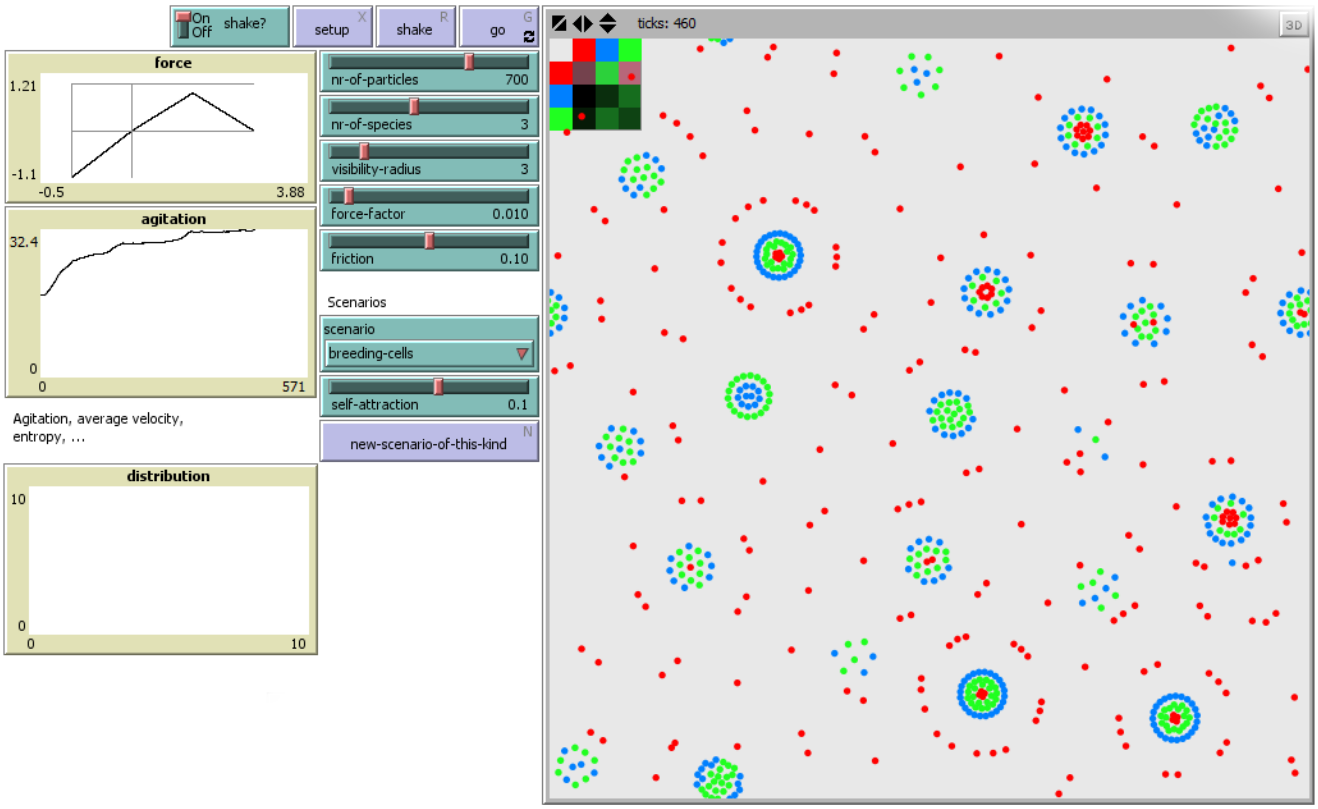
(e) Verklaar de convergentie in (20c) (je hebt ong. 5-15 regels nodig).

(f) Kan in geval(20c) convergentie bewezen worden? Zo ja, geef een bewijs. Zo nee, beargumenteer waarom convergentie niet bewezen kan worden.

(g) Kan in geval (20c) beweerd worden dat convergentie met kans 1 plaats zal vinden? Waarom wel / niet? Dit antwoord mag je gokken, maar we lezen wel graag je motivatie.

Deel V. Massieve multi-agent systemen

12 Particle life en kunstmatige chemie



Bron: Netlogo, schaduw-implementatie van Mohr's particle life, met links-boven de aantrekkings/afstotings-matrix. Om een té groot contrast met de rest van de tekst te vermijden, is de achtergrond tijdelijk grijs gemaakt—normaal is deze zwart.

1. (Particle life.) Eén van de meest eenvoudige kunstmatige chemieën, genaamd Clusters, werd rond 2017 informeel, op het internet, geïntroduceerd door Jeffrey Ventrella. Het nadeel van Ventrella's model, echter, is dat door te tijd heen er allerlei toeters en bellen aan zijn gekomen. Daarom bekijken we Tom Mohr' s particle life [YT]. Mohr's model is eenvoudiger en inzichtelijker dan dat van Ventrella.

   In Mohr's systeem is de aantrekkingskracht, F, tussen twee deeltjes afhankelijk van de huidige afstand, r, tussen die twee deeltjes, een vaste minimum afstand, β, en een aantrekkingsfactor a:

$$F(r,a) = \begin{cases} \dfrac{r}{\beta} - 1 & \text{als } r < \beta, \\ a\left(1 - \dfrac{|2r - 1 - \beta|}{1 - \beta}\right) & \text{als } \beta < r < 1, \\ 0 & \text{anders.} \end{cases}$$

   De parameter β is een constante. In Mohr's simulatie geldt bijvoorbeeld bijna altijd $\beta = 0.3$. Verder bezit a voor elk tweetal type deeltjes a een specifieke constante waarde. Bijvoorbeeld,

als er 5 typen deeltjes zijn (bijvoorbeeld rood, blauw, groen, oranje, paars), dan zijn er 5×5 verschillende waarden van a.

Teken vier grafieken van F, als functie van r, met parameters achtereenvolgens $(a, \beta) = (1, 0.3)$, $(0.5, 0.3)$, $(-0.5, 0.3)$, en $(-1, 0.3)$. Licht deze toe. Voor welke radii worden deeltjes aangetrokken? En voor welke radii worden deeltjes afgestoten? Wanneer is de aantrekkings- danwel afstotings-kracht maximaal? Voor welke radii wordt er geen kracht uitgeoefend?

2. (Particle life.) Stel, er zijn $N = 3$ typen deeltjes, te weten rood, blauw, en groen.

   (a) Geef een aantrekkings-matrix waarin alle deeltjes elkaar aantrekken. Wat is de meest algemene vorm van zo'n matrix?

   (b) Waarin alle deeltjes elkaar afstoten.

   (c) Waarin alleen deeltjes van hetzelfde type elkaar aantrekken.

3. (Particle life.) Geef een 4×4 matrix waarin deeltjes van type n aangetrokken worden tot deeltjes van type $n + 1 \pmod 4$, maar afgestoten worden door deeltjes van type $n - 1 \pmod 4$. Deeltjes van hetzelfde type trekken elkaar niet aan, en stoten elkaar niet af.

4. (Particle life.) Is de aantrekkings-matrix altijd symmetrisch? Zo ja, waarom? Zo nee, waarom niet?

5. (Particle life.) Stel, er zijn $N = 3$ typen deeltjes (rood, blauw, groen), en de aantrekkingskracht tussen twee deeltjes kan $k = 5$ verschillende waarden aannemen, bijvoorbeeld

$$a \in \{-0.8, -0.4, 0.0, 0.4, 0.8\}.$$

   Hoeveel verschillende soorten dynamieken danwel aantrekkings-matrices zijn er dan mogelijk?

   Hoeveel verschillende aantrekkings-matrices zijn er als $N = 10$ en

$$a \in \{-1.0, -0.9, \ldots, -0.1, 0.0, 0.1, \ldots, 0.9, 1.0\}?$$

6. (Particle life.)

   (a) Wat voor dynamiek op micro-niveau geeft een aantrekkings-matrix als alle rij-sommen nul zijn?

   (b) Wat voor dynamiek op micro-niveau geeft een aantrekkings-matrix als alle kolom-sommen nul zijn?

   (c) Wat voor dynamiek geeft een aantrekkings-matrix als alle rij- en kolom-sommen nul zijn?

7. (A life like system.) Thomas Schmickl *et al.* [29] ontdekten een eenvoudige regel waarmee kleine bewegende deeltjes kunnen samenkomen om patronen te vormen. Die patronen lijken op levende organismen [video]. Deze patronen kunnen groeien, zich voortplanten en zelfs hun eigen omgeving creëren, net als ecosystemen in het echte leven. De studie kijkt naar zowel het grootschalige gedrag van deze structuren als de kleinschalige regels welke ze creëren. Schmickl noemde hun systeem een life like system, en zo zullen wij het ook maar aanduiden.

   Deeltjes bewegen altijd vooruit met een stapgrootte v. Verder reageren ze op andere deeltjes door hun oriëntatie te veranderen:

$$\Delta\phi = \alpha + \beta \cdot (L + R) \cdot \mathrm{sgn}(R - L).$$

waarbij $0 \leq \alpha, \beta \leq 180$ parameters zijn, uitgedrukt in graden, L het aantal andere deeltjes representeert in de linker-semicirkel, en R het aantal andere deeltjes representeert in de rechter-semicirkel. De "sgn" staat voor de tekenfunctie: $\text{sgn}(x) = 1$ als $x > 0$, $\text{sgn}(x) = -1$ als $x < 0$, en anders is de signum-functie gelijk aan 0.

(a) Laat $(\alpha, \beta) = (0, 1)$. Bereken $\Delta\phi$ voor $(L, R) \in \{(l, r) \in \mathbb{Z} \times \mathbb{Z} \mid 1 \leq l, r \leq 10\}$. (Waarschijnlijk is het verstandig dit even te programmeren in een script.)

(b) Wat kan gezegd worden over de verandering in oriëntatie van een deeltje, als het aantal deeltjes in de linker-semicirkel het aantal deeltjes in de rechter-semicirkel geleidelijk gaat overstijgen?

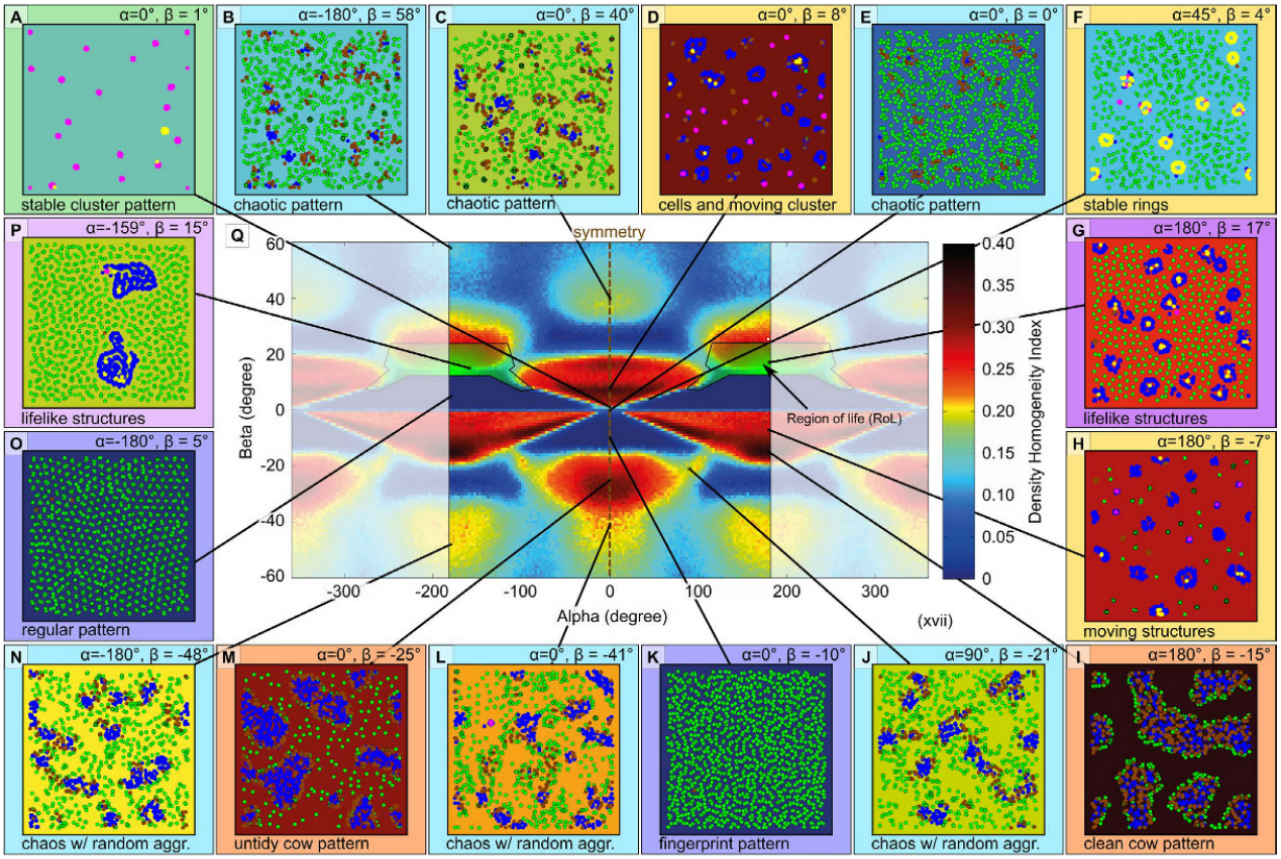


Fig. 5: HGI-waarden in de (α, β)-parameter-ruimte.

8. (A life like system.) In tegenstelling tot Mohr's particle life gedragen deeltjes in Schmickl *et al.*'s model zich identiek. Hoe ontstaan dan kleuren?

9. (A life like system.) In het artikel van Schmickl *et al.* wordt een zogenaamde *dichtheids-homogeniteits-index* (Eng.: density homogeneity index) gebruikt:

$$\text{DHI}_{\Theta,r} = \frac{\text{het aantal cellen met } \Theta \text{ of meer deeltjes binnen straal } r}{\text{het totaal aantal cellen}}$$

(a) Welke waarden bezitten Θ en r?

(b) Wat geeft een DHI aan, en waar wordt deze informatie voor gebruikt?

(c) Bekijk Fig. 5 op de pagina hiervoor. Welke kleur in de (α, β)-parameter-ruimte correspondeert met een hoge DHI? Controleer dit door de zwarte lijnen te volgen naar eigenlijke configuraties—of door naar de achtergrond te kijken van de eigenlijke configuraties: de kleur van de achtergrond correspondeert namelijk met de kleur op de DHI schaal.

(d) Geef de labels (hoofdletters) en/of namen van vier configuraties met een hoge DHI. Wat hebben zij gemeen?

(e) Geef de labels (hoofdletters) en/of namen van drie configuraties met een lage DHI. Wat hebben zij gemeen?

(f) Welke regionen in de parameter-ruimten worden gezien als interessant?

(g) Hoe worden deze regionen genoemd? Welk acroniem (welk afkorting) wordt daarvoor gebruikt?

(h) Welke kleur bezitten voornoemde regionen?

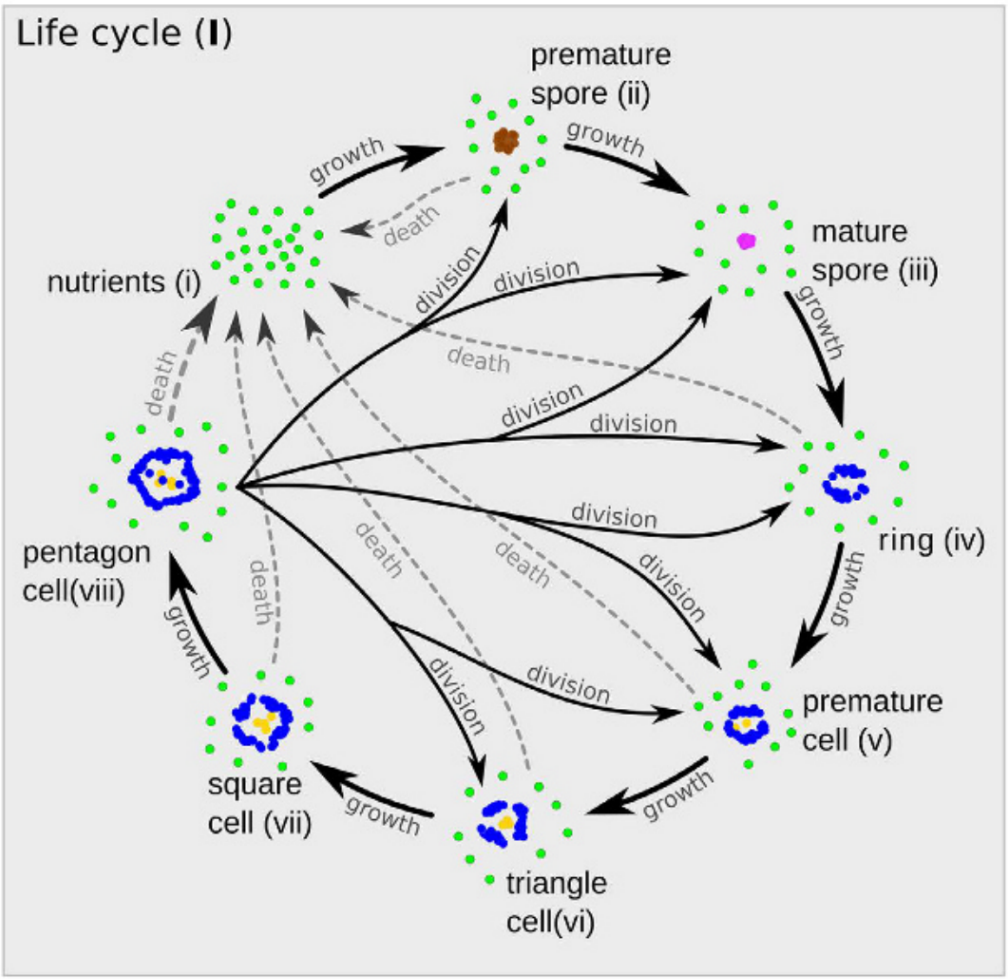


Fig. 6: De levens-cyclus van amoebe-achtige structuren.

10. (A life like system.) Verlopen overgangen in dit systeem synchroon of a-synchroon? Als overgangen synchroon verlopen, wat zou er dan, in het gedrag van het systeem, veranderen als

we in het systeem overgangen a-synchroon laten verlopen? Als overgangen a-synchroon verlopen, wat er dan veranderen als we in het systeem overgangen synchroon laten verlopen?

11. (A life like system.) Is dit systeem probabilistisch (van het toeval afhankelijk) of deterministisch? Waarom?

    Je zou de vraag ook zo kunnen formuleren: is elke run reproduceerbaar van uit een complete specificatie van begin-condities? Dan is het systeem deterministisch. Is elke run reproduceerbaar vanuit een complete specificatie van begin-condities, alleen met dezelfde seed van de random number generator? Dan is het systeem probabilistisch.

12. (A life like system.) Bekijk Fig. 6 op de vorige pagina. Daar geldt $r = 5$, $\alpha = 180^0$, $\beta = 17^0$, stapgrootte $v = 0.67$.

    (a) Waar kennen we deze cyclus van? (Verwijs naar een onderwerp uit één van de andere colleges.)

    (b) Hoe rechtvaardigen de auteurs het bestaan van deze cyclus in hun model?

13. Tim Hutton's Self-Reproducing Cells. [14] [YT] is een model van kunstmatige chemie.

    (a) Bekijk het hele YouTube filmpje (2:56 minuut). Beschrijf in eigen woorden wat er gebeurt.

    (b) Kleuren staan voor atoom-typen. Wat representeert de gele kleur? Wat representeren de atomen van verschillende kleur binnenin?

    (c) De eerste drie alinea's van Sectie 3.1 van het artikel beschrijven, in globale termen, de werking van Hutton's model. Vertaal deze beschrijving naar het Nederlands, en herformuleer waar nodig. Vat je beschrijving vervolgens samen in ongeveer 100 woorden.[19]

14. Fig. 7 bevat alle reactie-regels uit Hutton's model.

R1: e1a37 → e5a10	R13: x9y9 → x8 + y8	R24: a20 + a11 → a21a22
R2: a10 + e6 → a37e3	R14: a11a36 → a11a12	R25: e18a22 → e32 + a23
R3: e6e3 → e2e3	R15: f1 + a12 → f13a37	R26: e18a21 → e32 + a24
R4: x2y1 → x7y4	R16: x13y1 → x14y15	R27: a24 + a37 → a26a27
R5: x4 + y3 → x5y7	R17: a11 + x15 → a11x16	R28: a27a23 → a37 + a28
R6: x5 + x0 → x6x6	R18: x14y16 → x27y16	R29: a26a36 → a29a30
R7: x6 + y7 → x3y4	R19: x27a11 → x17 + a11	R30: a29a36 → a31a30
R8: x6y4 → x1 + y2	R20: x17y16 → x17y13	R31: a30 + a28 → a25a33
R9: x7y1 → x2y2	R21: x13e8 → x14e15	R32: a31a25 → a32 + a36
R10: f2a37 → f9a11	R22: e13a37 → e18a19	R33: a32a30 → a34a36
R11: a11 + f3 → a11f9	R23: e13a19 → e18a20	R34: a34a33 → a37 + a37
R12: x2y8 → x9y1		

Fig. 7: Reactie-regels uit Hutton's model.

    (a) Welke regels zijn verantwoordelijk voor het eerste stadium van celdeling? (Zie Fig. 3 van het artikel.)

[19] Het gebruik van automatische vertalers en generatieve AI is toegestaan, met dien verstande dat copy-pasten verboden is. Het is ook niet aan te bevelen. Gegenereerde zinnen zijn vaak redundant, lang en formeel. Bovendien wordt gecopy-paste tekst herkend door LLM-checkers. Wél is het toegestaan producten van LLM's te gebruiken als half-fabricaat.

(b) Welke regels zijn verantwoordelijk voor base-duplicatie? (Fig. 4 van het artikel.)

(c) Welke regels zijn verantwoordelijk voor het hechten van de voltooide genstring (gemarkeerd met een f-atoom) aan het membraan? (Fig. 5).

(d) Welke regels zijn verantwoordelijk voor het openritsen van de genstring?

(e) Welke regels zijn verantwoordelijk voor het uit elkaar trekken van de genstring? (Fig. 7 en 8.)

(f) Het effect van deze treksequentie op de cel wordt getoond in Fig. 9. Om de beweging van het aanhechtingspunt te faciliteren, wordt de cel gedwongen om in twee "lobben" te krullen, die elk één kopie van het genoom bevatten. Wanneer het aanhechtingspunt de bovenkant van het genoom bereikt (het e-uiteinde) kan het samensmelten met het membraan aan de andere kant, waardoor het uiteindelijke scheidingsproces wordt ingezet. Welke regels zijn hiervoor verantwoordelijk?

(g) Nadat membranen zijn samengesmolten vindt celdeling plaats. Welke regels zijn hiervoor verantwoordelijk?

(h) Welke regels zijn verantwoordelijk voor celgroei (in dit model komt dit neer op de groei van het membraan)? (Fig. 11.)

15. Lees Sec 4.1 uit het oorspronkelijke artikel, waarin Experiment 1 wordt beschreven.

(a) Waarom veroorzaakten mutaties, volgens Hutton, meestal cel-degeneratie in eerdere AChem-experimenten?

(b) Waarom evolueert de Qβ-replicatie in vitro naar de kortste mogelijke vorm?

(c) Hoe dragen moleculaire parasieten bij aan het verlies van genetische informatie?

(d) Hoe helpt een semi-doorlaatbaar membraan bij het behouden van genetische informatie?

(e) Wat is het doel van het verwijderen van reactie R3 in het experiment?

(f) Wat is de functie van de string *ecdbdbabccccadf* in het experiment?

(g) Waarom wordt de mutatiekans ingesteld op 1 op een miljoen per mogelijke gebeurtenis?

(h) Wat is de rol van overstromingen ("flooding") elke 8,000 iteraties?

(i) Waarom werd het voorouderlijke genoom vervangen door een licht gewijzigde versie in Fig. 8 op de volgende pagina (Figuur 18 in het oorspronkelijke artikel)?

(j) Wat is een neutrale mutatie en waarom is dat belangrijk?

(k) Hoe kunnen sommige onsuccesvolle soorten zich toch meerdere keren vermenigvuldigen voordat ze uitsterven?

(l) Waarom zou een enzymafbraak-mechanisme nuttig zijn in dit systeem?

16. Bekijk Fig. 8 op de pagina hierna (Figuur 18 in het oorspronkelijke artikel). Deze figuur licht Experiment 1 toe. Vat in eigen woorden samen wat de figuur poogt te illustreren.

17. Vat de portee van Experiment 2 in eigen woorden samen.

18. Vat de conclusie van Hutton's artikel in eigen woorden samen.

19. Geef in de onderstaande tabel de overeenkomsten en verschillen tussen Particle life, Schmickl *et al.*'s life like system, en Hutton's kunstmatige chemie.

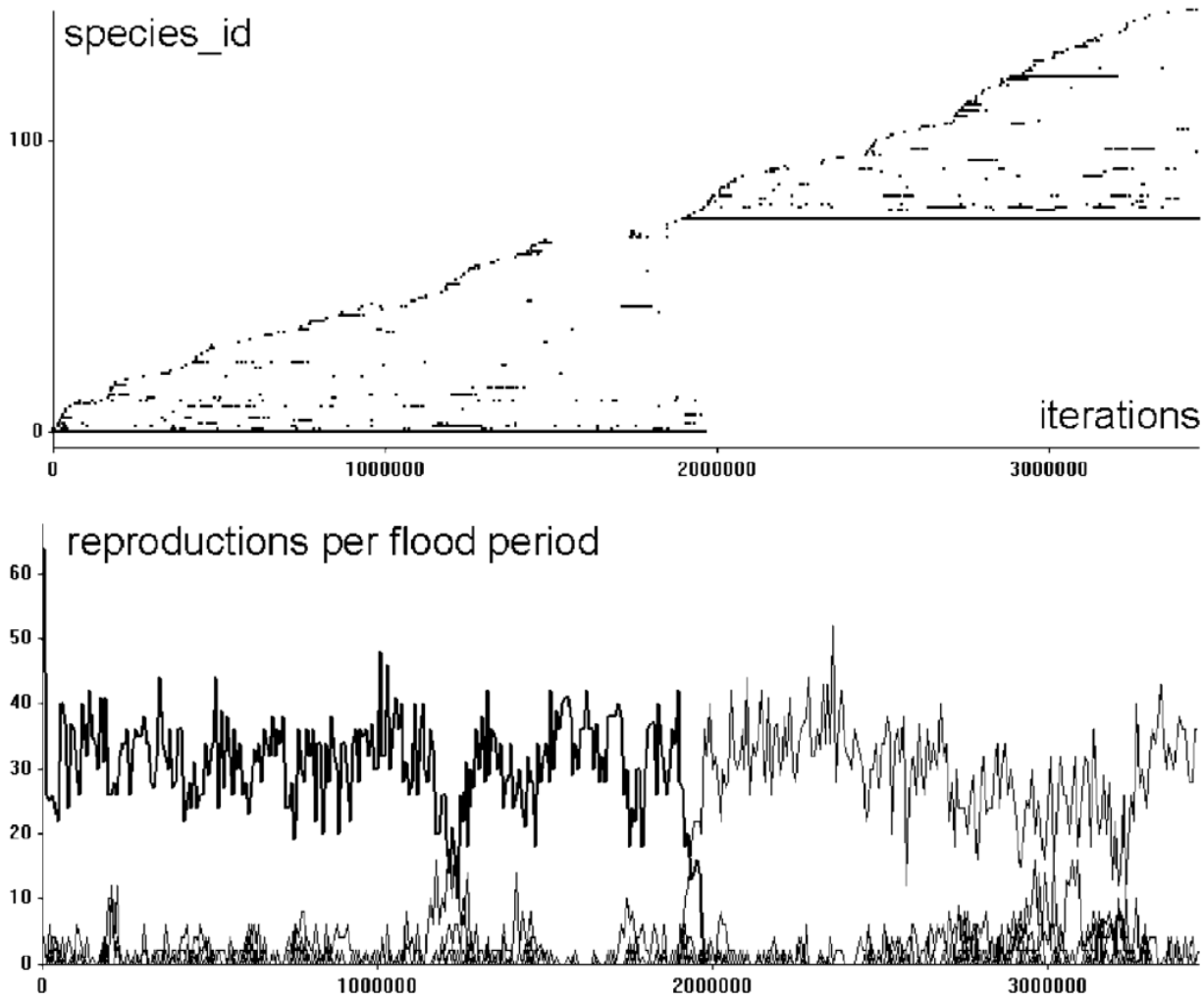
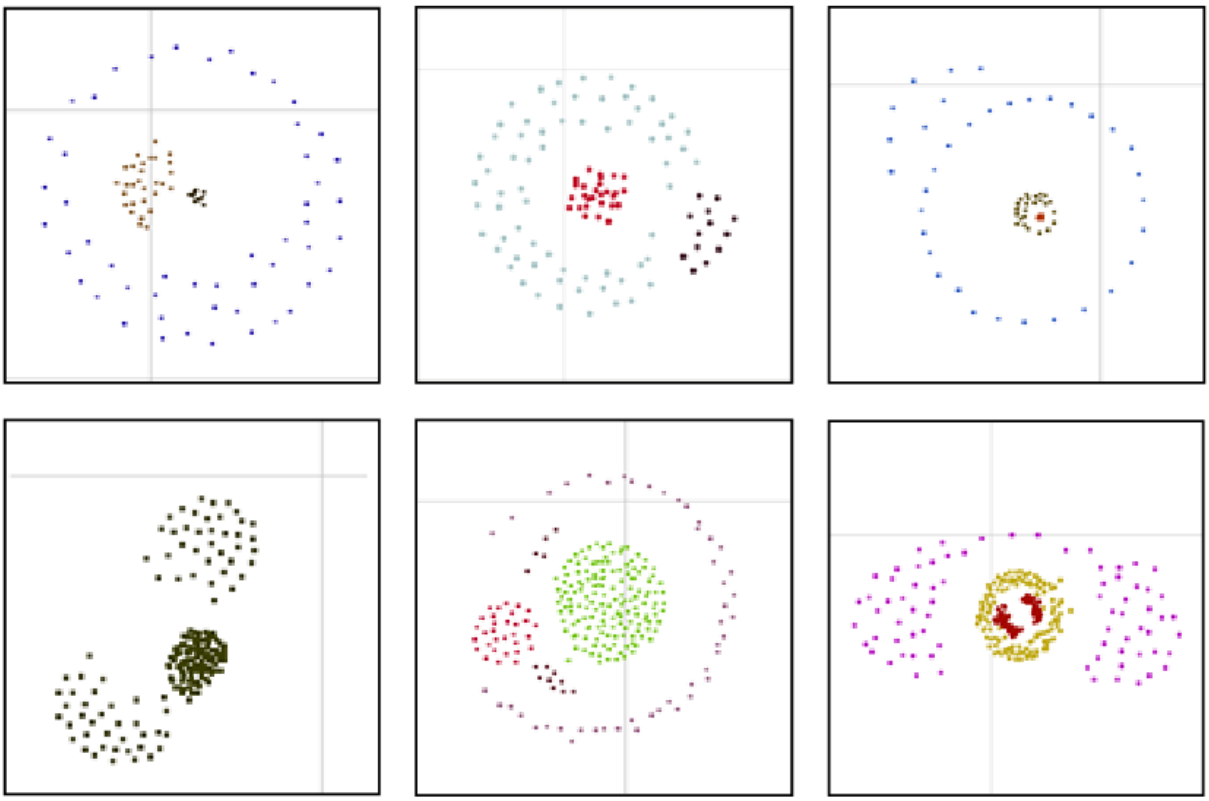


Fig. 8: Experiment 1: informatie blijft bewaard ondanks mutaties.

	Particle life	Schmickl's life like system	Hutton's kunstmatige chemie
Deeltjes			
Kleur			
Aggregatie-niveau			
Bindingen			
Aantrekking			
Mutatie			

13 Sayama's zwerm-chemie

Hiroki Sayama's Swarm Chemistry [28], is een uitbreiding van Craig Reynolds' klassieke Boids-model voor zwermvorming [26]. In tegenstelling tot Reynolds' model zijn deeltjes nu onderverdeeld in klassen (ook wel: typen, of: molecuulsoorten). Hier zie je visuele representaties van zes verschillende (en onafhankelijke) scenarios.



Elke klasse bezit een kleur, en wordt gekarakteriseerd door acht parameters. Gedurende een experiment blijven, per klasse, de parameters onveranderd.

parameter	*symbool*	*maximale waarde*
perceptie-radius	R	300 pixels
nominale snelheid	V_{nom}	20 pixels per tijdseenheid
maximum snelheid	V_{max}	40 pixels per tijdseenheid
cohesie-factor	c_1	1 eenheid per tijdseenheid^2
alignment-factor	c_2	1 eenheid per tijdseenheid
separatie-factor	c_3	100 pixels^2 per tijdseenheid^2
kans op willekeurig zwerven ("whim")	c_4	0.5
tempobehoud-factor	c_5	1

Sayama's algoritme werkt nu als volgt. Op enig moment bezit elk deeltje een positie-vector x, en een snelheids-vector v. Initialiseer de ruimte door alle deeltjes willekeurig te plaatsen en ze hun nominale snelheid mee te geven. Vervolgens wordt voortdurend per tijdseenheid (i.e., elke "tik")

de plaats- en snelheidsvectoren van alle deeltjes bepaald, echter zonder de deeltjes al daadwerkelijk te verplaatsen. Daarna worden, in dezelfde tijdseenheid alle deeltjes gelijktijdig verplaatst volgens hun nieuwe berekende snelheidsvector. Binnen één tijdseenheid zijn er dus twee achtereenvolgende globale stappen: voor alle deeltjes een individuele berekening (perceptie), en daarna voor alle deeltjes een simultane ruimtelijk update (move).[20]

Stap één: individuele berekening. 1. Van elk deeltje wordt eerst de versnellings-vector a bepaald. Deze hangt af van de waargenomen buren. Laat N bestaan uit alle buren van het deeltje, in straal R. Als het deeltje geen buren bezit, versnelt het in een willekeurige richting. Dit wordt "stray" genoemd, van "zwerven":

$$a = (r_{[-0.5,0.5]}, r_{[-0.5,0.5]}).$$

Hierbij representeert $r_{[-0.5,0.5]}$ een willekeurig getal uit het interval $[-0.5, 0.5]$.
Als het deeltje daarentegen wél buren bezit, dan:

(a) Bereken de gemiddelde positie, $\bar{x}$, en de gemiddelde snelheid, $\bar{v}$, van de waargenomen buren.

(b) Versnel in de richting die rekening houdt met positie en snelheid van buur-deeltjes:

$$a = c_1(\bar{x} - x) + c_2(\bar{v} - v) + c_3 \sum_{y \in N} \frac{x - y}{\|x - y\|^2}.$$

Dus a is ook een vector, en buren worden genoteerd met de vector y. Het algoritme schrijft voor dat deze versnellings-verctor a vervolgens kan worden verstoord met kans c_4 (Sayama noemt dit "whim"). In dat geval:

$$a = a + (r_{[-0.5,0.5]}, r_{[-0.5,0.5]}).$$

Dus als er buren zijn, wordt de versnellings-vector soms verstoord, en als er geen buren zijn wordt de versnellings-vector altijd verstoord. De versnellings-vector, a, is nu bepaald.

2. Vervolgens wordt de nieuwe snelheidsvector bepaald: $v_{\text{nieuw}} = v + a$. Omdat de nieuw berekende snelheid mogelijk te drastisch kan zijn veranderd, of te groot is, wordt deze nog in twee sub-stappen gecorrigeerd.

(a) Breng de norm (lengte, amplitude) zo nodig terug naar $V_{\max}$,—zonder uiteraard v_{nieuw} van richting te veranderen. Dit kan middels de formule

$$v_{\text{nieuw}} = \min\left\{\frac{V_{\max}}{\|v_{\text{nieuw}}\|}, 1\right\} v_{\text{nieuw}}.$$

Om deze formule te begrijpen is het goed om scalairen (i.e., gewone getallen) en vectoren uit elkaar te houden. Hoe in het algemeen vectoren kunnen worden afgekapt tot een bepaalde lengte, staat verder uitgelegd in Appendix A.3 op pagina 163.

(b) Behoud de nominale snelheid met een wegings-factor c_5:

$$v_{\text{nieuw}} = c_5 v_{\text{nieuw,nominaal}} + (1 - c_5) v_{\text{nieuw}}.$$

De vector $v_{\text{nieuw,nominaal}}$ is feitelijk gelijk aan v_{nieuw}, maar dan teruggebracht tot de nominale amplitude. De zojuist gegeven formule is het makkelijkst te begrijpen als de

[20] Dit is een gelijksoortig update-mechanisme als bij Conway's game of life.

scenarios $c_5 = 0$ en $c_5 = 1$ worden bestudeerd: in het eerste geval verandert v_{nieuw} niet, dus kan de amplitude ervan (de absolute snelheid, zonder te letten op de richting) behoorlijk variëren. In het tweede geval zal de absolute snelheid van v_{nieuw} volledig worden teruggebracht naar de nominale waarde. De originele update-formule is nu

$$v_{\text{nieuw}} = c_5 \frac{V_{\text{nom}}}{\|v_{\text{nieuw}}\|} v_{\text{nieuw}} + (1 - c_5) v_{\text{nieuw}}.$$

Voor verdere uitleg wordt weer verwezen naar Appendix A.3 op pagina 163.

De nieuwe snelheidsvector, v_{nieuw}, is nu bepaald.

Stap twee, simultane update: Geef alle deeltjes nu gelijktijdig de nieuw berekende snelheid, en de daar uit volgende nieuw berekende plaats: $v = v_{\text{nieuw}}$ en vervolgens $x = x + v$.

Dit completeert de beschrijving van Sayama's algoritme. Goed beschouwd kunnen stappen 2a en 2b worden samengebracht. Dan hoeft $\|v_{\text{nieuw}}\|$ maar één keer te worden berekend. We laten dit verder als oefening (cf. Opgave 3).

Meer details zijn overigens terug te vinden op Sayama's projectpagina, en in de slides van dit vak. Het oorspronkelijk achterliggende artikel kan zo nodig (bijvoorbeeld bij twijfel) worden gedownload van de vak-site [28]. Het is 10 pagina's lang met veel figuren, dus het valt mee.

Opgaven.

1. Noem én beschrijf drie verschillen tussen Swarm Chemistry en Boids. Illustreer deze verschillen ook met concrete voorbeelden.

2. Breid de tabel uit (het antwoord van) Opgave 19 op pagina 73 uit met een kolom voor Sayama's model. Dus:

	Particle life	Schmickl's life like system	Hutton's kunstmatige chemie	Sayama's swarm chemistry
Deeltjes	N typen	Homogeen, i.e., alle deeltjes zijn hetzelfde, en gedragen zich hetzelfde	N typen. Elk type bezit een eindig aantal toestanden	...
Kleur	Type deeltje	Aantal deeltjes in de omgeving	Type deeltje	...
⋮	⋮	⋮	⋮	⋮

3. Laat zien dat stappen 2a en 2b (het afkappen en verder corrigeren van de snelheids-vector) kunnen worden samengebracht. Wat is hiervan het voordeel?

4. (Rekensom.) Er wordt aangenomen dat er 3 klassen deeltjes bestaan, met parameters

Klasse	*perceptie*	V_{nom}	V_{max}	*cohesie*	*alignment*	*separation*	*pace keeping*
A	8	1.5	4	0.05	0.15	0.1	0
B	6	3	2	-0.02	0.05	0.2	0.5
C	7	4	3	-0.02	0.05	0.2	0.5

Voor het gemak wordt aangenomen dat er nooit verstoring plaatsvindt (i.e., de "whim"-parameter is nul). (Verstoring brengt namelijk willekeur in de vraagstelling, wat zou resulteren is meerdere mogelijke goede antwoorden door toeval, wat onwenselijk is.) Op tijdstip t bevindt het systeem zich in de volgende situatie:

	Klasse	*plaats* (sx, sy)	*snelheid* (vx, vy)
Deeltje 1	A	$(2, 3)$	$(1, 2)$
Deeltje 2	B	$(4, 5)$	$(1, 1)$
Deeltje 3	C	$(5, 4)$	$(1, 3)$

Bereken de resulterende bewegings-vector van het Type A-deeltje op tijdstip $t + 1$. Randvoorwaarden kunnen daarbij worden genegeerd. Veronderstel bijvoorbeeld een oppervlak zonder andere deeltjes, en zonder buitengrenzen.

5. (Variatie op Som 4 op de vorige pagina.) Als de vorige som, met het volgende verschil

Klasse	*perceptie*	V_{nom}	V_{max}	*cohesie*	*alignment*	*separation*	*pace keeping*
A	8	1.5	2	0.05	0.15	0.1	0.5
B	...	...	...	...	...	...	...
C	...	...	...	...	...	...	...

Dus alleen deeltjes van Type A bewegen zich anders. Bereken de resulterende bewegings-vector van het Type A-deeltje op tijdstip $t + 1$, als dezelfde uitgangsposities worden ingenomen als in Som 4.

6. (Rekensom.) Er wordt aangenomen dat er 3 klassen deeltjes bestaan, met parameters

Klasse	*perceptie*	V_{nom}	V_{max}	*cohesie*	*alignment*	*separation*	*pace keeping*
A	8	1.5	2	0.05	0.15	0.4	0.4
B	6	3	2	0.02	0.15	0.2	0.5
C	7	4	3	0.01	0.1	0.2	0.5

Er is geen zg. "swerve" parameter. Op tijdstip t bevindt het systeem zich in de volgende situatie:

	Klasse	*plaats* (sx, sy)	*snelheid* (vx, vy)
Deeltje 1	A	$(3, 1)$	$(1, 2)$
Deeltje 2	B	$(4, 5)$	$(1, 2)$
Deeltje 3	C	$(5, 4)$	$(1, 3)$

Bereken de resulterende bewegings-vector van het Type A-deeltje op tijdstip $t + 1$.

7. (Rekensom.) Er wordt aangenomen dat er 4 klassen deeltjes bestaan, met parameters

Klasse	*perceptie*	V_{nom}	V_{max}	*cohesie*	*alignment*	*separation*	*pace keeping*
A	2	1.5	5	0.05	0.15	0.4	0.2
B	6	3	2	0.02	0.15	0.2	0.5
C	7	4	3	0.01	0.01	0.2	0.6
D	6	1	5	0.02	0.15	0.2	0.3

Er is geen zg. "swerve" parameter. Op tijdstip t bevindt het systeem zich in de volgende situatie:

	Klasse	*plaats* (sx, sy)	*snelheid* (vx, vy)
Deeltje 1	A	$(4, 4)$	$(1, 3)$
Deeltje 2	B	$(6, 6)$	$(1, 2)$
Deeltje 3	C	$(5, 4)$	$(2, 3)$
Deeltje 4	D	$(6, 1)$	$(2, 2)$

Bereken de resulterende bewegings-vector van het Type A-deeltje op tijdstip $t + 1$.

8. Ontwerp een eigen voorbeeld-recept voor Sayama's Swarm Chemistry. Dit recept dient minimaal drie verschillende klassen te bevatten.

   Om ideeën op te doen wordt aangeraden Sayama's Swarm Chemistry simulator te gebruiken. Deze is beschikbaar op de officiële project-pagina. Gebruik daarvoor het JAR-bestand 1.2.0 of 1.3.0. Beiden zijn prima. Om deze te kunnen draaien is de Java Runtime Environment (JRE) nodig. De JRE alleen is voldoende—de SE Development Kit is niet nodig.

   Verder is het handig een LLM te vragen een voorzet te doen, bijvoorbeeld middels de prompt

   > *https://bingdev.binghamton.edu/sayama/SwarmChemistry biedt een aantal voorbeeldrecepten. Verzin een een nieuw, niet-bestaand, recept, met een interessante dynamiek.*

   LLMs zullen vermoedelijk met plausibel uitziende maar in werkelijkheid halfbakken regelsets komen. Het is aan jou of je deze voorstellen wil gaan verbeteren, of dat je toch zelf klassen gaat definiëren. Experimenteer vrijuit, voordat je met een definitief recept komt.

   Ben je er zo'n beetje uit, doe dan het volgende:

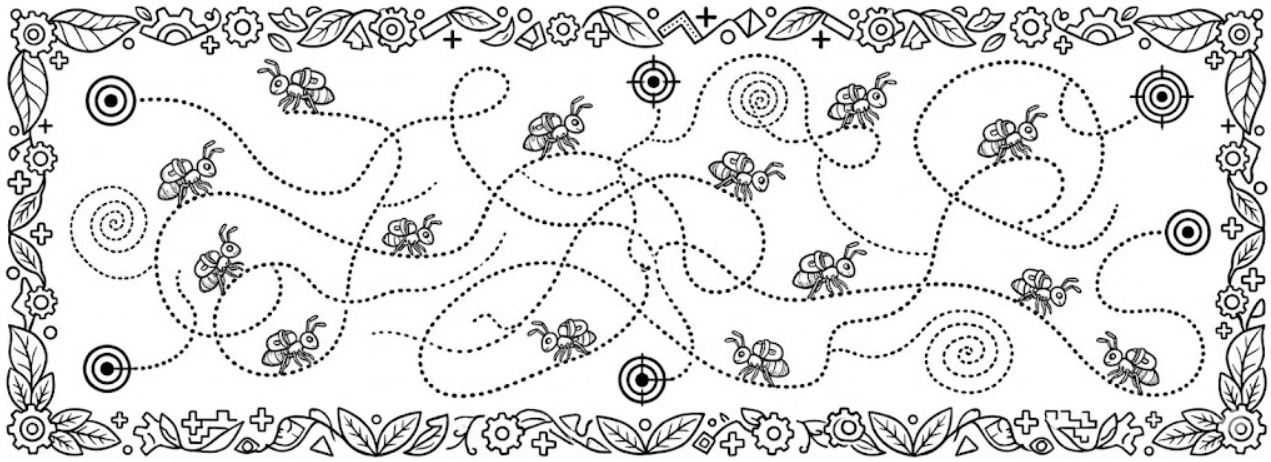
   (a) Benoem de parameterwaarden van elk ingredient (zoals cohesion, alignment, etc.).

   (b) Maak vier scherm-afbeeldingen van het resultaat: één aan het begin, één na 5 stappen, één na 10 stappen, en één na 50 stappen.

   (c) Bespreek je recept. Onderbouw in 200-500 woorden waarom je voor deze parameter-combinaties hebt gekozen. Gebruik daarbij zowel theoretische begrippen van Sayama's model, als observaties uit je experiment.

14 Mierenkolonie-optimalisatie

Mierenkolonieoptimalisatie (ACO) is een algoritme geïnspireerd door het foerageer-gedrag van mieren. Individuele mieren laten feromonen (geurstofjes) achter om paden naar voedsel te markeren. Kortere routes worden vaker belopen, waardoor de feromoon-concentratie stijgt en meer mieren worden aangetrokken.



Een associatieve impressie van Dorigo's mierenkolonieoptimalisatie-algoritme (Bron: GPT Image 1.5).

Mierenkolonieoptimalisatie en soortgelijke probabilistische zoekalgoritmen zijn essentieel om NP-moeilijke problemen aan te pakken, omdat dergelijke algoritmen heuristieken (i.e., zoekmethoden) bevatten die handig door de oplossingen-ruimte laat zoeken. In plaats van alle opties te berekenen, benadert ACO in veel gevallen optimale oplossing via collectieve, probabilistische padvinding. Op de benaderde oplossing ook echt optimaal is weten we typisch in dat soort gevallen niet (want anders hadden we zo'n oplossing wel direct, met een recht-toe-recht-aan algoritme kunnen uitrekenen; het probleem is dat voor NP-moeilijke problemen tot dusver geen efficiënte recht-toe-recht-aan algoritmes bekend zijn).

1. (NP-volledigheid.) Mierenkolonie-optimalisatie werd voor het eerst toegepast op het handelsreiziger-probleem.

   Het handelsreiziger-probleem is een belangrijke vertegenwoordiger van de klasse van NP-volledige problemen. Deze klasse is zeer groot en—niet alleen vanwege haar grootte—enorm belangrijk. De klasse van NP-volledige problemen bevat talloze even moeilijke problemen die naar alle waarschijnlijkheid alleen in exponentiële tijd kunnen worden opgelost.[21] (Preciezer: in een tijd de exponentieel afhangt van de grootte van het probleem.) NP-complete

[21] Naar alle waarschijnlijkheid. Dit is het bekende P =? NP vraagstuk, dat tot op de dag van het genereren van dit document, te weten 30 maart 2026, niet is opgelost. Als P = NP, dan kunnen NP-volledige problemen in polynomiale tijd worden opgelost. Er kan gerust worden gesteld dat de informatica-gemeenschap geen flauw idee heeft of P = NP. Als een individuele informaticus zou worden gedwongen geld in te zetten op dit vraagstuk, zou deze in de praktijk bijna altijd kiezen voor P ≠ NP. Dit is omdat alle tekenen die kant opwijzen. Maar een bewijs hiervan is tot aan dit jaar, 2026, altijd uitgebleven.

problemen kunnen onderling in niet al te lange tijd in elkaar worden vertaald. (Preciezer: in een tijd die polynomiaal afhangt van de grootte van het probleem.) Dat betekent dat, als we een efficiënt algoritme voor één NP-volledig probleem hebben, we een efficiënt algoritme voor *alle* NP-volledige problemen hebben.

(a) Wanneer is een beslis-probleem NP-oplosbaar? Geef een korte definitie. Het is toegestaan gebruik te maken van internet. Een definitie moet vervolgens wel in eigen woorden worden gegeven.

(b) Wanneer is een beslis-probleem NP-volledig? Geef een korte definitie.

(c) De NP-volledigheid van het handelsreiziger-probleem werd rond 1970 aangetoond. Dit werd gedaan door een ander beslis-probleem, waarvan al bekend was dat het NP-volledig is, in polynomiale tijd te relateren aan het handelsreiziger-probleem. In dit geval vervulde het Hamilton-cykel beslis-probleem deze rol. Van het Hamilton-cykel beslis-probleem was toen dus al bekend dat het NP-volledig is.

Leg uit in welke richting de probleem-reductie zal moeten plaatsvinden om te laten zien dat het handelsreiziger-probleem NP-volledig is. Meer concreet luidt de vraag als volgt: moet, om te laten zien dat het handelsreiziger-probleem NP-volledig is, aangetoond worden dat het handelsreiziger-probleem in polynomiale tijd kan worden gereduceerd naar het Hamilton-cykel beslis-probleem, of moet, juist omgekeerd, aangetoond worden dat het Hamilton-cykel beslis-probleem in polynomiale tijd kan worden gereduceerd naar het handelsreiziger-probleem? Motiveer je antwoord. (We vragen je overigens niet de reductie zelf te geven.)

2. (Mierenkolonieoptimalisatie). Gegeven zijn de tabellen 1, 2, en 3 op pagina 86. Verder $\beta = 0.5$ en $v(i,j) = 1/d(i,j)$.

(a) Completeer Tabel 3.

(b) Geef alle mogelijke toeren van een mier vanaf A bij exploitatie. Hoeveel toeren zijn dit?

(c) Hoeveel toeren zijn er mogelijk bij exploratie?

(d) Stel een mier staat op F en heeft A, B, C, E, en G al bezocht. Bereken

$$\Pr\{J \text{ wordt bezocht} \mid A, B, C, E, G \text{ zijn al bezocht}\}.$$

Zal de mier bij exploitatie J bezoeken? Waarom (niet)?

(e) Bereken de kans dat de mier bij exploratie J bezoekt.

(f) (Moeilijke vraag.) Bewering:

> Exploitatietoeren worden ook het vaakst afgelopen bij exploratie.

Becommentarieer.

3. (Mierenkolonie-optimalisatie.) In mierenkolonie-optimalisatie worden in het algemeen drie parameters gebruikt, te weten α, β en ρ.[22]

- $\alpha \in [0,1]$: neiging om te exploreren, dat wil zeggen, de neiging van een individuele mier om, proportioneel naar aantrekkelijkheid van een weg, een willekeurige weg[23] in te slaan (0 niet, 1 altijd).

[22] Bij het woord parameter ligt de klemtoon op de eerste a. Dus: parámeter.

[23] Onder een weg wordt een (bestaande) lijn van punt tot punt verstaan. Normaal wordt dit in grafentheorie een *kant* (Eng.: *edge*) genoemd.

- $\beta \geq 0$: weging van weg-lengte t.o.v. feromoon in de bepaling van de aantrekkelijkheid van een weg (0 irrelevant, $\to \infty$ relevant).
- $\rho \in [0, 1]$: verdampings-snelheid van feromoon (0 niet, 1 instantaan). In feite is dit gewoon een leerfactor.

Beschrijf wat er gebeurt in de volgende situaties:

(a) α is laag.
(b) α is hoog.
(c) β is laag.
(d) β is hoog.
(e) ρ is laag.
(f) ρ is hoog.

4. (Mierenkolonie-optimalisatie.) Beschouw de volgende bewering

   "Het handelsreizigerprobleem kan worden opgelost met mierenkolonie-optimalisatie."

   (a) Deze bewering is waar, immers het is bewezen dat mierenkolonie-optimalisatie PAC-compleet is.
   (b) Deze bewering is waar. Het aantal mogelijke oplossingen is immers begrensd. Alle mogelijkheden worden doorlopen, zodat op een gegeven moment het algoritme de optimale route moet vinden.
   (c) Deze bewering is onwaar. Mierenkolonie-optimalisatie kan alleen maar worden gebruikt om een oplossing te *benaderen*. Het is immers onbekend of het optimalisatiealgoritme op enig moment al een goede oplossing heeft gevonden.
   (d) Deze bewering is onwaar. Mierenkolonie-optimalisatie kan alleen maar worden gebruikt om een oplossing te *benaderen*. Het is mogelijk dat een benadering een oplossing blijkt te zijn. In zo'n geval kan er worden gestopt.

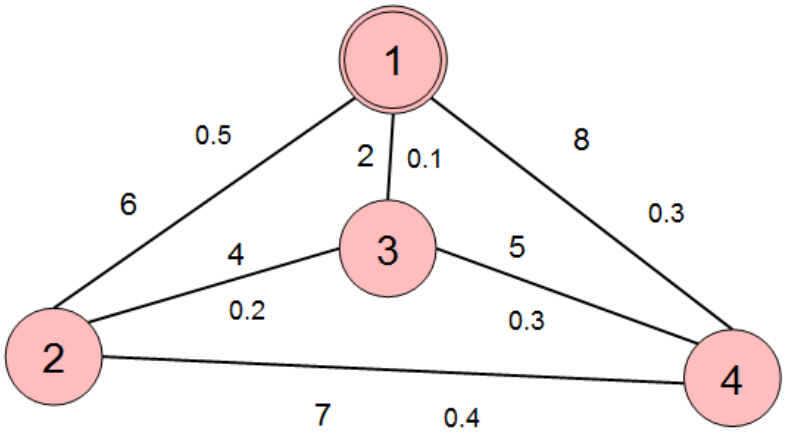


Fig. 9: Netwerk van steden, met stad 1 als start- en aankomstplaats.

.

5. We beschouwen de aanpak van het handelsreizigers-probleem (handelsreiziger-probleem) met behulp van mierenkolonie-optimalisatie. Gegeven is het netwerk van steden in Fig. 9. Afstanden worden gerepresenteerd door gehele getallen, en de hoeveelheid feromoon wordt gerepresenteerd door gebroken getallen.

(a) Wat is de uitkomst van het handelsreiziger-probleem-beslis-probleem met $M = 20$? Licht je antwoord toe.

(b) Wat is de uitkomst van het handelsreiziger-probleem-beslis-probleem met $M = 19$? Licht je antwoord toe. (Let op, de beantwoording van deze deelvraag ligt subtieler.)

(c) Wat is de uitkomst van het handelsreiziger-probleem-optimalisatieprobleem? Licht je antwoord toe.

(d) Bereken de aantrekkelijkheid van elke kant (link, weg, verbinding) als $\beta = 0.5$ en $v(i, j) = 1/d(i, j)$.

(e) Stel, een mier is, na stad 1 bezocht te hebben, gearriveerd in stad 3. Welke stad zal deze mier bezoeken bij exploitatie?

(f) Geef de volledige toer van een mier bij exploitatie.

(g) Stel, een mier is gearriveerd in Stad 3, na Stad 1 bezocht te hebben. Wat is de kans dat deze mier Stad 2 zal bezoeken bij exploratie?

(h) Wat is de kans dat, in dezelfde positie, deze mier stad 4 zal bezoeken bij exploratie?

(i) Geef alle mogelijke toeren van een mier bij exploratie. Hoeveel zijn dit er?

(j) Laat $\alpha = 0.9$. Geef alle mogelijke volledige toeren van een mier bij een mix van exploratie en exploitatie.

(k) Stel, generatie 11 levert een overwinnaar op die toer $1 - 2 - 3 - 4 - 1$ liep. Bereken van alle kanten (ook wel: links, of: edges) $\Delta m(i, j)$.

(l) Laat $\rho = 0.9$. Bereken op basis van het vorige antwoord van alle kanten de nieuwe feromoonwaarden.

(m) (Zes punten.) Stel dat na generatie 17 het feromoon er bij ligt zoals in Fig. 9. Stel verder dat generatie 18 bestaande uit 100 mieren volgens de gebruikelijke mix van exploratie en exploitatie over dit netwerk loopt. Hoe groot is de kans dat ten minste één van deze 100 mieren een optimale toer (toer met minimale lengte) aflegt?

6. Op een $N \times N$ grid, $N \geq 1$, ligt bij aanvang een hoeveelheid van f feromoon per cel. Daarna wandelen $K \geq 0$ mieren op dat grid die per tik per mier $D \geq 0$ feromoon deponeren ergens op dat grid. Nadat alle mieren hebben bewogen en feromoon hebben gedeponeerd, verdampt het feromoon met een snelheid van p, $0 \leq p \leq 1$. Dus als $p = 0.1$ dan verdampt er na elke tik 10% van het aanwezige feromoon.

   (a) Geef een formule voor de totale hoeveelheid feromoon op het grid na T tikken, $T \geq 0$.

   (b) Is er een bovengrens voor de totale hoeveelheid feromoon? Zo ja, geef een kleinste bovengrens. Zo nee, geef de (ultieme) groeisnelheid van de totale hoeveelheid feromoon.

7. We beschouwen de aanpak van het handelsreizigers-probleem (handelsreiziger-probleem) met behulp van mierenkolonie-optimalisatie.

   Het handelsreiziger-probleem beschouwt toeren. Een *toer* is een gesloten route die alle steden *precies* één keer aandoet. Het kan best zijn dat er een kortere route bestaat die alle steden *tenminste* één keer aandoet (denk aan een hub met zeer korte verbindingswegen naar elke stad), maar dergelijke routes tellen als oplossing niet mee.

   Als er over het handelsreiziger-probleem gesproken wordt, dan doelt men meestal op optimalisatie ("wat is de kortste toer zodanig dat ... ?"). In de complexiteitstheorie wordt het

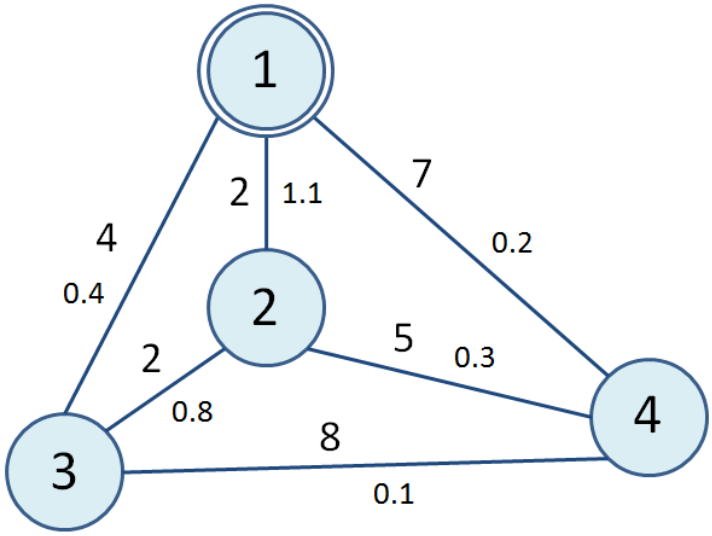


Fig. 10: Netwerk van steden, met stad 1 als start- en aankomstplaats.

.

handelsreiziger-probleem meestal beschouwd als beslis-probleem, i.e., als een ja/nee-vraag ("is er een toer, korter dan M, zodanig dat ... ?")

(a) Formuleer het handelsreiziger-probleem nauwkeurig als optimalisatie-probleem.

(b) Laat M gegeven zijn. Formuleer het handelsreiziger-probleem nauwkeurig als beslis-probleem.

(c) Licht de rol van M toe.

(d) Gegeven is het netwerk van steden in Fig. 10. Afstanden worden gerepresenteerd door gehele getallen, bijvoorbeeld $d(2,4) = 5$, en de hoeveelheid feromoon t.b.v. mierenkolonie-optimalisatie wordt gerepresenteerd door gebroken getallen, bijvoorbeeld $m(2,4) = 0.3$. Wat is de uitkomst van het handelsreiziger-probleem-beslis-probleem met $M = 20$? Licht je antwoord toe. (Mierenkolonie-optimalisatie speelt hier overigens nog geen rol.)

(e) Wat is de uitkomst van het handelsreiziger-probleem-beslis-probleem met $M = 8$? Licht je antwoord toe. (Let op, de beantwoording van deze deelvraag ligt subtieler.)

(f) Wat is de uitkomst van het handelsreiziger-probleem-optimalisatieprobleem? Licht je antwoord toe.

(g) Bereken de aantrekkelijkheid van elke kant (link, weg, verbinding) als $\beta = 2.0$ en $v(i,j) = 1/d(i,j)$.

(h) Stel, een mier is gearriveerd in stad 3, na stad 1 bezocht te hebben. Welke stad zal deze mier bezoeken bij exploitatie?

(i) Geef de volledige toer van een mier bij exploitatie.

(j) Stel, een mier is gearriveerd in stad 3, na stad 1 bezocht te hebben. Wat is de kans dat deze mier stad 2 zal bezoeken bij exploratie?

(k) Wat is de kans dat, in dezelfde positie, deze mier stad 4 zal bezoeken bij exploratie?

(l) Geef alle mogelijke toeren van een mier bij exploratie.

(m) Laat $\alpha = 0.9$. Geef alle mogelijke volledige toeren van een mier bij een mix van exploratie en exploitatie.

(n) Stel, generatie 11 levert een overwinnaar op die toer $1 - 2 - 3 - 4 - 1$ liep. Bereken van alle kanten (ook wel: links, of: edges) $\Delta m(i, j)$.

(o) Laat $\rho = 0.9$. Bereken op basis van het vorige antwoord van alle kanten de nieuwe feromoonwaarden.

(p) Stel dat na generatie 23 het feromoon er bij ligt zoals in Fig. 10. Stel verder dat generatie 24 bestaande uit 10 mieren volgens de gebruikelijke mix van exploratie en exploitatie over dit netwerk loopt. Is het mogelijk dat de overwinnaar van deze generatie een sub-optimale (niet optimale) toer flitst? Zo ja, hoe groot is die kans? Zo nee, waarom is dat niet mogelijk?

Tabellen

Tabellen behorende bij Opgave 2 op pagina 81.

	A	B	C	D	E	F	G	H	I	J
A		0.9	8.4	5.9	2.3	7.6	3.8	1.8	9.5	6.1
B			8.2	6.4	0.6	9.4	9.0	4.2	7.1	9.6
C				1.6	1.6	3.3	4.6	3.5	6.8	7.9
D					3.5	7.0	8.8	4.1	5.0	9.4
E						5.6	5.9	2.4	3.2	7.7
F							6.3	5.4	6.8	2.4
G								1.3	3.0	2.7
H									3.7	9.5
I										9.1

Tab. 1: De lengte van een kant.

	A	B	C	D	E	F	G	H	I	J
A		0.8	3.1	2.0	1.8	2.9	1.9	0.7	3.3	2.2
B			2.9	2.3	0.8	3.5	3.6	2.3	2.6	3.2
C				1.4	1.3	1.8	2.0	1.6	3.0	3.4
D					2.2	2.5	3.6	1.9	1.9	3.4
E						2.5	2.5	0.9	1.8	3.2
F							2.5	2.3	3.2	1.4
G								1.0	2.0	1.7
H									1.8	3.8
I										3.4

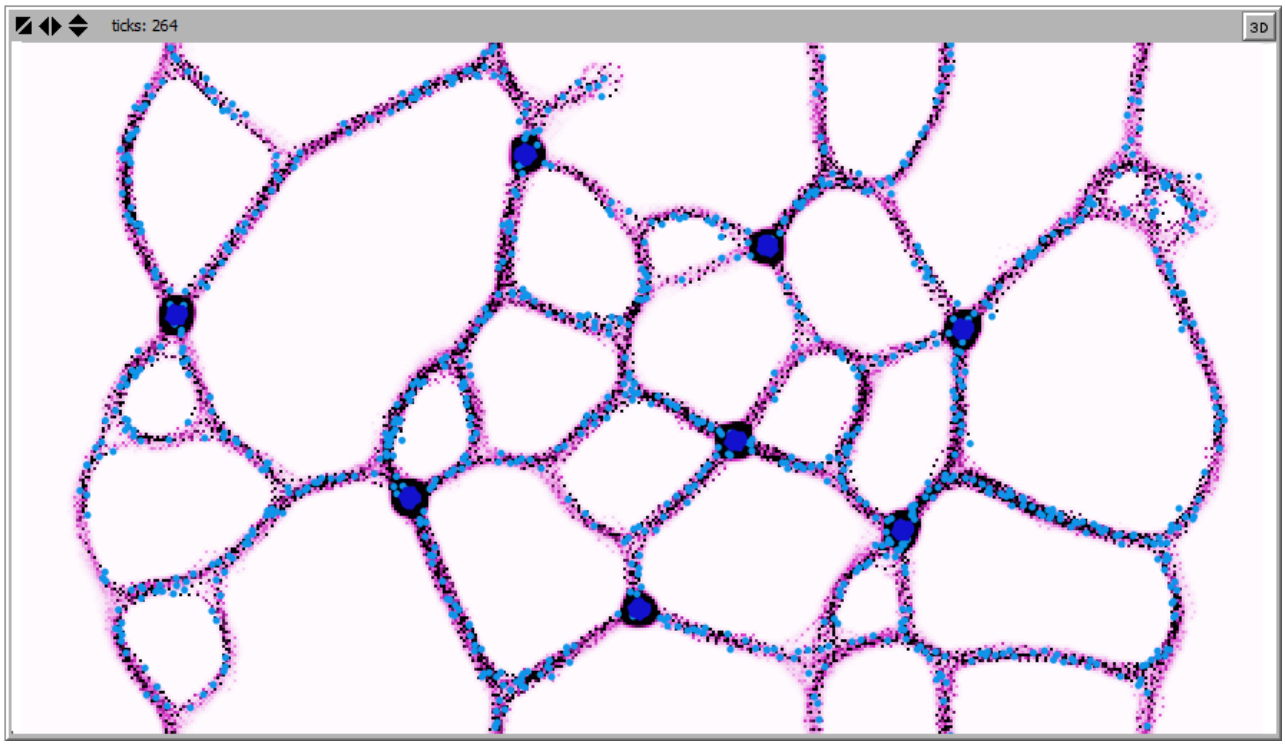
Tab. 2: De hoeveelheid feromoon op een kant.

	A	B	C	D	E	F	G	H	I	J
A		0.845	1.068	0.825	1.192	1.050	0.970	0.528	1.069	0.894
B			1.015	0.908	1.067	1.144	1.202	1.118	0.976	1.030
C				1.123	1.025	?	?	?	1.153	1.213
D					1.170	0.948	1.212	0.940	0.849	1.108
E						1.053	1.028	0.577	1.011	1.152
F							0.999	0.993	1.231	0.912
G								0.885	1.147	1.039
H									0.934	1.234
I										1.128

Tab. 3: De aantrekkelijkheid van een kant.

15 Slijmzwammen

Slijmzwammen zoals Physarum polycephalum zijn uniek omdat ze zonder centraal zenuwstelsel complexe logistieke puzzels oplossen. Voor NP-volledige problemen, waarbij het aantal mogelijke oplossingen bij elke uitbreiding explosief toeneemt, zijn traditionele computers vaak traag. Een slijmzwam vindt echter efficiënt de kortste route tussen voedselbronnen door een optimaal "buizennetwerk" aan te leggen. Onderzoekers gebruiken dit biologische "algoritme" om computermodellen te verbeteren voor complexe taken, zoals het ontwerpen van treinnetwerken of chip lay-outs.



Een inverted screenshot[24] van de Netlogo Physarum app, zoals gedemonstreerd op het hoorcollege. Blauwe punten zijn voedselplekken, wat in het model niet meer betekent dan dat er continu feromoon word gedeponeerd. Lichtblauwe schijfjes vertegenwoordigen deeltjes; zwart-tot-roze banen representeren feromoonsporen.

Het woord "algoritme" staat tussen aanhalingstekens, omdat een biologisch organisme niet echt volgens een algoritme werkt. Toch kan het functioneren van een eenvoudig organisme, met wat goedvinden, worden *gezien* als een algoritme: prikkels worden verwerkt en leiden tot een voorspelbare reactie, gestuurd door interne regels en de omgeving. Zo'n visie getuigt van de opvatting ook eenvoudige levensvormen informatie systematisch gebruiken. Tegelijkertijd roept het vragen op: doet deze benadering recht aan het doel en de complexiteit van leven? Voor nu laten we die vraag even rusten.

[24] De kleuren zijn omgekeerd (zwart wordt wit, etc.)

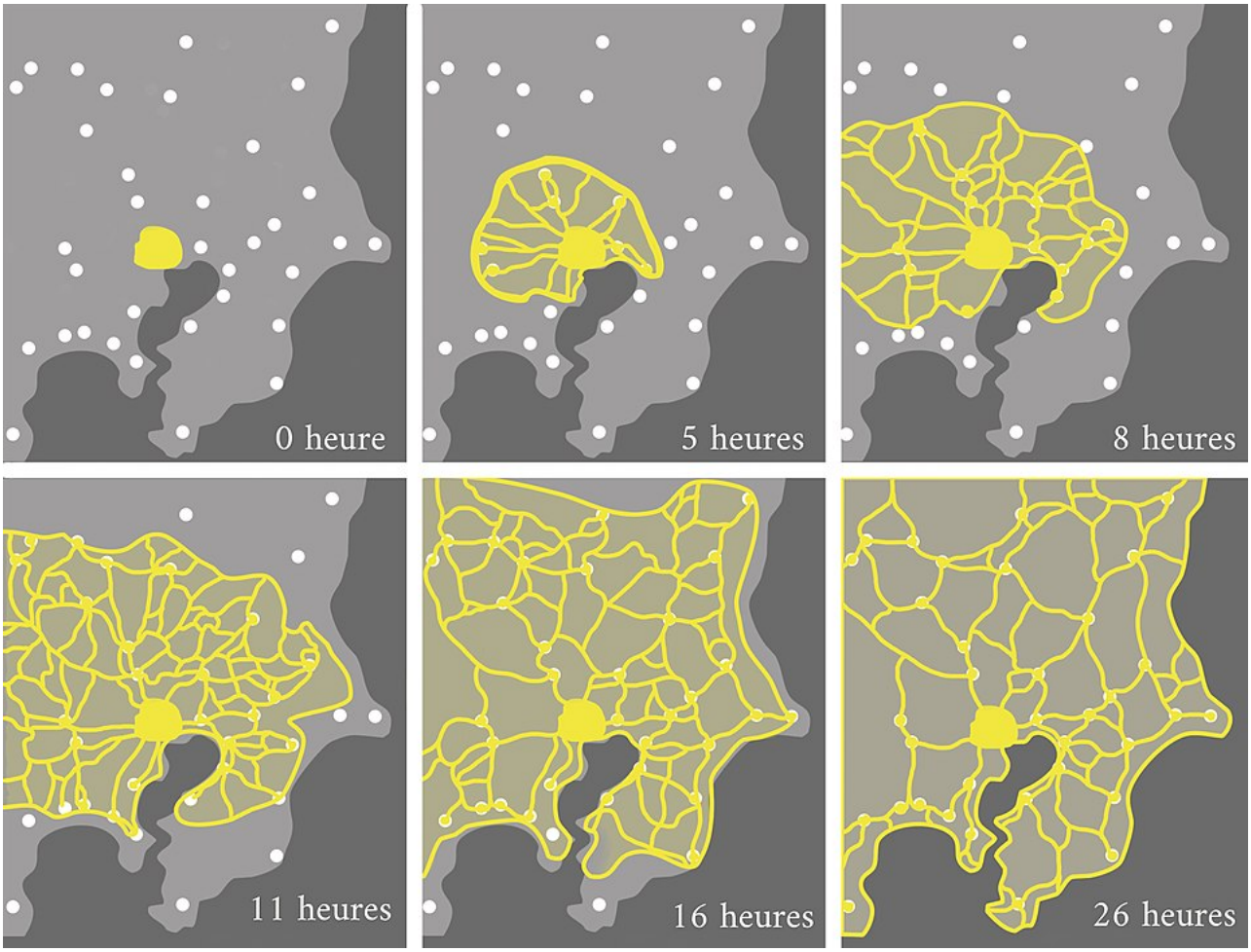


Fig. 11: Een netwerk, aangelegd door een computationeel model van Physarum (Bron: Wikipedia).

1. Het *Physarum polycephalum* is een felgele slijmzwam die groeit op schaduwrijke, koele, en vochtige plekken, zoals rottende bladeren en boomstronken. De physarum bestaat uit één grote enkele cel met meerdere kernen. Een interessante eigenschap is dat de morfologie (ontwikkeling in vorm) afhankelijk is van voedsel. Wanneer er geen voedsel is, zal de zwam zich met een minimale dikte verspreiden. Als er voedsel gevonden is, zullen kanalen naar het voedsel dikker worden en andere kanalen afsterven.

   In de jaren 2010-2014 is met name door de onderzoekers Jones en Adamatzky (Bristol) onderzoek gedaan naar de vraag of het gedrag van slijmzwammen kan worden nagebootst door een massief multi-agent systeem met eenvoudig gedrag op agent-niveau (Fig. 11). Het doel was algoritmen te ontwikkelen voor doolhof-exploratie en het heuristisch zoeken naar oplossingen in bekende NP-volledige problemen, zoals het TSP (het handelsreizigerprobleem). Voor het beantwoorden van de volgende vragen zul je op het internet moeten zoeken. We bekijken het werk van Jones en Adamatzky.

   (a) Om te beginnen is het goed te weten dat er ook een "Japanse school" is. Zie bijvoorbeeld de publicatie in Science (2010) door Tero, Takagi, Saigusa, Ito, etc., getiteld "Rules for Biologically Inspired Adaptive Network Design".

   Beschrijf kort het verschil in aanpak tussen de "Engelse school" (Jones, Adamatzky, en anderen) en de "Japanse school" in het simuleren van slijmzwamgedrag. Beschrijf met name de wiskundige verschillen.

(b) In de modellen van Jones *et al.* ziet een deeltje (op een aantal variaties na) er altijd op dezelfde manier uit. Het bezit een aantal sensoren (geursprieten) en die sensoren staan in een bepaalde hoek t.o.v. het lichaam. Beschrijf de opbouw van zo'n deeltje. Geef het aantal sensoren, de lengte van de sensoren, en de hoek die de sensoren maken met het lichaam. Benoem in dit verband de parameters SO en SA, en geef typische waarden van deze parameters.

(c) In de modellen van Jones *et al.* gedraagt een deeltje zich (op een aantal variaties na) altijd op dezelfde manier. Er is sprake van een sensorische fase (snuif en draai zo nodig) en een motorische fase (neem indien mogelijk een stap). Beschrijf de sensorische fase. Geef aan wanneer en hoe er afhankelijk van de hoeveelheid feromoon op de sensoren gedraaid wordt. Benoem in dit verband de parameter RA, en geef typische waarden van deze parameter.

(d) Beschrijf de motorische fase. Geef aan wanneer en hoe ver er gelopen wordt.

(e) Een mier staat op een plek waar de feromoonwaarden van alle drie de sensoren gelijk zijn. Wat zal de mier doen?

(f) Beschrijf alle gevallen waarin een mier naar links zal draaien.

2. (Feromoon in het Physarum-model van Jones en Adamatzky.)

(a) Beschrijf wat er in het Physarum-model gebeurt met het gedeponeerde feromoon. Verspreidt het zich? Verdampt het? Zo ja, waarom, hoe, en hoe snel? Zo nee, waarom niet?

(b) Feromoon wordt toegevoegd door optellen (zie boven). Waarom vindt verdamping dan niet plaats door aftrekken?

Reken bijvoorbeeld eens door wat er gebeurt als er op een $k \times k$ veld N mieren zijn die ieder per tik (of tijdsmoment) een verwachte hoeveelheid van $f > 0$ feromoon dumpen en er per tik van elke patch $F > 0$ feromoon wordt afgetrokken. Reken vervolgens eens door wat er gebeurt als er per tik $0 \leq d \leq 1$ feromoon verdampt. (In de berekening mag je er van uitgaan dat de verdampingsfase afgezonderd en als laatste proces wordt uitgevoerd in éń tijdseenheid (i.e., tik), dus nadat alle mieren feromoon hebben gelegd.)

(c) Beschrijf wat (je denkt dat) er zal gebeuren als SO (te) klein wordt.

3. (Slijmzwammen 3D.) Het idee om het gedrag van de gele slijmzwam met deeltjes te simuleren (Jones *et al.*) i.p.v. met buizen (Tero *et al.*) is vrij nieuw, zie Opgave 1. Op YouTube zijn beelden te vinden van werk van Jones *et al.* waar dit proces in 3D gesimuleerd wordt. Voorzover ons bekend is hierover nog niet gepubliceerd, ook niet door Jones *et al.* zelf. In deze opgave proberen we het gedrag van de deeltjes zelf te generaliseren naar 3D.

(a) In 2D bezit een deeltje (mier) drie sensoren: links, midden, en rechts. Hoeveel sensoren zou een deeltje in 3D kunnen, of moeten, bezitten? Hoe zouden deze sensoren gepositioneerd moeten zijn? Waarom?

(b) Bediscussieer het in het vorige onderdeel gevonden aantal sensoren, k, met het aantal ogen van de meeste organismen, 2. Geldt bij jou $k > 2$? Zo ja, waarom is $k = 2$ niet genoeg?

(c) We gaan nog even terug naar 2D. Laat F, L, en R de waargenomen feromoonhoeveelheden zijn. (Oorspronkelijk werden deze genoteerd als resp. FF, FL, en FR, maar we wijken voor het gemak van deze notatie af.) Verder wordt voor het gemak aangenomen dat bij waarneming deze drie grootheden altijd paarsgewijs van elkaar verschillen. (Ga na dat dat niet zo'n vreemde aanname is als we er van uitgaan dat we te maken hebben met

reëelwaardige grootheden.) Hoeveel ordeningen van F, L, en R zijn nu mogelijk? Maak een tabel waarin de rijen mogelijke ordeningen van F, L, en R bevatten, en de laatste kolom de actie op basis van die ordening. Bijvoorbeeld: de rij $F > L > R$ levert geen draaiactie op.

(d) Wijzig de 2D tabel zó, dat F zo'n beetje over de diagonaal loopt. (Of kijk in het antwoord van het vorige onderdeel.) Probeer een patroon te ontdekken. (Hint: kijk naar F.) Probeer aan de hand van dit patroon een draairegel op te stellen die abstraheert van sensorvariabelen $\neq \mathbf{F}$. Dus: probeer een draairegel op te stellen die het alleen heeft over de (waarden van) sensorvariable F, de (waarden van) sensorvariabelen die hoger scoren dan F, en de (waarden van) sensorvariabelen die lager scoren dan F.

(e) Generaliseer het draaigedrag nu naar 3D.

(f) Na de sensorische fase (snuiven) volgt een motorische fase (draaien en lopen). Generaliseer de motorische fase naar 3D. (Hint: zoek het niet te ver.)

16 Deeltjeszwerm-optimalisatie

Deeltjeszwerm-optimalisatie (particle swarm optimization, PSO) is een meta-heuristiek welke het sociale gedrag van groepen, zoals vogelzwermen, nabootst. Individuele deeltjes "vliegen" door een zoekruimte, geleid door hun eigen beste resultaat én de successen van de hele groep. Door deze collectieve intelligentie convergeert de zwerm efficiënt naar het globale optimum binnen (soms) complexe en (bijna altijd) hoog-dimensionale zoekruimtes.

1. Gegeven is een deeltje met plaats $s = (1, 1)$, en met snelheid

$$v = \begin{pmatrix} 2 \\ -1 \end{pmatrix}.$$

   De beste oplossing welke het deeltje tot nu toe zelf heeft gevonden bezit coördinaten $P = (6, 4)$. De tot nu toe beste globale oplossing bevindt zich op positie $G = (3, 4)$. Verder is de traagheid van deeltjes gelijk aan $I = 0.9$, en is de aantrekkingskracht van de globale oplossing gelijk aan $A_G = 0.4$. Er is geen ruis of verstoring. Bereken de nieuwe snelheidsvector.

2. (Als Vraag 1, met andere getallen.) Gegeven is een deeltje met plaats $s = (-1, 1)$, en met snelheid

$$v = \begin{pmatrix} -3 \\ 2 \end{pmatrix}.$$

   De beste oplossing welke het deeltje tot nu toe zelf heeft gevonden bezit coördinaten $P = (-7, 8)$. De tot nu toe beste globale oplossing bevindt zich op positie $G = (4, -5)$. Verder is de traagheid van deeltjes gelijk aan $I = 0.7$, en is de aantrekkingskracht van de globale oplossing gelijk aan $A_G = 0.2$. Er is geen ruis of verstoring. Bereken de nieuwe snelheidsvector.

3. (Als Vraag 1, met andere getallen, en alleen relatieve vectoren.) Gegeven is een deeltje dat met snelheid

$$v = \begin{pmatrix} 2 \\ -2 \end{pmatrix}$$

   beweegt. De tot nu toe beste oplossing van het deeltje bevindt zich op relatieve afstand $v_P = (1, 5)$. De tot nu toe beste globale oplossing bevindt zich op relatieve afstand $v_G = (3, 7)$. Verder is de traagheid van deeltjes gelijk aan $I = 0.8$, en de aantrekkingskracht van de globale oplossing gelijk aan $A_G = 0.7$. Er is geen ruis of verstoring. Bereken de nieuwe snelheidsvector.

4. Als Vraag 3, met ruis:

$$v_{\text{nieuw}} = Iv + (1 - I)[r_P A_P v_P + r_G A_G v_G], \tag{6}$$

   waarbij de randomgetallen r_P en r_G op enig moment de waarden respectievelijk 0.3 en 0.6 bezitten.

5. In Eq. 6 werd ruis (ook wel: willekeur, of randomness) geïntroduceerd. Goed beschouwd wordt dit gedaan volgens een alles-of-niets principe. Bijvoorbeeld, als $r_P = 0$, telt de particuliere oplossing, v_P, door toeval niet meer mee. Eigenlijk tellen bij ruis de particuliere en globale oplossingen nooit helemaal mee daar bijna altijd $r_P < 1$ en $r_G < 1$.

Het zou mooi zijn de mate van ruis met een parameter λ te variëren. De waarde $\lambda = 0$ zou dan corresponderen met 0% ruis, en de waarde $\lambda - 1$ zou dan corresponderen met 100% ruis. Hoe zou zoiets kunnen worden bewerkstelligd?

6. Het inbouwen van ruis in Eq. 6 op de pagina hiervoor vindt op een vrij conservatieve manier plaats. Immers, bij alle positieve waarden van r_P en r_G word de correctierichting

$$v_P + v_G$$

nog altijd gerespecteerd. Suggereer een wat radicalere manier om ruis in te bouwen. Uiteraard zijn er meerdere goede antwoorden.

7. Parameters van PSO kunnen onder meer de volgende zijn.[25] Er zijn ook wat redelijke waarden bij gezet.

Populatiegrootte	P	500
Initialisatieradius	R	15
Inertia	I	0.9985
Voorkeur voor de de tot nu toe globaal best gevonden oplossing	A_G	0.5
Ruis	λ	1.0
Snelheidslimiet	L	10

Als $R = 0$, dan is de afspraak dat de deeltjes uniform over de zoekruimte worden verspreid. Anders uniform in een schijf met straal R rondom, zeg, de oorsprong, of anders waar de programmeur aangeeft dat een run moet starten (bv. met een muisklik op het canvas).

Beantwoord voor elk van de parameters de volgende vragen:

- Wat zijn redelijke minimum- en maximumwaarden?
- Wat gebeurt er in in de dynamiek als een parameter wordt gezet op (zeer) lage waarden?
- Dezelfde vraag voor (zeer) hoge waarden.
- wat is het uiteindelijke effect voor convergentie in elk van die situaties?

Motiveer elk antwoord.

We beseffen dat er, goed beschouwd, ongeveer 36 vragen worden gesteld: er zijn zes parameters, en zes vragen per parameter. Ongeveer, want soms zijn tussenliggende gevallen ook nog interessant om te bespreken. Het werkt misschien het snelst als per parameter een analyse wordt gemaakt, en in die analyse elk van die zes punten wordt meegenomen.

[25] Bij het woord parameter ligt de klemtoon op de eerste a. Dus: parámeter.

17 Het vuurvliegjes-algoritme

Hierbij meteen het algoritme, omdat het anders niet op één pagina past. Op de volgende pagina volgt een wat wat intuïtievere uitleg.

Het vuurvliegjes-algoritme. Laat s_1 en s_2 de plaatsvectoren van twee vuurvliegjes zijn, waarbij $s_i \in [0, M]^2$, $i = 1, 2$, voor vaste $M \in \mathbb{N}$. Dus de vliegjes bewegen zich in een continu en begrenst vlak. (Continu: willekeurig kleine stapjes zijn mogelijk, dus geen hokjes zoals bij bijvoorbeeld cellulaire automaten.)

Het is de bedoeling dat elke tijdseenheid, elk van de vuurvliegjes zich verplaatst naar een zo licht mogelijke plek in het landschap $[0, M]^2$. De hoeveelheid licht op een plek is gewoonlijk gelijk aan de waarde van de doelfunctie, f, op die plek.

Stel bijvoorbeeld nu dat

$$f(s_j) > f(s_i),$$

wat betekent dat Vliegje j zich (op dat moment) op en lichtere plek bevindt dan Vliegje i. In dat geval blijft Vliegje j op haar plek, terwijl Vliegje i naar Vliegje j toebeweegt middels de update-formule

$$s_i^{\text{nieuw}} = s_i + \beta e^{-\gamma r_{ij}^2}(s_j - s_i) + \alpha\epsilon.$$

Hierbij zijn de parameters α, β, en γ schaalfactoren (i.e., knoppen om aan te draaien), en ϵ is ruis. Volgens de auteur van het vuurvliegjes-algoritme mag ϵ elke toevalsvariabele zijn, zolang de verwachting, μ, maar 0 is, en de standaardafwijking, ρ, in verhouding staat tot de overige parameters α, β, en γ [34, Sec. 10.3]. De auteur suggereert een uniforme verdeling op $[-1/2, 1/2]$, alsook de normale verdeling met parameters $(\mu, \rho) = (0, 1)$. Het lijkt hem niet erg uit te maken.

Parameters:[a]

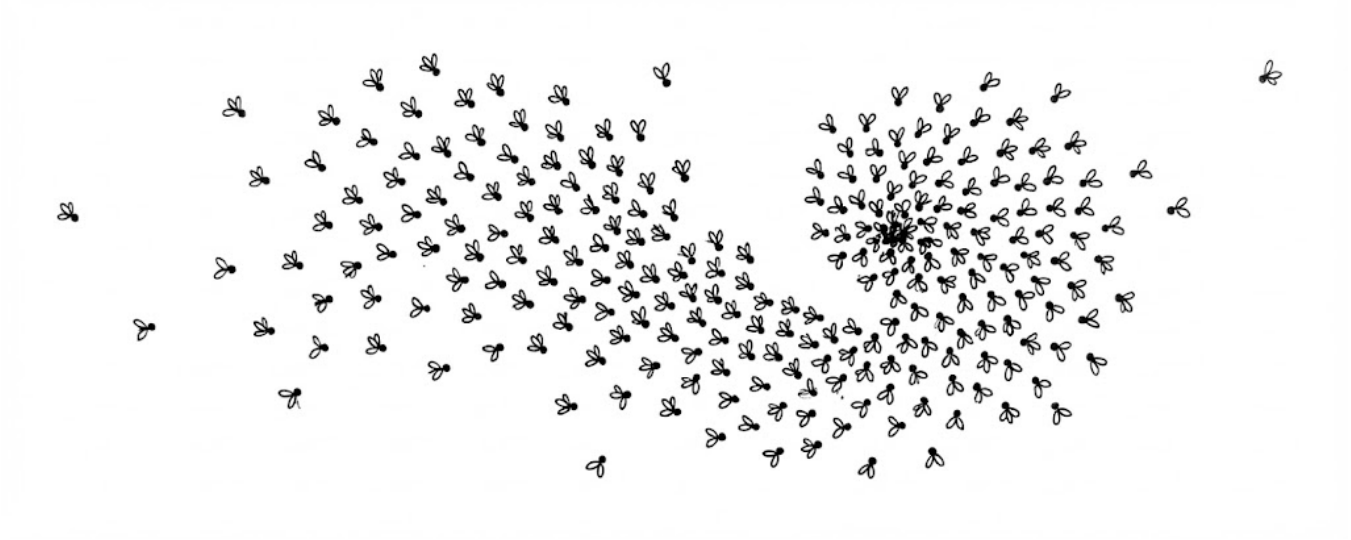
- De parameter α bepaalt de inbreng van ruis. Dus $\alpha = 0$ is geen ruis (ϵ word in dat geval dus genegeerd), $\alpha = 1$ neemt ϵ voor wat 'ie is, en bijvoorbeeld $\alpha = 2$ geeft de ruis ingebracht door ϵ een twee maal zo grote amplitude.
- De parameter β is een schaalparameter, die zorgt dat verplaatsingen niet te groot of te klein zijn.
- De parameter γ beïnvloedt de variatie van aantrekking (in dit geval de aantrekking van j door i), afhangende van de afstand tussen i en j, genoteerd als r_{ij}.

  Volgens de auteur is γ cruciaal in het bepalen van de convergentiesnelheid, en hoe het algoritme zich in het algemeen gedraagt (dit laatste is natuurlijk een open deur). In theorie is elke waarde $\gamma \geq 0$ geschikt, maar in de praktijk hangt γ volgens de auteur samen met "de karakteristieke lengte van het systeem dat moet worden geoptimaliseerd". Op basis van deze gegevens concludeert de auteur dat $\gamma \in [0.1, 10]$.

[a] Bij het woord parameter ligt de klemtoon op de tweede a. Dus: parámeter.

Nu een wat intuïtevere uitleg: het vuurvliegjes-algoritme is een meta-heuristiek (i.e., een zoekmethode) die het knippergedrag van vuurvliegjes nabootst. Het idee is dat de lichtintensiteit van een vuurvliegje de kwaliteit van een oplossing op die locatie representeert (het vuurvliegje is opgewonden of zo). Helderdere vliegjes trekken minder heldere soortgenoten aan, en die aantrekkingskracht neemt af met de afstand. Zo wordt voorkomen dat alle vuurvliegjes blindelings naar één enkel punt sprinten. Op deze manier behouden de vliegjes een gezonde balans tussen het grondig verkennen van hun eigen lokale omgeving (exploratie) en het gezamenlijk convergeren naar de beste oplossing (exploitatie). Eén en ander zorgt voor een dynamische spreiding, waardoor de zwerm de zoekruimte niet lukraak, maar op een wat diffusere—en misschien wel intelligentere—manier uitkamt. (Dit is nu niet onmiddellijk in te zien maar moet je nu maar even aannemen.) Zo bewegende, arriveert de zwerm via een collectieve dynamiek in de meeste gevallen naar een globaal optimum.

Omdat het vuurvliegjes-algoritme een meta-heuristiek is, is er altijd een kans dat de zwerm convergeert naar een lokaal optimum, dat minder goed is dan het globale optimum. Zo'n minder goed lokaal optimum wordt een *sub-optimum* genoemd.



Een associatieve impressie van het vuurvliegjesalgoritme (Bron: Gemini 3).

1. Bespreek de effecten van hoge en lage parameterwaarden van α, β, en γ. Hints:

   - α bepaalt de hoeveel ruis in verplaatsingen. Een lage waarde van α, te weten $\alpha = 0$, zorgt er voor dat het systeem als geheel deterministisch is.
   - β bepaalt de stapgrootte van verplaatsingen. Een minimale waarde van β, te weten $\beta = 0$, zorgt er voor dat deeltjes een pure zg. toevals-wandeling (Eng.: random walk) uitvoeren: elke tijdseenheid doen ze zomaar een stapje in welke richting dan ook.
   - γ bepaalt de de variatie van aantrekking. Een minimale waarde van γ, te weten $\gamma = 0$, zorgt er voor dat
   $$e^{-\gamma r_{21}^2} = e^{-0} = 1,$$
   zodat de aanpassing lineair is, ongeacht de afstand tussen s_1 en s_2.

2. (Rekensommen.)

(a) Gegeven is $\alpha = \beta = 1$, en $\gamma = 10^{-3}$. Verder is ϵ is de standaardnormale verdeling met $(\mu, \rho) = (0, 1)$, op elk van de twee coördinaten in de zoekruimte. (Dus 2x de normale verdeling.)

Bereken de nieuwe waarden van $s_1 = (1, 3)$ en $s_2 = (4, 7)$, als $f(1, 3) = 40$, $f(4, 7) = 52$, en ϵ op enig moment de waarde $(-1, 0.5)$ aanneemt. Duid nieuwe waarden van s_i, $i = 1, 2$.

(b) Gegeven is $\alpha = \beta = 2$, en $\gamma = 10^{-4}$. Verder is ϵ is de uniforme verdeling op $[-0.5, 0.5]$, op elk van de twee coördinaten in de zoekruimte.

Bereken de nieuwe waarden van $s_1 = (3, 1)$ en $s_2 = (5, 7)$, als $f(3, 1) = 20$, $f(5, 7) = 19$, en ϵ op enig moment de waarde $(0, 0.25)$ aanneemt. Duid de nieuwe waarden van s_i, $i = 1, 2$, en geef aan de hand daarvan commentaar op de keuze van $\beta = 2$.

3. (Verbeteringen en uitbreidingen.) In [33] worden verschillende verbeteringen van het vuurvliegjes-algoritme voorgesteld. Welke zijn dat? Wat zijn de effecten?

4. De slide "Fairness. Stel, er zijn 100 deeltjes" werd afgesloten met "zo maar een vraag": bereken de kans dat, na M runs, elk paar minstens één keer met elkaar is vergeleken. We gaan deze vraag beantwoorden in etappes.

   (a) Waarom is deze vraag belangrijk?

   (b) Beschouw het volgende experiment: selecteer eerst N elementen zonder teruglegging uit X, met $|X| = S$. Voor elk van de N geselecteerde elementen, selecteer K elementen, verschillende van het eerste element, ook zonder teruglegging uit X. Wat is de kans dat een specifiek geordend paar (u, v) wordt getrokken?

   (c) En wat is de kans dat een specifiek ongeordend paar $\{u, v\}$ wordt getrokken?

   (d) Wat is de exacte kans dat de exacte kans dat een specifiek paar $\{u, v\}$ minstens één keer voorkomt? Hint: gebruik het principe van in- en exclusie voor twee verzamelingen.

   (e) Wat is de kans dat, als dit experiment M keer wordt herhaald, elk ongeordend paar tenminste één keer wordt geselecteerd? (Hint: gebruik weer het principe van in- en exclusie.)

18 Een bestiarium van algoritmen

Binnen het vakgebied van door de natuur geïnspireerde algoritmen worden vaak metaforen gebruikt. Bijvoorbeeld, in Cuckoo Search worden eerst de levensloop, het foerageer-gedrag en de sociologie van koekoeken beschreven voordat de kern van het algoritme zelf wordt besproken. Door deze beeldspraak worden nature-inspired algoritmen soms onnodig vaag gepresenteerd, waardoor de technische essentie verborgen blijft.

Vaak worden bestaande methoden slechts in een nieuw jasje gestoken. Daarom moet dit onderzoeksveld met behoedzaamheid worden betreden, zodat de werkelijke waarde van een door-de-natuur-geïnspireerd algoritme correct wordt ingeschat.

1. Het is bekend dat mierenkolonie-optimalisatie (ant colony optimization, ACO) en deeltjeszwerm-optimalisatie (particle swarm optimization, PSO) onderzoekers hebben geïnspireerd tot het opstellen van alternatieve algoritmen die, eveneens als ACO en PSO, zijn geïnspireerd op verschijnselen in de natuur. Dit heeft geleid tot een wildgroei aan algoritmen.

   Om zowel te informeren als te vermaken, houden Claus Aranha (Tsukuba Universiteit) en Felipe Campelo (UFMG) een overzicht bij van dit soort algoritmen, genaamd An Evolutionary Computation Bestiary.

   (a) Verzin een dier, plant of andere biologische entiteit, en bekijk of aan de hand daarvan een algoritme is voorgesteld.

   (b) Onder het kopje “More info” wordt verwezen naar een parodie op de hele metafoor-gekkigheid. Welke parodie is dat? Welke vorm heeft deze parodie? Wat is,

globaal genomen, de boodschap van het product van deze parodie? En wat is de boodschap van de parodie zélf?

2. Open het artikel Mitigating Metaphors: A Comprehensible Guide to Recent Nature-Inspired Algorithms van Michael A. Lones [22].

   (a) Wat betekent het werkwoord (gebruikt als zelfstandig naamwoord, Eng.: *gerund*) "mitigating" in "mitigating metaphors"?

   (b) Bekijk Figuur 1. Wat is het meest geciteerde algoritme? Hoe vaak is dit algoritme geciteerd volgens Lones? En hoe vaak is het nu, anno 2026, geciteerd?

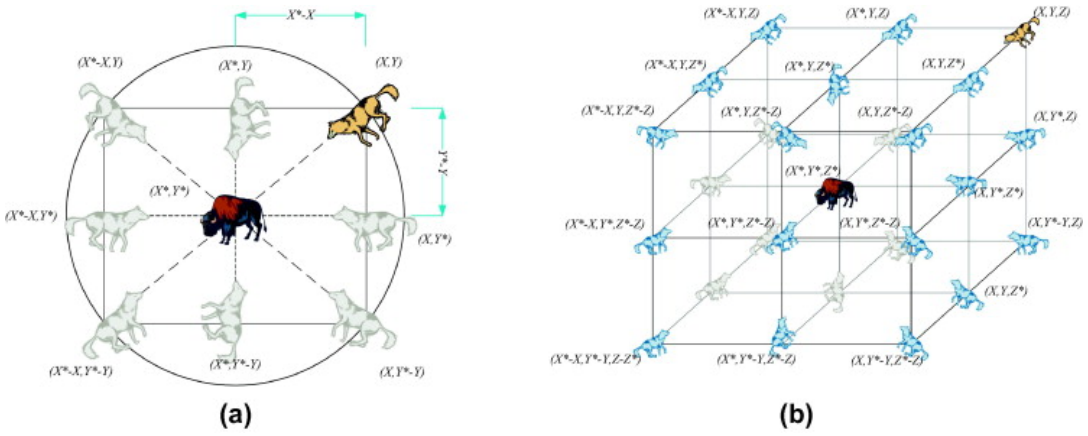
   (c) Welke, door Lones besproken, algoritmen maken gebruik van herstarts?

   (d) Welke algoritmen vormen spiraal-vormige zoeksporen rond (lokale) optima?

   (e) Welke algoritmen bezitten een duidelijke gelijkenis met Particle Swarm Optimization (PSO), doordat een populatie van zoekprocessen naar elkaar toe beweegt middels vectorgebaseerde bewerkingen?

   (f) Vat de conclusie van Lones' artikel in het Nederlands samen. Je mag online translation en LLMs gebruiken, als daarna maar een edit-slag wordt uitgevoerd. Je uiteindelijke samenvatting dient feitelijk correct te zijn.

3. Onderstaande afbeelding laat zien hoe een prooi (een bison?) word beïnvloedt in een omgeving met roofdieren (hyena's?). Uit welk artikel is onderstaande afbeelding afkomstig? (Tip: gebruik Google Lens, of Microsoft Lens.) Check hoe vaak dit artikel is geciteerd. Bespreek de betekenis van het aantal citaten.



(a) (b)

4. Open het artikel Metaphor-based metaheuristics, a call for action: the elephant in the room van Claus Aranha, Christian L. Camacho Villalón, Felipe Campelo (corresponderend auteur, en al dan niet toevallig ook de persoon die het bestiarium bijhoudt, zie Vraag 1), Marco Dorigo (de grondlegger van ACO), Rubën Ruiz, Marc Sevaux, Kenneth Sörensen, en Thomas Stützle [3].

   (a) Wat voor vorm bezit dit artikel? (Het antwoord is te vinden in de eerste alinea.)

   (b) Vat de boodschap van elke alinea samen in een sleutel-frase.

   (c) De auteurs van het pamflet zijn er van overtuigd dat het in het beste belang van iedereen is, dat artikelen waarvan de enige claim op wetenschappelijke bijdrage is dat ze een nieuwe bron van inspiratie hebben gevonden, niet langer worden gepubliceerd. Om deze reden roepen de auteurs op om hun redactionele beleid aan te passen, en een verklaring van de volgende strekking toe te voegen aan hun richtlijnen voor submissie. Hoe luidt deze verklaring?

Deel VI. Evolutionaire algoritmen

19 Simulated annealing

Simulated annealing (SA) past uitstekend bij evolutionaire algoritmen omdat beide stochastische zoekstrategieën zijn voor complexe optimalisatie-problemen. Beide meta-heuristieken delen principes van willekeurige variatie en het ontsnappen aan lokale optima, en beiden bootsen ze natuurlijke processen na om efficiënt door grote zoekruimtes te navigeren.

1. Teken een transparant blok, zoals in Fig. 12.

   (a) In simulated annealing is de kans op het accepteren van een slechtere oplossing gelijk aan

   $$P\{T, \Delta U)\} = e^{-\frac{\Delta U}{KT}}. \tag{7}$$

   Hierbij staat T voor de temperatuur, en ΔU voor het absolute verschil in uitkomst tussen de huidige oplossing en de gevonden oplossing. K is een constante. Wat is het bereik van deze grafiek op de z-as?

   (b) Wat (gebeurt er in de berekening) als $T = 0$?

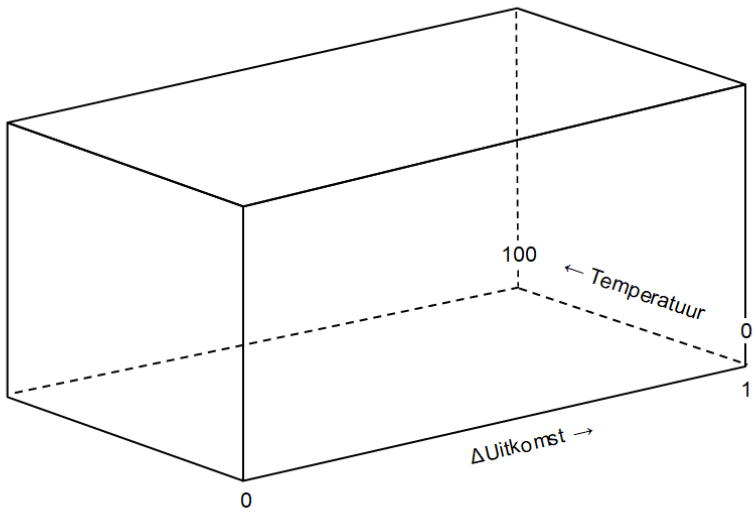


Fig. 12: Ruimte om de grafiek van $P\{T, \Delta U)$ te tekenen.

.

   (c) Bereken

   $$\lim_{T \downarrow 0} P\{T, \Delta U)\}.$$

   Dit kan formeel worden gedaan met een klassiek ϵ/δ-argument, of informeel worden gedaan met handgewapper (Eng.: hand waving).[26] Dat laatste kan zo: als in (Eq. 7 op de

[26] Hand waving is een officieel begrip in de wiskunde en informatica.

pagina hiervoor) de waarde van T naar 0 zakt bij vaste ΔU, dan gaat de waarde van de exponent van de e-macht naar $-\infty$. Voor de e-macht zelf betekent dat dan ...

(d) Teken, in Fig. 12 op de vorige pagina, de functie (7) voor waarden van T die dichtbij 0 liggen. Het is toegestaan de waarden op de lijn $T = 0$ zelf te tekenen.

(e) Teken, in dezelfde figuur, de functie (7) voor waarden $\Delta U = 0$.

(f) Schets (!) in dezelfde figuur, de functie (7) voor waarden $\Delta U = 1$. Neem maar aan dat $K = 1$. Maak vervolgens de figuur af, door alleen de doorsnijding van de grafiek met de randen van het blok te tekenen.

(g) Wat is de waarde van $P\{T, \Delta U)\}$ in (Eq. 7 op de pagina hiervoor) als zowel $T \downarrow 0$ als $\Delta U \downarrow 0$?

(h) Stel, simulated annealing vindt een slechtere oplossing bij $T = 0$. Geef de kans dat deze slechtere oplossing geaccepteerd wordt.

(i) Geef pragmatische redenen (i.e., gebruiksredenen) waarom een exponentiële functie gebruikt wordt voor de kans, en bijvoorbeeld geen lineaire functie (wat eenvoudiger zou zijn).

Eén reden heeft te maken met het feit dat zowel de temperatuur als het verschil in uitkomst in principe geen bovengrenzen hebben. I.e., de achterkanten van het blok zijn kunstmatig getrimd op $T = 100$, respectievelijk $\Delta U = 1$.

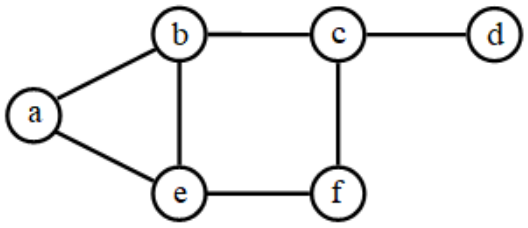


Fig. 13: Een ongerichte graaf G.

.

2. Het knopenbedekkingsprobleem, is het probleem om in een ongerichte graaf (een netwerk met ongerichte verbindingen) een zo klein mogelijke verzameling knopen (punten, vertices) K aan te wijzen, zodanig dat elke knoop in K zit, of een buur van K is.

   Als voorbeeld kan een netwerk van steden worden genomen, waarbij de letter K dan de verzameling van steden met een ziekenhuis representeert. Het zou fijn zijn als in elke stad een ziekenhuis staat, of direct verbonden is met een stad waarin een ziekenhuis staat. En uit kostenoverwegingen zou het fijn zijn als het aantal ziekenhuizen, $|K|$, minimaal is.

   We gaan een simulated annealing-algoritme schrijven dat het knopenbedekkingsprobleem zo goed mogelijk probeert op te lossen.

   (a) Bekijk de graaf G uit Fig. 1. Een *overdekking*, is een verzameling van knopen, K, zodanig dat elke knoop in K zit, of een buur van een element uit K is.

   Bepaal of de volgende verzamelingen overdekkingen van G zijn:

$A = \{a, c, f\}$; $B = \{d, e\}$; $C = \{c, e\}$; $D = \{b, f\}$.

(b) Een overdekking heet *minimaal* als elk punt in die overdekking nodig is. Dus als er een punt wordt uitgenomen, is het geen overdekking meer. Welke van bovenstaande verzamelingen zijn minimale overdekkingen?

(c) Een overdekking heet een *minimum* overdekking als het een overdekking is met zo weinig mogelijk knopen. Welke van bovenstaande verzamelingen zijn minimum overdekkingen?

(d) Is elke minimale overdekking een minimum overdekking? En omgekeerd?

(e) Een *partiële overdekking* is een verzameling van knopen, K, welke op zich geen overdekking is, maar kan worden uitgebreid tot een overdekking.

Geef alle partiële overdekkingen van G.

(f) Geef een algoritme (geen code!) waarmee een partiële overdekking kan worden uitgebreid tot een echte overdekking.

(g) Simulated annealing heeft altijd een functie nodig die vanuit een bestaande oplossing een nieuwe oplossing genereert. Stel zo'n functie voor. Dus stel een functie voor die vanuit een minimale (maar niet noodzakelijk minimum) overdekking, een nieuwe alternatieve minimale overdekking genereert. Deze functie wordt berekend door een algoritme. Dus als je een algoritme geeft, is de functie ook gegeven.

(h) Schrijf een simulated annealing-algoritme dat het knopenbedekkingsprobleem zo goed mogelijk probeert op te lossen. Beschrijf het algoritme in woorden, en niet in code.

20 Genetische algoritmen

Een genetisch algoritme is een slimme zoek-methode gebaseerd op de evolutietheorie van Darwin. De computer maakt verschillende oplossingen aan en laat deze met elkaar concurreren. De beste versies worden gecombineerd (gekruist) of licht aangepast (gemuteerd). Zo evolueert het systeem stap voor stap naar een goede, en misschien wel een meest optimale oplossing.

Genotypen (chromosomen) in een mating pool, klaar om eventueel, en in een licht gewijzigde vorm, over te gaan naar een volgende generatie, ook weer een mating pool (Bron: Gemini 3).

1. Laat P een populatie zijn ter grootte n. Van de individuen is de fitness bekend. Geef een algoritme om fitness-proportionele selectie op deze individuen toe te passen.

   Het is niet nodig je algoritme in Java o.i.d. te formuleren. Een voorschrift in natuurlijke taal is ook goed, en verdient eigenlijk de voorkeur.

   Bijvoorbeeld: een voorschrift om bijvoorbeeld de mediaan van n getallen te berekenen kan bijvoorbeeld de volgende zijn:

   > "Eén manier om de mediaan van n getallen $a_1, \ldots a_n$ te berekenen, is door ze te sorteren. Bij oneven n is het middelste getal de mediaan. Bij even n is het gemiddelde van de twee getallen bij het midden de mediaan".

2. Een genotype van een populatie van individuen wordt gerepresenteerd door bitstrings ter lengte N.

   (a) Opwarmertje: hoeveel verschillende genotypen zijn er?

   (b) Twee ouders bezitten een identiek genotype. Geef aan hoeveel verschillende soorten kinderen er kunnen ontstaan.

      i. Met éénpunts kruising (one-point crossover).

      ii. Met tweepunts kruising (two-point crossover).

      iii. Met een uniforme kruising (uniform crossover), waarbij het bitmask (crossover mask) bestaat uit vier nullen gevolgd door allemaal enen. (Neem aan dat $N \geq 4$.)

(c) De eerste ouder bestaat uit allemaal nullen terwijl de tweede ouder uit allemaal enen bestaat. Geef aan hoeveel verschillende soorten kinderen er kunnen ontstaan.

i. Met éénpunts kruising.
ii. Met tweepunts kruising ter lengte k. Teken eerst zelf voor $N = 8$ en $k = 3$.
iii. Met alle vormen van twee-punts kruising.
iv. Met een uniforme kruising (uniform crossover), waarbij het bitmask (crossover mask) bestaat uit vier nullen gevolgd door allemaal enen. (Neem aan dat $N \geq 4$.)

(d) Laten we twee genotypen *i-equivalent* noemen als ze op precies i plekken verschillen. Twee ouders zijn i-equivalent. Geef aan hoeveel verschillende soorten kinderen er kunnen ontstaan.

i. Met éénpunts kruising.
ii. Met tweepunts kruising ter lengte k.

3. Gegeven zijn de volgende genotypen, met daarachter de gemiddelde fitness. Selectie is fitness-proportioneel. Bij selectie wordt geen rekening gehouden met de multipliciteit van een genotype, i.e., hoe vaak dat genotype voorkomt.

0000: 6	0001: 20	0010: 4	0011: 18
0100: 9	0101: 12	0110: 14	0111: 6
1000: 12	1001: 7	1010: 5	1011: 26
1100: 13	1101: 36	1110: 8	1111: 4

(a) Bereken Pr{0***}, Pr{00**}, Pr{000∗}, Pr{***0}, Pr{**00}, en Pr{∗000}.

(b) Elke keer wordt de volgende generatie genotypen gevormd door (alleen) kruising, waarbij alleen op de binnenste drie plekken gesneden wordt. Bereken de kans dat bij een eenmalige kruising het genotype 0000 in de volgende generatie wordt geplaatst.

(c) De populatiegrootte is telkens 100. Bereken de kans dat de volgende generatie tussen de 5 en 10 exemplaren van het genotype 0000 bevat. De getallen 5 en 10 tellen ook mee. Hint: binomiale verdeling. Gebruik het web, een rekenmachine of een app.

4. Deze opgave behandelt fitness-proportionele selectie bij genetische algoritmen. Laat X een verzameling van genotypen zijn en $f : X \to R$ een fitnessfunctie.

Definieer voor elke niet-lege deelverzameling $H \subseteq X$

- H's totale fitness $\sum H =_{Def} \sum_{x \in H} f(x)$.
- H's (gemiddelde) fitness $f(H) =_{Def} (\sum H)/|H|$.

(a) Laat $p_s(x)$ de kans zijn dat een element $x \in X$ geselecteerd wordt bij fitness-proportionele selectie. Toon aan:

$$p_s(x) = \frac{1}{|X|} \frac{f(x)}{f(X)}.$$

Hint: $p_s(x) = f(x)/(X$'s totale fitness), en $\sum X = |X| f(X)$.

(b) Laat zien dat de kansen inderdaad sommeren tot 1, i.e., laat zien dat $\sum_{x \in X} p_s(x) = 1$.

(c) Laat zien dat zg. "fitness-proportionele selectie" inderdaad fitness-proportioneel selecteert, i.e., laat zien dat, als $f(x) = a \cdot f(y)$ voor $a \neq 0$ en $x, y \in X$, dat dan ook $p_s(x) = a \cdot p_s(y)$. Geldt deze identiteit trouwens ook voor $a = 0$?

(d) Definieer $p_s(H) =_{Def}$ de kans dat een element uit H wordt geselecteerd. Toon aan:

$$p_s(H) = \frac{|H|}{|X|}\frac{f(H)}{f(X)}.$$

(e) Bewijs Holland's schemastelling:

$$p_s(H) \geq \frac{|H|}{|X|}\frac{f(H)}{f(X)}(1-\epsilon_m)(1-\epsilon_c).$$

waarbij ϵ_m gelijk is aan de kans dat schema H beschadigd wordt door mutatie, en ϵ_c gelijk is aan de kans dat schema H beschadigd wordt door cross-over.

5. Beschouw een GA met populatiegrootte $n \geq 1$. De eerste generatie bestaat uit genotypen

$$G = \{g_1, \ldots, g_n\}$$

met fitness respectievelijk $f_1 < f_2 < \cdots < f_{n-1} < f_n$, waarbij $f_i \in [0, \infty)$. (Dus de fitness van g_1 is f_1, de fitness van g_2 is f_2, enz.) Een mating pool ter grootte n wordt gevuld door n keer toernooi-selectie op G toe te passen, telkens met toernooi-grootte $k \geq 1$.

(a) Hoe groot is de kans dat g_1 na één toernooi (bijvoorbeeld het eerste toernooi) als winnaar uit de bus komt? Let op: als een toernooi-groep ter grootte k geselecteerd wordt hoeft g_1 daar niet per sé in te zitten.

(b) Hoe groot is de kans dat g_1 na N toernooien niet in de mating pool terecht is gekomen?

(c) Hoe groot is de kans dat g_i na één toernooi als winnaar uit de bus komt?

(d) Hoe groot is, uitgedrukt in i, k, m en n, de kans dat g_i na n toernooien precies m keer, $0 \leq m \leq n$, in de mating pool terecht is gekomen? (Hint: binomiale verdeling met p gelijk aan antwoord vraag 5c.)

(e) Bereken de verwachte gemiddelde fitness van de mating pool.

Hint: gebruik het gegeven dat de verwachting van een binomiale verdeling met parameters n en p gelijk is aan np. Bereken hiermee voor elke i het aantal verwachte exemplaren van g_i in de mating pool. Gebruik vervolgens het gegeven dat de verwachting van een som gelijk is aan de som van de verwachtingen.

6. Beschouw een GA met populatiegrootte $N \geq 1$ en een gemiddelde fitness, f, van in generatie t (er geldt $t \geq 0$, met $t = 0$ de index van de startgeneratie). Neem verder aan dat generatie t een aantal van $w \geq 1$ winnaars bevat. Deze winnaars bezitten allen een fitness van F. Uiteraard geldt dan $F \geq f$.

We leggen een mating pool aan met behulp van fitness-proportionele selectie (de gewogen roulettewiel methode) en we doen niet aan elitisme, dat wil zeggen dat de winnaar(s) niet op voorhand in de mating pool wordt (worden) geplaatst.

(a) Bereken de kans dat een winnaar in de mating pool wordt geplaatst, uitgedrukt in f, F, w en N.

- Hint 1: herinner je dat de mating pool wordt gevuld door N keer fitness-proportioneel een individu uit generatie t te selecteren.
- Hint 2: je kunt naar het antwoord toewerken door achtereenvolgens de volgende grootheden te berekenen. Daarbij zul je veelvuldig gebruik willen maken van complementaire kansen.

i. De totale fitness, T, in generatie t.
ii. De kans dat, bij één selectiemoment, dus bij één keer spinnen van het gewogen roulettewiel, een specifieke winnaar, bv. winnaar nr. 1 wordt gekozen.
iii. De kans dat, bij één selectiemoment, een willekeurige winnaar wordt gekozen.
iv. De kans dat, bij één selectiemoment, geen winnaar wordt gekozen.
v. De kans dat alle N selectiemomenten geen winnaar wordt gekozen.
vi. De kans dat, na N selectiemomenten, tenminste één winnaar werd gekozen.

(b) Bewijs met behulp van onderdeel 6a, en door gebruik te maken van

$$\lim_{n\to\infty}(1+\frac{x}{n})^n = e^x$$

dat de kans dat een winnaar door fitness-proportionele selectie in de mating pool wordt geplaatst, bij toenemende populatiegrootte convergeert naar een waarde die altijd groter is dan $1 - e^{-1} \approx 0.63$.

(c) Noem tenminste vier nadelen van fitness-proportionele selectie. (Hint: zie dictaat Wiering. Drie nadelen staan al bij elkaar.)

7. (GA voor doolhoven, Marco Wiering, 2007)

Stel je hebt een agent en je wilt een doolhof probleem oplossen m.b.v. een genetisch algoritme. De doolhof bestaat uit N toegankelijke velden. De agent kan de 4 acties: $\{Oost, Noord, West, Zuid\}$ uitvoeren in elk veld. De agent weet altijd in welk veld hij is. Er is tenslotte 1 doelveld waar de agent naar toe moet.

(a) Bedenk een representatie voor het oplossen van dit probleem. Hoe groot is de zoekruimte (mogelijk aantal individuen)?

(b) Bedenk een mutatie operator voor dit probleem.

(c) Bedenk een crossover operator welke twee individuen combineert naar een nieuwe individu.

(d) Bedenk een fitness functie voor dit probleem.

(e) Zou de GA efficient zijn voor het oplossen van dit probleem? Licht je antwoord toe.

8. (a) Beschrijf toernooi-selectie.

(b) Noem tenminste vier voordelen van toernooi-selectie.

(c) Beschouw een GA met populatiegrootte $N \geq 1$ en toernooi-selectie met groepsgrootte $k \geq 0$. Neem verder aan dat individuen in dit geval een unieke absolute fitness bezitten, en dat bij een paarsgewijze vergelijking altijd het meest fitte individu wint.

Bereken de kans dat het meest fitte individu bij toernooi-selectie in de mating pool terecht komt.

(d) Bewijs dat de kans dat het meest fitte individu door toernooi-proportionele selectie met groepsgrootte k in de mating pool wordt geplaatst, bij toenemende populatiegrootte convergeert naar $1 - e^{-k}$.

9. Deze opgave behandelt de notie “schema” in genetische algoritmen.

Omdat een schema geassocieerd wordt met een hypothese noteren we deze meestal met een hoofdletter H. Een schema H correspondeert eenduidig met de verzameling van chromosomen die onder dat schema vallen. Bijvoorbeeld als $H = 02 * 1$ dan ook $H = \{0201, 0211, 0221\}$.

De populatie X op tijdstip t wordt genoteerd als X_t. We spreken ook wel over "de generatie t" of "de te generatie". Analoog wordt H_t gedefinieerd als $X_t \cap H$.

(a) Bepaal de kans dat een schema H behouden blijft na fitness-proportionele selectie uit generatie t. I.e., bepaal $p_s(H_{t+1})$. (Hint: bekijk het laatste onderdeel van Werkcollege-opgave 4.)

(b) Beschouw chromosomen ter lengte L over een alfabet ter lengte K. Hoeveel verschillende chromosomen zijn er?

(c) Hoeveel verschillende schemas?

(d) Hoeveel verschillende schemas "passen" op één chromosoom? (Hint: voor elk allel zijn er twee keuzes: het allel zelf of een sterretje.)

(e) Geef definities van de orde, $o(H)$, en de definiërende lengte, $\delta(H)$, van een schema H. (Gebruik bronnen.)

(f) Hoeveel (super-) schemas passen op een (sub-) schema H met $o(H) = m$?

(g) Een *kritieke plek* in een schema is een plek zonder sterretje. Bepaal de kans p_m dat een éénpunts-mutatie plaatsvindt op een kritieke plek in een schema. Antwoord met $p_m = \ldots$ en motiveer. (Hint: gebruik orde dan wel definiërende lengte.)

(h) Bepaal de kans ϵ_m een schema H verdwijnt door een éénpunts-mutatie. Antwoord met $\epsilon_m < p_m$ of $\epsilon_m \leq p_m$ of $\epsilon_m > p_m$ of $\epsilon_m \geq p_m$ en motiveer.

(i) Bepaal de kans dat een schema H bewaard blijft na een éénpunts-mutatie. Motiveer.

(j) Een *definiërende plek* in een schema is een plek op of tussen uiterste niet-sterretjes. Laat H een schema zijn. Bepaal de kans p_c dat een kruising plaatsvindt op een definiërende plek. Antwoord met $p_c = \ldots$ en motiveer.

(k) Bepaal de kans ϵ_c dat een schema H verdwijnt door een crossover. Motiveer.

(l) Bepaal de kans dat een schema H behouden blijft na een crossover. Motiveer.

(m) Een volledige slag bestaat uit selectie, crossover en mutatie, allen met een zeker kans. Bepaal de kans dat een schema H behouden blijft na selectie, crossover en mutatie. Schrijf:

$$p(H_{t+1}) = \ldots$$

Op de plaats van de gelijkheid mag ook een ongelijkheid of strikte ongelijkheid staan.

10. Mu/lambda evolutionaire strategie, afgekort tot μ/λ-ES, is een familie van ge-abstraheerde zoekalgoritmen, gebaseerd op het principe van natuurlijke selectie.[27] De eerste ideeën werden gepubliceerd door Rechenberg en Schwefel, in 1960. Veel op de natuur geïnspireerde algoritmen (Hoofdstuk 18) blijken te kunnen worden teruggebracht tot instanties van μ/λ-ES. Samen met ACO (Ant Colony Optimization), PSO (Particle Swarm Optimization) en Simulated Annealing, vormt de klasse van μ/λ-ES algoritmen dan ook de vier pijlers van evolutionaire algoritmen.

μ/λ-ES werkt als volgt. Zet μ en λ vast, bijvoorbeeld $\mu = 25$ en $\lambda = 100$. Het idee is om herhaaldelijk een populatie van μ genotypen om te zetten in een nieuwe populatie van ook weer

[27] Het acroniem ES staat voor *evolutionary strategy*, en niet voor het zichzelf ook wel plausibel klinkende *evolutionary search*.

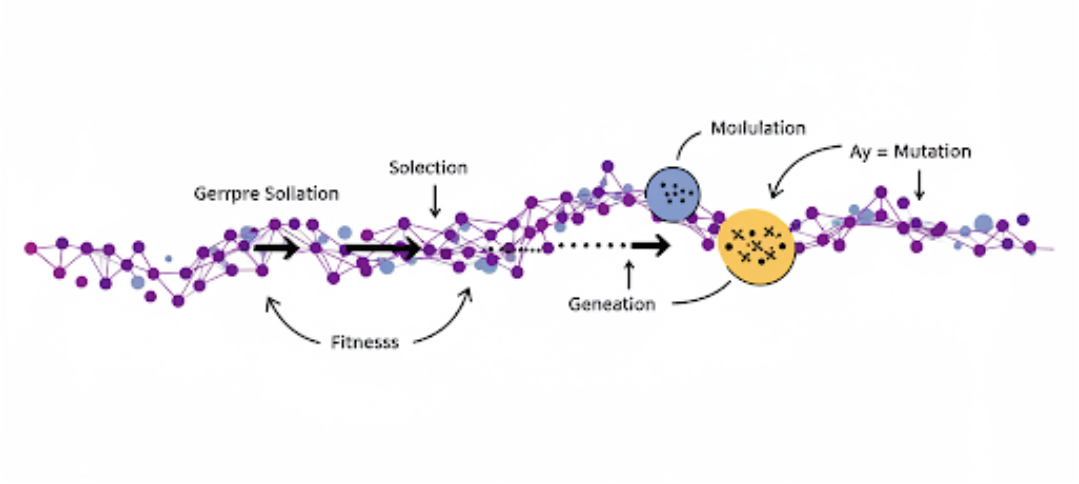


Fig. 14: Een associatieve impressie van μ/λ-evolutionaire strategie (Bron: Gemini 3).

μ genotypen, met hopelijk gemiddeld genomen een hogere fitness, en hopelijk met behoud van top-individuen. Een nieuwe generatie wordt gecreëerd door steeds willekeurige ouders te combineren (kruisen en/of muteren, afhangende van wat is afgesproken), tot aan λ kinderen.

In de notatie van μ/λ-ES worden de μ en de λ gescheiden door plus- danwel een komma-teken. Als de μ en de λ worden gescheiden door een plus-teken, worden de μ ouders samen met de λ gegenereerde kinderen in een selectiepoel geplaatst. De selectiepoel is dan dus $\mu + \lambda$ elementen groot. Bij een komma worden alleen de gegenereerde kinderen in de selectiepoel geplaatst. De selectiepoel is in dat geval dan dus λ elementen groot. De nieuwe generatie wordt tenslotte gevormd door de beste μ elementen uit de selectiepoel.

Merk op dat ouders aselect gekozen worden om kinderen te produceren. Er is geen fitness-gebaseerde selectie of iets dergelijks. Hetzelfde geldt voor de samenstelling van de nieuwe generatie. De nieuwe elementen, bestemd voor de volgende generatie, worden niet fitness-proportioneel uit de selectiepoel geplukt—althans niet volgens de originele definitie van μ/λ-ES. (Er wordt in de literatuur van de evolutionaire algoritmen heel wat geschmierd.)

Beantwoord de volgende vragen. Beschrijf ook de selectiepoel, dat is, de verzameling van individuen waar uiteindelijk uit wordt geselecteerd om de volgende generatie te vullen. Raadpleeg zo nodig [30] danwel het actuelere maar niet wezenlijk andere [4].

(a) Beschrijf $(1 + 1)$-ES. Aanwijzing: op elk moment bestaat de populatie uit één individu. Het proces zal neerkomen op hill climbing.

(b) Beschrijf $(1 + \lambda)$-ES. Aanwijzing: nog steeds geldt $\mu = 1$, maar er worden nu λ kinderen uit dat ene ouder-individu gegenereerd.

(c) Beschrijf $(1, \lambda)$-ES. Kan het ouder-individu terug voorkomen in de volgende generatie?

(d) Waarom is $(1, 1)$-ES niet zinvol? Aanwijzing: hier wordt één ouder weggegooid zonder het te laten concurreren met het geproduceerde kind.

(e) Beschrijf $(\mu + \lambda)$-ES.

(f) Beschrijf (μ, λ)-ES. Waarom moet $\lambda \geq \mu$?

(g) Bij $(\mu/\rho + \lambda)$-ES worden elke generatie ρ ouders uit een populatie van μ individuen gekozen, bijvoorbeeld middels fitness-proportionele selectie. Deze elite van ρ ouders wordt gebruikt om λ kinderen te produceren.

Gelden er restricties op ρ? Hoe ziet de selectiepoel er uit?

(h) Beschrijf $(\mu/\rho, \lambda)$-ES.

11. Leg uit waarom (μ, λ)-ES: te prefereren is boven $(\mu + \lambda)$-ES als de fitnessmaat doorheen het evolutieproces kan veranderen.

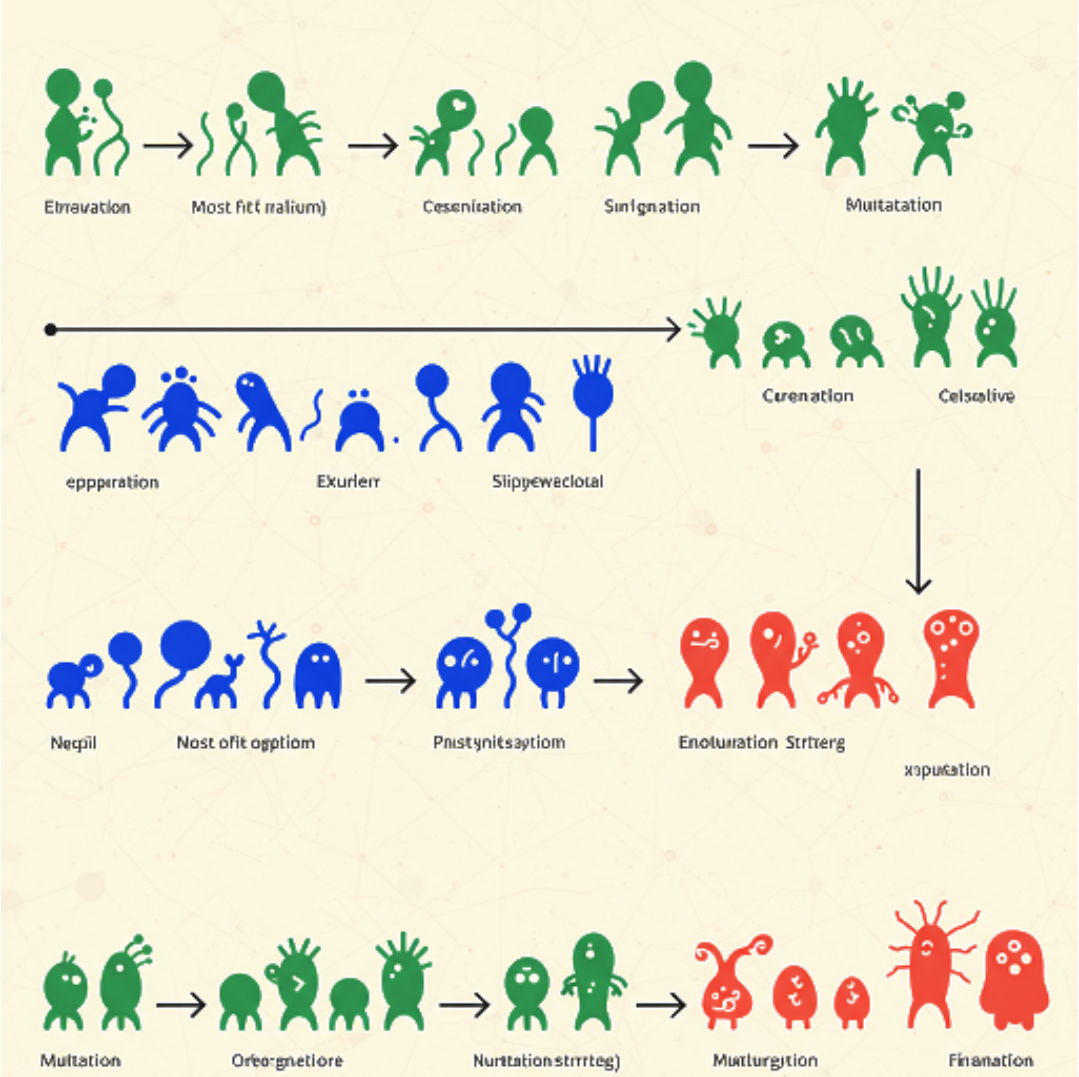


Fig. 15: Een andere (en wel zeer) associatieve impressie van μ/λ-evolutionaire strategie (Bron: Gemini 3).

12. De volgorde-statistiek (Eng.: *order statistics*) van de continue uniforme verdeling U op het eenheidsinterval $[0, 1]$ bezit de volgende eigenschap. Worden n elementen $X_1, \ldots, X_n$ getrokken uit $[0, 1]$, en daarna op volgorde $X_{(1)} < \cdots < X_{(n)}$ gezet, dan geldt[28]

$$E[X_{(k)}] = \frac{k}{n+1}.$$

Deze gelijkheid bewijzen we zelf verder niet [9].

[28] Als er wordt getrokken uit een verzameling reële getallen kan dit altijd beschouwd worden als trekken met teruglegging. Immers, zolang maar niet meer dan aftelbaar veel reële getallen worden getrokken, is de kans dat getallen dubbel worden getrokken gelijk aan nul.

(a) Gegeven $\mu = 25$ en $\lambda = 100$. De fitness van de huidige populatie is uniform verdeeld over $[0, 100]$. Bereken de verwachte gemiddelde fitness van individuen in de volgende generatie.

(b) Dezelfde vraag, met $\mu = 25$ en $\lambda = 200$.

(c) Dezelfde vraag, met $\mu = 25$ en $\lambda \geq 25$ onbepaald. Gebruik de formule

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

(d) Teken een grafiek van de formule, gevonden onder antwoord (12c), voor $\lambda \in \mathbb{N} \cap [25, 200]$. Verklaar de functie-waarde voor $\lambda = 25$. Leg uit waarom de grafiek monotoon stijgt. Leg uit waarom de grafiek limiet 100 bezit.

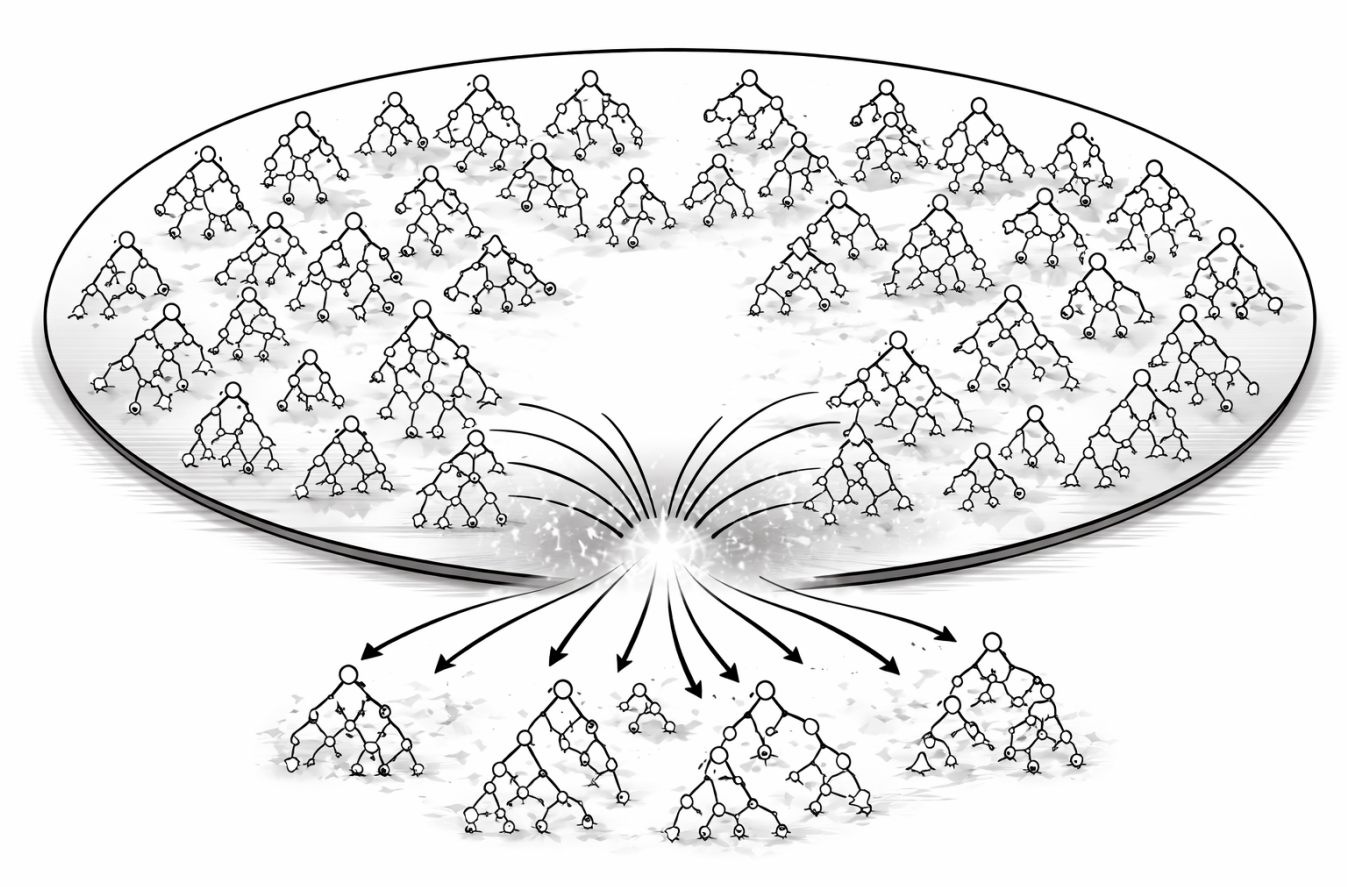
13. Gegeven is een populatie van genotypen met fitness respectievelijk 1, 1, 2, 3, 3, 3, 5, 5, 8, 8, 8, en 9. Er wordt 30 keer een exemplaar getrokken met teruglegging. De beste 12 exemplaren vormen de volgende generatie. Geef de verwachte gemiddelde fitness κ van de volgende generatie.

(a) Middels simulatie (ook wel: “Monte-Carlo simulatie”).

(b) Exact [12, Sec. 2.2., p. 23].

21 Genetisch programmeren

Genetisch programmeren (GP) is een evolutionair algoritme waarbij de populatie niet uit lineaire chromosomen bestaat, maar uit syntactische bomen van computerprogramma's. Mutatie en recombinatie worden dan toegepast op boomstructuren.



Artistieke impressie van genetisch programmeren. Wat hier niet helemaal klopt is dat syntactische bomen hier allemaal ongeveer even groot zijn. In werkelijkheid is er, door kruising en mutatie, een aanzienlijk verschil in grootte. (Afbeelding gegeneerd door Gemini 3.)

Omdat elk gegenereerd programma daadwerkelijk moet worden uitgevoerd om de fitness te bepalen, is de doorlooptijd afhankelijk van de executietijd van de gegenereerde computerprogramma's. Hierdoor is evolutie in genetisch programmeren tijdrovender dan evolutie bij genetische algoritmen.

1. Het boek van John Koza uit 1992 was een enorme doorbraak, en legde het fundament voor verder wetenschappelijk onderzoek in dit gebied [16].

   Vergelijk Koza's originele flow chart (Fig. 16 op pagina 118) met een modernere flow chart (Fig. 17 op pagina 119). Geef tenminste drie verschillen. Hint: je kun letten op het aantal runs, het detail van fitness-evaluatie, de variatie in genetische operaties, de kansen P_r, P_c, P_m, P_a, de specificatie van crossover output, en de afhandeling van terminatie.

2. (Bonusvraag.) Na zijn enorme succes in 1992 zat John Koza niet stil. Hij publiceerde nog drie dikke vervolgboeken (GP II tot en met GP IV), waarin hij steeds ingewikkelder problemen oploste, zoals het automatisch ontwerpen van elektrische circuits en zelfs antennes [17, 19, 20]. Ook bracht hij video's uit om te laten zien hoe de computerprogramma's stap voor stap "groeiden" [18].

   De impact van deze boeken is gigantisch. Koza's boeken zijn tienduizenden keren geciteerd, wat laat zien dat zijn werk de basis vormde voor een compleet nieuw vakgebied in de kunstmatige intelligentie. Zelfs nu, in het tijdperk van GenAI, worden zijn methodes nog steeds gebruikt om computers creatief te laten denken.

   Leg uit hoe de introductie van de zogenaamde *automatically defined functions* (ADFs) in Koza's tweede boek een oplossing bood voor de *curse of dimensionality* binnen genetisch programmeren (en zoek dus zo nodig eerst uit wat de curse of dimensionality is). Leg ook uit waarom ADFs essentieel zijn voor het evolueren van complexe hiërarchische structuren in vergelijking met de standaard boomstructuren uit zijn eerste werk.

3. (LISP.) Bekijk de History of programming languages poster van O'Reilly (lokale kopie). Welke talen zijn zo'n beetje gelijktijdig met LISP ontstaan? Geef van elke taal de expansie van het acroniem. Waarin onderscheidt LISP zicht het meest? Wie wordt beschouwd als de grondlegger, misschien zelfs de uitvinder, van LISP? Is het LISP of LISP? Wat was de bedoeling van deze taal, i.e., met wel doel werd LISP ontworpen? Welke romantische quote over LISP kan worden toegeschreven aan de invloedrijke Nederlandse informaticus Edsger Dijkstra? Wordt het nu nog vaak gebruikt?

4. (LISP.)

   (a) Is `ATOOM` een atoom? Is `T` een atoom? Is `(T)` een atoom? Is `117` een atoom? Is `NIL` een atoom? Is `()` een atoom?

   (b) Is `(ATOOM)` een lijst? Is `()` een lijst? Is `(EEN TWEE) DRIE` een lijst? Is `(NIL)` een lijst? Is `(())` een lijst? Is `NIL` een lijst?

   (c) Is `()` een s-expressie? Is `(() () () ())` een s-expressie? Is `(A B C)` een s-expressie? Is `(A, B, C)` een s-expressie? Is `(A (A (A B C) C) C)` een s-expressie? Is `,` een s-expressie?

   (d) Herschrijf de volgende expressies als s-expressie: $x - 5$, $x + (y + z)$, $x + y + z$, $\sqrt{x - 5}$, $\sqrt{x - 5}/4$, $\sqrt{x/4 - 5}$, $\sqrt{(4 + (x - 5))}/(y - (-w))$

   (e) Herschrijf de volgende drie programma's als s-expressie:

   ```
   IF FOOD THEN MOVE ELSE TURN.
   IF FOOD THEN { IF TASTY THEN EAT ELSE TURN } ELSE TURN.
   ```

   ```
   IF FOOD THEN {
      IF TASTY THEN {
         IF SCARCE THEN SAVE ELSE EAT
      }
      ELSE TURN
   }
   ELSE TURN.
   ```

   (f) Bepaal van de volgende strings of het s-expressies zijn. Zo ja, evalueer ze. Hou er rekening mee dat de evaluatie van een geldige s-expressie een zg. run-time error kan opleveren.

   De string `NIL`, de string `T`, de string `112`, de string `(112)`, de string `(* 5 (- 2 3))`, de string `(+ (- 9 5) (- 2 3))`, de string `(-2 3)`, de string `(* 5)`, de string `(* 2 3 4)`.

5. We werken met Common LISP. Geef syntax-bomen voor de volgende s-expressies:

   (a) `(+ (- (* 4 (- 2) 7) 3)(+ 6 1))`

   (b) `(+ (rand) (rand))`, waarbij "rand" een nul-plaatsige functie is.

   (c) `(and (or (and p (not r) q) p)(or r p))`

   (d) `(if (food-ahead) (left) (if (food-ahead) (right) (left)))`, waarbij "food-ahead," "left," en "right" nul-plaatsige functies zijn.

6. We werken met Common LISP. Evalueer de volgende s-expressies. Bij een foutmelding volstaat het "Error." op te schrijven.[29]

   (a) `'(+ 1 (* 2 3))`

   (b) `(+ 1 (* 2 3))`

   (c) `'(+ 1 '(* 2 3))`

   (d) `(+ 1 '(* 2 3))`

7. (Homogeniteit vs. diversiteit.) Bespreek de verschillen tussen

   `(IF (FOOD-AHEAD) (MOVE) (LEFT))` en `(IF-FOOD-AHEAD (MOVE) (LEFT))`.

   Geef i.h.b. de typen van alle zeven lijsten. Kies uit typen "numeriek," "test," en "actie".

8. (Recombinatie.)

   (a) Hoeveel kruisingen zijn er mogelijk tussen

   ```
   (if (< x 3) (if (< y 5) x 7) z)
   ```

   en zichzelf? Top-expressies mogen ook kruisen, en er mag ook op dezelfde plek gekruist worden. Verder mag worden aangenomen dat alle niet-Boolse termen hetzelfde type hebben.

   (b) Geef het aantal mogelijke kruisingen van

   ```
   (IF (NOT (wall-ahead)) (move) (turn)) en
   (IF (NOT (food-ahead)) (right) (eat)).
   ```

   waarbij alleen op lijst-niveau gekruist mag worden. (Dus bijvoorbeeld identifiers als `move` en `right` mogen niet worden gekruist.) Top-expressies mogen wel kruisen.

   (c) Geef het aantal mogelijke kruisingen van

   ```
   (if (food-ahead)
      (eat)
      (if (< (rand) 0.5)
         (left)
         (right)))
   ```

   en

   ```
   (while (not food-ahead) (move)).
   ```

[29] Gebruik zo nodig een LISP interpreter. (Google: online LISP interpreter.) Veel online LISP interpreters "evalueren stil", wat wil zeggen dat ze op top-niveau niet de evaluatie van ingevoerde expressies laten zien. Eigenlijk is dat niet "des LISP", je hoort altijd de evaluatie van elke ingevoerde expressie terug te krijgen. Om "stille" interpreters de evaluatie van een top-expressies te laten tonen moet er dan nog een expliciete `(PRINT ..)` instructie omheen worden gezet.

waarbij alleen op lijst-niveau gekruist mag worden.

(d) Geef het aantal mogelijke kruisingen van

```
(IF (FOOD-AHEAD) (RIGHT) (LEFT)) en
(IF (NOT (FOOD-AHEAD)) (RIGHT) (MOVE))
```

waarbij alleen op lijst-niveau mag worden gekruist. (Dus bijvoorbeeld de identifiers `LEFT` en `RIGHT` mogen niet worden gekruist.)

(e) De s-expressie E is gelijk aan `(/ (SQRT (+ 4 (- X 5))) (- Y (- W)))`. Hoeveel kruisingen van E met zichzelf zijn er mogelijk zonder E weer terug te krijgen?

9. Gegeven is het LISP-programma

```
(if (test-1)
    (action-1)
    (if (test-2)
        (if (test-3)
            (action-3)
            (action-1)
        )
        (if (test-1)
            (action-1)
        )
    )
)
```

(a) Teken een syntactische boom van dit programma. Uit hoeveel knopen bestaat de syntactische boom? Hoeveel bladeren bezit je syntactische boom?

(b) Geef het aantal mogelijke kruisingen van dit programma met zichzelf. Kruisingen op dezelfde knoop zijn toegestaan.

(c) Wat is de omvang van een grootst mogelijke kruising in (9b)? (I.e., een kruising welke een zo groot mogelijk kind oplevert.) Hoeveel van dergelijke kruisingen zijn er?

10. (Recombinatie.)

(a) De instructie `PROGN` zorgt er voor dat alle elementen sequentieel (in volgorde) worden uitgevoerd. Evaluatie van bijvoorbeeld `(PROGN (PRINT "Hallo") (PRINT "Wereld"))` laat LISP eerst `"Hallo"` en dan `"Wereld"` afdrukken. Geef het aantal mogelijke kruisingen van

```
(PROGN (PICK-UP)
 (IF-CARRYING-FOOD
  (PROGN
   (MOVE-TO-ADJACENT-PHEROMONE-ELSE
    (MOVE-TO-ADJACENT-FOOD-ELSE
     (MOVE-TO-ADJACENT-FOOD-ELSE
      (MOVE-TO-ADJACENT-FOOD-ELSE (PICK-UP)))))
   (PROGN
    (PROGN
     (PROGN (PROGN (MOVE-TO-ADJACENT-FOOD-ELSE (PICK-UP)) (PICK-UP))
```

```
     (PROGN (MOVE-TONEST) (DROP-PHEROMONE)))
    (PICK-UP))
   (PROGN (MOVE-TO-NEST) (DROP-PHEROMONE))))
(MOVE-TO-ADJACENT-FOOD-ELSE
 (IF-CARRYING-FOOD
  (PROGN
   (PROGN (DROP-PHEROMONE)
    (MOVE-TO-ADJACENT-PHEROMONE-ELSE
     (IF-CARRYING-FOOD
      (MOVE-TO-ADJACENT-FOOD-ELSE (PICK-UP))
      (MOVE-TO-ADJACENT-FOOD-ELSE (PICK-UP)))))
   (MOVE-TO-NEST))
  (IF-FOOD-HERE (PICK-UP)
   (IF-CARRYING-FOOD
    (PROGN
     (IF-FOOD-HERE
      (MOVE-RANDOM)
      (IF-CARRYING-FOOD (MOVE-RANDOM) (PICK-UP)))
     (DROP-PHEROMONE))
    (MOVE-TO-ADJACENT-PHEROMONE-ELSE (MOVE-RANDOM)))))))
```

met zichzelf. Geef aan hoe je aan je antwoord komt. (Hint: PROGN: 11 voorkomens; IF-*: 7; MOVE-*: 15; DROP-*: 5; PICK-UP: 9.)

(b) Wat is de omvang van een grootst mogelijke kruising in (10a)? (I.e., een kruising welke een zo groot mogelijk kind oplevert.) Hoeveel van dergelijke kruisingen zijn er?

(c) De diepte van een s-expressie wordt als volgt gedefinieerd: de diepte van een atoom is nul; de diepte van een lijst is het maximum van de diepte van de lijst-elementen + 1. Geef de diepte van de volgende s-expressies: `ATOOM`, `(A B)`, `(A (A B))`, `(A ((A B) B))`, `NIL`, `()`, `(NIL)`, `(())`, en `((()))`.

(d) Welke kruising(en) in (10a) levert kinderen met een maximale diepte? (Hint: de diepte van bovenstaand programma is 10.)

11. Je wilt met GP een logische formule vinden die een noodzakelijke voorwaarde uitdrukt voor het accepteren van een hypotheek. De terminal-set ligt vast en ziet er ongeveer als volgt uit:

$$\{ \text{is-BKR-geregisteerd, heeft-vast-inkomen, is-gehuwd, } \ldots \}$$

In de ontwikkeling van een GP heb je de keuze tussen twee functie-sets:

(a) $\{\neg, \wedge\}$

(b) $\{\neg, \wedge, \rightarrow, \vee, \equiv\}$

Welke van de twee ga je gebruiken en waarom? Neem in overweging dat de logische operatoren $\neg$ en $\wedge$ volstaan om elke propositie-logische bewering uit te drukken. (Ga dit laatste zo nodig na.)

12. Het is mogelijk de individuen van een populatie qua computergeheugen wat economischer bij te houden door meervoudige exemplaren te laten vertegenwoordigen één representant, gevolgd door een getal, genaamd de multipliciteit. De multipliciteit geeft aan hoe vaak een exemplaar voorkomt in een populatie. Bespreek kort de voors en tegens van een dergelijke aanpak.

13. (GP met behulp van grammatica's.)

    (a) Een *codon* is een stukje chromosoom dat wezenlijke informatie in zich draagt. Stel we willen bit-strings gebruiken als chromosomen, en stel dat we van een chromosoom verwachten dat deze een rijtje letters kan representeren. Hoe lang moet een codon dan minimaal zijn om dit aan te kunnen?

    (b) Geef een mapping van codons naar letters.

    (c) Codeer met de het in (13b) gegeven antwoord de string "dabbaac".

    (d) Gegeven is de volgende grammatica.

```
  <prog> ::= <prog> ; <action> | <action>
<action> ::= if <test> <actie> <else> | <atomic>
  <else> ::= | else <action>
  <test> ::= food-here? | bug-here? | <expr> <comp> <expr>
<atomic> ::= left | right | move
  <expr> ::= <expr> <op> <expr> | <var> | <num>
  <comp> ::= eq | lt | gt
    <op> ::= + | - | * | /
  <var>  ::= x | y | z
  <num>  ::= 0.5 | 1 | 5 | 10 | 50 | 100 | 500 | 1000
```

    (e) Geef omschrijvingen van de noties "token," "terminal," en "non-terminal".

    (f) Hoeveel verschillende terminals bevat deze grammatica?

    (g) Hoeveel verschillende non-terminals?

    (h) Nummer alternatieven bij herschrijfregels, oplopend vanaf 0. Welke herschrijfregel bevat de meeste alternatieven?

    (i) Geef de programmacode welke hoort bij de volgende strings: "12734351516464"; "12111114411111"; "35666666666666"; "010101".

    (j) Geef de cijferstrings die horen bij de volgende programma's:

    - `move`
    - `if x lt 5 then left`
    - `if x lt 5 then left else right`
    - `if x lt 5 then left else right; move`

    (k) Geef alternatieve strings voor voorgaande programma's. Hoeveel keuzemogelijkheden zijn er voor elk programma als een string lengte 100 heeft, en cijfers kan bevatten die lopen van 0 tot en met 7?

    (l) Hoe lang moet een codon minimaal zijn om alle herschrijvingen in deze grammatica aan te kunnen?

    (m) Welk probleem ontstaat er bij te korte codons?

    (n) Welke expressie levert de string "10514" op? Is dit een executeerbaar programma?

    (o) Om een executeerbaar programma te generen uit een grammatica is het vaak toegestaan *maxwrap* keer van voor af aan de cijferstring te beginnen, waarbij *maxwrap* een constant natuurlijk getal is.

    Welke expressie levert de string "10514" op met $maxwrap = 2$?

(p) Is er een cijferstring met cijfers die lopen van 0 tot en met 7 die meerdere wraps nodig heeft om een executeerbaar programma te genereren? Zo ja, geef een voorbeeld. Zo nee, beargumenteer waarom zo'n string niet bestaat.

14. Fig. 16 en Fig. 17 presenteren twee stroomdiagrammen die de handelingen en hun volgorde in GP weergeven. Het eerste diagram is afkomstig uit Koza's "Genetic Programming: on the Programming of Computers by Means of Natural Selection"; het tweede diagram is afkomstig van `http://www.genetic-programming.com/gpflowchart.html`. Deze site vermeldt voorzover ik kan nagaan verder niet wie er verantwoordelijk is, of wil zijn, voor het diagram.

    (a) Beschrijf drie wezenlijke verschillen tussen beide diagrammen.

    (b) Bespreek vervolgens elk van deze verschillen. Geef bij elk verschil aan welke benadering je het meest aanspreekt: die van Koza of die van `genetic-programming.com`.

15. We willen met genetisch programmeren een LISP programma produceren dat pacman succesvol door een doolhof met monsters laat manoeuvreren. Koza heeft in 1992 ook zoiets gedaan, deze oplossing gebruiken we *niet*.

    (a) Definieer een fitness maat.

    (b) Definieer LISP functies en -terminals geschikt voor het genetisch programmeren van pacman.

    Je mag (alleen) het volgende aannemen. Pacman kan links / rechts / op / neer. Als pacman een onmogelijke beweging wil maken, omdat er bijvoorbeeld een muur staat, gebeurd er niets. Pacman kan dots / ghosts in straal vier zien en weet tegelijkertijd of dit boven / beneden / links / rechts is. Pacman weet of spoken blauw zijn. Er is geen fruit.

    (c) Geef een voorbeeld van een programma dat voldoet aan (15b). Laat het bij voorkeur iets zinvols doen.

    (d) Koza (1992) gebruikt primitieven (atomaire instructies) die een vrij hoog niveau hebben, bijvoorbeeld "advance to nearest uneaten pill" en "retreat from nearest ghost".

    Bespreek het gebruik van "intelligentere" primitieven. Beargumenteer waarom deze succesvoller (of minder succesvol) zijn dan het gebruik van elementairder primitieven zoals voorgesteld in (15b).

16. We willen met genetisch programmeren emergent gedrag laten ontstaan in een mierenkolonie die op zoek is naar voedsel.

    (a) Verklaar de woorden "emergent" en "feromoon".

    (b) Geef een grammatica in Backus-Naur vorm, om o.a. het volgende programma te kunnen genereren

```
ifelse food-here? [ eat ] [
   if not-pheromone-here? [ drop-pheromone fd 1 rt 20 ]
   if turtle-in-my-sight? [ set haste 2 ]
   face closest-food
   face closest-turtle
   rt 180
   fd haste * 2
]
```

Er gelden de volgende aanvullende beperkingen. Forward-expressies (`fd`) in het interval $[-5 * \text{haste}, 5 * \text{haste}]$ moeten liggen, en dat argumenten van `rt` in ieder geval integers zijn in het interval $[-180, 180]$. Er wordt *niet* gevraagd dat expressies echte *alle* waarden in een interval kunnen aannemen. Enkele mogelijke waarden volstaan in je grammatica. Acties zoals `turn-around` en `face-closest-food` en tests zoals `food-here?` en `turtle-in-my-sight?` zijn atomair.

(c) Geef een minimale lengte voor binaire codons om willekeurige programma's in je grammatica te kunnen genereren. Verklaar je antwoord.

(d) Geef een voorbeeld van een bitstring die met alle waarden van *maxwrap* volgens je gegeven grammatica geen programma oplevert. (Of beargumenteer waarom zo'n bitstring niet bestaat.)

17. (LISP.)

(a) Herschrijf de volgende twee expressies als s-expressie: $-\sqrt{12 - x}$ en $\sqrt{\sqrt{-4}}$.

(b) Herschrijf de volgende twee programma's als s-expressie. Zorg voor een nette en consistente indentatie.

Programma 1:

```
IF FOOD THEN TURN ELSE MOVE.
```

Programma 2:

```
IF FOOD THEN {
   IF TASTY THEN EAT
   ELSE { IF SCARCE THEN SAVE ELSE EAT }
}
ELSE { IF SCARCE THEN EAT ELSE TURN }.
```

(c) Bepaal van de volgende strings of het s-expressies zijn. Zo ja, evalueer ze. Hou er rekening mee dat de evaluatie van een geldige s-expressie een run-time error kan opleveren.

De string `889`, de string `(889)`, de string `(- (4 5) 1)`, de string `(+ (- 9 5) (- 3))`, de string `(-7 3)`, de string `(* 5)`, de string `(* 2 3 4)`.

18. (Homogeniteit en kruising in genetisch programmeren.)

(a) Bespreek de verschillen tussen

```
(WHILE (FOOD-AHEAD) (EAT))
```

en

```
(WHILE-FOOD-AHEAD (EAT)).
```

Bespreek ook de zin (het doel) van de laatste constructie en de werking ervan. (Je antwoord beslaat ong. 1-2 regels.)

(b) Geef het aantal mogelijke kruisingen van `(WHILE (FOOD-AHEAD) (EAT))` met

```
(IF (NOT (FOOD-AHEAD)) (RIGHT) (WHILE (CAN-EAT) (EAT))).
```

(c) Geef het aantal mogelijk kruisingen als de twee stukjes LISP uit onderdeel 18b ook nog eens een keer met zichzelf kunnen kruisen. (Mogelijkheden waarbij op één genotype twee keer hetzelfde kruispunt wordt geselecteerd, tellen ook.)

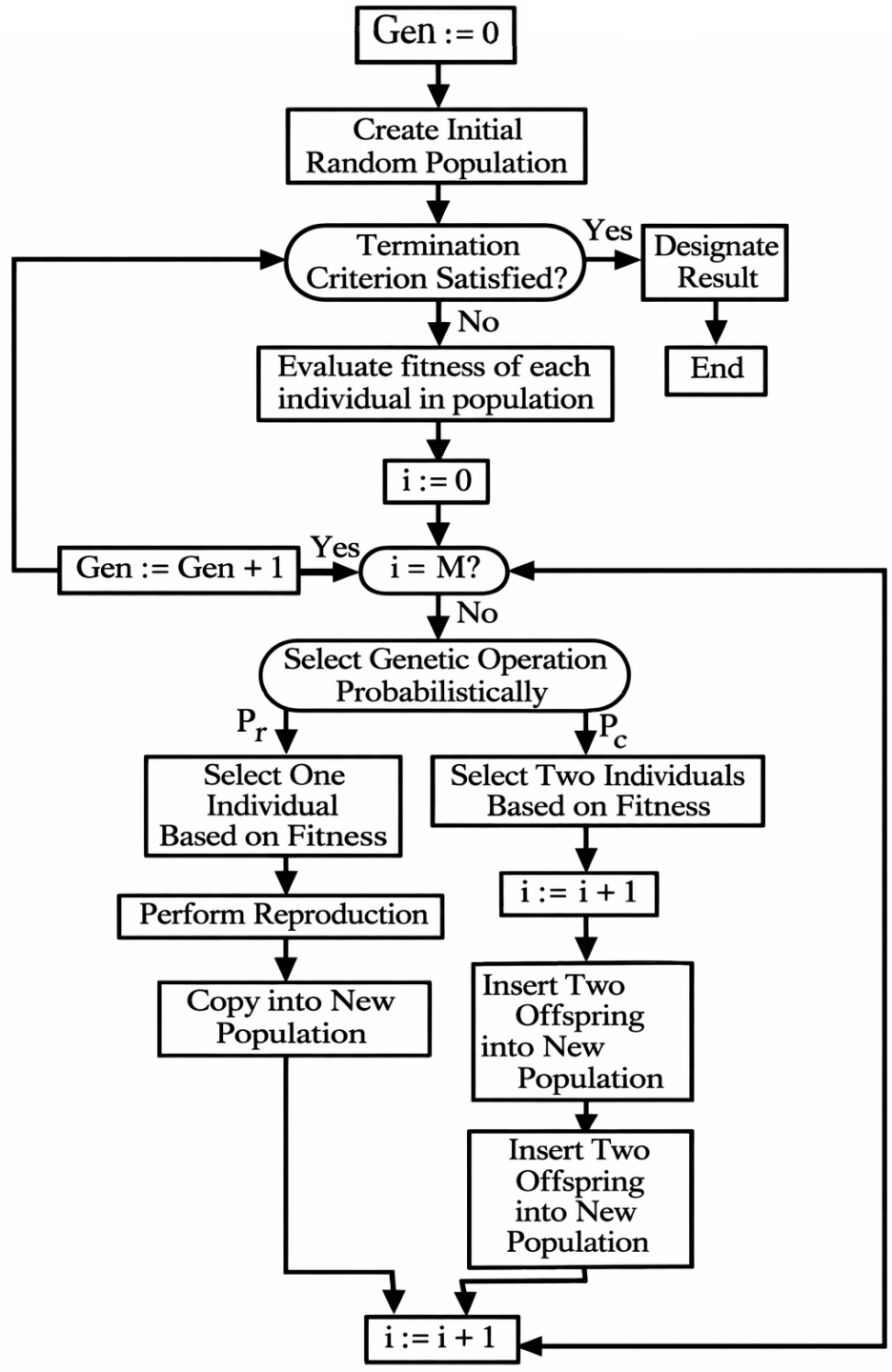


Fig. 16: De originele GP flowchart (Koza, 1992, scherper gemaakt met Gemini 3)

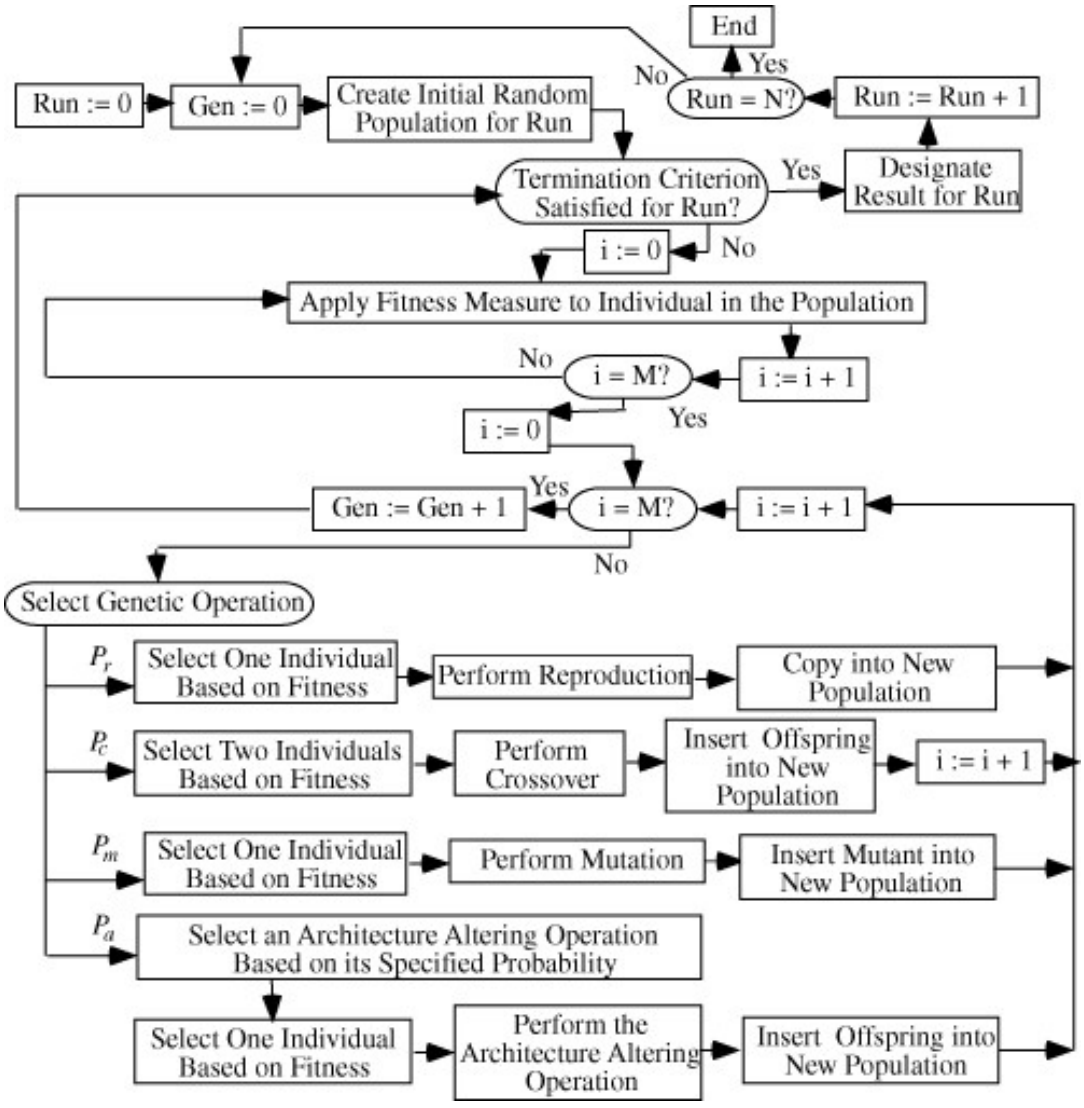
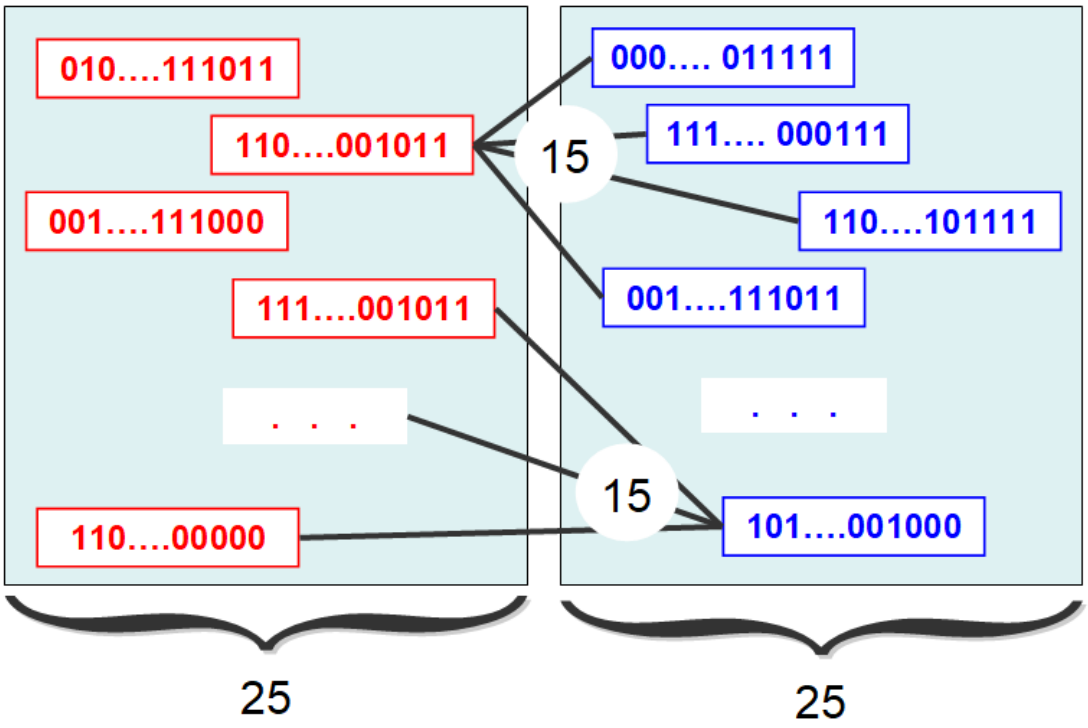


Fig. 17: GP flowchart (genetic-programming.com, 2026). Deze verschilt van Koza's originele flow chart.

22 Co-Evolutie

Co-evolutie is het proces waarbij twee soorten elkaars evolutie voortdurend beïnvloeden. Verandert de één? Dan moet de ander zich ook aanpassen om te overleven. De meerwaarde van co-evolutie ligt in de constante biologische wapenwedloop. Omdat organismen zich aanpassen aan een veranderende tegenstander, blijven ze zichzelf verbeteren. Het is een “gezamenlijke biologische dans”.

In een baanbrekend artikel “”Co-evolutionary dynamics in a minimal substrate”, tonen Watson en Pollack aan dat co-evolutie voorkomt dat een evolutionair proces vastloopt in middelmatigheid. Zonder deze dynamiek zou de evolutie stoppen bij een “goed genoeg” resultaat. Door voortdurende competitie worden grenzen verlegd, wat leidt tot steeds betere oplossingen.



De experimentele opzet van Watson & Pollack. Twee populaties, met verschillende competenties. (Denk voor het gemak aan jagers en prooien.) Rood concurreert niet met elkaar; blauw ook niet. Individuen (fenotypen) worden gekarakteriseerd door een vast aantal bit strings ter lengte 100 (genotypen). Elke bit string representeert een vast aspect van het genotype van een individu, bijvoorbeeld lengte, gewicht, of visie-radius. Bit strings bestaande uit uitsluitend nullen representeren dan slechtst mogelijke oplossingen; bit strings bestaande uit uitsluitend enen representeren dan optimale oplossingen. In dit figuur is zo’n bundel precies één bit string groot (anders werd de figuur te gecompliceerd). In één ronde vergelijkt elk individu zichzelf herhaaldelijk met een willekeurig groepje van 15 individuen uit de andere populatie.

1. Op het hoor- en werkcollege over co-evolutie wordt een belangrijke publicatie van Watson en Pollack over co-evolutie [32] behandeld. Het is belangrijk deze publicatie goed te bestuderen. Om je daarbij te helpen zijn er vragen opgesteld. Als je deze vragen beantwoord ben je in ieder geval een keer serieus door het artikel heen gegaan.

   1. INTRODUCTION

   - Wat is volgens de auteurs het basis-idee achter co-evolutie?[30]
   - Waarom is volgens de auteurs co-evolutionair leren in veel machinaal leerdomeinen het enige alternatief?
   - De auteurs noemen drie voordelen van co-evolutionair leren. Welke zijn dat?
   - De auteurs noemen drie problemen bij co-evolutionair leren. Welke zijn dat?
   - Aan het eind zeggen de auteurs voor een bepaalde aanpak te kiezen. Welke aanpak is dat, en waarom wordt voor die aanpak gekozen?

   2. A MINIMAL SUBSTRATE

   2.1. SCALARS

   - Waarom kiezen de auteurs er voor om getallen te nemen als te evolueren objecten?
   - Hoe werkt de score-functie?
   - Wat is bijvoorbeeld de score van a tegen S als $a = 4$ en $S = \{2, 4, 6, 8\}$?

   2.2. OBJECTIVE FITNESS, AND SUBJECTIVE FITNESS

   - Wat is objectieve fitness?
   - Wat is subjectieve fitness?
   - Geef een voorbeeld van een geval waarin bij een individu met een hoge absolute fitness toch een lage relatieve fitness scoort.

   2.3. MULTIPLE DIMENSIONS

   - Wat is een scalar? Wat is een vector?
   - Waarom is het zinvol individuen als 2-dimensionale vectoren (ook wel: geordende paren) te representeren?
   - Hoe is $score_2(\ ,\ )$ gedefinieerd? (Hint: boven eq. 2 staat "One way to do this is to choose that dimension in which the two players are ..."). Bereken $score_2((2, 7), (5, 6))$.
   - De definitie van $f_2(a, S)$ hoeft verder niet te worden beschreven. Dit wordt op college behandeld.

   2.4. INTRANSITIVE SUPERIORITY

   - Wat betekent "transitiviteit" in de wiskunde?
   - Is de relatie "$\leq$" transitief? De relatie "$=$"? De relatie "is vader van"?
   - Wat verstaan de auteurs onder "intransitiviteit"?
   - Geef een voorbeeld van een spel tussen twee person waarbij winst intransitief is. (Hint: 2e alinea.)

[30] Een *metriek* is een manier om afstanden te meten. De afstand tussen twee woorden kan bijvoorbeeld worden gemeten als de minimale hoeveelheid bewerkingen die nodig is om het ene woord in het andere te veranderen, waarbij de mogelijke bewerkingen zijn: verwijderen van een teken; invoegen van een teken; vervanging van een teken door een ander teken. (Dit is de zg. *Levenshtein afstand*.)

- Waarom is (volgens de auteurs) intransitiviteit interessant in co-evolutie?
- Hoe is $score_2(\ ,\ )$ gedefinieerd? (Hint: boven eq. 3 staat "That is, when two players, (a_x, a_y) and (b_x, b_y), enter a game ..."). Bereken $score_3((2,7),(5,6))$.
- De definitie van $f_3(a, S)$ hoeft verder niet te worden beschreven. Dit wordt op college behandeld.

3. EXPERIMENTAL SET-UP

- Welke keuzes moeten worden gemaakt in de opzet van experimenten 1 tot en met 3?
- Welke keuzes zijn gemaakt in de opzet van experimenten 1 tot en met 3?

3.1. MUTATION BIASES

- Wat is een bit string?
- Wat is een unaire representatie?
- Hoe kan een geheel getal tussen 0 en 100 (incl. 0 en 100) als bit string ter lengte 100 worden gerepresenteerd?
- Wat verstaan de auteurs onder een negatieve mutatie-bias?

4. EXPERIMENTS AND RESULTS

- De drie experimenten hoeven verder niet te worden beschreven. Deze worden op het college behandeld.

5. CONCLUSIONS

- Wat bedoelen de auteurs met "mediocre stable state"?
- In welk opzicht heeft het paper bijgedragen aan het begrijpen van de mechanismen achter het ontstaan van zg. mediocre stable states?
- In welk opzicht vinden de auteurs hun benadering het meest geslaagd?

2. Leg uit waarom de fitness van een speerwerper (of hardloper, of zwemmer) goed kan worden gerepresenteerd als een reëel getal, maar de fitness van een schermer (of judoka, of tennisser) niet.

– **Normeren**: druk elk getal uit als een proportie:

$$x_i' =_{Def} \frac{x_i}{\sum_{j=1}^{n} x_j}$$

– **Proportioneel**: druk een telling x_i uit als een proportie:

$$x_i' =_{Def} \frac{x_i}{y_i},$$

waarbij y_i het totaal aantal waarnemingen is.

Tab. 4: Normeren en relatieve frequentie.

- **Absoluut**: Absolute prestatie-maat.
- **Genormeerd absoluut**: Absolute fitness, geschaald naar breuken die optellen tot 1.
- **Relatief**: Bijvoorbeeld: 3 successen uit 5 interacties.
- **Proportioneel relatief**: Op basis van het zojuist gegeven voorbeeld: 3/5.
- **Genormeerd relatief**: Proportioneel relatieve fitness, geschaald naar breuken die optellen tot 1.

Tab. 5: Verschillende noties van fitness.

3. In deze opgave bekijken we twee populaties, A en B, die betrokken zijn in een proces van co-evolutie. Deze populaties bestaan elk uit 10 individuen. We veronderstellen verder dat deze 20 individuen op enig moment de volgende (absolute) fitness bezitten:

Populatie A :	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
Absolute fitness :	3	3	8	1	1	1	3	5	6	3
Populatie B :	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
Absolute fitness :	3	5	2	7	2	1	6	9	4	4

Het doel van deze opgave is inzicht te krijgen in het proces van competitie, fitness-bepaling en selectie. Eerst op basis van absolute fitness, dan op basis van relatieve fitness.

Om dit te kunnen doen is het nodig dat je om kunt gaan met noties als (absolute) fitness, genormeerde (absolute) fitness, relatieve fitness, proportioneel relatieve fitness, genormeerde relatieve fitness en fitness-proportionele selectie, en dat ga je leren in deze opgave. Zie ook Tabel 5.

(a) De *genormeerde absolute fitness* (ook wel simpelweg: genormeerde fitness) van een individu is gedefinieerd als de absolute fitness van dat individu, gedeeld door de totale absolute fitness.

De genormeerde fitness telt op tot 1, en kan worden gebruikt als kans-indicator voor bijvoorbeeld fitness-proportionele selectie. Dus als de genormeerde fitness van een individu gelijk is aan f, dan is de kans dat dit individu fitness-proportioneel wordt geselecteerd voor voortplanting, ook gelijk aan f.[31]

Geef de genormeerde fitness voor alle individuen in A. Je antwoord mag bestaan uit tien onvereenvoudigde breuken.

(b) Vaak is het zo dat de absolute fitness van individuen onbekend is. Een vergelijking kan worden gemaakt met sport. Er zijn sportonderdelen waarbij deelnemers een absolute score kunnen halen. Denk aan speerwerpen, 1km tijdrit bij wielrennen, of 400m hardlopen bij atletiek. Absolute scores zijn daar mogelijk, bijvoorbeeld 93 meter, 1:04 minuten, en 47.8 seconden. In andere sportonderdelen kan naar de aard van de sport geen absolute score worden behaald. Denk aan tennis, judo of boksen. Voor deze sporten moeten deelnemers in rondes tegen elkaar uitkomen om te bepalen wie de sterkste is.

Terug naar de theorie. Als indicator voor absolute fitness kan de relatieve fitness worden gebruikt. De *relatieve fitness* van een individu a ten opzichte van een groepje S wordt

[31] Voor proportionele fitness, zie Opgave 1 uit het hoofdstuk over evolutionaire algoritmen, op blz. 102.

bepaald door de formule

$$f(a,S) =_{Def} \sum_{s\in S} \text{score}(a,s), \text{ waarbij } \quad \text{score}(a,s) = \begin{cases} 1 & \text{als } f(a) > f(s), \\ 0 & \text{anders.} \end{cases}$$

De relatieve fitness van een individu a ten opzichte van een groepje S is dus het aantal leden in S dat strict gedomineerd wordt door a.
Bereken:

i. $f(a_8, \{b_3\})$
ii. $f(a_8, \{b_7\})$
iii. $f(a_8, \{b_9\})$
iv. $f(a_8, \{b_{10}\})$
v. $f(a_8, \{b_2\})$
vi. $f(a_8, \{b_2, b_3, b_7, b_9, b_{10}\})$
vii. $f(a_8, \{b_1, b_3, b_5, b_6, b_9, b_{10}\})$
viii. $f(a_8, \{b_2, b_4, b_7, b_8\})$
ix. $f(a_8, \{b_1, b_4, b_5, b_6, b_7\})$
x. $f(a_8, \emptyset)$
xi. $f(a_8, B)$
xii. $f(a_3, B)$

(c) Laat $S = \{b_5, b_9\}$. Bereken voor elk element van A de relatieve fitness.

(d) Deze opgave laat zien dat de relatieve fitness erg kan afhangen van de referentiegroep (de groep uitgekozen tegenstanders). Een ongelukkige selectie van tegenstanders kan er namelijk voor zorgen dat elementen uit A zich niet goed van elkaar kunnen onderscheiden. Illustreer dit met $S = \{b_6, b_8\}$ door voor elk element van A de relatieve fitness uit te rekenen.
Welke elementen in A onderscheiden zich niet meer van elkaar?

(e) De *proportioneel relatieve fitness* wordt gedefinieerd als

$$\hat{f}(a,S) =_{Def} \frac{f(a,S)}{|S|}$$

Bijvoorbeeld: a_7 scoort 1 uit 2 tegen $\{b_6, b_7\}$, en a_8 scoort 2 uit 4 tegen $\{b_6, b_7, b_8, b_9\}$. Element a_8 bezit dus een hogere relatieve fitness dan a_4. Ze bezitten echter *dezelfde* proportioneel relatieve fitness, namelijk 1/2.
Laat S weer gelijk zijn aan $\{b_5, b_9\}$. Geef voor elk element van A de proportioneel relatieve fitness.

(f) De *genormeerde relatieve fitness* wordt gedefinieerd als

$$\bar{f}(a,S) =_{Def} \frac{\hat{f}(a,S)}{\text{totale proportioneel relatieve fitness}}$$

Normeren betekent: terugbrengen tot grootheden die samen optellen tot 1. Laat S weer gelijk zijn aan $\{b_5, b_9\}$. Geef voor elk element van A de genormeerde relatieve fitness. Geldt

$$\bar{f}(a,S) =_{?} \frac{f(a,S)}{\text{totale relatieve fitness}} \; ?$$

4. Een verzameling individuen met een gelijke relatieve fitness wordt een *equivalentieklasse* genoemd.[32] Het *onderscheidend vermogen* van een deelverzameling $S \subseteq A$ is het aantal klassen waarin B door S wordt onderverdeeld. We gebruiken de populaties uit Som 3 als voorbeeld. Als

$$S = \{a_7, a_8, a_9\},$$

dan zorgt S ervoor dat elementen in B een relatieve fitness krijgen van resp. 0, 1, 0, 3, 0, 0, 2, 3, 1, en 1. Dat zijn vier verschillende getallen, dus B wordt onderverdeeld in vier klassen.

[32] De notie *equivalentieklasse* is een algemeen begrip en in het bijzonder niet afhankelijk van begrippen uit de co-evolutie.

(a) In welke vier klassen (deelverzamelingen) wordt B onderverdeeld? Uit welke elementen bestaan die klassen?

(b) We zeggen dat een verzamelingen klassen een andere klassenverzameling *verfijnt* als de nieuw gevormde klassen gelijk zijn aan of bevat zijn in de oorspronkelijke klassen. De klassenverzameling

$$\{\{b_1, b_3, b_5, b_6\}, \{b_2, b_9, b_{10}\}, \{b_7\}, \{b_4, b_8\}\},$$

bijvoorbeeld, is een verfijning van

$$\{\{b_1, b_3, b_5, b_6\}, \{b_2, b_9, b_{10}, b_7\}, \{b_4, b_8\}\}.$$

Stel nu dat S' groter is dan S. Dus $S \subset S' \subseteq A$. Bewijs dat de equivalentieklassen voortgebracht door S' de equivalentieklassen voortgebracht door S verfijnen.

5. Nog steeds de populaties A en B als in Opgave 3.

(a) Geef de verwachte relatieve fitness van a_8 bij steekproef-grootte $|S| = k$, waarbij $S \subseteq B$.

(b) Geef de verwachte proportioneel relatieve fitness van a_8 bij steekproef-grootte $|S| = k$.

(c) Hoe zal de variantie (spreiding) van relatieve fitness afhangen van de steekproef-grootte $|S|$? (Je hoeft de variantie niet uit te rekenen.)

6. Geef definities van de volgende begrippen: absolute fitness, genormeerde absolute fitness, relatieve fitness, proportioneel relatieve fitness.

7. Individuen zijn nu punten $(a_x^1, a_y^1), \ldots (a_x^n, a_y^n)$ in R^2. We kunnen deze individuen op verschillende manieren vergelijken. Twee daarvan zijn de volgende:

Max-d We schrijven $a < b$ als en slechts als individu b individu a domineert in de coördinaat (of de eigenschap, of het attribuut) waar ze het **meest** verschillen.

Min-d We schrijven $a < b$ als en slechts als individu b individu a domineert in de coördinaat (of de eigenschap, of het attribuut) waar ze het **minst** verschillen.

Laten we als individuen onszelf nemen met als attributen (i) ons geboortejaar en (ii) onze geboortemaand (uitgedrukt als getal).

(a) Bezit je een absolute fitness? Zo ja, wat is deze?

(b) Vergelijk de ranking van jezelf met je buurman of buurvrouw volgens *Max-d*. Daarna met twee van je buren.

(c) Vergelijk de ranking van jezelf met je buurman of buurvrouw volgens *Min-d*. Daarna met twee van je buren.

(d) Neem aan dat attributen (coördinaten, dimensies) staan voor de eigenschappen van een pokerspeler: tightness (zeer tight, speelt niets, tot extreme loose, speelt alles), aggressiveness (zeer agressief, gaat altijd door, tot zeer passief, bindt snel in). Je mag ook twee eigenschappen van een voetballer, schaker, of iets anders dat concurreert met gelijksoortige anderen nemen. Welk vergelijksmethodiek spreekt je het meest aan? *Min-d* of *Max-d*? Waarom?

(e) Van een ordening zou je redelijkerwijs mogen verwachten dat deze transitief is: $a < b < c$ impliceert $a < c$. Een ordening heet *intransitief* of *niet transitief* als er a, b en c zijn te vinden, zó dat $a < b < c$ maar $a \not< c$. Geef een tegenvoorbeeld waaruit blijkt dat *Min-d* niet transitief is.

Een ordening heet *sterk intransitief* als er a, b en c zijn te vinden, zó dat $a < b < c$ en $c < a$. Is *Min-d* sterk intransitief? Waaruit volgt dat?

8. In deze opgave leer je werken met de noties objectieve preferentie en subjectieve preferentie. Deze noties worden als volgt gedefinieerd:

$$P_{obj}(a, b) =_{Def} f(a) < f(b).$$

Laat $A' \subseteq A$ en $B' \subseteq B$.

$$P_{subj}^{B',\,A'}(a, b) =_{Def} f(a, B') < f(b, A') \tag{8}$$

Dus b wordt *subjectief geprefereerd* boven a als (en slechts als) b op groepje A' meer overwinningen behaalt dan a op groepje B'.[33]

(a) Nog steeds de populaties A en B als in Opgave 3. Bereken:

i. $P_{subj}^{\{b_2\},\{a_2\}}(a_1, b_1)$
ii. $P_{subj}^{\{b_2,b_4\},\{a_2,a_4\}}(a_1, b_1)$
iii. $P_{subj}^{\{b_2,b_4,b_6\},\{a_2,a_4,a_6\}}(a_1, b_1)$
iv. $P_{subj}^{\{b_2,b_4,b_6,b_8,b_{10}\},\{a_2,a_4,a_6,a_8,a_{10}\}}(a_1, b_1)$
v. $P_{subj}^{\{b_1,b_3,b_5,b_7,b_9\},\{a_2,a_4,a_6,a_8,a_{10}\}}(a_1, b_1)$
vi. $P_{subj}^{\{b_2,b_4,b_6,b_8,b_{10}\},\{a_1,a_3,a_5,a_7,a_9\}}(a_1, b_1)$
vii. $P_{subj}^{\emptyset,\{a_2\}}(a_1, b_1)$
viii. $P_{subj}^{\emptyset,\emptyset}(a_1, b_1)$
ix. $P_{subj}^{B,\emptyset}(a_1, b_1)$
x. $P_{subj}^{B,A}(a_1, b_1)$
xi. Als (8(a)i-8(a)x), maar dan met a_4 en b_4.
xii. Als (8(a)i-8(a)x), maar dan met a_7 en b_7.

(b) Geef nu zelf definities van de volgende begrippen: objectieve preferentie, subjectieve preferentie. (Hint: absoluut vs. relatief.)

(c) Vind individuen a, b en groepjes A' en B' zó dat

$$P_{subj}^{B',\,A'}(a, b) \neq P_{obj}(a, b).$$

(d) Probeer te omschrijven in welke omstandigheden (8c) zich precies voordoet.

(e) Omschrijf in welke omstandigheden $P_{subj}^{B',\,A'}(a, b) = P_{obj}(a, b)$.

9. Een natuurlijk getal n in $[0, 100]$ kan (wat tegennatuurlijk en omslachtig) worden gerepresenteerd als een bit string ter lengte 100 met precies n enen, en de rest nullen,

(a) Hoeveel bit strings ter lengte 100 bestaan er? Hoeveel is dat als 10-macht?

(b) Hoeveel representaties van het getal 0 bestaan er? Hoeveel van het getal 1? Hoeveel van het getal 50? Hoeveel van het getal n?

[33] Een wellicht aantrekkelijker notatie zou de volgende kunnen zijn: $P(a, b)$ voor objectieve preferentie, en $P|_{B',\,A'}(a, b)$ ["P beperkt tot B', A'"] voor subjectieve preferentie. Echter, we kiezen ervoor dezelfde notatie als Watson en Pollack te hanteren.

(c) Het genotype van een individu wordt gerepresenteerd door een 100-bit string. De absolute (en voor ons of de applicatie onzichtbare fitness) van een individu wordt gepresenteerd door het aantal enen.

Neem aan dat B gevuld is met willekeurige bit strings ter lengte 100. Wat is de kans om een bit string in B aan te treffen met fitness n?

(d) We krijgen een willekeurige bit string ter lengte 100 aangeboden. Wat is de kans dat de fitness van deze string in $[45, 55]$ ligt?

(e) Hoe groot is de kans dat een individu met fitness k door mutatie zwakker wordt?

(f) Stel dat $a \in A$ met absolute fitness f, en stel dat a met een groepje van 5 uit B vergeleken wordt. Neem aan dat B gevuld is met willekeurige bit strings ter lengte 100. Bereken de kans dat a een relatieve fitness van k scoort.

10. In [32] wordt in Sec. 2.4 in Eq. 3 de *score3*-ordening geïntroduceerd. In het college werd hiernaar verwezen als de Min-d ordening. De *score3*-ordening (de Min-d ordening) is zoals bekend intransitief.

    Bespreek of de *score3*-ordening reëel of instrumenteel is, dat wil zeggen, bespreek of volgens jou de *score3*-ordening in de realiteit voor zou kunnen komen (geef in dat geval een voorbeeld), of dat deze slechts een kunstmatig product is met als enig doel een intransitieve ordening te produceren (geef in dat geval een onderbouwing). (Antwoord in 5-15 regels.)

11. Deze vraag grijpt weer terug naar het artikel "Co-evolutionaire dynamiek in een minimaal substraat" [32].

    (a) In [32] wordt er voor gekozen om genotypen te representeren als bit strings ter lengte 100. Bereken de kans dat een genotype met m enen, waarbij $0 \leq m \leq 100$, na mutatie verbeterd.

    (b) Wat is negative mutation bias?

    (c) R.A. Fisher, auteur van het invloedrijke "The genetical theory of natural selection," (1930), heeft argumenten aangedragen die aannemelijk zouden moeten maken dat negative mutation bias voorkomt bij door-geëvolueerde soorten. Hoe luiden deze argumenten? (Fischer wordt aangehaald in [32, Sec. 3.1], einde middelste alinea.)

    (d) Watson en Pollack [32] betogen dat de fitness van een genotype typisch niet kan worden teruggebracht naar één dimensie:

    > *"It is important to realize that we cannot necessarily reduce multiple dimensions to a single scalar value that will represent a player's quality. We cannot let the fitness of a player be represented by some weighted sum of its component dimensions, for example."*[32, p. 703]

    Waarom kan dat volgens hen niet?

    (e) Watson en Pollack betogen verder dat intransitieve ordening vaak een rol speelt bij co-evolutionair falen:

    > *"The concept of intransitive superiority is central to issues in coevolutionary failure"* [32, p. 704]

    Watson en Pollack verwijzen daarbij naar werk van Cliff en Miller uit 1995, [7], `https://ics-websites.science.uu.nl/docs/vakken/b2nb/stuff/cm95.pdf`. Bespreek wat Cliff en Miller in dat werk hebben gezegd over de rol van intransitieve ordeningen bij het falen van co-evolutionair leren.

12. Zij A een 1-dimensionale cellulaire automaat in de vorm van een ring, met n cellen, 2 mogelijke toestanden per cel ("aan" 1 of "uit" 0), en omgevings-diameter R. Zoals altijd is ook de toestand van de cel zelf van belang bij het bepalen van de nieuwe toestand. Laat $M \in \mathbb{N}$ een groot natuurlijk getal zijn. Het *dichtheids-classificatieprobleem* voor cellulaire automaten is het probleem een regelverzameling te vinden die het volgende doet.

    - Als 50% of meer van de cellen aan staat, dan moeten binnen M generaties alle cellen aan staan.
    - Als meer dan 50% van de cellen uit staat, dan moeten binnen M generaties alle cellen uit staan.

    Het dichtheids-classificatieprobleem is een moeilijk probleem, omdat locale regels gebruikt worden om een globaal probleem op te lossen. Tot op heden is het dichtheids-classificatieprobleem in algemene zin, dus zonder specifieke waarden voor n, k, R en M, nog niet opgelost.

    (a) Omschrijf een opzet om met computationele co-evolutie oplossingen (of goede benaderingen van oplossingen) te vinden voor het dichtheids-classificatieprobleem. Beantwoord daarbij in ieder geval de volgende vragen. Houd er rekening mee dat sommige vragen meerdere keren zullen moeten worden beantwoord, namelijk één antwoord voor elk populatie-type.

    - Hoeveel populatie-typen?
    - Waar worden populaties door bevolkt?
    - Hoe groot zijn elk van de populaties?
    - Is er sprake van competitie of coöperatie tussen populaties?
    - Beschrijf de data-structuur voor genotypen.
    - Hoe worden initiële populaties bevolkt ("gevuld")?
    - Tegen hoeveel tegenstanders meet een individu zijn fitness?
    - Hoe wordt fitness gemeten? (Beschouw ook de mogelijkheid om prestaties over de laatste h generaties heen te onthouden (en eventueel te middelen) om toevallige prestatie-schommelingen uit te vlakken. Hiervoor moet een individu een geheugen krijgen ter lengte h.)
    - Hoe worden mutatie en (eventueel) kruising gedefinieerd?

    (b) In de praktijk blijkt pure co-evolutie met dit probleem niet (onmiddellijk) te werken. Probeer te voorspellen waarom. (Hint: een gedeelte van het antwoord ligt besloten in de volgende vraag.)

    (c) Het blijkt dat het dichtheids-classificatieprobleem toch met co-evolutie kan worden aangepakt als de mating pool van één of meer populaties minder of helemaal niet door fitness wordt gestuurd. Probeer te voorspellen voor welk(e) populatietype(n) dit betreft, en wat het effect zal zijn.

Deel VII. Competitie en coöperatie

23 Speltheorie

1. (Invloed van communicatie in het IPD.) Speel het Iterated Prisoner's Dilemma met je buurman (buurvrouw) in twee episoden van elk tien ronden. Daartoe leg je gelijktijdig één of twee vingers op de rand van de tafel. (Graag zonder toneel, anders kan het werkcollege chaotisch worden.) Houd je vingers op de tafel totdat voor beiden duidelijk is welke acties er gekozen zijn. Uitbetalingen bedragen resp. $R = 3$, $T = 5$, $S = 0$ en $P = 1$. Het doel is je eigen opbrengst te maximaliseren. (Het gaat er niet om wie er na het einde van een episode het meeste punten heeft behaald.)

   (a) Episode 1: zonder overleg.

   (b) Episode 2: vóór deze episode en vóór elke ronde mag er overleg plaatsvinden.

   Herhaal het hele experiment indien nodig. Bemerk je invloed van communicatie? Waarom (niet)? Kun je nog iets bijzonders opmerken over de laatste ronde in episode 1 of 2? Is er nog verschil tussen de laatste ronde in Episode 1 en de laatste ronde in Episode 2?

2. For each of the following situations, define the corresponding game matrix and find all mixed Nash equilibria. Answers may be verified with `https://ics.uu.nl/docs/vakken/ias/main.php?page=nash` but you must write down the calculations.

   (a) *Coordination.*[34] Two mobile phone companies (Samsong and Suny) must decide whether to use 4G wireless broadband Wimax or LTE Advanced.[35] Both players will sell more phones if their protocols are compatible. If they both choose for Wimax the payoffs will be 2 each. (In tens of millions of euros.) If they both choose for LTE Advanced the payoffs will be 1 for each. If they choose different protocols the payoffs will be -1 for each.

   (b) *The welfare game* (Rasmusen, 1989). This game models a government that wishes to aid an unemployed civilian if he searches for work but not otherwise, and an unemployed civilian who searches for work only if he cannot depend on government aid, and who may not succeed in finding a job even if the tries. The payoffs are $3, 2$ (for government, unemployed civilian) if the government aids and the unemployed civilian tries to work; $-1, 1$ if the government does not aid and the unemployed civilian tries to work; $-1, 3$ if the government aids and the unemployed civilian does not try to work; and $0, 0$ in the remaining case.

   (c) *Wage game.*[36] Each of two companies has one job opening. Suppose that Company i ($i = 1, 2$) offers wage w_i, where

   $$0 < w_1/2 < w_2 < 2w_1 \text{ and } w1 \neq w2.$$

   Imagine that there are two employees, each of whom can apply to only one company. The employees simultaneously decide whether to apply to Company 1 or Company 2. If only

[34] Rasmusen (1989). *Games and information: an introduction to game theory*, 2nd edition. Basil Blackwell, Oxford.
[35] `http://en.wikipedia.org/wiki/Format_war#2010s` and `http://en.wikipedia.org/wiki/4G`.
[36] Montgomery (1991) "Equilibrium wage dispersion and interindustry wage differentials," in: *Quarterly Journal of Economics* **106**, pp. 163-197.

one employee applies to a given company, that employee gets the job; if both employees apply to one company, the company hires one employee at random (with probability 1/2) and the other employee is unemployed (and has a payoff of zero).

(d) *Marketing game.* Two companies sell the same product. Each percent of market share yields a net payoff of 10,000 Euro. Without advertising both companies share 50% of the market. The cost of advertising is equal to 10,000 Euro but leads to an increase in market share of 20% at the expense of the other company. The companies make their advertising decisions simultaneously and independently. The total market for the product is of fixed size.

3. (Spel-evenwichten.) Het Oost-Europese spel pummie-pummie is een spel voor twee spelers, A en B, met de volgende uitbetalings-matrix:

		(Speler B)	
		C	D
Speler A	C	$1(-1)$	$3(0)$
	D	$4(2)$	$0(-1)$

Voor het gemak nemen we aan dat A en B uitbetaald worden door een derde partij, genaamd de bank. Negatieve uitbetalingen zijn betaling aan de bank. De bank heeft oneindige reserves. In onderdelen (3a-3g) volgen spelers een pure strategie. In de overige onderdelen volgen spelers een gemixte strategie.

(a) Geef alle Pareto-optima.

(b) Stel, je bent A en speelt tegen B. Je speelt C en krijgt 1 uitbetaald. Wat heeft B gespeeld? Als B dit de volgende keer weer zou spelen, zou je dan nog steeds C spelen? Waarom (niet)?

(c) Stel, je bent A en speelt tegen B. Je speelt D en krijgt 4 uitbetaald. Wat heeft B gespeeld? Als B dit de volgende keer weer zou spelen, zou je dan nog steeds D spelen? Waarom (niet)?

(d) Geef alle pure Nash-equilibria.

(e) Welke spel-evenwichten zijn niet optimaal? ("Evenwicht" in de zin van Nash, optimaal in de zin van "Pareto".)

(f) Stel dat spelers in een sub-optimaal evenwicht verzeild zijn geraakt. Hoe zouden ze hier weer uit kunnen komen?

(g) A heeft de afgelopen 100 ronden het gedrag van B geobserveerd en schat $\Pr_B(C) = q_0$. Als B zo blijft spelen, moet A dan in het vervolg C of D spelen?

(h) Vanaf dit onderdeel hanteren A en B een gemixte strategie met respectievelijk parameters p en q. Bepaal de verwachte winst van A in termen van p en q.

(i) Bepaal de verwachte winst van B in termen van p en q.

(j) Bereken $\partial \text{Payoff}_A(p,q)/\partial p$. Bepaal voor alle hoekpunten $(0,0)$, $(0,1)$, etc. of A reden heeft zijn strategie te wijzigen.

(k) Bereken $\partial \text{Payoff}_B(p,q)/\partial q$. Bepaal voor alle hoekpunten $(0,0)$, $(0,1)$, etc. of B reden heeft zijn strategie te wijzigen.

(l) Bepaal alle gemixte Nash-equilibria.

4. (Drie-weg actie.) Beschouw het volgende (symmetrische) spel.

		(Speler B)		
		b_1	b_2	b_3
	a_1	4(4)	0(5)	0(6)
Speler A	a_2	5(0)	3(3)	0(1)
	a_3	6(0)	1(0)	2(2)

In onderdelen (4a-4g) volgen spelers een pure strategie. In de overige onderdelen volgen spelers een gemixte strategie.

(a) Bepaal alle Pareto-optimale gezamenlijke strategieën.

(b) Bepaal alle Nash-equilibria.

(c) Welk optimale gezamenlijke strategieën zijn equilibria?

(d) Stel, spelers hebben bij aanvang van het spel geen kennis over de precieze waarden in de uitbetalings-matrix. Het enige dat zij kunnen doen is de drie verschillende strategieën uitproberen. Stel dat elke speler al zijn strategieën uniform exploreert. Op welke JS (gezamenlijke strategie) komen spelers op den duur uit?

(e) Hetzelfde scenario als de vorige vraag, alleen exploreren beide spelers precies 10 ronden. Is er verschil?

(f) Stel, beide spelers hebben volledige kennis van de payoff matrix. Ze vertrouwen elkaar niet. Op welke JS komen spelers op den duur uit?

(g) Je schrijft een computer-programma dat dit spel tegen zichzelf zal spelen. Op welke JS komt dit programma uit?

(h) Bepaal alle Nash-equilibria. (Hint: A's strategie wordt nu bepaald door drie kansen, die je kunt modelleren met twee parameters, bv. p_1 en p_2.)

5. Vind alle (dus pure en gemengde) Nash-evenwichten van de spelen met de volgende uitbetalingsmatrix.

(a)

	L	R
T	(0,0)	(2,5)
B	(5,2)	(1,1)

(b)

	L	R
T	(0,0)	(-1,1)
B	(1,-1)	(-3,-3)

(c)

	L	R
T	(-1,1)	(2,2)
B	(2,0)	(0,0)

(d)

	L	R
T	(-1,-1)	(0,1)
B	(0,1)	(2,-3)

(e)

	L	R
T	(0,3)	(6,2)
B	(2,0)	(4,1)

(f)

	L	R
T	(0,0)	(0,1)
B	(0,0)	(0,0)

6. (2-persoons asymmetrische niet-nulsom spelen.) Beschouw de meest algemene variant van een twee-persoons spel met simultane zetten en kwantitatieve uitbetaling.

		(Speler B)	
		C	D
Speler A	C	$R(r)$	$S(t)$
	D	$T(s)$	$P(v)$

A en B hanteren een gemixte strategie met parameters p en q. Bepaal alle Nash-equilibria. (Het aantal verschilt voor verschillende waarden van $R, T, \ldots, r, t, \ldots$) Verdeel, op basis van je antwoorden, dit spel onder in verschillende categorieën en benoem deze.

7. Consider the following payoff matrix for the row player:

$$\begin{array}{c} \\ \text{T} \\ \text{B} \end{array} \begin{array}{c} \begin{array}{cc} \text{L} & \text{R} \end{array} \\ \begin{pmatrix} x & -1 \\ -1 & 1 \end{pmatrix} \end{array}$$

where x is a real number, and T stands for "top," L stands for "left," etc. For which value(s) of x does this game possess a saddlepoint, if any? (Cf. Flake, pp. 285-287, 458, 463.)

8. (Drie-weg actie.) Beschouw het volgende (symmetrische) spel.

		Speler B		
		b_1	b_2	b_3
	a_1	4(4)	0(5)	0(6)
Speler A	a_2	5(0)	3(3)	0(1)
	a_3	6(0)	1(0)	2(2)

   (a) Bepaal alle Pareto-optimale strategie-profielen.

   (b) Bepaal alle pure Nash-equilibria.

   (c) Leg uit waarom het voor Speler A nooit interessant is a_1 te spelen. (En waarom het voor Speler B nooit interessant is b_1 te spelen.)

   (d) Bepaal alle (pure en gemixte) Nash-equilibria. (Hint: gebruik het antwoord van het vorige onderdeel.)

   (e) Welke Nash-equilibria zijn Pareto-optimale strategie-profielen?

9. Gegeven is het volgende twee-persoons competitieve symmetrische niet-nulsom spel op basis van volledige informatie met simultane zetten en kwantitatieve beloningen:

		Speler B	
		b_2	b_3
Speler A	a_2	$(2, 5)$	$(3, -1)$
	a_3	$(1, 1)$	$(4, 4)$

   (a) Bepaal het Pareto front.

(b) Bereken alle pure Nash evenwichten.

(c) Bereken alle pure en gemixte Nash evenwichten.

(d) Stel, de "2" linksboven wordt vervangen door een x, waarbij x een reëel getal is. Bepaal voor dit algemenere geval het Pareto front.

(e) Idem. (De "2" linksboven wordt vervangen door een x.) Bereken alle pure en gemixte Nash evenwichten.

(f) Stel, de "2" linksboven wordt vervangen door een "1". Bepaal $\partial\text{Payoff}_A/\partial p$ en $\partial\text{Payoff}_B/\partial q$.

(g) Idem. (De "2" wordt vervangen door een "1".) Teken een eenheidsvierkant met op de x-as p (gemixte strategie van A) en op de y-as q (gemixte strategie van B). Teken in dit vierkant voor enkele punten (p, q) de gradient. (De richting waarin beide spelers hun strategie willen veranderen, gegeven dat de andere speler dat niet doet.)

- Teken vooral op cruciale punten de gradient. Cruciale punten zijn hoeken, randen, en punten waar de gradient eventueel nul is (Nash-equilibrium). Markeer de laatsten met een stip.
- Het toegestaan gebruik te maken van `https://ics.uu.nl/docs/vakken/maa/2012-13/netlogo_gradient_dynamics.php`.

10. (Matching pennies.) Twee spelers, Speler A en Speler B kiezen gelijktijdig (voor zichzelf) "kop" of "munt". Ze kiezen bewust een kant—de munt wordt niet opgegooid o.i.d. Het doel van Speler A is met Speler B te coördineren; het doel van Speler B is juist om met Speler A te "anti-coördineren". De uitbetalingsmatrix is als volgt.

		Speler B	
		"kop"	"munt"
Speler A	"kop"	$1(-1)$	$-1(1)$
	"munt"	$-1(1)$	$1(-1)$

(a) Zoek op wat een nulsom spel (zero sum game) is. Is matching pennies een nulsom spel?

(b) Waarom is het niet verstandig voor A om de hele tijd "kop" op te gooien?

(c) Waarom is het niet verstandig voor B om de hele tijd "kop" op te gooien?

(d) Stel, B heeft na enig spelen ontdekt dat A ongeveer 95% van de tijd "munt" gooit. (Verder heeft B geen patroon in A's gedrag kunnen ontdekken.) Welke actie kan B in de volgende ronde het best kiezen?

(e) Op den duur merkt Speler A (die tot nu toe overwegend voor "munt" heeft gekozen) dat Speler B overwegend voor "kop" kiest. Hoe verwacht je dat Speler A op basis van deze waarneming zijn strategie zal aanpassen?

(f) Stel, beide spelers hebben 20 ronden gespeeld, als volgt

Speler A:	H	H	H	T	T	T	H	H	T	T	T	T	H	H	H	T	T	T	H	H
Speler B:	H	T	T	T	H	H	H	T	H	T	H	H	H	T	T	T	H	H	H	T

waarbij H staat voor ""kop"," en T staat voor "munt".

(g) Hoeveel nutseenheden (geld) bezitten beide spelers na afloop?

(h) Stel dat beide spelers voor de volgende ronden een actie ("kop" of "munt") kiezen die gebaseerd is op de frequentie waarmee hun tegenstander alle voorgaande ronden voor "kop" heeft gekozen. Voorbeeld: B heeft de afgelopen 20 ronden 11 keer voor "kop" gekozen. Op basis daarvan zal A in ronde 21 voor "kop" kiezen.

Als B zijn strategie op dezelfde manier bepaald, welke actie zal B dan in ronde 21 spelen?

(i) Laat $0 \leq p \leq 1$ de frequentie zijn waarmee A de laatste n ronden "kop" heeft gespeeld. Dus

$$p = \frac{h_A^n}{h_A^n + t_A^n},$$

waarbij h_A^n het aantal keren is dat A de afgelopen n ronden voor "kop" heeft gekozen, en t_A^n het aantal keren is dat A de afgelopen n ronden voor "munt" heeft gekozen.

Stel dat A nu in ronde $n+1$ "munt" speelt. Hoe groot is p dan, uitgedrukt in de oude waarde van p en n?

(j) Stel dat A in ronde $n+1$ "kop" speelt. Hoe groot is p dan, uitgedrukt in de oude waarde van p en n?

(k)
- Teken de strategieruimte van A en B. Dit is een eenheidsvierkant met op de x-as de gemixte strategie p van A en op de y-as de gemixte strategie q van B.
- Teken stippellijnen $p = 1/2$ en $q = 1/2$.
- Teken een strategie-profiel (p, q) als vette punt in de strategieruimte van A en B, zó dat $p > 1/2$ en $q > 1/2$.

Neem aan dat beide partijen hun gemixte strategie aanpassen volgens onderdeel (10h).

Laat door berekening zien dat het punt (p, q) na een volgende ronde is opgeschoven naar een punt (p', q') zó dat $0 \leq p < p' \leq 1$ en $0 \leq q' < q \leq 1$.

- Trek een pijl van (p, q) naar (p', q'). [De coördinaten van (p, q) naar (p', q') weet je niet precies, het gaat vooral om de *richting* van de pijl.]

(l) Teken een strategie-profiel (p, q) in de strategieruimte van A en B, zó dat $p < 1/2$ en $q < 1/2$, en trek een pijl naar de plek van dit profiel na een volgende ronde. Idem met $p < 1/2$ en $q > 1/2$, en $p > 1/2$ en $q < 1/2$.

(m) Beschrijf de dynamiek van strategieën als beide partijen hun gemixte strategie aanpassen volgens onderdeel (10h). Schrijf als dat helpt een simulatie, bijvoorbeeld in Netlogo, om dit antwoord te kunnen geven. (Om de dynamiek te zien schreef ik zelf een Netlogo simulatie. Deze bestond uit 23 regels, waarvan 7 daadwerkelijk betrekking hadden op het algoritme zelf. Het is dus weinig werk om een simulatie te schrijven.)

24 Herhaalde spelen

In dit hoofdstukje duiken we in de dynamische wereld van herhaalde spelen, een essentieel concept binnen de speltheorie en de ontwikkeling van multi-agent systemen (MAS). Waar een eenmalig spel vaak leidt tot een rigide uitkomst (denk aan het bekende prisoner's dilemma waarin wederzijds wantrouwen domineert), opent herhaling de deur naar intelligenter gedrag.

Voor AI-agenten die in een gedeelde omgeving opereren, verandert de horizon: acties hebben niet alleen direct resultaat, maar beïnvloeden ook de toekomstige reputatie en de bereidheid van anderen om samen te werken. In deze context is een algoritme niet enkel een rekenmachine, maar een sociale actor die moet leren balanceren tussen kortstondig eigenbelang en langdurige stabiliteit.

1. Twee spelers spelen herhaaldelijk het gevangenendilemma met de standaard uitbetalingen 0, 1, 3 en 5. De eerste speler bedient zichzelf van All-D, de tweede van Unforgiving.

   Bepaal de gemiddelde opbrengst voor beide strategieën, als beiden 100 ronden spelen.

2. Twee spelers spelen herhaaldelijk het gevangenendilemma met standaard uitbetalingen. De eerste speler bedient zichzelf van Pavlov, de tweede van All-D.

   Bepaal de gemiddelde opbrengst voor beide strategieën, als beiden 100 ronden spelen.

3. Twee spelers spelen herhaaldelijk het gevangenendilemma met de standaard uitbetalingen. De eerste speler bedient zichzelf van TFT, de tweede van RAND.

   Bepaal de verwachte gemiddelde opbrengst voor TFT als beiden 100 ronden spelen.

4. Twee spelers spelen herhaaldelijk het gevangenendilemma met de standaard uitbetalingen 0, 1, 3 en 5. De eerste speler bedient zichzelf van Pavlov, de tweede van TFT. Beide spelers beginnen met verzaken. (Normaal beginnen Pavlov en TFT met samenwerken, nu niet.)

   (a) Geef de totale opbrengst van elke speler na 1000 ronden.

   (b) Geef de verwachte totale opbrengst na 1000 ronden als beide spelers zich bedienen van random 50%.

   (c) Dezelfde vraag, maar nu als één speler zich bedient van random 50%, terwijl de andere speler zich bedient van TF2T, waarbij TF2T begint met samenwerken. (TF2T = werk samen tenzij partner 2× achter elkaar verzaakt heeft.)

   Hint: Bereken de kans op een dubbel-D in een random-rij ter lengte 1000 met door de bank genomen evenveel C's als D's, en beschouw daarbij het eerste element van zo'n random rij apart. Bereken daarna het verwachte aantal dubbel-D's in zo'n rij.

5. Het SIEPD.

   (a) Voor welke $\alpha \geq 0$ is de uitbetalingsmatrix voor het SIEPD qua Pareto-optima en Nash-equilibria (qua plek) gelijk aan de uitbetalingsmatrix van het gevangen-dilemma?

   (b) Geef voor de volgende configuraties aan of de centrumcel meewerkt in de volgende generatie.

Het paar $U_{\max}$ staat daarbij voor de toestand van de cel met de hoogste opbrengst van alle cellen in de omgeving.[37]

C	D	C
C	D	C
C	C	D

$\alpha = 1.2$
$U_{\max} = (C, 7.3)$

C	C	C
D	C	D
C	C	D

$\alpha = 1.4$
$U_{\max} = (D, 7.0)$

6. (Varianten van het gevangenendilemma.) Neem Tabel 6 over en kruis de juiste vakjes aan. De kolom CSIEPD hoeft niet te worden ingevuld. Wanneer je vind dat er redenen zijn voor zowel een blank als een kruis, zet dan een symbool in het desbetreffende vakje en licht direct onder de tabel toe wat de redenen zijn voor zowel een blank als een kruis.

   Tel, als je klaar bent, de posities van de aangekruiste vakjes in machten van twee op, en zet aan het eind van de rij een checksum, dit is om het nakijken te vergemakkelijken.

	IPD	IEPD	SIEPD	SSIEPD	CSIEPD
het is handig een grand table te hebben					
TFT is een mogelijke strategie					
het aantal deelnemers is eindig					
deelnemers bezitten een omgeving					
deelnemers kunnen toestanden aannemen					
deelnemers in vier mogelijke toestanden					
de populatie is homogeen					
oneindig veel mogelijke strategieën					

Tab. 6: Varianten van het gevangenen-dilemma.

[37] En, ja, elke cel hoort bij z'n eigen omgeving.

25 De replicator-dynamiek

De replicator-vergelijking beschrijft hoe verschillende soorten $1, \ldots, n$ dankzij onderlinge interactie groeien. In feite wordt alleen de ontwikkeling van de proporties $P = (P_1, \ldots, P_n)$ bijgehouden. Met de replicator-vergelijking wordt gewoonlijk de continue versie bedoeld, maar er bestaat ook een discrete versie. Deze discrete versie wordt bijvoorbeeld behandeld in Flake's TCBoN, pp. 297-299. (Dat daar de co-evolutionaire replicator-vergelijking wordt toegepast in een competitie-coöperatie context is voor nu niet zo heel relevant.) We herhalen wat daar staat:

Een zg. *relatieve score matrix* beschrijft wat een interactie tussen soort i en j oplevert voor soort i, namelijk R_{ij}:

$$R = \begin{pmatrix} R_{11} & \ldots & R_{1n} \\ \vdots & \ddots & \vdots \\ R_{n1} & \ldots & R_{nn} \end{pmatrix}.$$

Er wordt afgesproken dat altijd $R_{ij} \geq 0$.[38] Op elk moment bevinden de soorten zich in een bepaalde verhouding $P = (P_1, \ldots, P_n)$, waarbij $P_1 \geq 0, \ldots, P_n \geq 0$ en $P_1 + \cdots + P_n = 1$. De opbrengst, of *score* S_i, voor soort i is dan:

$$S_i = \sum_{j=1}^{n} P_j R_{ij}.$$

De replicator-vergelijking schrijft nu voor dat de ongenormaliseerde nieuwe verhouding gelijk moet zijn aan $Q = (P_1 S_1, \ldots, P_n S_n)$. Maar Q sommeert niet tot 1. Om te zorgen dat de proporties wel weer sommeren tot 1 wordt de ongenormaliseerde nieuwe verhouding door het totaal $\sum_{j=1}^{n} P_j S_j$ gedeeld. Dit geeft de nieuwe verhouding

$$P_i^{\text{nieuw}} = \frac{P_i S_i}{\sum_{j=1}^{n} P_j S_j}.$$

De P_i's sommeren nu weer tot 1. Door elke keer weer nieuwe proporties uit oude proporties te generen ontstaat een oneindige rij van proporties die soms naar een limiet-proportie P^* convergeert, en andere keren niet.

1. (a) Het komt wel eens voor dat in de zich ontwikkelende rij van proporties een soort, i, uitsterft. Beredeneer dat soort i dan niet meer kan terugkeren.

   (b) Beredeneer dat geen enkele soort kan uitsterven als elke rij van R tenminste één positief getal bevat, en de start-proporties allemaal positief zijn.

   (c) Gegeven is de discrete replicator-vergelijking met initiële proporties $P = (0.2, 0.5, 0.3)$ en relative score-matrix

   $$R = \begin{pmatrix} 0 & 1 & 3 \\ 2 & 0 & 1 \\ 1 & 3 & 0 \end{pmatrix}.$$

   Bereken één iteratie met de hand.

[38] Als dat niet kan worden waargemaakt dan wordt er bij de rewards een positieve standard *birth rate* β opgeteld. Zo zijn de uiteindelijke rewards toch weer positief. Zie verder Opgave 3.

(d) Gegeven is de discrete replicator-vergelijking met initiële proporties $P = (0.1, 0.2, 0.7)$ en relative score-matrix

$$R = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 0 & 1 \\ 3 & 4 & 0 \end{pmatrix}.$$

Bereken één iteratie met de hand.

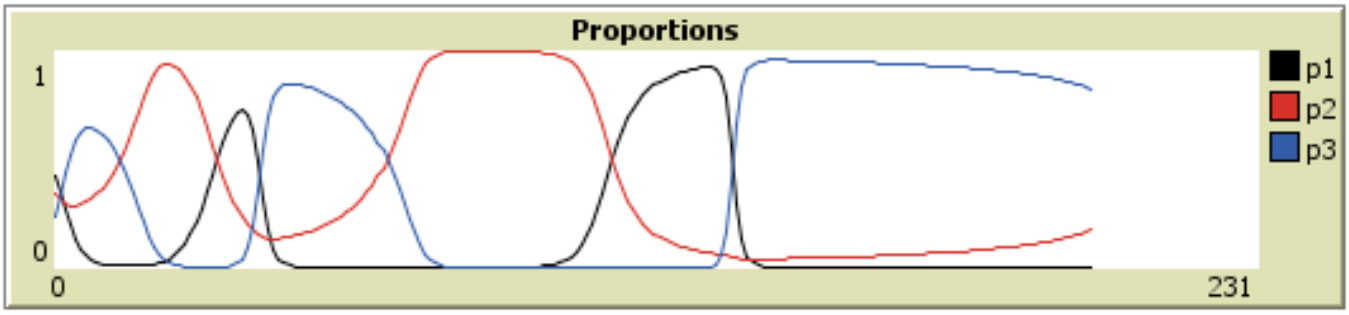
2. Gegeven is de discrete co-evolutionaire replicator-vergelijking[39] met initiële proporties $P = (0.8, 0.1, 0.1)$ en relative score-matrix $R = ((0, 1, 2), (2, 1, 1), (1, 2, 0))$. (Elke binnenlijst is een rij.)

   (a) Bereken één iteratie met de hand.

   (b) Bereken de limiet-proporties P^*. Geef aan hoe je aan je antwoord bent gekomen, en maak dat plausibel. Neem bv. een plot op van het convergentieproces en/of hecht programmacode aan.

3. (De discrete replicator-vergelijking, iets andere insteek.) De (discrete en continue) replicator-vergelijking modelleert hoe de verhoudingen tussen n soorten in een populatie met $q_1 + \cdots + q_n = q$ individuen kan evolueren, als die individuen onderling met elkaar concurreren.

   Hieronder zie je een voorbeeld



De ontwikkeling van drie soorten met een discrete replicator-vergelijking.

Je ziet dat in een replicator-dynamiek de verhoudingen tussen soorten (in ieder geval in het begin) nogal kunnen fluctueren. Meestal gaat het er niet niet zo wild aan toe en is er snel convergentie. In die zin is dit dus een a-typisch plaatje :-)

   (a) Laat $p_j = q_j/q$ het percentage, of de proportie, van soort i zijn. In de afbeelding hierboven zijn bijvoorbeeld startproporties $p = (0.42, 0.34, 0.24)$ genomen. Ga na dat in het algemeen $p_1 + \cdots + p_n = 1$.

   (b) Om soorten tegen elkaar uit te kunnen spelen is een zg. *relatieve score matrix* nodig die beschrijft wat een treffen tussen twee individuen van type i en j oplevert, of gemiddeld oplevert, voor het individu van type i. We noteren dat als R_{ij} (de "r" voor reward). Als we deze informatie voor alle mogelijke interacties tussen soorten hebben we een relatieve score matrix:

$$R = \begin{pmatrix} R_{11} & \dots & R_{1n} \\ \vdots & \ddots & \vdots \\ R_{n1} & \dots & R_{nn} \end{pmatrix}.$$

[39] Zie bijvoorbeeld Flake, pp. 297-299.

In de afbeelding hierboven werd $R = ((1,3,1),(1,2,3),(4,1,3))$ gebruikt, dus $R_{31} = 4$. Stel dat in dit voorbeeld op $t = 10$ de proporties gelijk zijn aan $p = (0.1, 0.4, 0.5)$. Stel dat een individu van soort 1 een willekeurig ander individu tegenkomt. Wat is dan de verwachte opbrengst? Hint: $0.1 \times R_{21} + \ldots$. Dezelfde vraag voor soorten 2 en 3.

(c) Laat zien dat de verwachte opbrengst voor soort i per interactie met een willekeurig individu in het algemeen gelijk is aan

$$f_i = \sum_{j=1}^{n} p_j R_{ij}.$$

De verwachte opbrengst wordt ook wel *fitness* genoemd, vandaar de letter f.

(d) Geef voor het voorbeeld hierboven een formule voor de fitness per individu per soort bij gegeven proporties. Hint:

$$\begin{cases} f_1 = p_1 R_{11} + p_2 R_{12} + \ldots \\ f_2 = p_1 R_{21} + \ldots \\ f_3 = \ldots \end{cases} \quad \Rightarrow \quad \begin{cases} f_1 = p_1 + 3p_2 + \ldots \\ f_2 = p_1 + \ldots \\ f_3 = \ldots \end{cases}$$

(e) Het idee van de replicator-vergelijking is dat een individu van soort i tussen twee generaties in $1 + \beta + f_i(t)$ nakomelingen produceert, waarbij β de intrinsieke, en f_i de fitness-afhankelijke geboorte-ratio is. Om de replicator-vergelijking te laten werken moet altijd $1 + \beta + f_i(t) \geq 0$ voor alle i. Dit kan met een voldoend grote β desnoods worden afgedwongen. Zo doende

$$q_i(t+1) = q_i(t)[1 + \beta + f_i(t)]. \tag{9}$$

Dit noemen we de *discrete stap-vergelijking*.

Laat middels de discrete stap-vergelijking zien dat een soort uitsterft als <u>en slechts als</u> op enig moment de fitness van die soort gelijk is aan $-1 - \beta$.

(f) Laat

$$\Delta q_i(t) = q_i(t+1) - q_i(t)$$

de groei van soort i zijn van tijdstip t naar tijdstip $t+1$. Laat zien dat soort i even hard groeit als dat zij groot is, maal $\beta + f_i(t)$:

$$\Delta q_i(t) = q_i(t)[\beta + f_i(t)].$$

(g) Laat $\bar{f}(t)$ de gemiddelde fitness zijn op tijdstip t, dus

$$\bar{f}(t) = p_1 f_1(t) + \cdots + p_n f_n(t).$$

Wat is de gemiddelde fitness van de populatie in het lopende voorbeeld op tijdstip $t = 10$? Hint: gebruik het antwoord op vraag 3b.

(h) Leid de *discrete replicator-vergelijking* af:

$$p_i(t+1) = p_i(t) \frac{1 + \beta + f_i(t)}{1 + \beta + \bar{f}(t)}$$

Hint: $p_i(t+1) = q_i(t+1) / \sum_{j=1}^{n} q_j(t+1)$. Schrijf uit en deel teller en noemer door $q(t)$. Je komt dan bij proporties uit.

(i) Laat zien dat soort i groeit als en slechts als $p_i(t) > 0$ en de fitness van deze soort groter is dan gemiddeld.

(j) Wat als β erg groot is?

4. Deze opgave gaat over de zg. replicator dynamics. Gegeven is een populatie met drie fenotypen E_1, E_2, en E_3. Fenotype E_1 bezit fitness $f_1 = 9$, fenotype E_2 bezit fitness $f_2 = 6$, en fenotype E_3 bezit fitness $f_3 = 3$. Stel nu dat in het begin alle individuen even vaak voorkomen in de populatie, dus $x_1 = x_2 = x_3 = 1/3$.

   (a) Bepaal de gemiddelde fitness $\hat{f}(\vec{x})$ van de populatie.

   (b) Laat met de replicator vergelijking zien dat met aanpassingssnelheid α de relatieve frequenties tot 1.0 blijven sommeren.

   (c) Laat zien dat het gebruik van $\alpha = 1.0$ verkeerde waarden kan opleveren.

   (d) Itereer twee tijdstappen met de replicatorvergelijking het systeem. Dus reken alle relatieve frequenties uit na twee updates. Neem als aanpassingssnelheid $\alpha = 0.1$.

26 Populatiedynamiek

1. (Het modelleren van een epidemie middels een differentievergelijking.) Een epidemie kan worden gemodelleerd door middel van een stelsel differentievergelijkingen.

$$H(t+1) = H(t) - aH(t)S(t)$$
$$S(t+1) = S(t) + aS(t)H(t) - bS(t)$$

met regelparameters a en b. De expressies $H(t)$, $S(t)$ en $I(t)$ staan voor het aantal gezonde, zieke en immune personen op tijdstip t.

We beginnen met een bepaalde toestand $(H(0), I(0), S(0))$, en maken gebruik van de vergelijkingen om de evolutie van het systeem te bepalen. We nemen aan dat de populatiegrootte constant blijft. Merk op dat H, I en S niet negatief mogen worden.

   (a) Druk de totale populatiegrootte uit in $H(t)$, $S(t)$ en $I(t)$.

   (b) Druk $I(t+1)$, het aantal immuun geworden personen in generatie $t+1$, uit in $H(t)$, $S(t)$ en $I(t)$, $H(t+1)$ en $S(t+1)$.

   (c) Leid een vergelijking af voor $I(t+1)$, het aantal immuun geworden personen in generatie $t+1$.

   (d) Leg uit waarom een toestand $(H(t), S(t), I(t))$ met $H(t) = S(t) = 0$ stabiel is.

   (e) Leg uit wat er gebeurd als $a = 0$; als a klein is.

   (f) Leg uit wat er gebeurd als a (te) groot is.

   (g) Leg uit wat er gebeurd als $b = 0$; als b klein is.

   (h) Leg uit wat er gebeurd als b (te) groot is.

2. (Het modelleren van een epidemie middels een cellulaire automaat.) Zie Fig. 18. Completeer één configuratie van de volgende generatie. De regels staan in het dictaat. Neem daarbij aan dat

$$\Pr(s_{t+1}(x) = I \mid s_t(x) = Z) = 0.25$$

voor elke cel x. Gebruik twee munten om dit te simuleren.

3. Een model voor de populatiegrootte van bacteriën op een petri-schaaltje op tijdstip t is gegeven door de recursieve betrekking

$$P_{t+1} = aP_t(1 - P_t). \tag{10}$$

   (a) Stabiliseert de populatie voor $a = 2.50$? (Hint: ga niet itereren met een rekenmachine maar gebruik de gegevens uit Flake's boek, de handouts, of het dictaat.)

   (b) Voor $a = 3.33$?

   (c) Waarom levert het itereren met een rekenmachine niet altijd een antwoord op?

   (d) In welke gevallen is dat?

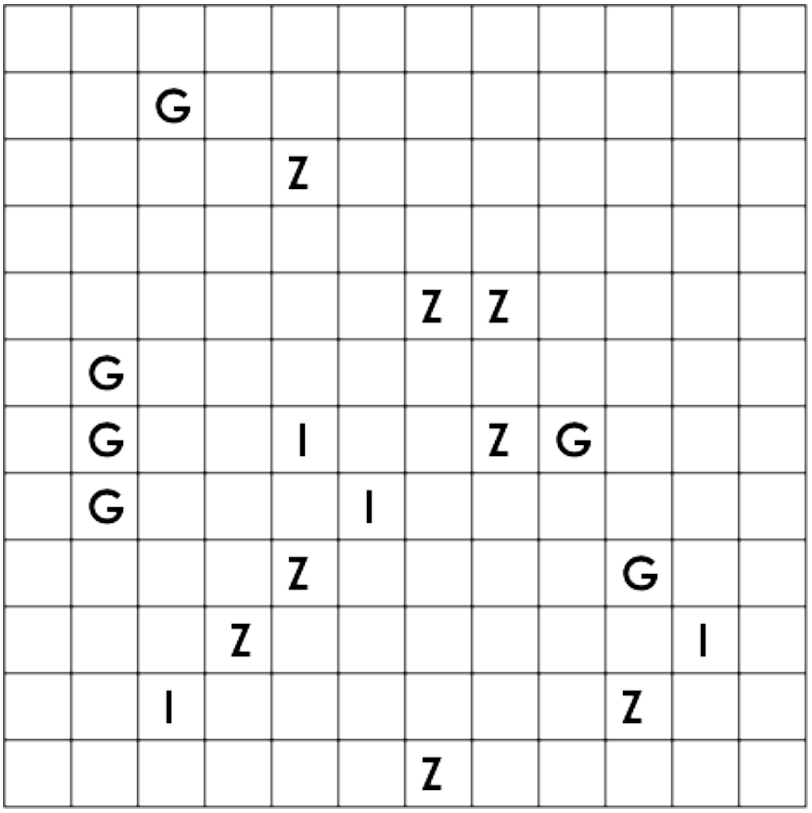


Fig. 18: Het modelleren van een epidemie middels een cellulaire automaat

4. Beschouw het volgende model voor groei:

$$\frac{dP}{dt} = \lambda P \left(1 - \left(\frac{P}{K}\right)^{\eta}\right) \tag{11}$$

   (a) Vind de evenwichtspunten van dit systeem, en geef aan of ze stabiel of labiel zijn.

   (b) Zet $K = 1000$ en $\lambda = 0.2$ en schets in een grafiek de groei voor drie verschillende waarden van η.

   (c) De logistieke functie (Eng.: *logistic map*) werkt op relatieve populatiegrootte die loopt van 0.0 to 1.0. Werk de logistieke functie om tot een functie op de absolute populatiegrootte. Neem aan dat de maximale populatiegrootte gelijk is aan K.

   (d) Bespreek de verschillen en overeenkomsten met de logistieke functie.

5. (Lotka-Volterra.) Laat x staan voor hoeveelheid prooi en y voor hoeveelheid jager.

$$\begin{cases} dx/dt = x(a - by) \\ dy/dt = y(-c + dx) \end{cases} \tag{12}$$

   met positieve a, b, c, en d. Zie Fig. 19.

   (a) Bepaal de triviale dekpunten van de populaties gegeven door (12).

   (b) Voorspel de dynamiek als $x_0 = 0$ en $y_0 > 0$. Dus voorspel wat er gebeurd als er in het begin wel vossen zijn, maar geen konijnen. Probeer dit ook uit te drukken met behulp van Eq. (12) door daar geschikte waarden in te vullen.

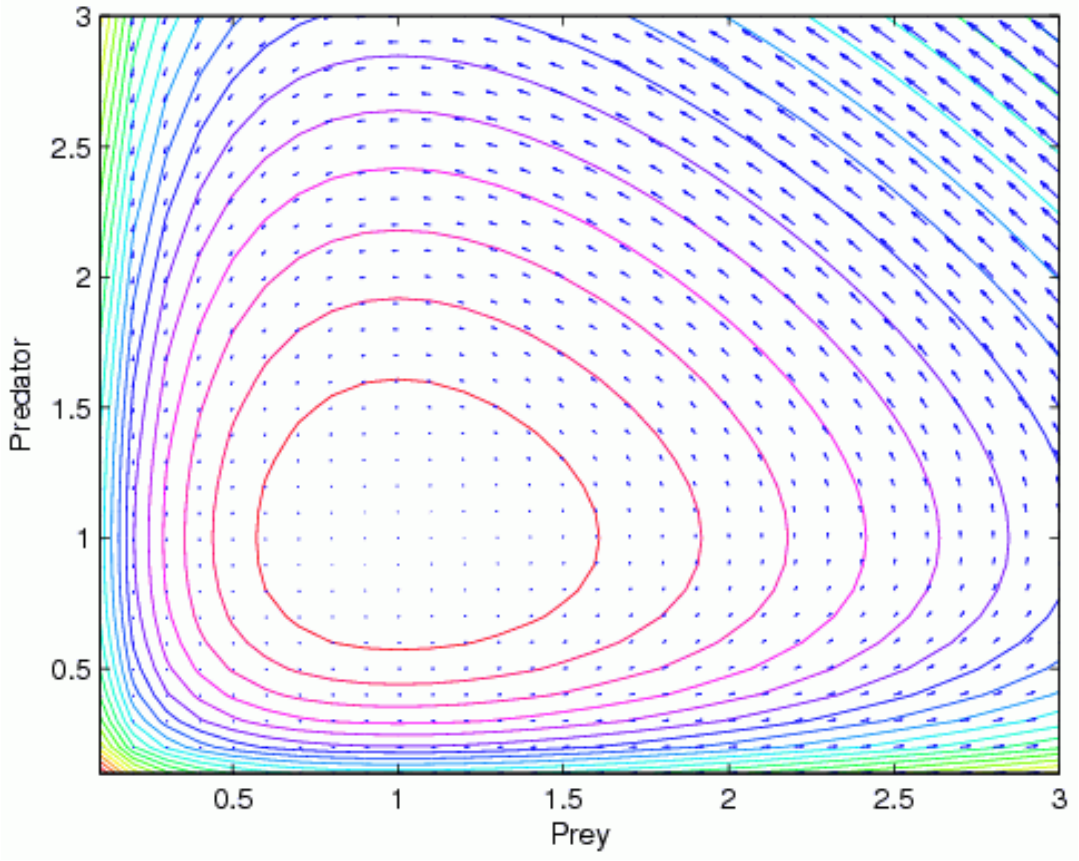


Fig. 19: Interactie tussen populaties volgens Lotka-Volterra

(c) Voorspel de dynamiek als $x_0 > 0$ en $y_0 = 0$.

(d) De eigenwaarden van de Jacobiaan[40] zijn belangrijk voor de bepaling van de dynamiek rond een dekpunt (Eng.: stability analysis).

- $n0 =_{\text{Def}}$ aantal eigenwaarden met reële gedeelte gelijk nul.
- $n+ =_{\text{Def}}$ aantal eigenwaarden met positief reëel gedeelte.
- $n- =_{\text{Def}}$ aantal eigenwaarden met negatief reëel gedeelte.

Een dekpunt kan worden geclassificeerd op drie manieren:

i. Als $n- > 0$ en $n+ = 0$, dan is het dekpunt **st**abiel (Eng.: *attractor*).
ii. Als $n- = 0$ en $n+ > 0$, dan is het dekpunt **l**abiel (Eng.: *repellor*).
iii. Als $n- > 0$ en $n+ > 0$, dan is er vanuit één as aantrekking, en vanuit een andere as afstoting (Eng: *saddle point*).

In andere gevallen weten we niets.

Bepaal van alle dekpunten in welke klasse deze vallen. Dus bepaal eerst de Jacobiaan van de rechterkant van (12), en bereken dan de eigenwaarden van de Jacobiaan. Het reële gedeelte bepaalt of het punt aantrekt of afstoot, en het imaginaire gedeelte geeft aan of er oscillerend gedrag is.

(e) Bekijk de trajecten (Eng: *orbits*) in een Lotka-Volterra systeem met drie soorten (Flake, Fig. 12.4, p. 189). Bekijk de baan in sub-figuur (c). Deze snijdt zich, of lijkt zich te snijden, bij de 2e omloop. Kan een periodiek traject in een Lotka-Volterra zichzelf

[40] The Computational Beauty of Nature, Sec. 13.2, p. 207.

tussentijds snijden? Waarom (niet)? Zo ja, wat is er dan aan de hand? Zo nee, waarom kan dat niet, en hoe kan het systeem eventueel worden aangepast om doorsnijding toch mogelijk te maken?

6. De logistieke map is een voorbeeld van een discreet dynamisch systeem. Een stelsel Lotka-Volterra vergelijkingen is een voorbeeld van een continu dynamisch systeem. Welke van de volgende beweringen zijn ***niet*** waar?

   i) Discrete systemen kunnen zich chaotisch gedragen in $[0, 1]$.

   ii) Discrete systemen kunnen zich chaotisch gedragen in $[0, 1]^2$.

   iii) Discrete systemen kunnen zich chaotisch gedragen in $[0, 1]^3$.

   iv) Continue systemen kunnen zich chaotisch gedragen in $[0, 1]$.

   v) Continue systemen kunnen zich chaotisch gedragen in $[0, 1]^2$.

   vi) Continue systemen kunnen zich chaotisch gedragen in $[0, 1]^3$.

   (a) *iv)* en *v)*.

   (b) *v)* en *vi)*.

   (c) *iv)* en *vi)*.

   (d) Het goede antwoord staat er niet bij.

27 Chaos-theorie

Chaos-theorie
bestudeert systemen waarbij een kleine verandering in het begin leidt tot een enorme afwijking verderop in het proces. Hierdoor lijkt de toekomst onvoorspelbaar en chaotisch, terwijl toch alles strikt volgens vaste regels verloopt.

Je ziet hier een still uit Alberto Cagnetta's Youtube video "Bunimovich Stadium billiard". Duizenden deeltjes beginnen, naast elkaar geplaatst, gelijktijdig, onafhankelijk en met dezelfde snelheid naar het Noorden te bewegen; alleen hun horizontale start-coördinaat verschilt. Wanneer ze een rand treffen, kaatsen ze terug. In het begin gaan ze nog gelijk op, maar als snel lopen hun sporen uiteen, en is niet meer te zeggen welke deeltjes nu oorspronkelijk buren waren.

In populatiedynamiek verklaart
chaos-theorie grillige schommelingen zonder externe ruis (i.e., zonder de exogene inwerking van toeval). Bij genetische algoritmen voorkomt chaos dat zoekprocessen vastlopen in lokale optima. Het biedt zo een essentieel raamwerk voor het begrijpen van niet-lineaire groei en adaptatie.

Helaas is er geen wetenschappelijke
consensus over de precieze definitie van chaos.
Wel is men het eens over de volgende vier eigenschappen.

1. Chaos is een *deterministisch* proces: er is geen toeval.
2. Chaos is gevoelig voor *perturbaties* (kleine veranderingen in startwaarden). Dit wordt ook wel het *het vlinder-effect* genoemd.
3. Het proces *mixt*. Dat wil zeggen dat er een punt s_0 is dat, als het geïtereerd wordt, dicht licht in de toestandruimte. Anders gezegd: de baan van s_0, i.e.,

   $$s_0, f(s_0), f(f(s_0)), f(f(f(s_0))), \cdots = s_0, s_1, s_2, s_3, \ldots$$

   komt willekeurig dicht bij elk punt in de toestandruimte. De functie f representeert hierbij de overgangsfunctie van het chaotische proces.

   Preciezer: voor elk punt y in de toestandruimte X, en elke willekeurig kleine afstand $\epsilon > 0$, is er een $N \in \mathbb{N}$, zodanig dat $d(s_N, y)) \leq \epsilon$.
4. De verzameling periodieke punten ligt dicht in de toestand-ruimte.

   Preciezer: voor elk punt x in de toestandruimte X, en elke willekeurig kleine afstand $\epsilon > 0$, is er een periodiek punt $y \in X$ zodanig dat $d(x, y) \leq \epsilon$. Het periodiek zijn van y kan worden uitgedrukt door het feit dat er een $N \in \mathbb{N}$ is zodat $f^N(y) = y$.

Er kan bewezen worden dat (2) volgt uit (3) en (4).

Het stadium van Bunimovich is een uitstekend voorbeeld—en een uitstekende verbeelding—van een elementair chaotisch proces. Het is echter moeilijk om in dit model te rekenen: al snel hou je je meer bezig met natuurkundige berekeningen dan met de geheimen van chaos-theorie. Daarom behandelen onderstaande opgaven eenvoudiger chaotische systemen.

1. Beschouw de *decimale schuif-afbeelding*

$$f : [0,1] \to [0,1] : x \mapsto 10x \pmod 1.$$

Uitleg over modulo rekenen:

- De uitdrukking
$$x \pmod m$$
is gelijk aan het getal dat je krijgt als m zo vaak mogelijk afgetrokken wordt van x, zonder de uitkomst negatief te laten worden. Spreek uit: "de rest van x bij delen door m". Bijvoorbeeld: $8 \pmod 3 = 2$ en $8.4 \pmod 3 = 2.4$.
- De uitdrukking $x = y \pmod m$ impliceert dat x en y een veelvoud van m van elkaar verschillen. Dus $x = y + nm$ voor zekere $n \in \mathbb{Z}$. Spreek uit: "x is gelijk aan y modulo m". Bijvoorbeeld: $8.4 = 5.4 \pmod 3$. Of: "20:12 uur dinsdag = 20:12 uur zaterdag (mod 24 uur)".

Bewering: f is een chaotische afbeelding. We gaan dit checken door Devaney's vier condities voor chaos na te lopen. We gaan dus de volgende eigenschappen controleren: (1) f is deterministisch; (2) f is gevoelig voor perturbaties (kleine veranderingen in startwaarden); (3) f mixt (er is een punt dat, geïtereerd, dicht ligt in de toestand-ruimte); (4) f's verzameling periodieke punten ligt dicht in de toestand-ruimte.

(a) Waarom is f deterministisch?

(b) Beschrijf wat f "doet" met haar argument. Hint: schrijf het argument in decimale notatie. Wat doet f met de decimalen?

(c) Bereken $f(0.012345)$, $f(0.812345)$, en $f(0.12345)$. Is f injectief? Waarom wel / niet?

(d) Teken de grafiek van f.

(e) Een dekpunt van een functie g is een punt x in het domein van g waarvoor $g(x) = x$. Geef in de grafiek van f de dekpunten aan van f, door een lijn $y = x$ te trekken, en de dekpunten te markeren met een bolletje. Beredeneer wat de dekpunten moeten zijn, of reken de dekpunten gewoon direct uit.

(f) Herinner dat
$$f^k(x) =_{Def} \underbrace{f(f(...f}_{k}(x))).$$
Een punt $x \in [0,1]$ met periode k is een punt waarvoor geldt dat $f^k(x) = x$. Dus na k keer kom je weer terug bij x.[41]
Beredeneer dat voor elke $k \in \mathbb{N}$ een punt $x \in [0,1]$ met periode k bestaat. Kijk daarvoor naar groepjes decimalen ter lengte k. Hoe ziet een punt met periode k er uit qua decimale ontwikkeling?

[41] Verwar een punt met periode k overigens niet met een mogelijke periode k van de functie f *zelf*: $f(x+k) = f(x)$ voor alle x.

(g) Hoeveel punten met periode k zijn er in het algemeen (dus niet specifiek van het type uit het vorige onderdeel)?

(h) Beredeneer dat $[0, 1]$ overaftelbaar veel a-periodieke punten bevat.

Hint 1: Als je het wilt beredeneren, maak dan gebruik van het feit dat elementen van $\mathbb{Q}$ precies díe getallen zijn waarvan de decimale ontwikkeling zich op een gegeven moment gaat herhalen.

Hint 2: $\mathbb{Q}$ is aftelbaar, en $[0, 1]$ niet.

(i) Beredeneer dat er overaftelbaar veel punten in $x \in [0, 1]$ zijn waarvoor de reeks

$$x, f(x), f^2(x), f^3(x), f^4(x), \ldots$$

dicht ligt in $[0, 1]$.[42] Dus beredeneer, of toon aan, dat er overaftelbaar veel getallen $y \in [0, 1]$ te vinden zijn zó dat voor elke willekeurig nauwkeurigheid $\epsilon > 0$ er altijd een $k \in \mathbb{N}$ te vinden is zo dat $|f^k(x) - y| \leq \epsilon$. Maak daarbij gebruik van het feit dat overaftelbaar veel reële getallen zogenaamd "rijk" zijn.

Ligt van elk a-periodiek punt de ontwikkeling dicht in $[0, 1]$? Waarom wel/niet?

(j) Beredeneer dat f extreem gevoelig kan zijn voor willekeurig kleine verstoringen in de start-waarde. (Het zg. vlinder-effect.)

(k) Beredeneer dat f mixt.

2. We gaan experimenteren met het zogenaamde *spinnenweb-diagram* (Eng.: *cob web diagram*) van de logistieke afbeelding. Zoek eerst een geschikte web demo.[43]

Schuif de slider r (in sommige apps: a) langzaam naar rechts.

(a) Voor welke waarden van $r \in [0, 4]$ convergeert de dynamiek naar een vast punt?

(b) Voor welke waarden van $r \in [0, 4]$ convergeert de dynamiek naar een cyclus met periode 2?

(c) Wanneer naar een cyclus met periode 4?

(d) Periode 8?

(e) Bestaat er een waarde voor r waarbij de dynamiek een stabiele periode 3 vertoont?

(f) Zijn er ook waarden van r waarbij de dynamiek een stabiele periode 5 vertoont?

(g) Beschrijf de dynamiek voor $r = 4$.

(h) Is er ook chaotisch gedrag voor $r \neq 4$?

3. De grootte van een populatie die groeit in een ruimte met begrensde reserves kan worden gemodelleerd middels de logistieke afbeelding

$$\kappa : [0, 1] \to [0, 1] : x \mapsto 4x(1 - x)$$

Iemand wil uitrekenen hoe groot de populatie is na $10,000$ iteraties, voor een startwaarde $s_0 = 0.75$.

(a) Dit kan: gewoon $10,000$ keer itereren en Bob's your uncle.

[42] Maak zo nodig Opgave 14 op pagina 61.

[43] De software-pagina van de cursus-site bevat ook een web-demo (zoek op "logistieke"). Een screen shot daarvan is afgebeeld in Fig. 20.

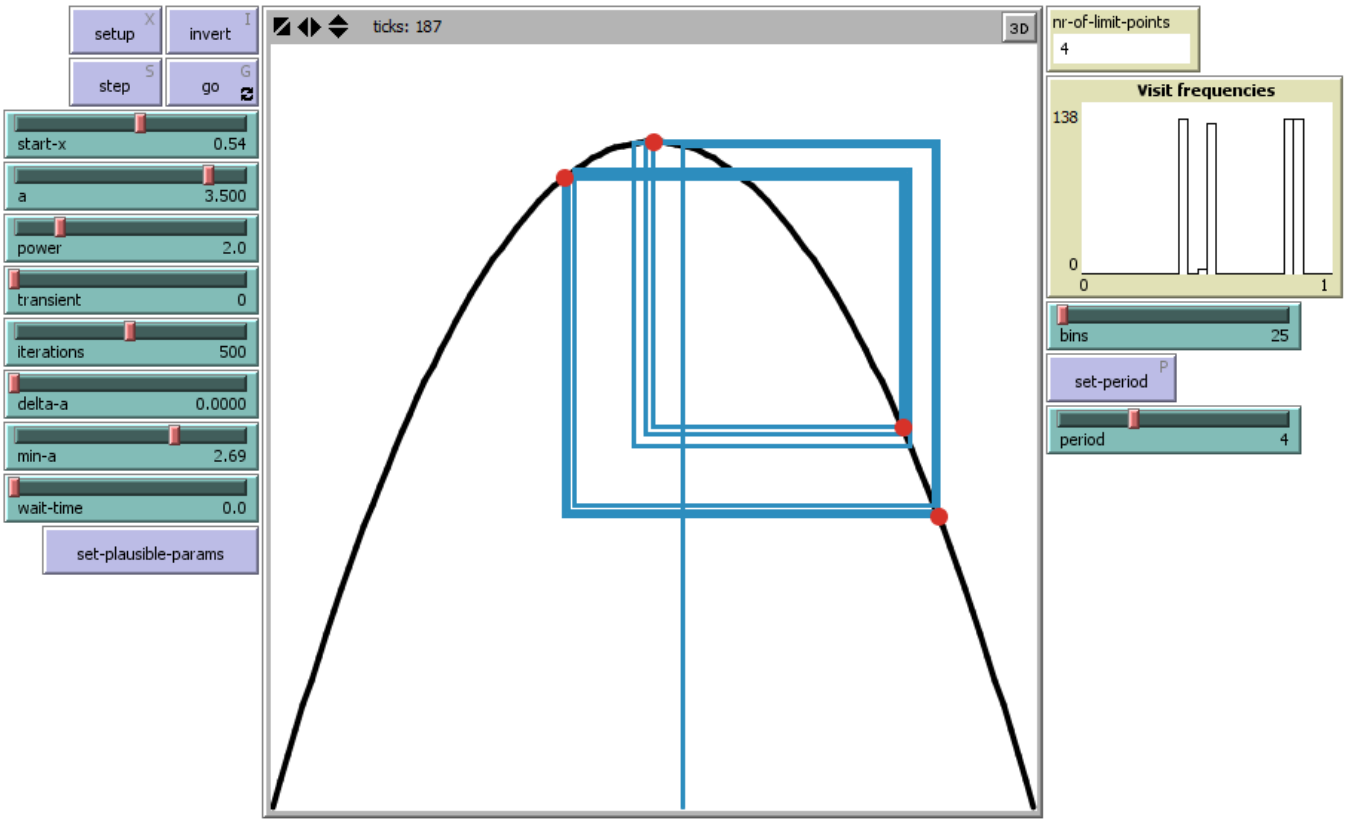


Fig. 20: Een Netlogo-implementatie van het spinnenweb-diagram van de logistieke afbeelding. De app is te vinden op de software-pagina van de cursus-site.

(b) Dit kan: maar $10,000$ keer itereren is niet nodig.

(c) Dit kan niet, want κ is chaotisch voor $r = 4$.

(d) Dit kan niet, want computers kunnen maar met een eindige precisie rekenen, en de imprecisie van de baan van 0.75 neemt exponentieel toe.

4. Beschouw de logistieke afbeelding $x \mapsto rx(1-x)$ op het interval $[0,1]$ waarbij $r \in [0,4]$, en bekijk Fig. 21. De verschillende waarden van r worden onderverdeeld in intervallen:

$$\begin{array}{lll} 1\text{: } r \in [0,1) & 4\text{: } r \in [3, 1+\sqrt{6} \approx 3.45) & 7\text{: } r \in [3.57, 4]. \\ 2\text{: } r \in [1,2) & 5\text{: } r \in [1+\sqrt{6}, 3.54) & \\ 3\text{: } r \in [2,3) & 6\text{: } r \in [3.54, 3.57) & \end{array}$$

Het gedrag bij oneindige iteratie kan voor deze intervallen als volgt worden omschreven:

A. Bijna altijd cyclisch met periode 4.
B. Bijna altijd chaotisch gedrag (soms weer periodiek).
C. Monotone convergentie naar $(r-1)/r$.
D. Bijna altijd cyclisch met periode 2.
E. Convergentie naar nul.
F. Alternerende convergentie naar $(r-1)/r$.
G. Bijna altijd cyclisch met periode 2^n, $n > 2$.

Koppel het juiste gedrag aan de juiste intervallen. (Hint: haal informatie uit TCBoN.)

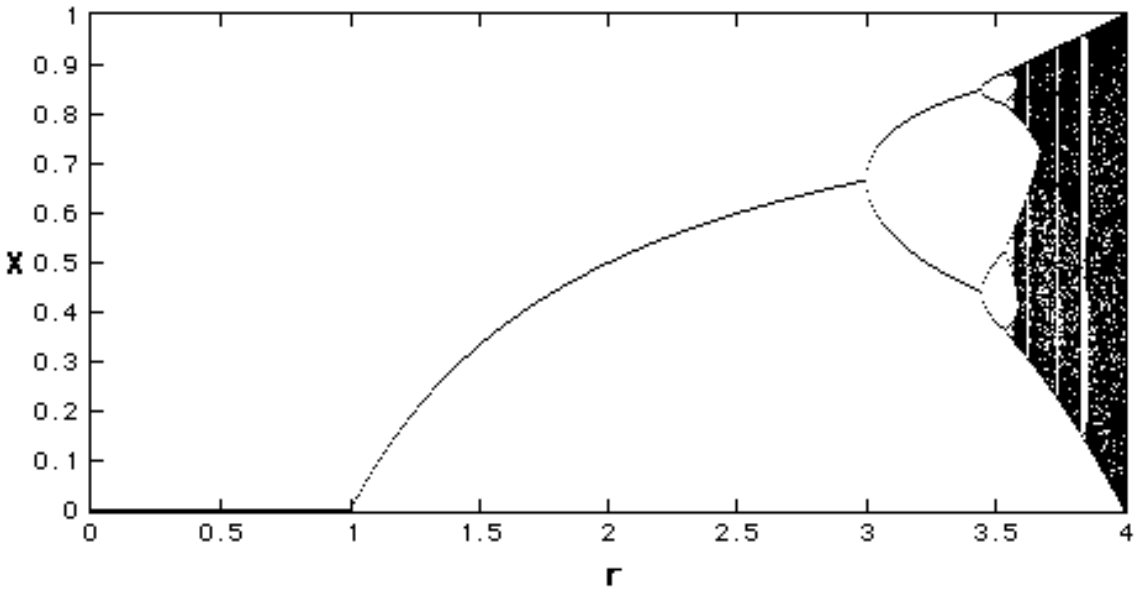


Fig. 21: Bifurcatiediagram van de logistieke afbeelding, $0 \leq r \leq 4$.

(a) 1:D, 2:A, 3:G, 4:E, 5:C, 6:F, 7:B.

(b) 1:G, 2:B, 3:E, 4:C, 5:F, 6:D, 7:A.

(c) 1:D, 2:A, 3:G, 4:B, 5:E, 6:C, 7:F.

(d) 1:E, 2:C, 3:F, 4:D, 5:A, 6:G, 7:B.

5. De familie van logistieke afbeeldingen wordt gegeven door:

$$f_r : [0,1] \mapsto [0,1] : x \mapsto rx(1-x), \quad r \in [0,4]$$

We zeggen dat f_r op startwaarde x_0 convergeert naar een stabiele cyclus met periode p als $\lim_{n\to\infty} f_r^{np}(x_0)$ bestaat en eindig is, en de rij bovendien terug convergeert naar deze limiet als de rij op het eind een beetje verstoord wordt. Anders gezegd: als we in de omgeving van de limiet opnieuw beginnen met itereren, dan zullen we weer op deze limiet uitkomen. Vandaar de naam stabiel.

Stilzwijgend bedoelen we meestal ook nog dat p minimaal is, dus dat f_r niet tegelijkertijd convergeert naar een stabiele cyclus met een kleinere periode.

(a) Bekijk een (gedetailleerd!) bifurcatiediagram van de logistieke afbeelding f_r en geef in vier decimalen nauwkeurig een waarde voor r waarvoor f_r een stabiele cyclus met periode $p = 2$ bezit. Hint: gebruik een online bifdiagram analyser.

Welke startwaarde heb je gekozen? Welke startwaarde mag niet worden gekozen omdat je anders beland in een cyclus met periode 1 (= dekpunt)?

(b) Voor welke r lijkt f_r naar een stabiele cyclus van periode 4 te convergeren? Weer in vier decimalen nauwkeurig. Welke startwaarden mogen (niet) worden gekozen?

(c) Voor welke r lijkt f_r naar een stabiele cyclus van periode 3 te convergeren? Welke startwaarden mogen (niet) worden gekozen?

(d) Probeer waarden voor r in het bifurcatie diagram te ontdekken waarvoor f_r convergeert naar een stabiele cyclus met periode 5, 6, 7, 8 en 9.

6. (a) Bereken de dekpunten van f_r. Let op: de waarde van een dekpunt hangt soms af van r.

   (b) Met enige kennis van differentiaalrekening en recurrente betrekkingen is aan te tonen dat, als x^* een dekpunt is van f, en f continu differentieerbaar is, dit dekpunt stabiel (ook: wel aantrekkend) is a.e.s.a. $|f'(x^*)| < 1$, en labiel (ook wel: afstotend) is a.e.s.a. $|f'(x^*)| > 1$. Zie ook Sec. 10.2 van TCBoN (Flake). We bewijzen dit nu niet, maar gebruiken dit gegeven nu slechts als feit. Onderzoek welke dekpunten van f_r stabiel zijn en welke labiel. Illustreer convergentie met een cobweb (spinnenweb) diagram voor $r = 2.5$ en startwaarde $x_0 = 0.2$.

   (c) Teken een grafiek met op de horizontale as $0 \leq r \leq 4$ en op de vertical as de dekpunten (dus x-waarden uit $[0, 1]$ waarvoor $f_r(x) = x$). Stippel de grafiek als de dekpunten labiel zijn.

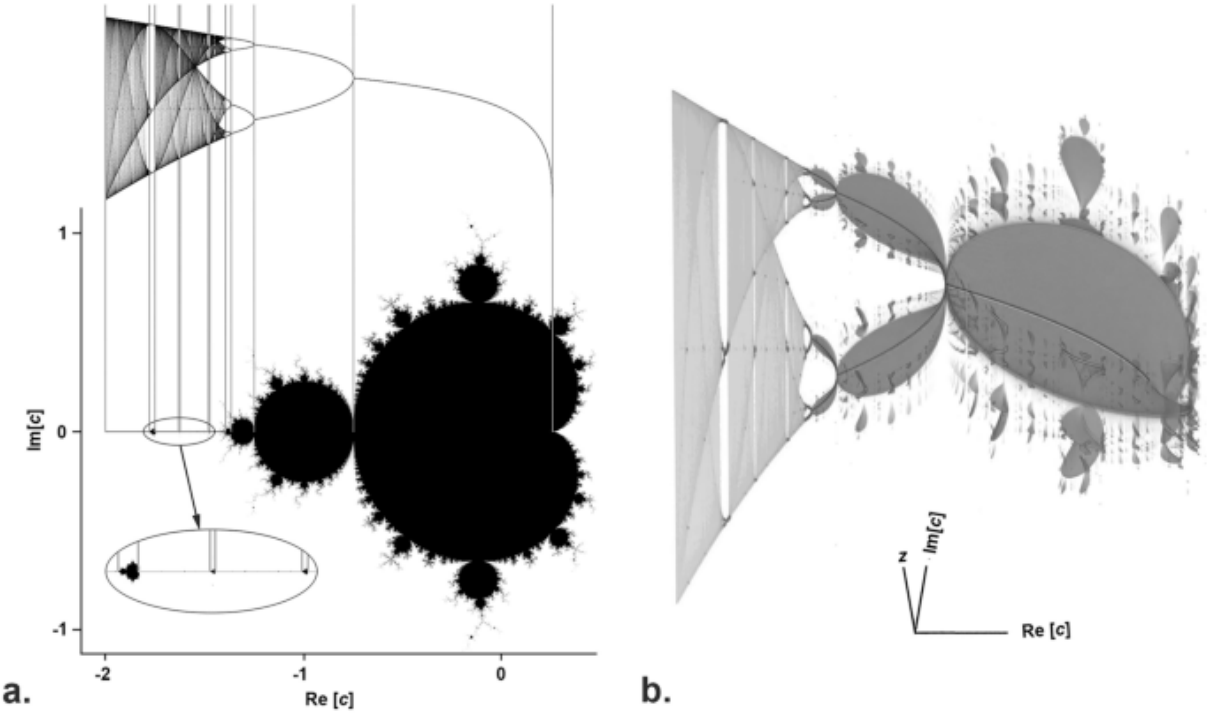
   (d) Bereken de punten met periode twee. Hint: bepaalde de dekpunten van f_r^2 en haal er twee factoren van de vorm $(x - a)$ uit, waarbij a een dekpunt van f_r is. Los vervolgens een kwadratische vergelijking op.

   (e) Breid de grafiek uit met op de y-as de perioden twee. Bepaal wanneer deze perioden aantrekken dan wel afstoten. Stippel de grafiek als de perioden labiel zijn.

7. (a) Laat zien dat de logistieke afbeelding

$$f_r : [0, 1] \mapsto [0, 1] : x \mapsto rx(1 - x), \quad r \in [0, 4]$$

   uitgebreid naar $\mathbb{C}$, en $r \in \mathbb{C}$, de Mandelbröt fractal voorbrengt. Bekijk zonodig eerst Veritasum's uitleg over het verband tussen de Mandelbröt fractal en het bifurcatiediagram van de logistieke afbeelding.



   Aanwijzing 1: de Mandelbröt fractal wordt normaliter gegenereerd door

$$g_c : \mathbb{C} \to\to \mathbb{C} : z \mapsto z^2 + c, \quad c \in \mathbb{C}.$$

   Probeer g om te zetten naar f door een geschikte substitutie van variabelen.

Aanwijzing 2: pas een lineaire substitutie $z = ax + b$ toe.

Aanwijzing 3:

$$\text{Als } z = ax + b \text{ dan } \begin{cases} x_{n+1} = rx_n(1 - x_n) \\ z_{n+1} = z_n^2 + c \end{cases} \Leftrightarrow \begin{cases} x_{n+1} = rx_n(1 - x_n) \\ ax_{n+1} + b = (ax_n + b)^2 + c \end{cases}.$$

Los a, b en c uit het rechter stelsel op. (Dit kan het makkelijkst door de geïsoleerde x_{n+1} uit de eerste vergelijking in de tweede vergelijking in te vullen. Laat de indices n vervolgens weg.)

(b) We willen $g_c : z \mapsto z^2 + c$ gebruiken als alternatieve representatie voor de logistieke afbeelding. Waartoe moet het domein van g_c dan worden beperkt, en waartoe moet c dan worden beperkt? Geef een functievoorschrift van g_c waar deze beperkingen tot uitdrukking komen.

Let op: hoewel het domein van f_r niet afhangt van r, hangt het het domein van g_c *wel* af van c.

Laat tot slot zien dat het gevonden functievoorschrift klopt door handmatig twee iteraties van f_3 met startwaarde $x_0 = 1/2$ te emuleren met g_c, en de gevonden z-waarden terug te vertalen naar x-waarden. Let op: de startwaarde z van g_c zal waarschijnlijk verschillen van de startwaarde van f_3. Dus waarschijnlijk $z_0 \neq 1/2$.

Concludeer / zie in dat $g_c : z \mapsto z^2 + c$, op een lineaire transformatie na, hetzelfde gedrag vertoont als de originele logistieke afbeelding. Voor welke c gaat g_c bijvoorbeeld chaotisch gedrag beginnen te vertonen? (Hint: zoek op internet naar: logistic map, onset to chaos, of naar rij A098587 in de On-Line Encyclopedia of Integer Sequences.)

8. (a) De *dyadische transformatie* wordt gegeven door

$$D : [0, 1] \to [0, 1] : x \mapsto \begin{cases} 2x, & x < 1/2, \\ 2x - 1, & \text{anders.} \end{cases}$$

Teken een grafiek van D. Geef ook duidelijk (met een open of dicht bolletje) aan wat D bij $x = 1/2$ doet.

(b) Het is makkelijker het gedrag van D te bestuderen als getallen binair geschreven worden. Voorbeelden:

$$\begin{array}{rcll} 5 & = & 1 \times 4 + 0 \times 2 + 1 \times 1 & = (101)_2 \\ 6 & = & 1 \times 4 + 1 \times 2 + 0 \times 1 & = (110)_2 \\ 7 & = & 1 \times 4 + 1 \times 2 + 1 \times 1 & = (111)_2 \\ 8 & = & 1 \times 8 + 0 \times 4 + 0 \times 2 + 0 \times 1 & = (1000)_2 \\ \frac{1}{2} & = & 1 \times \frac{1}{2} & = (0.1)_2 \\ \frac{3}{8} & = & 0 \times \frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{8} & = (0.011)_2 \\ \frac{27}{32} & = & \ldots & = (0.11011)_2 \end{array}$$

Schrijf $(10011)_2$ in het tientallig stelsel. Schrijf 0.90625 in het tweetallig stelsel. (Hint: $0.90625 = 29/32 = 16/32 + 13/32 = \ldots$.) Probeer dit eerst zonder online tools, anders krijg je geen inzicht.

(c) Geef nu, met behulp van een online number base converter binaire representaties van een aantal breuken, bijvoorbeeld 1/2, 1/3, 2/3, 1/5, 2/5, 3/5, en 4/5. Wat valt op in de binaire expansie? Hoe verklaar je dat? Hoe noteer je wat je ziet?

(d) Leg uit wat D doet op getallen in binaire representatie als deze kleiner of gelijk zijn aan een half. I.e., laat $x = (0.0b_2b_3b_4\ldots)_2$. (Immers, $x \leq 1/2$.) Hoe ziet $D(x)$ er uit?

(e) Leg uit wat D doet op getallen in binaire representatie als deze groter of gelijk zijn aan een half. I.e., hoe ziet $D(x)$ er uit als $x = (0.1b_2b_3b_4\dots)_2$?

(f) Beredeneer dat er voor elke $k \in \mathbb{N}$ wel een $x \in [0,1]$ bestaat waarvoor de rij

$$x,\ D(x),\ D(D(x)),\ D(D(D(x))),\ \dots$$

uiteindelijk in een cyclus met precies periode k belandt. (Bijvoorbeeld periode 6 en niet periode 5, 4, 3, 2, of 1.)

(g) Laat $P_k \subseteq [0,1]$, $k \in \mathbb{N}$ de verzameling punten zijn waarvoor D in een cykels met periode k belandt, $k \geq 1$. Beredeneer dat P_k dicht ligt in $[0,1]$. (We zeggen dat D bijna overal periodiek is.)

(h) Net zoals met decimalen is $(0.b_1b_2b_3b_4\dots)_2$ rationaal als en slechts als de rij $b_1b_2b_3b_4\dots$ eindig is, of zich in groepjes herhaalt. Dit hoef je verder niet te bewijzen.

Laat $A \subseteq [0,1]$ de verzameling punten zijn waarvoor D uiteindelijk in a-periodiek gedrag vervalt. Beredeneer dat A dicht ligt in het eenheidsinterval. Is A aftelbaar?

(i) Beredeneer dat er een punt x is waarvoor D mixt, dat wil zeggen dat de rij $x, D(x), D(D(x)), \dots$ dicht ligt in het eenheidsinterval. Hint: de verzameling eindige bit strings is aftelbaar en kan dus achter elkaar worden gezet.

(j) Laat $M \subseteq [0,1]$ de verzameling punten zijn waarvoor D mixt. Beredeneer dat M dicht ligt in het eenheidsinterval. Dus D mixt bijna overal.

(k) Beredeneer dat D extreem gevoelig kan zijn voor willekeurig kleine verstoringen in begin-waarden. (Het zg. vlinder-effect.)

9. (Dicht liggen in.) Laat $f : [0,1] \to [0,1]$ en $x_0 \in [0,1]$. Hoe kan geformuleerd worden dat de rij $x_0, f(x_0), f^2(x_0), f^3(x_0), f^4(x_0), \dots$ dicht ligt in $[0,1]$?

(a) Voor elke $p \in [0,1]$ en $\epsilon > 0$ is er een $n > 0$, zó dat $|\, f^n(x_0) - p \,| \leq \epsilon$.

(b) Voor elke $p \in [0,1]$ is er een $\epsilon \geq 0$ en een $n > 0$, zó dat $|\, f^n(x_0) - p \,| \leq \epsilon$.

(c) Er zijn een $p \in [0,1]$ en $\epsilon > 0$, zó dat voor elke $n > 0$ geldt dat $|\, f^n(x_0) - p \,| > \epsilon$.

(d) Er is een $p \in [0,1]$, zó dat voor elke $\epsilon > 0$ er een $n > 0$ is, zó dat $|\, f^n(x_0) - p \,| \leq \epsilon$.

10. (Quasi-periodieke functies.) Laat $[0,1]*$ het gesloten eenheidsinterval zijn, waarbij 1 geïdentificeerd is met 0 ("0 en 1 zijn aan elkaar geplakt"). Er ontstaat een cirkel. Een voorbeeld van een quasi-periodieke functie is de functie $f : [0,1]* \to [0,1]*$ met het voorschrift $x \to x + r \pmod 1$, waarbij de constante r een reëel irrationaal getal is.

(a) Beargumenteer dat f begrensd is.

(b) Bewijs (of beargumenteer) dat f continu is. (Als je een vak als calculus of analyse volgt, of hebt gevolgd, dan zou je dit moeten kunnen bewijzen. Als dat niet zo is, probeer dan te beargumenteren dat we kleine veranderingen in $f(x)$ kunnen garanderen door x zelf niet te veel te veranderen.

(c) Bereken n iteraties van f op x. Bereken n iteraties van f op $x + \epsilon$.

(d) Bewijs dat f geen dekpunt bezit.

(e) Bewijs dat f niet periodiek is.

(f) Bewijs (of beargumenteer) dat f niet gevoelig is voor perturbaties.

(g) Beargumenteer dat uit (10a-10f) zo ongeveer wel volgt dat f een quasi-periodieke functie is.

(h) Definieer een quasi-periodieke functie op de eenheidscirkel. Beargumenteer dat de functie inderdaad quasi-periodiek is. Het is hierbij mogelijk de antwoorden uit bovenstaande onderdelen in licht aangepaste vorm te hergebruiken.

11. Flake schrijft in Sec. 12.3 het volgende:

> *"If you think about how continuous dynamical systems work, with a point in an n-dimensional space representing the complete state of the system, then it should be clear that continuous chaos (as opposed to the discrete chaos found in the logistic or Hénon maps) requires three dimensions to flow through. Here is why. Suppose you had only two dimensions for the state space. For something to be chaotic, it not only must never repeat itself, but it also must return to a state that is very similar to a state that it has been at before. Since we are talking about continuous systems, we can represent the motion of the system by drawing a very long line on a sheet of paper. If you scribble long enough on some paper, it should become apparent that there is no way to meet these two constraints in only two dimensions because we are not allowed to intersect the line with itself; doing so would be to return to an old state, that is, to repeat. Hence, to have chaos, we really need to be able to "jump" over lines in a third dimension. This is why continuous chaos can exist only in three- or more dimensions."* (Flake, Sec. 12.3, p. 186)

Flake's observatie volgt uit de stelling van Poincaré en Bendixson. Deze stelling beweert het volgende:

> **Stelling (Poincaré en Bendixson).** Elke baan ("grafiek") van een continu differentieerbaar systeem van differentiaalvergelijkingen die binnen de grenzen van een compacte deelverzameling van $\mathbb{R}^2$ ligt,[44] benadert in de limiet een dekpunt of anders een gesloten omloop.

Elk dynamisch systeem zonder dekpunten zal dus in elke begrensde deelverzameling van $\mathbb{R}^2$ dus rondcirkelen. In het bijzonder sluit deze stelling uit dat de baan van een differentieerbaar systeem in een begrensde deelverzameling van $\mathbb{R}^2$ chaotisch gedrag zal vertonen. Kort gezegd stelt de stelling van Poincaré en Bendixson dat een dynamisch systeem in $\mathbb{R}^2$ nooit chaotisch gedrag zal kunnen vertonen.

Laten we de Stelling van Poincaré en Bendixson nu toepassen op de Tron Light Cycle game . De Tron Light Cycle game is een belangrijk computerspel uit de jaren '80.[45] Tron kan tegenwoordig nagespeeld worden op Mame[46] of in talloze platformversies van goede kwaliteit, zoals Armagetron, GLtron, en RevoTron.[47] Huidige versies van Tron zijn ook vaak terug te zien op mobiele telefoons en zijn, los daarvan, geschikt voor (korte) LAN[48] game sessies.

Stel, Tron wordt gespeeld door één individu, P, in het eenheidsvierkant $[0,1]^2$. Laat K de weg (spoor, of baan) zijn die wordt afgelegd door P.

[44] In $\mathbb{R}^2$ zijn compacte deelverzamelingen precies die deelverzamelingen die gesloten en begrensd zijn.

[45] In 1982 verscheen de film "Tron". Nu is dit een cultfilm. In december 2010 verscheen de film "Tron: Legacy," die ook in 3D werd uitgebracht.

[46] Multiple Arcade Machine Emulator.

[47] GLtron en RevoTron bezitten eigen Wikipedia pagina's, maar Armagetron lijkt op dit moment het best onderhouden en het meest populair.

[48] Local area network.

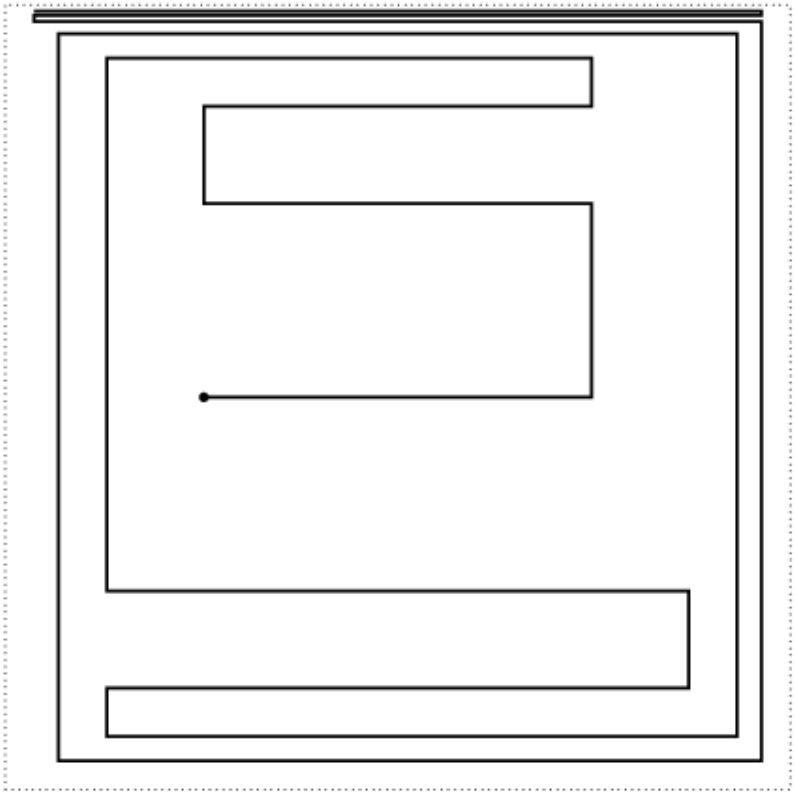


Fig. 22: "Chaotische" curve, K, in $[0,1]^2$.

(a) Stel, P moet altijd ϵ verwijderd blijven van zijn eigen spoor, waarbij $\epsilon > 0$. Kan P onbeperkt doorrijden, i.e., is K oneindig? Waarom (niet)? (Een informeel argument volstaat.)

(b) Stel P, mag zijn eigen spoor willekeurig dicht naderen (maar niet raken). Kan P onbeperkt doorrijden, i.e., is K oneindig? Waarom (niet)? Als K altijd eindig is, waarom zal P dan eindigen? Hoe zal dat dan gebeuren? Als K oneindig kan zijn, hoe zal K zich in dergelijke gevallen dan ontwikkelen? En welke eigenschappen zal de limiet-verzameling van K bezitten? (Een informeel antwoord volstaat.)

(c) Stel P, mag zijn eigen spoor willekeurig dicht naderen (maar niet raken). Iemand beweert dat P het vlak kan vullen door P een Peano-kromme[49] af te laten leggen. In dat geval zou $K = [0,1]^2$. Leg uit waarom deze bewering niet klopt.

12. Fig. 22 beeldt een continue curve, K, af, die bevat is in het eenheidsvierkant $[0,1]^2$. Laat $\eta \in (0,1)$ een willekeurig onberekenbaar getal zijn. Dat wil zeggen dat er geen computerprogramma bestaat dat η kan afdrukken.[50] De curve slingert in stappen die corresponderen met decimalen van η, als volgt. Eén stap is één lijnstuk. De n^{e} stap correspondeert met de n^{e} decimaal van η achter de komma. De curve slingert omhoog, daarbij de bovenkant naderend in afstanden van $1/2$, $1/4$, $1/8$, ..., tenzij in de n^{e} stap de n^{e} decimaal van η gelijk is aan 9. In dat geval beweegt de curve verticaal, van boven naar beneden, of, als de curve beneden is, van beneden naar boven, ook in afstanden van $1/2$, $1/4$, $1/8$, ... van de zijkanten, om vervolgens weer horizontale slagen te maken.

[49] Of een Hilbert-curve.

[50] Er kan bewezen worden dat $(0,1)$ dergelijke getallen bevat. Zie verder Hoofdstuk 1 "Computation" uit The Computational Beauty of Nature, en Hoofdstuk 9 van dit werkcollegecahier, dat ook over berekenbaarheid gaat.

De onberekenbaarheid van η zou er voor moeten zorgen dat K's baan chaotisch is. Dankzij de (aangepaste)[51] stelling van Poincaré en Bendixson weten we nu dat dit onmogelijk is.

(a) Stel, de decimale expansie van η bevat oneindig veel negens. Beschrijf de limiet-verzameling van K.

(b) (Ietsje moeilijker.) Stel, de decimale expansie van η bevat eindig veel negens, zeg 42. Beschrijf de limiet-verzameling van K. (Hint: gebruik het feit dat 42 even is.)

13. De *tent-afbeelding*, T,

– `http://www.google.nl/search?q=tent+map+wikipedia`
– `http://www.google.nl/search?q=tent+map+bifurcation+demo`
– `http://math.bu.edu/DYSYS/applets/nonlinear-web.html`,
– `http://math.fau.edu/MLogan/Pattern_Exploration/Chaos_Lab/BifFamily.html`

is, zoals dat heet, *topologisch geconjugeerd* aan de logistieke afbeelding. (Voor deze opgave is nu niet nodig te weten wat het betekent als een afbeelding topologisch geconjugeerd is aan een andere afbeelding.)[52] Een gevolg is dat de tent-afbeelding zich onder iteratie vergelijkbaar gedraagt als de logistieke afbeelding.

(a) Geef een functievoorschrift voor T. Gebruik voor de groeifactor het symbool c, en geef het domein van c.

(b) Bekijk het bifurcatiediagram van T via Fig. 23, of via één van de bovenvermelde applets. Wordt voor $c = 1.2$ bij itereren van T (i.e., het genereren van de rij $T(x), T(T(x)), T(T(T(x))), \dots$) de waarde $x = 0.53$ bereikt? Motiveer je antwoord met behulp van het bifurcatie-diagram.

(c) Laat zien (d.m.v. een berekening) dat voor $c = 1$ de functie T stabiel is. (Hint: laat zien dat T direct een dekpunt $T(x) = x$ bezit als $x \leq 1/2$, resp. na één iteratie op een dekpunt belandt als $x > 1/2$.)

(d) Laat zien (d.m.v. een berekening) dat voor $c \leq 1$ de functie T stabiel is. (Hint: laat zien dat itereren van T een strikt dalende rij oplevert als $x \leq 1/2$, en dat $T(x) \leq 1/2$ als $x > 1/2$.)

(e) Laat zien (d.m.v. een berekening) dat voor $c > 1$ de functie T chaotisch gedrag vertoont. (Hint: laat zien dat voor bijna elke $x \in [0, 1]$ het verschil tussen de beelden van x en $x + \epsilon$ voor $c > 1$ strikt groter is dan ϵ, en leg vervolgens uit waarom dit chaotisch gedrag impliceert.)

14. Het *bierspel* ("MIT's Beer Game", "Beer Distribution Game") is een voorraadketenbeheersimulatie (supply chain management simulation) die uitgevoerd kan worden door vier personen of software agents: de fabriek, de distributeur, de groothandel, en de retailer (winkel). De partijen leveren ook in die volgorde aan elkaar. Elke partij heeft als doel te voldoen aan de vraag naar bier van zijn afnemer. Om aan die vraag te voldoen vraagt elke partij op zijn beurt weer bier aan zijn leverancier. De vraag van klanten naar producten bij de retailer wordt gemodelleerd als een exogene factor (wordt buiten het systeem om door de simulatie bepaald). Het spel verloopt in tijdstappen van een week.

[51] Ook hier weer geldt dat Poincaré en Bendixson's stelling eigenlijk niet meteen kan worden toegepast, omdat Tron-sporen op sommige plaatsen geknikt is.

[52] Als je het toch wilt weten is dat alleen maar goed. Op de Engelstalige Wikipedia staat een mooie uitleg.

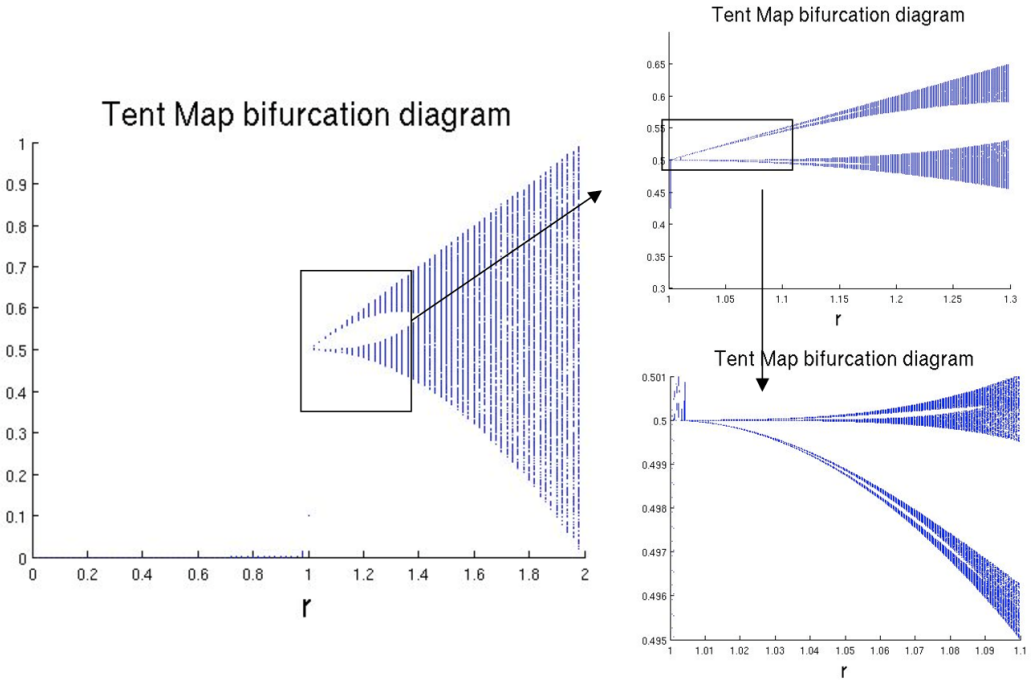


Fig. 23: Bifurcatiediagram van de tent-afbeelding.

- In The Computational Beauty of Nature besteedt Flake aandacht aan MIT's Beer Game.
- Een Netlogo simulatie is te vinden op `http://backspaces.net/models/beergame.html`.
- Een alternatief scenario voor de goederenstroom in de illegale drugshandel is te vinden op `http://backspaces.net/models/drugsupply.html`.

(a) Wat is de bedoeling van MIT's Beer Game?

(b) Wat is het verband tussen MIT's Beer Game en niet-lineaire dynamische systemen, zoals de logistic map?

(c) In `http://backspaces.net/models/beergame.html` kan de klant-vraag op verschillende manieren worden bepaald. Welke manieren zijn dat, en welke zouden in aanmerking kunnen komen voor het genereren van chaotisch gedrag? Waarom?

(d) Welke tegenmaatregelen of oplossingen heeft men in de praktijk voor voorraadschommelingen bedacht?

(e) Wat is bullwhip?

(f) Wat is just in time?

(g) Noem twee mechanismen voor het controleren van chaos. (Zie Flake, Ch. 13 en Slides Chaos en Populatiedynamiek.) Hoe werken deze twee mechanismen? Op welke onderdelen verschillen ze?

(h) Bespreek hoe mechanismen voor het controleren van chaos zich verhouden, of kunnen verhouden, tot MIT's Beer Game.

(i) Geef een korte omschrijving van het vak voorraadketenbeheer (Eng.: supply chain management). Noem enkele steekwoorden en/of sleutelbegrippen.

15. (a) Teken zelf een bi-furcatiediagram van de logistieke vergelijking met een programma dat kan plotten, zoals GNUplot, Mathematica, Maple, Octavia, Netlogo, of R. In de

aanloopfase wordt de logistieke functie geïtereerd maar niet getekend. Vanaf het eind van de aanloopfase (Eng.: transient phase) tot aan het totaal aantal iteraties worden punten getekend. Experimenteer met de lengtes van de aanloopfase en met het totaal aantal iteraties.

(b) De logistieke afbeelding en de tent map zijn beide chaotische afbeeldingen. Je kunt ze vloeiend in elkaar laten overlopen ("morphen") door de parameter β te laten overgaan van $\beta = 1$ (tent map) naar $\beta = 2$ (logistieke afbeelding):

$$f : [0,1] \to [0,1] : x \mapsto r\left(1 - |1 - 2x|^{\beta}\right), \quad 0 \leq r \leq 4,\ 1 \leq \beta \leq 2.$$

Construeer een geanimeerd bi-furcatiediagram door r op de x-as te zetten, de uitkomst na K iteraties op de y-as te zetten en β langzaam in de tijd te laten variëren.

Deel VIII. Appendices

A Wat handige wiskunde

A.1 Combinatoriek

Het basisprincipe van tellen. Als er twee experimenten worden uitgevoerd waarbij het eerste experiment m mogelijke uitkomsten heeft en het tweede experiment n mogelijke uitkomsten heeft, dan zijn er in totaal mn mogelijke uitkomsten.

Voorbeeld: er zijn $10 \cdot 10 \cdot 26 \cdot 26 \cdot 10 \cdot 10 = 6,760,000$ mogelijke nummerborden van het type CC-LL-CC, als C staat voor "cijfer", en L staat voor "letter".

Permutaties. Een aantal van n onderscheidbare objecten kunnen op $n! = 1 \times \cdots \times n$ verschillende manieren worden geordend.

Voorbeeld: de letters in het woord WINKEL kunnen op 6! verschillende manieren worden geordend: voor de W kunnen we kiezen uit zes plaatsen, voor de I kunnen we kiezen uit $6 - 1 = 5$ plaatsen, etc.

Voorbeeld: de letters van het woord EENDEN kunnen op

$$\frac{6!}{3!2!} = \frac{6 \cdot 5 \cdot 4}{2 \cdot 1} = 60$$

verschillende manieren worden geordend. Als we alle letters als verschillend beschouwen $E_1E_2N_1DE_3N_2$ zijn er 6! mogelijke ordeningen. De drie E's kunnen onderling op 3! verschillende manieren worden geordend, maar elk van die ordeningen komt op hetzelfde neer, want de E's zijn eigenlijk ononderscheidbaar. Dus delen we door 3!. Hetzelfde voor de N's.

Combinaties. Uit een groep van n onderscheidbare objecten kunnen

$$\binom{n}{k} = \frac{n!}{(n-k)!k!}$$

verschillende groepjes van k worden gekozen. Immers, als volgorde belangrijk is, kan het eerste element op n verschillende manieren worden gekozen, het tweede element op $n - 1$ verschillende manieren, etc., en het ke element op $n - (k - 1)$ verschillende manieren. Dus als volgorde belangrijk is, kan op $n!/(n - k)!$ verschillende manieren worden gekozen. Elk groepje komt in $k!$ volgordes voor. Dus om af te zien van volgorde delen we het aantal mogelijkheden door $k!$ en komen we op $n!/(n - k)!k!$ uit. (In de laatste notatie nemen we even aan dat vermenigvuldigen voor delen gaat.)

Voorbeeld: Een bestuur bestaat uit 3 leden. Uit een vereniging van 20 leden zijn dus $20!/(20 - 3)!3! = 1140$ mogelijke besturen te vormen.

Het verdelen van knikkers over vazen. Er zijn k^n mogelijke manieren om n onderscheidbare (genummerde) knikkers over k vazen te verdelen. Immers, voor elk van de n knikkers zijn er k keuzes. Als de knikkers er allemaal hetzelfde uitzien, dus ononderscheidbaar zijn, wordt het probleem moeilijker. Laten we eerst kijken hoeveel mogelijkheden er zijn als in de verdeling elke vaas tenminste één knikker moet bevatten. We leggen de n knikkers op een rijtje, en brengen $k - 1$ tussenschotjes aan, om de n knikkers te verdelen in k groepen. Voor $n = 10$ knikkers en $k = 4$ vazen kan een verdeling er bijvoorbeeld uitzien als

$$000 \mid 00 \mid 0000 \mid 0. \tag{13}$$

Er zijn $n-1$ plekken tussen n knikkers waar $k-1$ tussenschotjes kunnen worden geplaatst. Dit komt effectief neer op het kiezen van $k-1$ plekken uit $n-1$ plekken. Dus het aantal manieren om n ononderscheidbare knikkers te verdelen over k vazen waarbij geen enkele vaas leeg blijft is

$$\binom{n-1}{k-1}.$$

Als vazen wel leeg mogen blijven, mogen in zo'n configuratie als (13) schotjes direct naast elkaar liggen. Bijvoorbeeld

$$000 \,||\, 000000 \,|\, 0.$$

Hier is de tweede vaas leeg. Hoe tellen we al déze mogelijkheden? Wel, als we alle knikkers en schotjes bij elkaar optellen, komen we op $n+k-1$ uit: n knikkers en $k-1$ tussenschotjes. Maak vooralsnog een rijtje van evenzoveel, dus $n+k-1$, knikkers en verander daarna $k-1$ knikkers in schotjes. Dit experiment correspondeert nu 1-1 met het verdelen van n knikkers over k vazen waarbij vazen leeg mogen blijven. Dus het aantal verschillend mogelijkheden is gelijk aan

$$\binom{n+k-1}{k-1}.$$

Voorbeeld 1: iemand wil 20,000 Euro verdelen over 4 fondsen, in eenheden van 1000. Op hoeveel manieren kan dit? Antwoord: op

$$\binom{20+4-1}{4-1} = 1771$$

verschillende manieren.

Voorbeeld 2: 100 mieren verdelen zich over 10 holen. Hoeveel mogelijke toestanden geeft dit? Antwoord: dit geeft

$$\binom{100+10-1}{10-1} = 4,263,421,511,271$$

mogelijke toestanden. Vraag: als we alleen kijken of holen leeg zijn of niet, hoeveel mogelijke toestanden zijn er dan? Antwoord: voor elk hol zijn er twee mogelijkheden, te weten leeg of niet leeg. Er zijn 10 holen, dus het antwoord is nu: $2^{10}-1 = 1023$ mogelijke toestanden. Minus 1 omdat de toestand waarbij alle holen leeg zijn niet voor kan komen.

Zie verder [27].

A.2 De inverse van een matrix

Hoe je matrices bij elkaar optelt en met elkaar vermenigvuldigt, is hopelijk bekend. Maar hoe bepaal je de inverse van een matrix? Alleen vierkante matrices hebben een inverse, mits de determinant ervan ongelijk nul is. In dat geval geldt

$$A^{-1} = \frac{1}{\mathrm{Det}(A)}\mathrm{Adj}(A),$$

waarbij $\mathrm{Adj}(A)$ de zogenaamde geadjugeerde matrix van A is. Maar deze bereken je niet zo maar. Het berekenen van zowel de determinant (een reëel getal) als de geadjugeerde (een matrix) is een behoorlijk karwei.

Een relatief handzaam alternatief voor het bepalen van de inverse is door *Gauss-eliminatie* toe the passen op $(A \mid I)$ om $(I \mid A^{-1})$ te verkrijgen.

We gaan de inverse bereken van

$$A = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 2 \\ 0 & 2 & 4 \end{pmatrix}$$

1. We beginnen met $(A \mid I)$ op te schrijven.

$$\left(\begin{array}{ccc|ccc} 2 & 1 & 0 & 1 & 0 & 0 \\ 1 & 2 & 2 & 0 & 1 & 0 \\ 0 & 2 & 4 & 0 & 0 & 1 \end{array}\right)$$

Het is de bedoeling dat we links de eenheids-matrix maken. Dat kan door eerst het linker-benedengedeelte en dan het rechter-bovengedeelte "schoon te vegen". We werken van boven naar beneden door (i) de diagonaal op 1 te zetten (ii) een rij steeds een aantal malen af te trekken van de volgende rij. Dat laatste heet Gauss-eliminatie.

2. Diagonaal op 1 zetten. Rij 1 delen door 2:

$$\left(\begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 0 & 0 \\ 1 & 2 & 2 & 0 & 1 & 0 \\ 0 & 2 & 4 & 0 & 0 & 1 \end{array}\right)$$

3. Rij 1 aftrekken van Rij 2:

$$\left(\begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 3/2 & 2 & -1/2 & 1 & 0 \\ 0 & 2 & 4 & 0 & 0 & 1 \end{array}\right)$$

4. Diagonaal op 1 zetten. Rij 2 delen door 3/2:

$$\left(\begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 1 & 4/3 & 1/3 & 2/3 & 0 \\ 0 & 2 & 4 & 0 & 0 & 1 \end{array}\right)$$

5. Twee keer Rij 2 aftrekken van Rij 3:

$$\left(\begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 1 & 4/3 & 1/3 & 2/3 & 0 \\ 0 & 0 & 4/3 & -2/3 & -4/3 & 1 \end{array}\right)$$

6. Diagonaal op 1 zetten. Rij 3 delen door 4/3:

$$\left(\begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 1 & 4/3 & 1/3 & 2/3 & 0 \\ 0 & 0 & 1 & 1/2 & -1 & 3/4 \end{array}\right)$$

7. Nu kan het rechter-bovendeel schoongeveegd worden. Eén-één-derde van Rij 3 aftrekken van Rij 2:

$$\left(\begin{array}{ccc|ccc} 1 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 1 & 0 & -1 & 2 & -1 \\ 0 & 0 & 1 & 1/2 & -1 & 3/4 \end{array}\right)$$

8. Een half keer Rij 2 aftrekken van Rij 1:

$$\left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -1 & 1/2 \\ 0 & 1 & 0 & -1 & 2 & -1 \\ 0 & 0 & 1 & 1/2 & -1 & 3/4 \end{array}\right)$$

De inverse staat nu aan de rechterkant. De inverse van een matrix berekenen (zo die bestaat), blijft altijd bewerkelijk. Gauss-eliminatie is dan nog één van de minste bewerkelijke processen.

A.3 Een vector normaliseren of afkappen

Normaliseren

Als de lengte van een (niet-nul-) vector, zeg w, op, zeg, $M \geq 0$, moet worden gezet, dan kan dit middels

$$w_{\text{nieuw}} =_{Def} \frac{M}{\|w\|} w. \tag{14}$$

Na deze operatie geldt dus dat w_{nieuw} dezelfde kant op wijst als w, en dat $\|w_{\text{nieuw}}\| = M$.

1. Dat w_{nieuw} dezelfde kant op wijst als w, volgt uit het gegeven dat w_{nieuw} ontstaat uit w door het vermenigvuldigen met een scalar.

2. Dat $\|w_{\text{nieuw}}\| = M$ volgt eenvoudig:

$$\|w_{\text{nieuw}}\| = \left\| \frac{M}{\|w\|} w \right\| = \frac{M}{\|w\|} \|w\| = M.$$

Als w de nulvector is, en $M > 0$, dan kan w niet worden genormaliseerd. Immers, de nulvector heeft geen richting, dus waar zou w_{nieuw} met lengte M naar toe moeten wijzen? Dirt probleem vertaalt zich naar een verboden nuldeling in Eq. 14.

Als $\|w\| = 0$ én $M = 0$, dan is w_{nieuw} welgedefinieerd (want gelijk aan w), maar niet dankzij Eq. 14.

Afkappen

Het afkappen van w tot een lengte M is iets anders, dat kan ook. Afkappen betekent:

$$w_{\text{nieuw}} = \begin{cases} w & \text{als } \|w\| \leq M; \\ w \text{ genormaliseerd tot } M & \text{anders.} \end{cases}$$

Dit kan allemaal middels

$$w_{\text{nieuw}} =_{Def} \min\left\{ \frac{M}{\|w\|}, 1 \right\} w.$$

Na deze operatie geldt dus dat w_{nieuw} dezelfde kant op wijst als w, en dat $\|w_{\text{nieuw}}\| \leq M$.

1. Dat w_{nieuw} dezelfde kant op wijst als w, volgt uit het gegeven dat w_{nieuw} ontstaat uit w door het vermenigvuldigen met een scalar.

2. Dat $\|w_{\text{nieuw}}\| \leq M$ volgt eenvoudig. Gewoon de formule invullen, en vervolgens alle scalairen bij elkaar vegen:

$$\|w_{\text{nieuw}}\| = \left\| \min\left\{ \frac{M}{\|w\|}, 1 \right\} w \right\| = \min\left\{ \frac{M}{\|w\|}, 1 \right\} \|w\| = \min\left\{ \frac{M\|w\|}{\|w\|}, \|w\| \right\} = \min\{M, \cdot\} \leq M.$$

Referenties

[1] Scott Aaronson. The busy beaver frontier. *ACM SIGACT News*, 51(3):32–54, 2020.

[2] Noga Alon and Joel H Spencer. *The probabilistic method*. John Wiley & Sons, 2016.

[3] Claus Aranha, Christian L Camacho Villalón, Felipe Campelo, Marco Dorigo, Rubén Ruiz, Marc Sevaux, Kenneth Sörensen, and Thomas Stützle. Metaphor-based metaheuristics, a call for action: the elephant in the room. *Swarm Intelligence*, 16(1):1–6, 2022.

[4] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies–a comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

[5] Ben Brubaker. Amateur mathematicians find fifth "busy beaver" turing machine. *Quanta Magazine*, July 2, 2024.

[6] Christian L Camacho-Villalón, Marco Dorigo, and Thomas Stützle. An analysis of why cuckoo search does not bring any novel ideas to optimization. *Computers & Operations Research*, 142:105747, 2022.

[7] D. Cliff and G. F. Miller. Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations. In *Proc. of the Third European Conf. on Artificial Life*, LNCS 929, pages 200–218, San Francisco, California, USA, 1995. Springer-Verlag.

[8] Distinguished Lecturer in Parapsychology Daniel Maturana, Senior Ufologist Volology, David Fouhey, and Ghost Hunter. A spectral approach to ghost detection, 2013. Manuscript. The actual authors are D. Fouhey and D. Maturana from The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

[9] Herbert A. David and Haikady N. Nagaraja. *Order Statistics*. John Wiley & Sons, 2004.

[10] B. Drossel and F. Schwabl. Self-organized critical forest-fire model. *Phys. Rev. Lett.*, 69:1629–1632, 1992.

[11] Agoston E Eiben and James E Smith. *Introduction to evolutionary computing*. Springer, 2015.

[12] Diane L Evans, Lawrence M Leemis, and John H Drew. The distribution of order statistics for discrete random variables with applications to bootstrapping. *INFORMS Journal on Computing*, 18(1):19–30, 2006.

[13] Anahi Gajardo, Andre Moreira, and Eric Goles. Complexity of langton's ant. *Discrete Applied Mathematics*, 117(1-3):41–50, 2002.

[14] Tim J Hutton. Evolvable self-reproducing cells in a two-dimensional artificial chemistry. *Artificial life*, 13(1):11–30, 2007.

[15] Dervis Karaboga et al. An idea based on honey bee swarm for numerical optimization. Technical Report Technical report tr06, Erciyes university, engineering faculty, computer science, 2005.

[16] John R. Koza. *Genetic programming: On the programming of computers by means of natural selection*. Morgan Kaufmann, 1992.

[17] John R. Koza. *Genetic programming II: automatic discovery of reusable programs*. MIT press, 1994.

[18] John R. Koza. *Genetic programming II the next generation*. Video tape, MIT Press, 1994.

[19] John R. Koza. *Genetic programming III: Darwinian invention and problem solving*. Morgan Kaufmann, 1999.

[20] John R. Koza, Martin A Keane, Matthew J Streeter, William Mydlowec, Jessen Yu, and Guido Lanza. *Genetic programming IV: Routine human-competitive machine intelligence*. Springer, 2003.

[21] Christopher G Langton. Studying artificial life with cellular automata. *Physica D: nonlinear phenomena*, 22(1-3):120–149, 1986.

[22] Michael A Lones. Mitigating metaphors: A comprehensible guide to recent nature-inspired algorithms. *SN Computer Science*, 1(1):49, 2020.

[23] Pascal Michel. The busy beaver competition: a historical survey. *arXiv preprint arXiv:0906.3749*, 2009.

[24] Michael Mitzenmacher and Eli Upfal. *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge UP, 2017.

[25] Piergiorgio Odifreddi. *Classical Recursion Theory*. Elsevier, 1992.

[26] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[27] Sheldon Ross. *A first course in probability*. MacMillan, 10th edition, 1976.

[28] Hiroki Sayama. Swarm chemistry. *Artificial life*, 15(1):105–114, 2009.

[29] Thomas Schmickl, Martin Stefanec, and Karl Crailsheim. How a life-like system emerges from a simplistic particle motion law. *Scientific reports*, 6(1):37969, 2016.

[30] Hans-Paul Schwefel and Günter Rudolph. Contemporary evolution strategies. In *European Conference on Artificial Life*, pages 891–907. Springer, 1995.

[31] John Maynard Smith. Evolution and the theory of games. In *Did Darwin get it right? Essays on games, sex and evolution*, pages 202–215. Springer, 1982.

[32] Richard A. Watson and Jordan B. Pollack. Coevolutionary Dynamics in a Minimal Substrate. In Lee Spector, Erik D. Goodman, Annie Wu, W. B. Langdon, Hans M. Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund Burke, editors, *Proc. of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 702–709, San Francisco, California, USA, Juli 2001. Morgan Kaufmann.

[33] Jinran Wu, You-Gan Wang, Kevin Burrage, Yu-Chu Tian, Brodie Lawson, and Zhe Ding. An improved firefly algorithm for global continuous optimization problems. *Expert Systems with Applications*, 149:113340, 2020.

[34] Xin-She Yang. *Nature-inspired metaheuristic algorithms*. Luniver Press, 2nd edition, 2010.

B Antwoorden

1, p. 3: Schoonheid 1: schoonheid die de zinnen prikkelt ("ziet er mooi uit, klinkt goed, ruikt lekker, voelt goed aan"). Schoonheid 2: een, volgens Poincaré, diepgaander schoonheid die voortkomt uit de harmonieuze ordening van delen die samen een complexer geheel vormen.

2, p. 3: Wiskundigen, ingenieurs en informatici zijn, volgens Flake, te generaliseren naar, respectievelijk, theoretici, empiristen en, in de woorden van Flake: "simulationisten". Theoretici worden vaak gedwongen allerlei onrealistische aannamen te maken, empiristen worden vaak gedwongen om om te gaan met realistische omstandigheden, meetfouten en and praktische problemen. Volgens Flake heeft de simulationist met beide problemen te maken.

3, p. 3: Als het gesimuleerde gedrag lijkt op het gedrag van het complexe systeem, dan heeft de eenvoudige simulatie blijkbaar iets essentieels te pakken. Als we iets missen in het gesimuleerde gedrag, dan mist de simulatie juist iets essentieels.

4, p. 3: De metafoor van bomen in een bos: *"If this book is a forest, then the first way of reading it is akin to poking at individual trees. The second way is analogous to observing how nearby trees relate to each another. The third way would equate to standing back and taking in the whole forest at once."*

5, p. 3: Voor de persoon die hij tien of vijftien jaar geleden was.

Ik heb moeite gedaan het geboortejaar van Flake te achterhalen. Het is me niet gelukt. Gelet op zijn jaar van promoveren (1993) gok ik dat hij toen ongeveer 30 jaar oud zou kunnen zijn geweest. In 1997 (toen hij het boek schreef) zou hij dan dus 34 zijn. Min 10-15 jaar is ong. 19-24 jaar. Ik denk dat hij doelde op iemand van rond de twintig. Wat is jouw leeftijd?

(Update.) Op 24 April 2013 kreeg ik een mail van Gary Flake. Daarin schreef hij:

```
Van: Gary Flake [xxxx@xxxx.xxx] namens Gary William Flake [xxxx@xxxx.xxx]
Verzonden: woensdag 24 april 2013 22:39
Aan: Vreeswijk, G.A.W.
Onderwerp: Re: your year of birth

My target was my 15 year old self.  I started writing the book when I was
25 and finished it when I was about 30.

-- GWF
```

Als we zijn uitspraken en de informatie in zijn mail letterlijk nemen en combineren dan heeft hij dit book dus geschreven voor personen van 10 tot en met 25 jaar. Ik zou zeggen dat het boek het meest geschikt is voor personen vanaf, zeg, 16 jaar.

6, p. 3: De notie "eend" kan niet alleen worden begrepen door deze volledig te ontleden. We begrijpen de notie "eend" beter als we bestuderen hoe deze interacteert met zijn omgeving.

7, p. 3: Het volledig kunnen begrijpen en voorspellen van gedrag op individueel niveau garandeert niet dat de effecten van individueel gedrag kunnen worden voorspeld op de lange termijn. Ook garandeert het kunnen begrijpen en voorspellen van gedrag op individueel niveau niet dat individueel gedrag kan worden voorspeld in de aanwezigheid van andere individuen. Door interactie kan het gedrag van een individu dan ineens heel anders zijn. Een systeem van individuen gedraagt zich dus anders dan simpelweg een optelling van het gedrag van individuen.

8, p. 3: We can take a purely reductionist approach and attempt to understand things through dissection. We also can take a wider view and attempt to understand whole collections at once by observing how many agents, say the neurons in a brain, form a global pattern , such as human intelligence. Or we can take an intermediate view and focus attention on the interactions of agents. Through this middle path, the interactions of agents can be seen to form the glue that binds one level of understanding to the next level.

9, p. 3: *"All of these examples are emergent in that they contain simple units that, when combined, form a more complex whole."*

Eigen definitie: emergent gedrag van een complex systeem is gedrag dat niet te voorspellen is door alleen naar de onderdelen, de werking van hun onderdelen, of de interactie tussen hun onderdelen te analyseren.

De notie emergentie is overigens een controversieel begrip. Er is weinig overeenstemming over wat het nu precies is, en of het wel echt bestaat.

10, p. 3: Holisme en reductionisme.

11, p. 3: De natuur is zuinig. In het bijzonder zal de natuur zaken niet complexer maken dan nodig.

Zuinigheid of *parsimonie* is een basisprincipe in de wetenschappelijke methode. Volgens dit principe geldt dat als twee verklaringen plausibel zijn, de eenvoudigste (de verklaring waarvoor de minste aannames nodig zijn, ofwel het meest 'parsimone') de voorkeur heeft. Dit is precies wat Einstein zegt: *"Things should be as simple as possible, but not simpler."*.

De middeleeuwse Engelse wijsgeer William of Ockham (ook wel: Occam) zegt ook zoiets. Ockham stelt: *Entia non sunt praeter necessitatem multiplicanda:* "Men moet de zijnden (gepostuleerde objecten binnen een hypothese) niet zonder noodzaak verveelvoudigen". Dit principe staat bekend als "Occam's razor". Bijvoorbeeld de verklaring dat de opwarming van de aarde wordt veroorzaakt door antropogene broeikasgasemissies heeft op grond van Occam's razor de voorkeur boven de verklaring dat deze wordt veroorzaakt door een combinatie van zonneactiviteit, meetfouten en emissies door vulkanen als we geloven dat de eerste verklaring berust op minder aannamen.

14, p. 3: Parallellisme, recursie (incl. iteratie en feedback) en adaptatie (incl. leren en evolutie).

15, p. 4: *"The simplest type of question that we can ask about an interaction is what will X do next, in the presence of Y ? Notice that this question has a functional, algorithmic, or even computational feel to it, in that we are concerned not with "What is X ?" but with "What will X do?". In this respect, describing an interaction is very similar to discovering nature's "program" for something 's behavior.* ***The goal of this book is to highlight the computational beauty found in nature's programs****."*

16, p. 4: 1. *Het geheel is groter dan de som der delen.*

2. *Het meest interessante spul zit in het midden.*

3. *Wetenschap is gedoemd tot onzekerheid—maar dat is goed.* Is ook gerelateerd aan de zg. "no free lunch theorem".

17, p. 4: Computers helpt de wetenschap weer meer interdisciplinair te worden

> *"In many ways, the computer has reversed the trend toward fragmentation and has helped the sciences converge to a common point."*

Veel wetenschappelijk disciplines proberen gedrag van natuurkundige, biologische, economische en sociale system te begrijpen of inzichtelijker te maken door ze simuleren door middel van reductionistisch gedragsregels. Schijnbaar onbegrijpelijk gedrag van een complex systeem kan zo soms toch kan worden begrepen door dit te herleiden naar het simpele gedrag van eenvoudige individuen.

18, p. 4: The volgende woorden kunnen voorkomen in een woordenlijst van Hoofdstuk 1: adaptive, agents, algorithmic, algorithmically, algorithms, annealing, classifier, clusters, cognitive, complexity, computability, computable, computation, computational, computationally, connectionist, deterministic, emergent, fragmentation, holism, hybrid, inanimate, incomputable, indescribable, iterated, iterating, iteration, iterative, metaphor, multiagent, nonlinear, parallel, parallelism, pathological, proliferation, recurrent, recurrently, recursion, reductionism, reductionist, redundant, reinforced, reinforcement, self-similar, sequential, subatomic, subdiscipline, subdivision, symbiotic, transition, universal, unpredictable.

1, p. 6: Nederlandse samenvatting van Hoofdstuk 2 (Number Systems and Infinity) van *The computational Beauty of Nature* (Flake, 1998):

2. Het is mogelijk op (tenminste) twee manieren naar de computer te kijken: (1) als canvas waarop de programmeur zijn eigen werkelijkheid creëert door te programmeren (2) als medium, of bedding, waardoor programma's hun weg banen net zoals geluid zich zich een weg baant door de lucht.

2.1. Zeno's paradox illustreert wat er gebeurt met afstanden en getallen wanneer er in de oneindigheid over geredeneerd wordt. Zeno's paradox is een opstap voor de rest van het hoofdstuk.

2.2. Veel getalverzamelingen, bijvoorbeeld de verzameling van derde-machten, kunnen in 1-1 correspondentie worden gebracht met de verzamelingen van natuurlijke getallen. Dus er zijn bijvoorbeeld evenveel derde-machten als natuurlijke getallen. Dergelijke verzamelingen zijn *aftelbaar oneindig*.

2.3. De verzameling van rationale getallen (breuken) is ook aftelbaar oneindig.

2.4. Wortel twee is een irrationaal getal. De verzameling irrationale getallen is overaftelbaar oneindig. Het interval $(0, 1)$ bevat evenveel getallen als $\mathbb{R}$. Flake betoogt dat het geheugen van elke computer kan worden gerepresenteerd door een rationaal getal tussen de nul en één, met eindig veel nullen en enen als decimalen.

2a, p. 7: Een verzameling heet *eindig* dan en slechts dan als deze eindig veel elementen bevat .... Dit is een circulaire definitie! Hier hebben we niet veel aan. Laten we het nog eens proberen.

> *Een verzameling heet* eindig *dan en slechts dan als deze leeg is of precies één element meer bevat dan een andere eindige verzameling.*

De verzameling $\{1, 2, 3\}$, bijvoorbeeld, is eindig, want bevat één element meer dan $\{1, 3\}$, en de laatste verzameling is eindig. (Het element 2 is willekeurig gekozen.) De verzameling $\{1, 3\}$ is eindig omdat het één element meer bevat dan $\{3\}$, en de laatste verzameling is eindig. De verzameling $\{3\}$ is eindig omdat het één element meer bevat dan $\emptyset$, en de laatste verzameling is per definitie (per afspraak) eindig.

Een andere manier om eindigheid te definiëren is te zoeken naar eigenschappen van oneindige verzamelingen die eindige verzamelingen niet hebben. Eén zo'n eigenschap is dat een oneindige verzameling altijd deelverzamelingen bezit die gelijkmachtig zijn met de originele verzameling, denk aan $\mathbb{N} \subset \mathbb{Z}$. Dus in deze benadering definiëren we eerst de notie van een oneindige verzameling als een verzameling die een echte deelverzameling heeft die gelijkmachtig is met de originele verzameling. Vervolgens definiëren we een eindige verzameling als een een verzameling die niet oneindig is.

2b, p. 7: Hier is een definitie van een oneindige verzameling:

> Een verzameling heet *oneindig* als deze niet eindig is.

Dit lijkt flauw, maar het is een prima definitie. Immers, de notie "eindige verzameling" was immers al gedefinieerd.

2c, p. 7: Een verzameling heet *aftelbaar* als alle elementen op een (aan één kant mogelijk oneindige) rij kunnen worden gezet. Formeler heet een verzameling aftelbaar als $\mathbb{N}$ kan worden gebruikt om die verzameling volledig te nummeren. (Het is toegestaan nummers uit $\mathbb{N}$ niet te gebruiken. Bij eindige aftelbare verzamelingen is dit altijd het geval.)

Preciezer heet een verzameling X aftelbaar als deze leeg is, $X = \emptyset$ of er een surjectie $s : \mathbb{N} \to X$ bestaat. Jammergenoeg moet het geval $X = \emptyset$ apart worden genoemd, omdat een surjectie van $\mathbb{N}$ naar een lege set niet mogelijk is. (Ga na.) Een andere (maar gelijkwaardige) definitie van aftelbaarheid van X is het bestaan van een injectie $s : X \to \mathbb{N}$. Voor verzamelingen A en B geldt namelijk dat, als er een surjectie $s : A \to B$ bestaat, er ook een injectie $i : B \to A$ bestaat, en omgekeerd. Of dit voor randgevallen $A = \emptyset$ en/of $B = \emptyset$ ook nog op gaat, weet ik nu even niet, maar ook hier zouden deze gevallen in de definitie apart kunnen worden genomen om van alle problemen af te zijn `:-)`.

2d, p. 7: Een verzameling heet *aftelbaar oneindig* als deze aftelbaar en oneindig is.

2e, p. 7: Een verzameling heet *overaftelbaar* als deze niet aftelbaar is. (Termen als "niet aftelbaar" en "onaftelbaar" worden gek genoeg bijna nooit gebruikt.)

3a, p. 7: Misschien is het 't gemakkelijkst te begrijpen als we beginnen de kardinaalgetallen in volgorde op te noemen. Daadwerkelijk aftellen kan niet, want de verzameling kardinaalgetallen is niet aftelbaar, i.e., de verzameling kardinaalgetallen is overaftelbaar. Hier gaat 'ie (met wat suggestie door puntjes)

$$0, 1, 2, \ldots, 100, \ldots, \aleph_0, \aleph_1, \aleph_2, \ldots, \aleph_{100}, \ldots$$

We kunnen verder gaan met $\aleph_\omega, \aleph_{\omega+1}, \ldots$ (ω is het eerste telgetal na de natuurlijke getallen) maar voor nu is het genoeg. Tot aan $\aleph_0$ zijn de kardinaalgetallen eindig (voorbeelden: 6, 8, 3454) of aftelbaar (enige voorbeeld: $\aleph_0$). Antwoord: alles vanaf $\aleph_1$.

Dus hoewel er precies één aftelbaar kardinaalgetal is, zijn er meerdere en in feite overaftelbaar veel overaftelbare kardinaalgetallen. Dus het kardinaalgetal $|\mathbb{R}|$ (met de continuümhypothese gelijk aan $\aleph_1$) is niet het enige overaftelbare kardinaalgetal.

3b, p. 7: Alleen het begrip "aftelbaar oneindig" staat in direct verband met een kardinaalgetal, te weten $\aleph_0$ ("Aleph-nul").

Het begrip "overaftelbaar oneindig" representeert niet één kardinaalgetal maar een hele hoop, onder andere die van $\mathbb{R}$, $\mathcal{P}(\mathbb{R}) = 2^{\mathbb{R}}$, $\mathcal{P}(\mathcal{P}(\mathbb{R})) = \mathcal{P}^2(\mathbb{R})$, $\mathcal{P}^3(\mathbb{R})$, etc. Om precies te zijn zijn er overaftelbaar veel overaftelbaarheden, dat wil zeggen, overaftelbaar veel overaftelbare kardinaalgetallen. Dat laatste hoef je nu niet verder te onthouden of te begrijpen, maar ik wil het hier wel even hebben gezegd omdat het rijtje $\mathbb{R}$, $\mathcal{P}(\mathbb{R})$, $\mathcal{P}^2(\mathbb{R})$, $\mathcal{P}^3(\mathbb{R})$, ... onterecht aftelbaarheid suggereert. Er kan namelijk verder worden geteld dan $1, 2, 3, \ldots$, te weten met zogenaamde *ordinaalgetallen* (d.w.z. telgetallen):

$$1, 2, 3, \ldots, \omega, \omega+1, \omega+2, \omega+3, \ldots, 2\omega, 2\omega+1, 2\omega+2, 2\omega+3, \ldots$$

3c, p. 7: Onder $\aleph_1$:

$$\begin{array}{l} 0\ , 1, 2, \ldots, 100, \ldots,\ \aleph_0\ ,\ \aleph_1\ , \aleph_2, \ldots, \aleph_{100}, \ldots, \\ |\emptyset|,\ \ ,\ \ ,\quad ,\quad ,\quad , |\mathbb{N}|, |\mathbb{R}|,\quad ,\quad ,\quad , \end{array}$$

4a, p. 7: 3840×2160.

4b, p. 7: Eindig, maar dat is geen kardinaliteit.

Preciezer: als van elk object op aarde eenduidig kan worden vastgesteld of het een zandkorrel is of niet, dan is de kardinaliteit van de verzameling zandkorrels op aarde een natuurlijk getal n, waarbij we n kunnen schatten (iets als 7.5×10^{18}) maar vermoedelijk nooit exact kunnen bepalen.

4c, p. 7: $|\mathbb{R}|$. Het makkelijkste antwoord is gewoon strepen om $\mathbb{R}$ heen zetten, want $\mathbb{R}$ is qua kardinaliteit zo'n beetje per conventie vertegenwoordiger van alle verzamelingen gelijkmachtig met ("even groot als") $\mathbb{R}$. Dus gewoon strepen om $\mathbb{R}$ heen zetten mag.

Het antwoord $\mathfrak{c}$ is eigenlijk ietsje netter, want fraktur c is speciaal in het leven geroepen om de kardinaliteit van het continuüm te representeren. Het antwoord $2^{\aleph_0}$ is ook goed, want $|\mathbb{R}| = 2^{\aleph_0}$. Het antwoord $\aleph_1$ is goed als de continuümhypothese $\aleph_1 = |\mathbb{R}|$ wordt geaccepteerd. Het antwoord "overaftelbaar" is fout, want "overaftelbaar" is geen kardinaalgetal.

4d, p. 7: Het kardinaalgetal $|\mathbb{R}|$ ofwel $\mathfrak{c}$. Het antwoord $|(0,1)|$ is goed maar een beetje makkelijk en communiceert niets. Alle vragen in deze opgave zouden dan makkelijk zijn te beantwoorden door gewoon rechte strepen om de verzameling heen te zetten. Beter is aan te geven dat $|(0,1)|$, net zoals bij zo veel overaftelbare verzamelingen, toch weer gelijk is aan $|\mathbb{R}|$. Dit kan bewezen worden met een bi-jectie tussen $(0,1)$ en $\mathbb{R}$, bijvoorbeeld $x \mapsto (2x-1)/(1-|2x-1|)$—of een andere functie die $\mathbb{R}$ 1-1 comprimeert naar $(0,1)$.

4e, p. 7: $|\mathbb{R}|$. De in de opgave gesuggereerde vervlechting levert een bi-jectie door het eenheidsvierkant (dat gelijkmachtig is met $\mathbb{R} \times \mathbb{R}$) af te beelden op $(0, 1)$. Onder bepaalde redelijke voorwaarden (het zogenaamde *keuzeaxioma*) geldt eigenlijk voor elke oneindige verzameling X dat $X \times X \sim X$. Het bewijs daarvan valt buiten het bestek van dit dictaat.

5a, p. 7: De machtigheid van elk open interval is gelijk aan $|\mathbb{R}|$. Dit kan het makkelijkst met behulp van de stelling van Schröder-Bernstein . Een injectie van $\mathbb{R}$ naar $I(\mathbb{R})$: $x \mapsto [x, x + 1)$. Een injectie van $I(\mathbb{R})$ naar $\mathbb{R} \times \mathbb{R}$ naar $\mathbb{R}$:

het interval $(x, y) \mapsto$ het paar $(x, y) \mapsto$ de vervlechting van x en y.

5b, p. 7: $|\mathbb{R}|$. Eén manier om dit te laten zien is de volgende. Omdat $\{1, 2, 3, 4\} \subseteq \mathbb{R}$ geldt $|\mathbb{R} \times \{1, 2, 3, 4\}| \leq |\mathbb{R} \times \mathbb{R}| = |\mathbb{R}|$. En $|\mathbb{R}| = |\mathbb{R} \times 1| \leq ||\mathbb{R} \times \{1, 2, 3, 4\}|$.

5c, p. 7: $|\mathbb{R}|$. Elk gesloten interval $[a, b]$ correspondeert met vier intervallen met dezelfde eindpunten, te weten $[a, b]$ zelf en dan nog $[a, b)$, $(a, b]$ en (a, b). En $|\mathbb{R} \times \{1, 2, 3, 4\}| = |\mathbb{R}|$.

5d, p. 7: $|\mathbb{R}| = \mathfrak{c}$, dus $|2^{\mathbb{R}}|{=}2^{|\mathbb{R}|} = 2^{\mathfrak{c}}$. Niet $\aleph_2$, ook niet met de continuümhypothese, die slechts aanneemt dat $2^{\aleph_0} = \aleph_1$. Wel met de veralgemeniseerde continuümhypothese GCH, die aanneemt (of stipuleert) dat $2^{\aleph_\alpha} = \aleph_{\alpha+1}$ voor elk ordinaalgetal (elk telgetal) α.

5e, p. 7: De *verzameling* van alle open intervallen in $\mathbb{R}$ correspondeert 1-1 met het open halve vlak

$$H = \{(x, y) \in \mathbb{R} \times \mathbb{R} \mid x < y\}.$$

Dus H is overaftelbaar. Omdat $H \subseteq \mathbb{R} \times \mathbb{R}$ en al aangetoond is dat $\mathbb{R} \times \mathbb{R} \sim \mathbb{R}$, is H dus gelijkmachtig met een deelverzameling van $\mathbb{R}$. Met aanname de continuümhypothese geldt dat elke overaftelbare deelverzameling van $\mathbb{R}$ gelijkmachtig is met $\mathbb{R}$. Dus met aanname de continuümhypothese geldt dat $H \sim \mathbb{R}$, dus $|H| = |\mathbb{R}|$.

6a, p. 8: Tel $\mathbb{Z}$ af als: $0, 1, -1, 2, -2, 3, -3, 4, -4, \ldots$. Zo komen alle gehele getallen aan de beurt.

6b, p. 8: Er zijn meerdere bi-jecties tussen $\mathbb{N}$ en $\mathbb{Z}$ te construeren. Als we de bi-jectie in het antwoord van Onderdeel 6a willen reconstrueren, dus als we de bi-jectie

$$\begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \ldots \\ \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \ldots \\ 0 & 1 & -1 & 2 & -2 & 3 & -3 & 4 & -4 & 5 & -5 & \ldots \end{array}$$

willen reconstrueren, dan luidt het bijbehorende fucntievoorschrift als volgt: beeld alle even getallen, n, af op hun helft, en beeld alle oneven getallen, n, af op $(1 - n)/2$:

$$f : \mathbb{N} \to \mathbb{Z} : n \mapsto \begin{cases} n/2, & n \text{ is even,} \\ (1 - n)/2 & \text{anders.} \end{cases}$$

Dus even getallen worden op positieve getallen afgebeeld, en oneven getallen worden op niet-positieve (nul en negatieve) getallen afgebeeld. Deze functie is surjectief: laat $z \in \mathbb{Z}$ willekeurig. Als $z > 0$, dan is $2z \in \mathbb{N}$ een origineel van z: $f(2z) = 2z/2 = z$. Als $z \leq 0$, dan is $-2z + 1 \in \mathbb{N}$ een origineel van z:

$$f(-2z + 1) = \frac{1 - (-2z + 1)}{2} = z.$$

7, p. 8: Laat T de verzameling derde-machten zijn. Dus

$$T = \{\, t \in \mathbb{Z} \mid \text{er is een } z \in \mathbb{Z} \text{ zodanig dat } z^3 = t \,\}.$$

(Korter kan ik T helaas niet schrijven.) We kunnen proberen een surjectie $s : \mathbb{N} \to T$ te geven, of we kunnen proberen een injectie $i : T \to \mathbb{N}$ te geven. Het eerste lijkt het makkelijkst. Definieer:

$$s : \mathbb{Z} \to T : n \mapsto n^3.$$

We bewijzen eerst dat s surjectief is. Laat $t \in T$ willekeurig. Volgens de definitie van T is er dus een $z \in \mathbb{Z}$ zodanig dat $z^3 = t$. (Nu komt die lange definitie van T toch goed van pas!) De functie s is dus surjectief. We hadden al bewezen dat $\mathbb{Z}$ aftelbaar is. We mogen dus aannemen dat er een surjectie $s' : \mathbb{N} \to \mathbb{Z}$ bestaat. De compositie $s \circ s' : \mathbb{N} \to T$ is surjectief (want een samenstelling van surjecties is ook weer surjectief). Daarmee zijn we klaar.

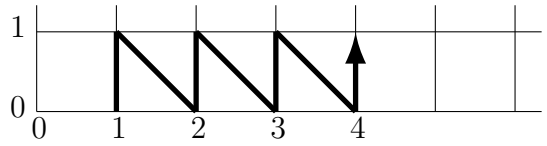
8a, p. 8: Zet op een rij: $0, 1, 2, 3, 4, 5, \ldots$. De functie $f : \mathbb{N} \to \mathbb{N} \cup \{0\} : n \mapsto n - 1$ is de bijbehorende surjectie.

8b, p. 8: Zet op een rij:

$$-m, -m+1, \ldots, -1, 0, 1, 2, 3, 4, 5, \ldots.$$

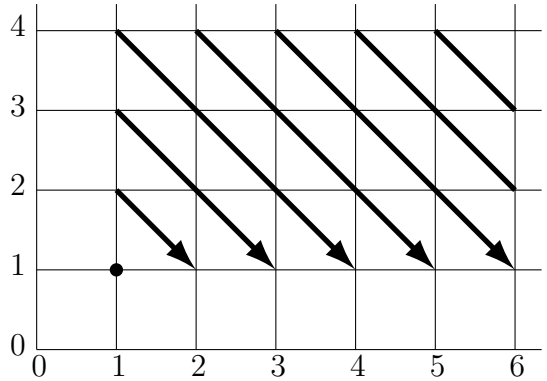
De functie $f : \mathbb{N} \to \{-m, -m+1, -m+2, \ldots, -2, -1, 0\} \cup \mathbb{N} : n \mapsto n - m - 1$ is de bijbehorende surjectie. Van $n - m$ moet nog 1 worden afgetrokken omdat het kleinste getal in $\mathbb{N}$ gelijk is aan 1.

9a, p. 8: Informeel kan dat worden aangetoond door een continue lijn te trekken die alle elementen van $\mathbb{N} \times \{0, 1\}$ aandoet. Bijvoorbeeld

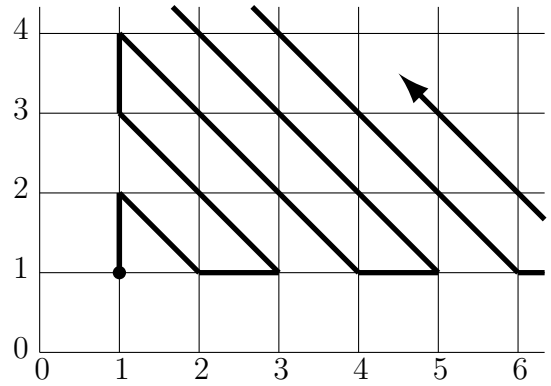


Deze lijn correspondeert met (of althans suggereert) de aftelling $(1, 0)$, $(1, 1)$, $(2, 0)$, $(2, 1)$, $(3, 0)$, $(3, 1)$, ...

9b, p. 8: Met losse streken:



Of door middel van één haal:



10, p. 8: Het begrip dubbele aftelbaarheid voegt niets nieuws toe, want komt op hetzelfde neer als aftelbaarheid: $\ldots, a_{-3}, a_{-2}, a_{-1}, a_0, a_1, a_2, a_3, \ldots$ kan bijvoorbeeld worden herordend als $a_0, a_1, a_{-1}, a_2, a_{-2}, a_3, a_{-3}, \ldots$.

11, p. 8: – 4: ja.

– n ($n \in \mathbb{N} \cup \{0\}$): ja.

– Eindig: nee, immers twee eindige verzamelingen kunnen een verschillend aantal elementen bevatten.

– Aftelbaar: nee, immers zowel eindige als aftelbaar oneindige verzamelingen zijn aftelbaar.

– $\mathbb{N}$: nee, dit is een verzameling.

– $|\mathbb{N}|$: ja, we spreken over de kardinaliteit van $\mathbb{N}$ (of gelijk welke verzameling dan ook tussen de rechte strepen staat).

– aftelbaar oneindig: ja, dit zijn alle verzamelingen gelijkmachtig met $\mathbb{N}$.

– $x, x \geq 0$: nee, een verzameling kan geen niet-gehele kardinaliteit bezitten.

– $\mathbb{R}$: nee, dit is een verzameling.

– $|\mathbb{R}|$: ja (zie $|\mathbb{N}|$).

– Overaftelbaar oneindig: nee, deze term geeft slechts aan dat een verzameling niet aftelbaar oneindig is. Twee overaftelbare verzamelingen kunnen een verschillende kardinaliteit bezitten, bv. $\mathbb{R}$ en $2^{\mathbb{R}}$. De stelling van Cantor zegt immers $|X| < |2^X|$ voor elke verzameling X.

– Ontelbaar: is geen officiële term. Ligt nog het dichtst bij niet-telbaar, dus: overaftelbaar. Dus geen kardinaliteit.

12a, p. 8: A is aftelbaar, en bezit dus een aftelling. Maak een aftelling van X door A opnieuw af te tellen en de elementen van X daarbij te "vergeten". Deze nieuwe aftelling is moeilijk in een formule te zetten. We doen het daarom niet.

We tonen aftelbaarheid van X anders aan door een injectie $X \to \mathbb{N}$ te geven. Dat is makkelijk: een injectie $A \to \mathbb{N}$ definieert meteen een injectie $X \to \mathbb{N}$ middels de zg. inbeddingsfunctie "$\to_i$" die zelf injectief is: $X \to_i A \to \mathbb{N}$. De samenstelling van twee injectieve functies is ook weer injectief.

12b, p. 8: Een aftelling g van $A \cup B$ ontstaat nu door eerst $B - A$ af te tellen en te vervolgen met de aftelling van de A. Als $B - A$ k elementen heeft en f een aftelling van A is, dan kan g formeel als volgt worden gedefinieerd:

$$g(n) = \begin{cases} \text{het } n^e \text{ element van } B - A & \text{als } n < k \\ f(n-k) & \text{als } n \geq k \end{cases}$$

12c, p. 8: (Vergelijk opdracht 3.6) Laat f een aftelling zijn van A, en g een aftelling van B. Definieer nu een aftelling h van $A \cup B$ als volgt: neem om en om het volgende element van A (volgens aftelling f) dat je nog niet bent tegengekomen en het volgende nieuwe element van B (volgens g).

12d, p. 8: Er geldt $A \cap B \subseteq A$, en A is aftelbaar. Een deelverzameling van een aftelbare verzameling is aftelbaar.

12e, p. 8: Laat a_0, a_1, a_3, ... een aftelling zijn van A, en b_0, b_1, b_2, b_3, ... van B. Dan kunnen de elementen van $A \times B$ weer als volgt worden gerangschikt:

$$\begin{array}{cccccc} \langle a_0, b_0 \rangle & \langle a_0, b_1 \rangle & \langle a_0, b_2 \rangle & \langle a_0, b_3 \rangle & \cdots \\ \langle a_1, b_0 \rangle & \langle a_1, b_1 \rangle & \langle a_1, b_2 \rangle & \langle a_1, b_3 \rangle & \cdots \\ \langle a_2, b_0 \rangle & \langle a_2, b_1 \rangle & \langle a_2, b_2 \rangle & \langle a_2, b_3 \rangle & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{array}$$

Tel weer af als in de breuken-procedure van Cantor.

12f, p. 8: Aftelling f van A is als volgt om te zetten in een aftelling h van $A \backslash B$: h is gelijk aan f, behalve dat de elementen uit B worden overgeslagen. Dit is trouwens nog niet zo heel makkelijk in een nieuw functievoorschrift $g : A \backslash B \to \mathbb{N}$ te vertalen. Maar in principe kan het.

13, p. 9: Allemaal door herhaald toepassen van het resultaat van de vorige opgave. Officieel gaat dit met volledige inductie naar het aantal gebruikte verzamelingstheoretische operatoren.

14, p. 9: Vereniging is verzamelingen achter elkaar zetten, doorsnede is deelverzameling-argument.

15, p. 9: Vereniging is met Cantor-veegmethode, doorsnede is deelverzameling-argument.

16a, p. 9: Overaftelbaarheid Cartesisch product is te bewijzen middels diagonalisatie-argument op de verzameling $\prod_{i=1}^{\infty} \{0, 1\}$.

16b, p. 9: Het Cartesisch product

$$\prod_{i=1}^{\infty} A_i$$

is aftelbaar onder de voorwaarde dat bijna alle (= alle op eindig veel na) A_i's leeg zijn of singleton. Dit zijn voldoende voorwaarden voor aftelbaarheid. Of dit ook noodzakelijke voorwaarden zijn is niet nagegaan.

17a, p. 9: Cantor's diagonaalmethode toepassen. Stel, er is wèl een aftelling van de verzameling van oneindige bitstrings. Laat (15) dan zo'n aftelling zijn.

$$\begin{array}{ll} 1. & 01110101110111011010\ldots \\ 2. & 11101100010101101011\ldots \\ 3. & 00011011100001011011\ldots \\ 4. & 10100110110100101100\ldots \\ 5. & 00010101110100110101\ldots \\ 6. & 01010101011010100101\ldots \\ & \ldots \end{array} \tag{15}$$

Licht de diagonaal uit

$$\begin{array}{ll} 1. & \mathbf{0}1110101110111011010\ldots \\ 2. & 1\mathbf{1}101100010101101011\ldots \\ 3. & 00\mathbf{0}11011100001011011\ldots \\ 4. & 101\mathbf{0}0110110100101100\ldots \\ 5. & 0001\mathbf{0}101110100110101\ldots \\ 6. & 01010\mathbf{1}01011010100101\ldots \\ & \ldots \end{array}$$

(De diagonaal lijkt scheef, maar dat is omdat cijfers hier hoger dan breder zijn.) Neem nu het negatief van de diagonaal:

$$101110\ldots \tag{16}$$

Deze oneindige bitstring kan niet voorkomen in (15), want elk n-de element van de aftelling verschilt van (16) op het n-de bit. Dus (16) kan niet voorkomen in (15). Maar dan was (15) toch geen complete aftelling van de verzameling van oneindige bitstrings. Omdat de aftelling verder willekeurig was, geldt dit diagonaalargument voor elke mogelijk aftelling. Er bestaat dus geen complete aftelling van de verzameling van oneindige bitstrings.

17b, p. 9: Omdat oneindige bitstrings 1-op-1 corresponderen met deelverzamelingen van $\mathbb{N}$.

17c, p. 9: Elke functie van $\mathbb{N}$ naar $\mathbb{N}$ correspondeert 1-op-1 met een oneindige rij van natuurlijke getallen. Met Cantor's diagonaalmethode kan nu makkelijk worden aangetoond dat de verzamelingen van dergelijke oneindige rijen niet aftelbaar kan zijn.

18a, p. 9: Elke deelverzameling van $\mathbb{N}$ correspondeert 1-op-1 met een oneindige bitstring. De verzameling van priemgetallen, bijvoorbeeld, is te schrijven als $011010100010100010\ldots$. Dus geldt $2^{\mathbb{N}} \sim \{0,1\}^{\mathbb{N}}$. Met een diagonaalargument is nu makkelijk aan te tonen dat $\{0,1\}^{\mathbb{N}}$ overaftelbaar is.

18b, p. 10: Beschouw eerst de echte beginstukken van $\mathbb{N}$. Noem $B_0 = \{0\}$, $B_1 = \{0,1\}$, en in het algemeen $B_i = \{0,1,\ldots,i\}$. Hieruit is als volgt een aftelinstuctie voor de eindige deelverzamelingen van $\mathbb{N}$ te destilleren: neem eerst de deelverzamelingen van B_0: dat levert $\emptyset$ en kijk vervolgens naar de deelverzamelingen van B_1: neem al die deelverzamelingen op die nog niet in de aftelling aanwezig zijn. Herhaal dit procédé voor elke volgende B_i (voor $i \geq 1$ levert dit 2^i

nieuwe deelverzamelingen op.)

$$\begin{array}{lrcl} B_0: & 0 & \longrightarrow & \emptyset \\ & 1 & \longrightarrow & \{0\} \\ B_1: & 2 & \longrightarrow & \{1\} \\ & 3 & \longrightarrow & \{0,1\} \\ B_2: & 4 & \longrightarrow & \{2\} \\ & 5 & \longrightarrow & \{0,2\} \\ & 6 & \longrightarrow & \{1,2\} \\ & 7 & \longrightarrow & \{0,1,2\} \\ & & \vdots & \end{array}$$

Een korte recursieve beschrijving van deze aftelling f is:

- $f(0) = \emptyset$
- $f(n) = f(n - 2^k) \cup \{k\}$ als $2^k \leq n < 2^{k+1}$

19, p. 10: Laat f de aftelling zijn van alle eindige deelverzamelingen van $\mathbb{N}$ uit de vorige opgave. Definieer nu een aftelling g van de co-finiete deelverzamelingen van $\mathbb{N}$ door: $g(X) = f(\mathbb{N} - X)$ voor elke co-finiete X.

20, p. 10: Drie dingen controleren: (i) $A \sim A$, (ii) als $A \sim B$ dan $B \sim A$, en tenslotte (iii) als $A \sim B$ en $B \sim C$, dan $A \sim C$. Welnu, eigenlijk alle drie de eigenschappen volgen gemakkelijk. Laten we de derde eigenschap controleren: stel $A \sim B$ en $B \sim C$. Te bewijzen $A \sim C$. Omdat $A \sim B$ en $B \sim C$, bestaan er bi-jecties $f : A \to B$ en $g : B \to C$. Nu is gemakkelijk te controleren dat $g \circ f : A \to C$, met $g \circ f(x) =_{Def} g(f(x))$ een bi-jectie is. Injectiviteit: stel $g(f(x)) = g(f(x'))$. Vanwege injectiviteit g volgt $f(x) = f(x')$. Vanwege injectiviteit f volgt $x = x'$. Surjectiviteit: Stel $z \in C$. Omdat g surjectief is bestaat er een $y \in B$ zo dat $g(y) = z$. Omdat f surjectief is bestaat er een $x \in A$ zo dat $f(x) = y$. Maar dan $g(f(x)) = g(y) = z$. Dus is $g \circ f$ ook surjectief.

21a, p. 10: Bi-jectie: $f : [0, 2] \to [0, 1] : x \mapsto x/2$.

21b, p. 10: Dezelfde bi-jectie: $f : [0, 2] \to [0, 1] : x \mapsto x/2$.

21c, p. 10: Bi-jectie: $f : [0, 2) \to (0, 1] : x \mapsto 1 - x/2$. De bi-jectie uit de vorige onderdelen kan niet worden gebruikt.

21d, p. 10: Bi-jectie: $f : (0, 1) \to (-1, 1) : x \mapsto 2x - 1$. Dit is inderdaad een bi-jectie, want er kan een inverse geconstrueerd worden.

21e, p. 10: Bi-jectie: $f : \mathbb{R}^+ \to (0, 1) : x \mapsto 1/(1 + x)$. Dit is inderdaad een bi-jectie, want er kan een inverse $g : (0, 1) \to \mathbb{R}+$ geconstrueerd worden: $g : (0, 1) \to \mathbb{R}^+ : x \mapsto 1/x - 1$.

21f, p. 10: Bi-jectie: $f : \mathbb{R}^+ \to (-1, 1) : x \mapsto x/(1 + |x|)$. Er zijn andere mogelijkheden, bv. met arctangens. Het voordeel van de arctangens is dat deze makkelijk te inverteren is: de inverse van arctangens is natuurlijk tangens (op $[-\pi/2, \pi/2]$ en nul daarbuiten).

22a, p. 10: Deze twee verzamelingen zijn gelijkmachtig. Het is moeilijk meteen een directe bi-jectie te geven. Makkelijker is het om de stelling van Schröder-Bernstein toe te passen. Deze zegt dat twee verzamelingen A en B gelijkmachtig zijn zodra er twee injecties $i_1 : A \to B$ en $i_2 : B \to A$ bestaan. De volgende twee afbeeldingen volstaan

i_1: $[0.1) \to [0, 1]$: $x \mapsto x$ (makkelijk)
i_2: $[0, 1] \to [0.1)$: $x \mapsto x/2$ (krimp, zodat het beeld binnen het co-domein blijft)

Controleer zelf dat beide afbeeldingen injectief zijn, en dat hun bereik inderdaad binnen het co-domein (de verzameling waar de afbeelding naar toe mag gaan) blijft. Met andere afbeeldingen kan het ook, maar dit zijn denk ik de meest voor de hand liggende.

22b, p. 10: Meteen in te zien, maar het geven van concrete injecties vereist "een beetje mikken".

i_1: $[-1, 2) \to [3, 400]$: $x \mapsto x + 4$ (-1 op of boven de 3 tillen)
i_2: $[3, 400] \to [-1, 2)$: $x \mapsto x/400$ (een groot interval krimpen; daarna is verschuiven niet meer nodig)

Controleer zelf dat beide afbeeldingen injectief zijn, en dat hun bereik inderdaad binnen het co-domein (de verzameling waar de afbeelding naar toe mag gaan) blijft. Met andere afbeeldingen kan het ook, maar dit zijn denk ik de meest voor de hand liggende.

22c, p. 10:

i_1: $[0.1] \to \mathbb{R}$ $: x \mapsto x$ (makkelijk)
i_2: $\mathbb{R} \to [0.1]$: $x \mapsto 1/2 + \arctan(x)/\pi$ (pas het bereik van arctan aan op $(0, 1)$)

22d, p. 10: Intuïtief is meteen in te zien dat dit waar is: krimp het eenheidsvierkant, en stop het in de eenheidsschijf. Omgekeerd, krimp de eenheidsschijf en stop het in het eenheidsvierkant. Moeilijker is het concrete injecties te geven. Maar dit gaat lukken als beide figuren overdreven worden gekrompen. Het is dus toch mogelijk om een vierkante pin in een rond gat te passen! (Vgl. de Engelse uitdrukking "to fit a square peg in a round hole".)

i_1: $I^2 \to D^2$: $(x, y) \mapsto (x - 1/2, y - 1/2)$ (verschuif het vierkant tot binnen de schijf)
i_2: $D^2 \to I^2$: $(x, y) \mapsto (x/2 + 1/2, y/2 + 1/2$ (verklein de schijf; verschuif hem daarna)

22e, p. 10: Intuïtief is meteen in te zien dat dit waar is: beiden zijn massieve voorwerpen in 3D. Voor een injectie van de snaar naar de doughnut krimpen we de snaar totdat 'ie doughnut past. Als we voldoende krimpen hoeft de snaar zelfs niet te worden gebogen of opgerold. Voor een injectie van de doughnut naar de snaar krimpen we de doughnut tot 'ie zo klein is dat 'ie in (een gedeelte van) de snaar past. Dat de doughnut een gat heeft maakt dan verder niet uit.

Veel en veel moeilijker is het om concrete injecties te geven. Eigenlijk is dat hier ook niet nodig, omdat een beschrijving al voldoende zou moeten overtuigen.

23, p. 11: Repareer de laatste fout door $1/2$ in het co-domein op een nieuw getal, zeg $1/3$, binnen $[0, 1)$ af te beelden. Nu is er een nette bi-jectie $1 \leftrightarrow 1/2$ en $1/2 \leftrightarrow 1/3$. Het probleem is nu dat $1/3$ in het domein niet meer kan worden afgebeeld op $1/3$ in het co-domein, immers $1/3$ in het co-domein wordt al geclaimd door $1/2$: $f(1/2) = 1/3$. Bij de tweede reparatie hebben we dus een nieuwe fout geïntroduceerd. De situatie herhaald zich. Door oneindig lang fouten te repareren wordt de gewenste bi-jectie gecreëerd.

Anders gezegd: laat $S = \{1, 1/2, 1/3, 1/4, \dots\}$. Beeld S af op, respectievelijk, 1/2, 1/3, 1/4, 1/5, ... Noem de laatste verzameling S'. Dus $S' \subset [0, 1)$. Tussen S en S' is nu een 1-1 correspondentie aangelegd: $1 \leftrightarrow 1/2$, $1/2 \leftrightarrow 1/3$, $1/3 \leftrightarrow 1/4$, etc.

De clou is nu dat de restanten $[0, 1]\backslash S$ en $[0, 1)\backslash S'$ gelijk zijn (ga na!), dus tussen de restanten bestaat een natuurlijke 1-1 correspondentie. De twee 1-1 correspondenties samenvoegen geeft de gevraagde bi-jectie.

Opmerking: er kan bewezen worden dat het onmogelijk is een bi-jectie aan te leggen die zich wat netter gedraagt dan de hier gevonden bi-jectie. Preciezer geformuleerd kan worden aangetoond dat er geen continue bi-jectie kan bestaan tussen $[0, 1$ en $[0, 1)$. We gaan hier niet verder op in.

24, p. 11: Voor het gemak werken we eerst het geval af dat $A = \emptyset$. In dit geval is $|A| = |\emptyset| = 0$ en $|2^A| = |\{\emptyset\}| = 1$, dus de stelling geldt.

Voor niet lege A laten we twee dingen zien. In de eerste plaats: 2^A heeft minstens dezelfde machtigheid als A. In de tweede plaats: A en 2^A hebben niet dezelfde machtigheid.

Het eerste is gemakkelijk: de functie $f : A \to 2^A$ die wordt gedefinieerd door $f(a) = \{a\}$, voor elke $a \in A$, is een injectie. Het tweede gaat als volgt. We nemen aan dat een bi-jectie F tussen A en 2^A gegeven is, en we construeren een deelverzameling B van A die geen F-beeld is van enig element in A. De constructie van $B \subseteq A$ gaat als volgt. Kies

$$B = \{b \in A \mid b \notin F(b)\}.$$

B is geen F-beeld van enig element van A. Veronderstel namelijk van wel. Neem aan dat we hebben: $F(c) = B$, voor zekere $c \in A$. Zit c in B? Stel van wel. Dan volgt uit de definitie van B dat $c \notin B$. Stel van niet. Dan volgt uit de definitie van B dat $c \in F(c)$, dat wil zeggen $c \in B$. Beide mogelijkheden leiden dus tot een tegenspraak. Uit het feit dat B geen F-beeld is van enig element van A volgt dat we mogen concluderen dat er geen bi-jectie tussen A en 2^A bestaat.

25a, p. 11: Iedere programmeertaal J (bijv. Java) is opgebouwd uit een alphabet A (bv. ASCII). Nu geldt dat elk J-programma een string is, opgebouwd uit elementen van A, i.e, $J \subseteq A^+$, waarbij A^+ gelijk is aan alle niet-lege strings opgebouwd uit A, i.e.,

$$A^+ = \{\, ax \mid a \in A, x \in \{A^+, \epsilon\}\},$$

waarbij ϵ de lege string is. Het is dus voldoende te bewijzen dat A^+ aftelbaar is. Laat $A^n \subseteq A^+$ alle strings zijn ter lengte n. Dan is A^n eindig en

$$A^+ = \bigcup_{n=1}^{\infty} A^n.$$

Om A af te tellen sommen we eerst alle elementen van A^1 op: `a, b, c, d,`.... Dat zijn er eindig veel, dus op enig moment zijn we klaar met het opsommen van A_1. Dan alle elementen van A^2: `aa, ab, ac, ad,`..., `ba, bb, bc, bd,`..., dan alle elementen van A^3: `aaa, aab, aac, aad,`..., dan alle elementen van A^4, enzovoort. Als programma $j \in J$ bijvoorbeeld n karakters lang is, dan zal j verschijnen in de aftelling van A^n.

1a, p. 12: Waar, want er zijn inderdaad slechts eindig veel uitzonderingen, te weten 1 t/m 10.

1b, p. 12: Onwaar, want er blijven nog steeds oneindig veel getallen over die niet even zijn.

1c, p. 12: Waar, want er zijn inderdaad slechts eindig veel uitzonderingen—eigenlijk maar één uitzondering, te weten het getal 2.

1d, p. 12: Waar. Er zijn hoogstens 201×201, dus eindig veel, vereenvoudigd breuken met een absolute waarde kleiner dan of gelijk aan 100.

1e, p. 12: Onwaar. Van 1 alleen al zijn er oneindig veel onvereenvoudigde breuken, te weten 2/2, 3/3, 4/4, ...

1f, p. 13: Er van uitgaande dat er altijd eindig veel mensen zijn, kunnen we geen antwoord geven, want de definitie in het kader gaat pas gelden als we spreken over eindig veel uitzonderingen in een aftelbare verzameling.

Toch wordt het voorbeeld "bijna alle mensen hebben drie oren" (of een soortgelijk voorbeeld) vaak aangehaald om de kracht van het begrip "bijna alle" te illustreren.

1g, p. 13: Waar. De enige niet-trouwe ("ontrouwe") getallen blijken 1, 2, 3, 4, 5, 6, 8, 12, en 24. FWIW.

Waarom zo'n relatief vergezocht voorbeeld? Omdat van bekendere voorbeelden de verzameling van eindige uitzonderingen bijna altijd[53] meteen is aan te wijzen. En hier dus niet.

2a, p. 13: Waar, want de verzameling van rationale getallen, $\mathbb{Q}$, is aftelbaar.

2b, p. 13: Waar, want de verzameling van algebraïsche getallen, $\mathbb{A}$, is aftelbaar.

2c, p. 13: De verzameling van alle functies van $\mathbb{N}$ naar $\mathbb{N}$ is een overaftelbare verzameling. Elke functie correspondeert namelijk 1-1 met een oneindige rij natuurlijke getallen, en van de verzameling van oneindige rijen van natuurlijke getallen is makkelijk in te zien dat deze overaftelbaar is. (Gebruik Cantor's diagonaalargument.)

De verzameling van functies met een functievoorschrift is daarentegen aftelbaar. Immers, elke functie met een functievoorschrift is te schrijven als een string, en de verzameling van mogelijke strings uit een vast alfabet is aftelbaar.

Om dezelfde reden zullen we later zien dat nagenoeg alle elementen uit $\mathbb{N}^{\mathbb{N}}$ (= alle functies van $\mathbb{N}$ naar $\mathbb{N}$) niet programmeerbaar zijn.

2d, p. 13: Waar, want de verzameling roosterpunten is aftelbaar.

2e, p. 13: Waar, het is de dubbele negatie van het vorige onderdeel. Dus naast "bijna alle" bestaat ook nog "bijna geen enkele", "praktisch geen", "nagenoeg geen", etc.

2f, p. 13: Niet waar volgens deze definitie, want de verzameling van punten op alle roosterlijnen is niet aftelbaar. Het antwoord "niet waar" gaat (hoop ik) tegen je intuïtie in. Dit probleem lossen we op in Opgave 8b op pagina 15.

[53] In de informele zin van het woord.

2g, p. 13: Niet waar volgens deze definitie, want de verzameling van punten op de schijf in de oorsprong met straal 10 is niet aftelbaar. Ook dit antwoord is tegenintuïtief. Dit wordt opgelost in Opgave 8c op pagina 15.

3, p. 13: De uitkomstenruimte bestaat uit bitstrings die aan één kan oneindig doorlopen:

0101011011010110110101 01 . . .
0101010101101101010111 01 . . .
1010101110101010101011 01 . . .
1011101001101010111010 10 . . . . . .

Deze verzameling is gelijkmachtig met $\mathbb{R}$. Dit kan worden bewezen door een 1-1 correspondentie aan te brengen tussen, bijvoorbeeld, alle oneindige bitstrings en alle deelverzamelingen van de natuurlijke getallen. Van de laatste verzameling weten we dat deze gelijkmachtig is met $\mathbb{R}$.

4a, p. 13: 2^{10}.

4b, p. 13: Deze verzameling is aftelbaar:

000000 . . .
1
01
001
0001
00001
. . .

Maak niet de fout door zo "af te tellen":

1
01
001
0001
00001
. . .
000000 . . .

De string met allemaal nullen komt dan nooit aan de beurt.

5a, p. 14: Deze uitkomstenruimte bevat precies 2^{10} elementen: met één worp 2 mogelijkheden (kop of munt), met twee worpen $2^2 = 4$ mogelijkheden (kop, kop; kop, munt; munt, kop; munt, munt), met tien worpen 2^{10} mogelijkheden (kop, kop, kop, kop, kop, . . .).

5b, p. 14: 2^n, $n \in \mathbb{N} \cup \{0\}$. Merk op dat voor $n = 0$ het antwoord ook goed is. Immers, als er niet wordt geworpen is de enige uitkomst een leeg rijtje ϵ. dus de uitkomstenruimte is dan gelijk aan $U = \{\epsilon\}$.

5c, p. 14: Hoeveel uitkomsten (muntstukrijen) zijn er mogelijk in een experiment waarin een onbepaald eindig aantal keer een muntstuk wordt geworpen?

Het antwoord is: aftelbaar oneindig veel. We weten vooraf niet hoeveel keer de muntstuk zal worden geworpen. Iemand kan $0\times$ werpen, dat zijn eindig veel ($2^0 = 1$) mogelijkheden. Iemand kan $1\times$ werpen. Dat zijn eindig veel (2) mogelijkheden. Iemand kan $2\times$ werpen. Dat zijn eindig veel (2^2) mogelijkheden. Zo doorgaand kunnen al deze mogelijkheden achter elkaar worden gezet. Op deze manier krijgen we een aftelbaar oneindige rij van realisaties, waarbij elke realisatie uiteindelijk ook aan de beurt komt.

Het antwoord "2^n, voor een vaste n" is niet goed. Er wordt duidelijk aangegeven dat het niet uitmaakt hoeveel keer de muntstuk wordt geworpen.

5d, p. 14: Hoeveel uitkomsten (muntstukrijen) zijn er mogelijk in een experiment waar iets of iemand gevraagd wordt herhaaldelijk een muntstuk werpen, totdat kop verschijnt?

Deze verzameling, laten we hem U noemen, bevat aftelbaar oneindig veel elementen, we kunnen ze namelijk allemaal op een rij zetten, zonder er één te vergeten.

munt, munt, munt, munt, munt, munt, munt, munt, munt, munt, ...
kop
munt, kop
munt, munt, kop
munt, munt, munt, kop
munt, munt, munt, munt, kop
...

De plek van de oneindige rij "munt, munt, munt, ..." (i.e., nooit kop) is willekeurig gekozen. Een aftelling waarbij deze op een andere plek staat is ook prima, bijvoorbeeld

kop
munt, kop
munt, munt, kop
munt, munt, munt, munt, munt, munt, munt, munt, munt, munt, ...
munt, munt, munt, kop
munt, munt, munt, munt, kop
...

Het volgende is echter onzin

kop
munt, kop
munt, munt, kop
munt, munt, munt, kop
munt, munt, munt, munt, kop
... munt, munt, munt, munt, munt, munt, munt, munt, munt, munt, ...

Het is dan namelijk niet duidelijk wanneer de oneindige rij "munt, munt, munt, ..." aan de beurt komt.

5e, p. 14: De uitkomstenruimte, laten we deze U noemen, bestaat uit oneindig lange strings die zijn opgebouwd met karakters uit $D = \{k, m\}$, waarbij "k" staat voor "kop" en "m" staat voor "munt".

$kkmmkmkmmkmmmkmmmkm\ldots$
$mmmkkmkmkmkkmkkmkmkm\ldots$
$mmkkkmkmkmkmkkmkkkmmk\ldots$
$kkmkkmmkkkkmmkkkkkmmkk\ldots$
$kmmkmkmkmmkmkmmkkkmm\ldots$
$\ldots$

U is overaftelbaar omdat diagonalisatie kan worden toegepast, en de kardinaliteit van U is gelijk aan gelijk aan die van $\mathbb{R}$. Het bewijs van dit laatste verloopt analoog aan het bewijs van het antwoord op Item 6d op pagina 14.

6a, p. 14: Deze uitkomstenruimte bevat precies 6^{10} elementen: met één worp 6 mogelijkheden, met twee worpen $6^2 = 36$ mogelijkheden, met tien worpen 6^{10} mogelijkheden.

6b, p. 14: 6^n, $n \in \mathbb{N} \cup \{0\}$. Merk op dat voor $n = 0$ het antwoord ook goed is. Immers, als er niet wordt geworpen is de enige uitkomst een leeg rijtje ϵ. dus de uitkomstenruimte is dan gelijk aan $U = \{\epsilon\}$.

6c, p. 14: Hoeveel uitkomsten (dobbelsteenrijen) zijn er mogelijk in een experiment waarin een onbepaald eindig aantal keer een dobbelsteen wordt geworpen?

Het antwoord is: aftelbaar oneindig veel. We weten vooraf niet hoeveel keer de dobbelsteen zal worden geworpen. Iemand kan $0\times$ werpen, dat zijn eindig veel ($6^0 = 1$) mogelijkheden. Iemand kan $1\times$ werpen. Dat zijn eindig veel (6) mogelijkheden. Iemand kan $2\times$ werpen. Dat zijn eindig veel (6^2) mogelijkheden. Zo doorgaand kunnen al deze mogelijkheden achter elkaar worden gezet. Op deze manier krijgen we een aftelbaar oneindige rij van realisaties, waarbij elke realisatie uiteindelijk ook aan de beurt komt. Het is belangrijk dat laatste op te merken!

Het antwoord "6^n, voor een vaste n" is niet goed. Er wordt duidelijk aangegeven dat het niet uitmaakt hoeveel keer de dobbelsteen wordt geworpen.

6d, p. 14: Hoeveel realisaties (dobbelsteenrijen) zijn er mogelijk in een experiment waar iets of iemand gevraagd wordt herhaaldelijk een dobbelsteen werpen, totdat een zes verschijnt?

Deze verzameling, laten we hem U noemen, bevat overaftelbaar oneindig veel elementen.

- Ten eerste alle eindige rijtjes die eindigen in een zes. Laten we deze verzameling V noemen. Makkelijk is in te zien dat deze verzameling aftelbaar is: tel eerst alle rijtjes ter lengte 1 af, dat is er één, nl. de eerste keer meteen een zes gooien. Tel dan alle rijtjes ter lengte 2 af, dat zijn er $5 \cdot 1 = 5$, nl. eerst geen zes, en dan wel een zes. Tel dan alle rijtjes ter lengte 3 af, dat zijn er $5^2 \cdot 1 = 25$, nl. de eerste twee keren geen zes en dan wel een zes. Enzovoort. Op die manier komen alle elementen uit U aan de beurt.
- Ten tweede alle oneindige rijen zonder zes (overaftelbaar), W. Immers, bij een dergelijk experiment is het mogelijk dat de zes nooit verschijnt. (De kans op zo'n gebeurtenis is echter nul.)[54]

[54] Iets kan met kans 1 plaatsvinden zonder noodzakelijk te zijn; iets kan met kans 0 plaatsvinden maar toch mogelijk zijn.

Op deze manier is U de disjuncte vereniging van V en W.

De overaftelbaarheid W, en daarmee U, kan worden aangetoond door diagonalisatie, maar daarmee is alleen nog maar overaftelbaarheid aangetoond. I.h.b. is de kardinaliteit van U daarmee nog niet bepaald.

Bewering: $U \sim \mathbb{R}$. Dit kan worden aangetoond door te laten zien dat

$$\mathbb{R} \sim [0,1] \sim Q' \sim W'$$

waarbij W' de verzameling is van alle dobbelsteenrijen uit W die niet eindigen in $555\ldots$, waarbij Q de verzameling is van representaties van elementen uit $[0,1]$ in het vijftallig stelsel, en waarbij Q' de verzameling is van elementen uit Q die niet eindigen in $444\ldots$. Op deze manier heeft elk element uit $[0,1]$ een unieke representatie in Q', immers doublures zoals $0.1444\cdots = 0.2$ komen niet voor in Q'. De relatie $\mathbb{R} \sim [0,1]$ mag als bekend worden verondersteld (op college behandeld en geoefend op werkcollege). De relatie $Q' \sim W'$ geldt omdat dobbelsteenrijen die niet eindigen in $555\ldots$ 1-1 corresponderen met elementen uit Q' door van alle cijfers één af te trekken en er dan een nul en een punt voor te zetten. Tenslotte geldt $W' \sim W \sim U$ omdat alle drie de verzamelingen oneindig zijn en er van links naar rechts steeds een aftelbare verzameling bijkomt. (I.h.a.: als X oneindig en Y aftelbaar, dan geldt $X \cup Y \sim X$, omdat een aftelling van een aftelbare $X' \subset X$ kan worden gemerged met een aftelling van Y en dan weer in X kan worden teruggestopt.)

Twee maal een injectie construeren van en naar een verzameling X waarvan we weten dat die gelijkmachtig is met $\mathbb{R}$, en dan verwijzen naar de stelling van Schröder-Bernstein, is ook goed. Voorwaarde is dan wel dat beide injecties dan ook echt tussen X en U lopen, of dan op toch op z'n minst tussen X en W'.

Normering toen dit een inleveropgave was: totaal 4 punten; 1 punt voor de observatie dat er faal- en succesrijen zijn, 1 punt voor overaftelbaarheid middels diagonaalargument, 1 punt voor het opmerken dat de kardinaliteit gelijk is aan die van $\mathbb{R}$, 1 punt voor een bewijs of bewijsschets van de kardinaliteit.

6e, p. 14: De uitkomstenruimte, laten we deze U noemen, bestaat uit oneindig lange strings die zijn opgebouwd met karakters uit $D = \{1,2,3,4,5,6\}$:

$$\begin{array}{l} 143546452442452264635434\ldots \\ 624244424211645244536163\ldots \\ 213413416313564352346311\ldots \\ 432413323163331643413342\ldots \end{array}$$

U is overaftelbaar omdat diagonalisatie kan worden toegepast. Preciezer, de kardinaliteit van U is gelijk aan gelijk aan die van $\mathbb{R}$. Het bewijs van dit laatste verloopt analoog aan (maar is niet hetzelfde als) het bewijs in de uitwerking van het antwoord van het vorige onderdeel. (Nu met strings met karakters uit $D = \{1,2,3,4,5,6\}$.)

Normering toen dit een inleveropgave was: totaal 3 punten; 1 punt voor overaftelbaarheid middels diagonaalargument, 1 punt voor het opmerken dat de kardinaliteit gelijk is aan die van $\mathbb{R}$, 1 punt voor een bewijs of bewijsschets.

7a, p. 14: Niet waar: van succesvolle realisaties zijn er "slechts" aftelbaar veel, namelijk $\{1,2,3,4,5\}^*6$, en van mislukkende realisaties zijn er overaftelbaar veel, namelijk $\{1,2,3,4,5\}^\infty$. Dit is natuurlijk een tegenintuïtief resultaat.

7b, p. 14: Waar:

$$\begin{aligned} P\{\text{nooit een 6}\} &= P\{\text{1e worp geen 6 en 2e worp geen 6 en 3e worp geen 6 } \dots\} \\ &= P\{\text{1e worp geen 6}\}P\{\text{2e worp geen 6}\}P\{\text{3e worp geen 6}\}\dots \\ &= \lim_{n\to\infty} \left(\frac{5}{6}\right)^n \\ &= 0. \end{aligned}$$

De tweede gelijkheid geldt omdat we mogen aannemen dat de uitkomst van elke volgende worp niet afhangt van de uitkomsten van vorige worpen. Dit antwoord correspondeert beter met onze intuïtie.

We herhalen: als niet alle elementen van een verzameling even zwaar wegen, betekent "bijna alle" : "de (kans)maat van de verzameling van uitzonderingen is nul".

8a, p. 14: Waar. In termen van kansen kunnen we zeggen dat, als we een willekeurig punt in $\mathbb{R}^2$ prikken, de kans nul is dat dit punt een roosterpunt is Zonder te verwijzen naar kansen kunnen we zeggen dat een punt, ten opzichte van een vlak, geen volume bezit.

8b, p. 14: Waar. In termen van kansen kunnen we zeggen dat, als we een willekeurig punt in $\mathbb{R}^2$ prikken, de kans nul is dat dit punt op een roosterlijn ligt. Zonder te verwijzen naar kansen kunnen we zeggen dat een lijn geen volume bezit ten opzichte van een vlak.

8c, p. 14: Waar, want de kans dat een willekeurig punt in $\mathbb{R}^2$ buiten de eenheidsschijf ligt, is 1. De kans dat een willekeurig punt in $\mathbb{R}^2$ buiten een schijf met straal 10^{100} ligt, is overigens ook 1.

9, p. 15: We moeten de kans berekenen dat een willekeurig getal priem is.

$$P\{\, x \text{ is priem} \mid x \leq n \,\} = \frac{\ln n}{n}.$$

Dus

$$\begin{aligned} P\{\, x \text{ is priem} \,\} &= \lim_{n\to\infty} P\{\, x \text{ is priem} \mid x \leq n \,\} \\ &= \lim_{n\to\infty} \frac{\ln n}{n} \\ &= 0. \end{aligned}$$

Die kans is nul, dus bijna geen enkel natuurlijk getal is priem (i.e., bijna alle natuurlijke getallen zijn niet priem).

10a, p. 15: Bijna zeker (kans is 1), maar niet zeker, want een oneindige rij zonder zessen is mogelijk.

10b, p. 15: Dit is zeker (wat sterker is dan bijna zeker): stel dat we willen voorkomen dat we eindigen met één etui met 2 of meer pennen. Dan kunnen we de eerste 10 pennen nog kwijt, elk in een aparte etui. Maar de 11e pen kunnen we dan niet meer kwijt, op straffe van het creëren van een etui met 2 of meer pennen. Dit verschijnsel staat bekend als het *duiventilprincipe*: als er 101 duiven in 100 hokjes moeten, zal er altijd een hokje met 2 of meer duiven zijn.

10c, p. 15: Dit is zeker. We passen weer het duiventilprincipe toe. Een aantal van 12 mensen kunnen elk nog in een verschillende maand jarig zijn, maar 13 mensen niet meer. Dus zeker 15 mensen niet.

10d, p. 15: Dit is zeker. Je mocht het antwoord opzoeken, maar hier is een kort bewijs. Stel jij maakt deel uit van die zes personen.

1. Stel, je kent er tenminste drie, zeg A, B, en C.
   (a) Als ook maar één van die drie elkaar kent, dan heb je je groepje van drie die elkaar wederzijds kennen. Bijvoorbeeld, als B en C elkaar kennen, dan vormen B, C en jijzelf dat groepje.
   (b) In het andere geval, als géén van die drie elkaar kent, dan is dát een groepje van 3 dat elkaar wederzijds niet kent.

   Geval 1 is afgehandeld.
2. In het andere geval ken je hoogstens twee personen die jou ook kennen. Meteen volgt dat je tenminste drie personen NIET kent of jou niet kennen. Nu kunnen we weer hetzelfde argument ophangen als bij (1), met de noties "kennen" en "niet kennen" omgedraaid.

Frank Plumpton Ramsey (1903-1930) was een briljant wiskundige. Zijn tijdgenoten vonden hem grappig maar hijzelf was wel eens somber. Ramsey overleed vermoedelijk aan Hepatitis B.

10e, p. 15: Na wat zoeken op internet blijkt dat de kans op het missen van de roos ongeveer gelijk is aan $1 - 0.0014 = 0.9986$. Die kans is dus niet gelijk aan 1. Dus de gebeurtenis "het missen van de roos in een dartspel" is niet bijna zeker, laat staan zeker.

10f, p. 15: Bijna zeker (kans is 1), maar niet zeker (want het kiezen van een punt is theoretisch mogelijk).

10g, p. 15: Als de aarde niet wordt opgeslokt door een uitdijende, uitdovende rode zon (5-7 miljard jaar vanaf nu), en de aap blijft maar rammen, dan is bijna zeker dat de betreffende regel verschijnt. Formeel volgt dit uit het volgende resultaat.

> In elke oneindige rij waarbij elk karakter aselect is gekozen, zal elke vooraf opgegeven eindige reeks bijna zeker ergens voorkomen.

En dit resultaat volgt weer meteen uit de zg. 2e stelling van Borel en Cantelli.

Maar je kunt het ook eenvoudig zelf aantonen. Stel, het aantal beschikbare karakters is 100. Dan is de kans dat de aap met de eerste 53 toetsaanslagen de betreffende regel er meteen uit ramt, gelijk aan

$$\left(\frac{1}{100}\right)^{53}$$

en dat is niet nul, zeg ϵ. De kans dat de aap met de tweede 53 toetsaanslagen (niet overlappend met de eerste 53 toetsaanslagen) de betreffende regel er uit ramt, is ook gelijk aan ϵ. Enzovoort.

$$\underbrace{MedF\ldots bFFgrk}_{53}\underbrace{MetbloA\ldots bF}_{53}\underbrace{3cfA\ldots H6}_{53}\ldots$$

Keren we het om. De kans dat de aap in n (disjuncte!) episoden van elk 53 toetsaanslagen geen succes heeft is dus

$$(1-\epsilon)^n$$

Immers, de episoden overlappen elkaar niet, dus beïnvloeden elkaar niet. De bijbehorende kansen zijn daarmee onafhankelijk en mogen dus met elkaar vermenigvuldigd worden.

Laten we n naar oneindig gaan (i.e., laten we de aap maar lang genoeg z'n gang gaan), dan convergeert bovenstaande kans naar nul. De kans dat bij onbeperkt typen de aap geen succes heeft is dus nul. Dus de kans dat de aap bij onbeperkt typen wél succes heeft is gelijk aan 1.

1(a)i, p. 19: Afkorting: A1 $\rightarrow$ 0<. De toestand blijft immers A.

1(a)ii, p. 19: Afkorting: 'K0 $\rightarrow$ B>. Het tape-symbool blijft immers 0.

1(a)iii, p. 19: Afkorting: Z0 $\rightarrow$ >. Immers, de toestand en het te schrijven symbool blijven hetzelfde.

1(a)iv, p. 19: Afkorting: H1 $\rightarrow$. Immers, er veranderd niets.

1(b)i, p. 19: Het maakt nogal uit of de tape naar links (<) of rechts (>) verschoven wordt.

1(b)ii, p. 19: Een 1 lezen in toestand A is hetzelfde als in toestand A een 1 lezen. Geen verschil.

1(b)iii, p. 19: Een 0 schrijven en dan in toestand B overgaan, is hetzelfde als eerst in toestand B overgaan, en dan een 0 schrijven.

1(b)iv, p. 19: De tape naar links schuiven en dan een 0 schrijven heeft een ander effect dan eerst een 0 schrijven en dán de tape naar links schuiven. Om deze reden worden per conventie tape-schuifsymbolen < en > altijd al laatste geschreven. Of in ieder geval, áchter de schrijfactie.

1(b)v, p. 19: Eerst schuiven en dan van toestand veranderen, heeft hetzelfde effect als eerst van toestand veranderen, en dan schuiven.

1(b)vi, p. 19: Hier verschijnt de schuifactie weer vóór de schrijfactie, en dan gaat het mis.

1b, p. 19: Algemene conclusie: het is prima om afkortingen van transities te gebruiken. Om misverstanden te voorkomen, is het echter verstandig om de schuif-actie consequent als laatste element op te nemen in een transitie-regel.

2a, p. 19: De executie-log voor $N = 2$:

State/see: A/1; do: <
State/see: A/1; do: <
State/see: A/0; do: B>
State/see: B/1; do: 0>
State/see: B/1; do: 0>
State/see: B/0; do: H<

Zoals je dus eigenlijk zou verwachten: hetzelfde als de executie-log voor $N = 3$, alleen een rij minder voor A/1 en een rij minder voor B/1. Dus geen gemene randgevalletjes.

2b, p. 19: De executie-log voor $N = 0$:

State/see: A/0; do: B>
State/see: B/0; do: H<

De lees/schrijf-kop is ook teruggekeerd op zijn start-cel, zoals dat per conventie geëist wordt. Het antwoord is dus: "ja".

3a, p. 19: Bijvoorbeeld: toestand A wordt gebruikt om de string van 1-en te scannen. Zodra de machine de 0 na die string 1-en treft, print deze een 1, en gaat in toestand B. Toestand B wordt gebruikt om terug te lopen over (de unaire representatie van) het getal, totdat de lees/schrijf-kop stuit op de 0 vóór de string 1-en. De machine schuift de band dan één cel naar links en zet zichzelf dan in toestand H, i.e., schakelt zichzelf dan uit.

A0	B1
A1	<
B0	H<
B1	>

3b, p. 19: Executie-log voor $N = 3$:

State/see: A/1; do: <
State/see: A/1; do: <
State/see: A/1; do: <
State/see: A/0; do: B1
State/see: B/1; do: >
State/see: B/1; do: >
State/see: B/1; do: >
State/see: B/1; do: >
State/see: B/0; do: H<

3c, p. 19: Voor deze transitie-tabel wel. Executie-log:

State/see: A/0; do: B1
State/see: B/1; do: >
State/see: B/0; do: H<

4, p. 19: Bijvoorbeeld: toestand A wordt gebruikt om de eerste 0 na (de unaire representatie van) het getal te vinden; toestand B wordt vervolgens gebruikt om de tweede 0 na het getal te vinden; toestand C wordt gebruikt om de derde 0 na het getal te vinden. Telkens als de machine de volgende 0 vindt, wordt de toestand eentje opgehoogd. Toestand D wordt gebruikt om weer over het (nu opgehoogde) getal terug te lopen naar de 0 vóór de string 1-en.

Hieronder is voor het gemak de tabel niet lexicografisch maar chronologisch opgeschreven. (I.e., de toestand/input-combinaties die als eerste voorkomen worden ook als eerste opgeschreven.)

A1	<
A0	1B<
B0	1C<
C0	1D
D1	>
D0	H<

Executie-log voor $N = 2$:

State/see: A/1; do: <
State/see: A/1; do: <
State/see: A/0; do: 1B<
State/see: B/0; do: 1C<
State/see: C/0; do: 1D
State/see: D/1; do: >
State/see: D/1; do: >
State/see: D/1; do: >
State/see: D/1; do: >
State/see: D/1; do: >
State/see: D/0; do: H<

De zojuist gegeven transitie-tabel geeft het correcte antwoord voor $N = 0$.

5, p. 20: Als een Turing-machine beperkt wordt tot een eindig stuk tape, dan zijn er eindig veel super-toestanden. Immers, met een eindig tape-alfabet en een eindige tape, zijn er slechts eindig veel tape-configuraties. Verder kan de lees/schrijf-kop op slechts eindig veel posities staan. Tot slot kan, per definitie van de notie Turing-machine, de machine zélf slechts eindig veel toestanden aannemen.

Vanuit elke super-toestand vertrekt hoogstens één pijl naar een andere super-toestand. Dus het aantal transities per super-toestand is eindig. (Als je het tape-symbool onder de kop niet laat meetellen in de super-toestand, zijn er, per super-toestand, hoogstens net zo veel transities naar andere super-toestanden als dat er tape-symbolen zijn. En een Turing-machine bezit eindig veel tape-symbolen, meestal alleen de symbolen 0 en 1 en nog wat markers.)

De Turing-machine degenereert daarmee tot een eindige automaat (Eng.: *finite state automaton*). De eindige automaat is behandeld in het vak Inleiding Taalkunde.

6, p. 21: Laat $f : \mathbb{N} \to \mathbb{N}$ een willekeurige functie zijn.

1. Het is makkelijk voor te stellen een Turing-machine te maken die elk x getal leest, en dan op de start-cel in toestand R_x belandt op een verder lege tape. De toestand R_x representeert dan het feit dat de machine het getal x heeft gelezen.

   Hoe dan? Bijvoorbeeld, de machine scant een getal tot het eind, en zet zichzelf op de eerste 0 achter het getal in toestand R_0. Vervolgens wist het de 1-en achterwaarts, ondertussen elke keer overgaand van toestand R_i naar R_{i+1}. Bij de begin-cel aangekomen staat de machine dan in toestand R_x, waarbij x gelijk is aan het aantal 1-en dat oorspronkelijk op de tape stond.

2. Voor dezelfde machine is het makkelijk voor te stellen dat er toestanden W_y zijn, $y \in \mathbb{N}$ die, als de machine op de start-cel staat, met een verder lege tape, precies y 1-en schrijft, en daarna terug-loopt naar de start-cel, om vervolgens toestand H aan te nemen. Dus in die situatie betekent toestand W_y dat getal y gaat worden geschreven.

3. Vul de transitie-tabel verder met toestand-overgangen $R_x \to W_y$, met $y = f(x)$. Dus $R_x \to W_{f(x)}$. Voor elke $x \in \mathbb{N}$ betekent deze overgang dus dat als getal x wordt gelezen, getal $f(x)$ wordt geschreven.

Hiermee is een Turing-machine geconstrueerd die functie f uitrekent. Omdat f willekeurig was, kunnen Turing-machines met oneindig veel toestanden alle functies $\mathbb{N} \to \mathbb{N}$ uitrekenen.

7, p. 22: Definieer de functie $f : \mathbb{N} \to \mathbb{N} : x \mapsto \mathbb{N}$ als $f(x) = 0 \Leftrightarrow$ machine met codering x stopt. Dankzij de vorige opgave bestaat er een Turing-machine met oneindig veel toestanden die deze functie implementeert. Daarmee is het stop-probleem beslisbaar geworden.

8a, p. 22: Alan Turing zelf, in zijn paper waarin hij de Turing-machine zélf voorstelde. Dit paper is getiteld: “On computable numbers, with an application’s to the Entscheidungsproblem” (1936-38). Turing noemde de machine toen nog “computing machine”. Later zou de deze machine de naam Turing-machine krijgen.

8b, p. 22: Ook weer Alan Turing, in datzelfde paper. (Bron: Odifreddi, e.a.)

8c, p. 22: Marvin Minsky, één der founding fathers van de AI, ontdekte in 1962 een universele $(7, 4)$ Turing-machine. Andere kleine universele Turing-machines zijn sindsdien gevonden door Yurii Rogozhin en anderen. Later werden de volgende tupels gevonden: $(15, 2)$ (binair tape alfabet), $(9, 3)$, $(6, 4)$, $(5, 5)$, $(4, 6)$, $(3, 9)$ en $(2, 18)$. De $(4, 6)$-machine van Rogozhin gebruikt slechts 22 instructies (i.e., de transitie-tabel is slechts 22 rijen hoog). Tot 2026 aan toe is er geen universele Turing-machine bekend met een lagere beschrijvende complexiteit. (Bron: Wikipedia.)

8d, p. 22: Een universele Turing-machine kan worden beschouwd als een computer in de moderne betekenis van het woord. Als een programma aan een computer wordt aangeboden, is het in feite nog statische data welke gedecodeerd en geïnterpreerd (of gecompileerd) dient worden, vóórdat het kan worden geëxecuteerd (i.e., uitgevoerd). Met andere woorden, een universele Turingmachine is geen machine voor een speciaal doel: het is in essentie programmeerbaar en dus universeel inzetbaar. In het bijzonder zijn alle universele Turing-machines even krachtig, en verschillen ze alleen in snelheid en efficiëntie. Omgekeerd is elke huidige computer (afgezien van fysieke storingen) gelijkwaardig aan een universele Turingmachine, als deze de mogelijkheid krijgt om een potentieel oneindig geheugen te gebruiken, dat wil zeggen, altijd in staat is om meer RAM toe te voegen tijdens een berekening wanneer nodig. Naast een potentieel oneindig geheugen geheugen zijn de enige noodzakelijke eigenschappen van een universele Turingmachine de mogelijkheid om coderings- en decoderings-bewerkingen uit te voeren. Zodra dit niveau van complexiteit is bereikt, kan de machine taken uitvoeren die ingewikkelder zijn dan waarvoor hij direct is gebouwd. Het spectaculaire van moderne computers en universele Turing-machines is dus dat er complexere programma’s op kunnen worden gedraaid dan waar mee de computer (of universele Turing-machine) zélf is geschreven. (Vrij naar [25, 133-134].)

1, p. 23: Vanuit een busy beaver kan altijd weer een nieuwe worden gemaakt door alle toestanden te hernoemen. Eigenlijk is dit flauw, want beide machines zijn, op hernoeming van toestanden na, identiek.

2, p. 24: Na 107 stappen.

3, p. 24: We moeten laten zien dat, voor elke $N \geq 1$, de verzameling van Turing-machines met N toestanden die stoppen, niet leeg, en eindig is.

- Voor elke $N \in \mathbb{N}$ zijn wel er machines te bedenken die stoppen. Dus de verzameling van machines van orde N die stoppen is niet leeg.
- Voor elke N is het aantal mogelijke Turing-machines met N toestanden eindig. Immers, het tape alfabet van elke Turing-machine is eindig, en voor elke toestand en gelezen tape-symbool zijn er maar eindig veel verschillende transities te bedenken.

  Omdat het aantal mogelijke Turing-machines met N toestanden eindig is, is het aantal mogelijke Turing-machines met N toestanden dat stopt, óók eindig.

Met elke stoppende Turing-machine kan een getal worden geassocieerd, namelijk het aantal stappen dat uitgevoerd wordt bij executie. Elke niet-lege en eindige verzameling van natuurlijke getallen bezit een uniek maximum. Dus de busy beaver-functie is wel-gedefinieerd

4, p. 24: De busy beaver functie is totaal: dit volgt meteen uit Opgave 3.

Laat $S(n) = k$. Dan bestaat er een n-state Turing machine TN die k stappen draait en dan stopt. Vanuit TN kunnen we een m-state Turing machine TM bouwen die $k + (m - n)$ stappen draait en dan stopt. We voegen $m - n$ toestanden toe aan TN. Laat elke toestand die een stop-toestand van TN was, ophouden een stop-toestand te zijn. Laat hem in plaats daarvan, bij elk invoer-teken, naar de eerste nieuwe toestand gaan, schrijf een 1 en ga naar rechts. Ga vanuit die eerste nieuwe toestand, bij elk invoer-teken, naar de tweede nieuwe toestand, schrijf een 1 en ga naar rechts. Ga door met alle nieuwe toestanden. Maak de laatste een stop-toestand. Deze nieuwe machine voert k stappen uit, net als TN deed, en dan nog eens $m - n$ stappen, en dan stopt hij. Dus $S(m) \geq S(n) + (m - n)$. Omdat $m > n$, is $m - n$ positief. Dus $S(m) > S(n)$.

5, p. 24: Stel, BB is berekenbaar. We beschrijven nu een algoritme voor het beslissen van Turing's stop-probleem.[55]

Als we een machine M krijgen waarvan me moeten beslissen of deze (uiteindelijk) stopt, kijken we eerst naar het aantal toestanden in de transitie-tabel van M. Als dit N is, dan berekenen we het maximale aantal stappen dat M kan maken, alvorens te stoppen. Dit is $\mathrm{BB}(N)$. We laten nu de machine lopen. Als deze binnen $\mathrm{BB}(N)$ stappen stopt, is de vraag beantwoord. Als machine M na $\mathrm{BB}(N)$ stappen nóg loopt, dan weten we per definitie van $\mathrm{BB}(N)$ dat M zal blijven lopen, en is de vraag ook beantwoord.

We hebben zojuist een algoritme beschreven voor het beslissen van Turing's stop-probleem, terwijl juist bekend is dat Turing's stop-probleem onbeslisbaar is. De oorspronkelijke aanname dat de busy beaver-functie berekenbaar is, is dus onjuist.

[55] Bij het woord algoritme ligt de klemtoon op de i. Dus: algorítme.

6, p. 25: Uit het ongerijmde. Stel dat er wél een axiomatische theorie bestaat zodanig dat, voor alle $N \geq 1$, de formule $\mathrm{BB}(N) = k$ een stelling is. Daarmee zouden we een algoritme hebben om voor elke $N \geq 1$, te bepalen wat $\mathrm{BB}(N) = k$ is. Als N gegeven is, begin dan alle stellingen te enumereren, totdat de stelling $\mathrm{BB}(N) = k$ verschijnt. En dat is dan het antwoord. Met dit algoritme zou de busy beaver functie berekenbaar zijn, maar dankzij Opgave 5 op pagina 24 weten we dat dit niet zo is. Dus onze oorspronkelijke aanname is onhoudbaar.

De intuïtieve betekenis van hetgeen net bewezen is, is de volgende. Ook al is de busy beaver-functie wel-gedefinieerd (Opgave 3 op pagina 24), er kan nooit een systematische manier zijn om vooruitgang te boeken in het bepalen ervan. Het bepalen van elke volgende waarde (als die al bekend kan worden) is een nieuwe uitdaging, die nieuwe inzichten en uiteindelijk zelfs nieuwe axioma's vereist. Zelfs als we zouden veronderstellen dat er voor elke N een redelijke theorie was die krachtig genoeg is om de waarde van $\mathrm{BB}(N)$ te bepalen, dan nóg zou er geen systematische manier kunnen zijn om die uitbreidingen te vinden. Want als die er waren, dan zou de busy beaver-functie berekenbaar worden, en uit een vorige opgave weten we dat dit niet zo is.

7, p. 25: Dit volgt meteen: [nog te schrijven, maar volgt i.d.d. meteen].

8a, p. 25: Stel f is berekenbaar, en groeit sneller dan BB_Σ. Dus er is een N zo dat $\mathrm{BB}_\Sigma(x) < f(x)$ voor alle $x \geq N$. We beschrijven nu een algoritme dat van elke Turing-machine M beslist of deze stopt.

Als we een machine M krijgen waarvan me moeten beslissen of deze (uiteindelijk) stopt, kijken we eerst naar het aantal mogelijke toestanden, k, van deze machine. Z.b.d.a. mag worden aangenomen dat $k \geq N$. Immers, als $k < N$, kan de machine middels het toevoegen van $N - k$ dummy-toestanden (toestanden die verder niks doen) worden uitgebreid naar een machine met k toestanden, die voor wat betreft gedrag verder identiek is aan de originele machine. In het bijzonder is het stop-gedrag (waar het ons om gaat) dan identiek aan de originele machine. Via deze truc hoeven we dus alleen nog maar na te denken over machines met N toestanden of meer, en dat is natuurlijk handig vanwege $\mathrm{BB}_\Sigma(x) \leq f(x)$ voor alle $x \geq N$.

Om te beslissen of M stopt, berekenen we eerst $f(k)$. Laat dit Z zijn. Laat M nu lopen. Als M binnen Z stappen stopt, is de vraag beantwoord. Als de machine na Z stappen nog steeds loopt, kan de machine ook niet meer stoppen. Immers, $\mathrm{BB}_\Sigma(k) < Z$. Dus in dat geval is de vraag ook beantwoord.

We hebben zojuist een algoritme beschreven voor het beslissen van Turing's stop-probleem, terwijl juist bekend is dat er geen algoritme bestaat voor het beslissen van Turing's stop-probleem. Onze oorspronkelijke aanname dat f berekenbaar is, en groeit sneller dan BB_Σ, is dus onhoudbaar. Omdat f willekeurig was groeit geen enkele berekenbare functie sneller dan BB_Σ.

8c, p. 25: Als wordt toegelaten dat aan het begin getallen tot en met x op de tape mogen staan, zal er qua aantal te nemen stappen allicht meer mogelijk zijn dan wanneer aan het begin niets op de tape mag staan.

8d, p. 25: Stel M implementeert f, en M heeft daarvoor N toestanden nodig. Als $x \geq N$, dan zal de berekening van M op argument x allicht hoogstens evenveel stappen nemen dan een busy beaver met x toestanden op argument x. Kortom, $\mathrm{BB}'_\Sigma(x) > f(x)$, voor $x \geq N$.

9a, p. 26: Zoek op internet hoe een Turing-machine een getal in binaire notatie omzet naar een getal in unaire notatie.

9b, p. 26: Dankzij het vorige onderdeel weten we dat er een Turing-machine bestaat die een getal in binaire notatie omzet naar een getal in unaire notatie. Laat die machine K_1 toestanden daar voor nodig hebben. Verder is er voor elke $N \geq 0$, een Turing-machine met

$$K_2 + \lceil \log_2 N \rceil$$

toestanden, die op een blanke tape het getal N schrijft, in binaire notatie, waarbij K_2 een constante is. Immers, voor elk bit van het getal n in binaire notatie kan een toestand gebruikt worden, en dan is er nog een klein, maar vast, aantal toestanden K_2 voor house keeping. Deze twee machines kun je combineren. Er zal een klein, maar vast, aantal toestanden K_3 nodig zijn om de transitie-tabellen van die machines aan elkaar te koppelen. Laat $K = K_1 + K_2 + K_3$. We hebben nu het bestaan aangetoond van een Turing-machine met

$$K + \lceil \log_2 N \rceil$$

toestanden, die op een blanke tape het getal N schrijft, in unaire notatie.

9c, p. 26: Laat M_f de Turing-machine zijn die f uitrekent. Stel dat M_f daarvoor K_f toestanden nodig heeft.

Laat M_n^f de Turing-machine zijn die eerst het getal n in unaire representatie op een lege tape schrijft, en vervolgens M_f emuleert om f op dat argument te berekenen. Deze machine heeft daarvoor

$$K + \lceil \log_2 N \rceil + K_f + L$$

toestanden nodig, waarbij L toestanden nodig zijn om de transitie-tabellen van M_N en M_f aan elkaar te breien. Ook hier is L weer een constante, die in dit geval afhangt van N noch f. Als $C =_{Def} K + +K_f + L$, volgt het gevraagde.

9d, p. 26: Voor voldoend grote n geldt $n > C + \lceil \log_2 n \rceil$. Omdat BB_Σ strict monotoon stijgt dus ook

$$\mathrm{BB}_\Sigma(n) > \mathrm{BB}_\Sigma(C + \lceil \log_2 n \rceil)$$

voor voldoend grote n. Dankzij Opgave 9c op pagina 26 was al bekend dat

$$\mathrm{BB}_\Sigma(C + \lceil \log_2 n \rceil) \geq f(n)$$

voor voldoend grote n. Samen geeft dit het gevraagde.

1, p. 27: De nul-functie $Z : x \mapsto 0$, de successor-functie $S : x \mapsto x + 1$, en de klasse van projectie-functies.

2, p. 27: Functie-compositie (of kortweg compositie), primitieve recursie (of kortweg recursie), en μ-minimalisatie (of kortweg minimalisatie).

3, p. 27: Met de projectie-functie π_2^3. Immers, een punt bestaande uit drie coördinaten wordt afgebeeld op haar tweede coördinaat. De functie $(x, y, z) \mapsto y$ is een elementaire bouwsteen, want elke projectie is een basis-functie.

4, p. 27: $I = pi_1^1$.

5, p. 28: $f(x) =_{Def} S(S(x))$. Eleganter: $f =_{Def} S \circ S$ (spreek uit: "f is S na S"). Kortweg: $f =_{Def} SS$. De compositie-operator "$\circ$" wordt dan weggelaten.

6, p. 28: $g =_{Def} SSS$. Nu we f hebben kunnen we ook schrijven: $g =_{Def} fS$, of $g =_{Def} Sf$.

7, p. 28: $e(x) =_{Def} S(S(S(0))) = S^3(0) = 3$.

8, p. 28: Eerst alle coördinaten projecteren op de eerste coördinaat, en dan die eerste coördinaat afbeelden op 0, en dan k keer ophogen met de successor-functie S:

$$(x_1, \ldots, x_n) \xrightarrow{\pi_1^n} x_1 \xrightarrow{Z} 0 \xrightarrow{S^k} k.$$

Samengevat: $c_k^n =_{Def} S^k \circ Z \circ \pi_1^n$. De index 1 is hier niet zo belangrijk, er had ook een andere index $1 \leq i \leq n$ genomen kunnen worden. Dus bijvoorbeeld $c_k^n =_{Def} S^k \circ Z \circ \pi_5^n$ zou ook een prima antwoord zijn. En misschien zijn constante-functies in het algemeen ook wel op andere manieren te genereren uit elementaire bouwstenen.

9a, p. 28:

$$\begin{cases} p(x,0) & = I(x) \\ p(x,S(y)) & = Sp(x,y). \end{cases}$$

9b, p. 28: We laten zien dat $2+2=4$ met alleen de elementaire bouwstenen 0, S en p:

$$\begin{array}{rll} 2+2 &= p(S(S((0)),S(S(0))) & \text{definitie van } + \text{ en de getallen 2,} \\ &= S(p(S(S(0)),S(0))) & \text{definitie van } p\text{, 2e clausule,} \\ &= S(S(p(S(S(0)),0))) & \text{definitie van } p\text{, 2e clausule,} \\ &= S(S(S(S(0)))) & \text{definitie van } p\text{, 1e clausule,} \\ &= 4 & \text{terugschrijven in gebruikelijke notatie.} \end{array}$$

10a, p. 28:

$$\begin{cases} m(x,1) & = I(x) \\ m(x,y+1) & = p(m(x,y),x). \end{cases}$$

De expressie $p(m(x,y),x)$ mag ook als $m(x,y)+x$ worden geschreven. Immers, van optelling is al bekend dat deze primitief recursief is.

10b, p. 29: Verstandig is om getallen eerst gewoon te schrijven als te doen gebruikelijk, en pas als het bewijs af is, de successor-notatie te gebruiken.

$$\begin{array}{rll} 3 \times 2 &= m(3,2) & \text{definitie van } \times\text{,} \\ &= p(m(3,1),3) & \text{definitie van } m\text{, 2e clausule,} \\ &= p(p(m(3,0),3),3) & \text{definitie van } m\text{, 2e clausule,} \\ &= p(p(0,3),3) & \text{definitie van } m\text{, 1e clausule,} \\ &= p(p(0,2)+1,3) & \text{definitie van } p\text{, 2e clausule,} \\ &= p((p(0,1)+1)+1,3) & \text{definitie van } p\text{, 2e clausule,} \\ &= p(((p(0,0)+1)+1)+1,3) & \text{definitie van } p\text{, 2e clausule,} \\ &= p(3,3) & \text{definitie van } p\text{, 1e clausule,} \\ &= (p(3,2))+1 & \text{definitie van } p\text{, 2e clausule,} \\ &= (p(3,1)+1)+1 & \text{definitie van } p\text{, 2e clausule,} \\ &= ((p(3,0)+1)+1)+1 & \text{definitie van } p\text{, 2e clausule,} \\ &= ((3+1)+1)+1 & \text{definitie van } p\text{, 1e clausule,} \\ &= 6. & \end{array}$$

Nu alle getallen en +1-operaties omzetten naar successor-notatie:

$$\begin{aligned}
0''' \times 0'' &= m(0''', 0'') && \text{definitie van } \times, \\
&= p(m(0''', 0'), 0''') && \text{definitie van } m, \text{ 2e clausule}, \\
&= p(p(m(0''', 0), 0'''), 0''') && \text{definitie van } m, \text{ 2e clausule}, \\
&= p(p(0, 0'''), 0''') && \text{definitie van } m, \text{ 1e clausule}, \\
&= p((p(0, 0''))', 0''') && \text{definitie van } p, \text{ 2e clausule}, \\
&= p((p(0, 0'))'', 0''') && \text{definitie van } p, \text{ 2e clausule}, \\
&= p((p(0, 0))''', 0''') && \text{definitie van } p, \text{ 2e clausule}, \\
&= p(0''', 0''') && \text{definitie van } p, \text{ 1e clausule}, \\
&= (p(0''', 0''))' && \text{definitie van } p, \text{ 2e clausule}, \\
&= (p(0''', 0'))'' && \text{definitie van } p, \text{ 2e clausule}, \\
&= (p(0''', 0))''' && \text{definitie van } p, \text{ 2e clausule}, \\
&= (0''')''' && \text{definitie van } p, \text{ 1e clausule}, \\
&= 0'''''.
\end{aligned}$$

11, p. 29:

$$\begin{cases} f(0) & = 0 \\ f(x+1) & = m(f(x), S(x)). \end{cases}$$

De expressie $m(f(x), S(x))$ mag ook als $f(x) \times (x+1)$ worden geschreven. Immers, van vermenigvuldiging is al bekend dat deze middels primitieve recursieve kan worden geconstrueerd.

12, p. 29:

$$\begin{cases} x^0 & = I(x) \\ x^(y+1) & = x^y \times x. \end{cases}$$

Machtsverheffing is hier niet meer in functie-notatie geschreven. Dat is OK, zolang iedereen maar beseft dat machtsverheffing hier het definiendum (i.e., de te definiëren functie) is, en netjes het schema van primitieve recursie gevolgd wordt.

13, p. 29:

$$\begin{cases} f(0) & = k \\ f(x+1) & = h(x, f(x)), \end{cases}$$

waarbij h een primitief recursieve functie is. Bijvoorbeeld, bij de definitie van faculteit was h gelijk aan vermenigvuldiging.

14, p. 29:

$$\begin{cases} f(0, \quad y_1, \ldots, y_n) = g(y_1, \ldots, y_n) \\ f(x+1, y_1, \ldots, y_n) = h(x, y_1, \ldots, y_n, f(x, y_1, \ldots, y_n)). \end{cases}$$

Korter:

$$\begin{cases} f(0, \quad \bar{y}) = g(\bar{y}) \\ f(x+1, \bar{y}) = h(x, \bar{y}, f(x, \bar{y})), \end{cases}$$

waarbij g en h een primitief recursieve functies zijn. Bijvoorbeeld, bij de definitie van optelling was h gelijk aan S, en bij de definitie van vermenigvuldiging was h gelijk aan optelling.

15, p. 29: Alleen de successor-functie!

16, p. 30:

$$A(1,4) \;\; = A(A(0,4),3) = A(13,3) = \text{onbekend.}$$

Het uiteindelijke antwoord is:

$\overset{m\rightarrow}{n\downarrow}$	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	3	5	7	9	11	13	15	17	19	21	23	25	27	29
3	5	13	29											
4	13													

De lege plekken kunnen niet worden berekend omdat elk van de voorgaande rijen maar loopt tot $m = 13$.

18, p. 31: Bereken $A(0,0)$, $A(1,1)$, $A(2,2)$, en $A(3,3)$.

$$\begin{cases} A(0,0) = 0 + 1 = 1 \\ A(1,1) = 2 + (1+3) - 3 = 3 \\ A(2,2) = 2 \times (2+3) - 3 = 7 \\ A(3,3) = 2^{(3+3)} - 3 = 2^6 - 3 = 64 - 3 = 61 \end{cases}$$

19, p. 31:

$$A(4,4) = \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{4+3} - 3 = 2^{2^{2^{2^{2^{(2^2)}}}}} - 3 = 2^{2^{2^{2^{(2^4)}}}} - 3 = 2^{2^{2^{(2^{16})}}} - 3 = 2^{2^{(2^{65536})}} - 3.$$

Dit is, kortom, een idioot groot getal.

20a, p. 32: $g(x) = 0$.

20b, p. 32:

$$g(x) = \begin{cases} 0 & \text{als } x = 0, \\ \perp & \text{anders.} \end{cases}$$

Immers, als $x > 0$ is er geen enkele y zodanig dat $f(x,y) = x = 0$.

20c, p. 32:

$$g(x) = \begin{cases} 0 & \text{als } x = 0, \\ \perp & \text{anders.} \end{cases}$$

Immers, voor $x = 0$ heeft $y = 0$ de minimale waarde voor $x + y$, en voor $x > 0$ is er geen enkele $y \in \mathbb{N}$ waarvoor $x + y = 0$. Dus dan is $\mu y f(x,y)$ ongedefinieerd.

20d, p. 32:

$$g(x) = \begin{cases} 0 & \text{als } x \text{ een veelvoud is van 3,} \\ \perp & \text{anders.} \end{cases}$$

Immers, als x *geen* veelvoud is van 3, dan is er geen enkele y waarvoor $x \;\%\; 3 = 0$.

20e, p. 32: $g(x) = 0$. Immers, voor alle x geldt $(y = 0)\ \%\ 3 = 0$.

20f, p. 32:

$$g(x) = \begin{cases} 0 & \text{als } x = 1 \text{ of } x = 3, \\ \bot & \text{anders.} \end{cases}$$

Immers, $3\ \%\ x = 3$, voor $x > 3$.

20g, p. 32: $g(x) = \bot$. Immers, voor alle x geldt $3\ \%\ 1 = 0$, maar $3\ \%\ 0 = \bot$.

20h, p. 32: Hetzelfde verhaal als bij Vraag 20g: voor alle x geldt $x\ \%\ 1 = 0$, maar $x\ \%\ 0 = \bot$.

20i, p. 32:

$$g(x) = \begin{cases} \bot & \text{als } x = 0, \\ 0 & \text{anders.} \end{cases}$$

Immers, $y\ \%\ x = 0$ voor alle $x > 0$ en $y = 0$.

20j, p. 32:

$$g(x) = \begin{cases} \bot & \text{als } x = 1, \\ 1 & \text{anders.} \end{cases}$$

Immers, het getal 0 is geen deler van 1, dus $0 \mid 1 = 0$. Verder deelt 1 alle getallen, dus is er geen minimale waarde van y, zodanig dat $1 \mid y$.

20k, p. 32:

$$g(x) = \begin{cases} \bot & \text{als } x = 0, \\ 0 & \text{anders.} \end{cases}$$

20l, p. 32: Leg eerst een tabel aan "laagste niet-delers van x, groter dan nul".

x	0	1	2	3	4	5	6	7	8	9	10	11	12	...
laagste $\nmid x$	$\bot$	2	3	2	3	2	4	2	3	2	3	2	5	...

Bijvoorbeeld, 4 is de laagste niet-deler van 6, omdat 1, 2, en 3 allemaal 6 delen. En 5 is de laagste niet-deler van 12 omdat 1, 2, 3, 4 allemaal 12 delen. Verder is er geen laagste niet-deler, groter dan nul, van nul.

Dus

x	0	1	2	3	4	5	6	7	8	9	10	11	12	...
laagste $\nmid x$	$\bot$	2	3	2	3	2	4	2	3	2	3	2	5	...
$\mu y f(x, y)$	$\bot$	1	2	1	2	1	3	1	2	1	2	1	4	...

Het functie-voorschrift van $\mu y f(x, y)$ kan met of zonder deze tabel makkelijk worden gegeven, zij het dat dit voorschrijft, in tegenstelling tot de tabel, weinig informatie prijsgeeft over de grillige aard van deze functie. Immers, op een eerste oogopslag is moeilijk te zien wat de minimalisatie-operator elke keer oplevert:

$$g(x) = \begin{cases} \bot & \text{als } x = 0, \\ \min\{y \in \mathbb{N} \mid (y+1) \nmid x\} & \text{anders.} \end{cases}$$

1a, p. 34: 9.

1b, p. 34: 49.

1c, p. 34: $u + 4$.

1d, p. 34: $5 + y^2$.

1e, p. 34: $x + 9$.

1f, p. 34: $9 + w^2$.

1g, p. 34: 25.

1h, p. 34: 50.

1i, p. 34: $49 + z$.

1j, p. 34: $\lambda x.x$. Immers, $\lambda w.w$ beeldt een argument af op zichzelf, en dat argument is toevallig de expressie $\lambda x.x$.

2a, p. 34: De expressie $(\lambda x.(\lambda y.x))8$ zegt: vervang elk vrij voorkomen van x in de expressie $\lambda y.x$ door het getal 8. En dan krijg je de constante functie $\lambda y.8$.

2b, p. 34: De expressie $(\lambda x.(\lambda y.x^y))3$ zegt: vervang elk vrij voorkomen van x in de expressie $\lambda y.x^y$ door een 3. En dan krijg je de functie $\lambda y.3^y$.

2c, p. 34: $\lambda x.x^3$.

2d, p. 34: $\lambda x.x^2$.

2e, p. 35: Er zijn twee manieren:

– Van binnenuit evalueren: $(\lambda x.((\lambda y.x^y)2))3 \to (\lambda x.x^2)3 \to 3^2 \to 9$.
– Van buitenaf evalueren: $(\lambda x.((\lambda y.x^y)2))3 \to (\lambda y.3^y)2 \to 3^2 \to 9$.

2f, p. 35: De enige manier om te evalueren is van buitenaf. De buitenste expressie $(\lambda x. \ldots)2$ schrijft voor om elk vrij voorkomen van x in de binnenste expresse $\lambda y.x^y$ te vervangen door een 2. En dan krijg je $\lambda y.2^y$.

2g, p. 35: $((\lambda x.(\lambda y.x^y))2)3 \to (\lambda y.2^y)3 \to 2^3 \to 8$.

2h, p. 35: $2^3 \to 8$.

3, p. 35: Bewijs van $\neg T = F$. Achter elke regel staat wat er is gebeurd. Zoals afgesproken gebruiken we $T = \lambda u.(\lambda v.u)$.

$$\begin{aligned}
\neg T &= \lambda x.((xF)T)T && \text{definitie van } \neg \text{ ingevuld,}\\
&= (\lambda x.((xF)T))(\lambda u.(\lambda v.u)) && \text{definitie van } T\text{, maar dan met andere variabelen,}\\
&= ((\lambda u.(\lambda v.u))F)T && \lambda u.(\lambda v.u) \text{ invullen voor } x,\\
&= (\lambda v.F)T && F \text{ invullen voor } u,\\
&= F. && \text{de constante functie } \lambda v.F \text{ toepassen op } T.
\end{aligned}$$

Er werd voor andere variabelen u en v gekozen om expressies beter uit elkaar te kunnen houden.

4, p. 35: Bewijs van $\neg F = T$. We gebruiken $F = \lambda u.(\lambda v.v)$.

$$\begin{aligned}
\neg F &= \lambda x.((xF)T)F && \text{definitie van } \neg,\\
&= (\lambda x.((xF)T))(\lambda u.(\lambda v.v)) && \text{definitie van } F\text{, maar dan met andere variabelen,}\\
&= ((\lambda u.(\lambda v.v))F)T && \lambda u.(\lambda v.v) \text{ invullen voor } x,\\
&= (\lambda v.v)T && F \text{ invullen voor } u,\\
&= T. && \text{de identiteits-functie } \lambda v.v \text{ toepassen op } T.
\end{aligned}$$

5, p. 35: Bewijs van of $TF = T$:

$$\begin{aligned}
\text{of } TF &= (\text{of } T)F && \text{haakjes associëren naar links,}\\
&= ([\lambda x.(\lambda y.(xT)y)]T)F && \text{definitie van “of”,}\\
&= (\lambda y.(TT)y)F && T \text{ voor } x,\\
&= (TT)F && F \text{ voor } y,\\
&= ([\lambda x.(\lambda y.x)]T)F && \text{expandeer het 1e voorkomen van } T,\\
&= (\lambda y.T)F && T \text{ voor } x,\\
&= T && \text{pas de constante functie } \lambda y.T \text{ op } F.
\end{aligned}$$

6, p. 35: Bewijs van of $FF = F$:

$$\begin{aligned}
\text{of } TF &= (\text{of } T)F && \text{haakjes associëren naar links,}\\
&= ([\lambda x.(\lambda y.(xT)y)]T)F && \text{definitie van “of”,}\\
&= (\lambda y.(FT)y)F && F \text{ voor } x,\\
&= (FT)F && F \text{ voor } y,\\
&= ([\lambda x.(\lambda y.y)]T)F && \text{expandeer het 1e voorkomen van } F,\\
&= (\lambda y.y)F && T \text{ voor } x\text{, maar er valt niets in te vullen,}\\
&= F && \text{pas de identiteits-functie } \lambda y.y \text{ op } F.
\end{aligned}$$

7, p. 35: Bewijs van “if T then a else $b = a$”:

$$\begin{aligned}
\text{if } T \text{ then } a \text{ else } b &= Tab && \text{definitie van het if-then-else statement,}\\
&= (Ta)b && \text{haakjes associëren naar links,}\\
&= ([\lambda x.(\lambda y.x)]a)b && \text{expandeer } T,\\
&= (\lambda y.a)b && a \text{ voor } x,\\
&= a && \text{pas de constante functie } \lambda y.a \text{ toe op } b.
\end{aligned}$$

Bewijs van “if F then a else $b = b$”:

$$\begin{aligned}
\text{if } F \text{ then } a \text{ else } b &= Fab && \text{definitie van het if-then-else statement,}\\
&= (Fa)b && \text{haakjes associëren naar links,}\\
&= ([\lambda x.(\lambda y.y)]a)b && \text{expandeer } T,\\
&= (\lambda y.y)b && a \text{ voor } x,\\
&= b && \text{pas de identiteits-functie } \lambda y.y \text{ toe op } b.
\end{aligned}$$

8, p. 36: Met gebruikmaking van :

$$
\begin{aligned}
& ((\lambda M.(\lambda N.((M(\lambda N.(\lambda f.(\lambda y.(f((Nf)y))))))N)))1)1 \\
= & ((\lambda M.(\lambda N.((M(\lambda N.(\lambda f.(\lambda y.(f((Nf)y))))))N)))(\lambda u.(\lambda x.(ux))))1 \\
= & (\lambda N.(((\lambda u.(\lambda x.(ux)))(\lambda N.(\lambda f.(\lambda y.(f((Nf)y))))))N))1 \\
= & (\lambda N.((\lambda x.((\lambda N.(\lambda f.(\lambda y.(f((Nf)y)))))x))N))1 \\
= & (\lambda N.((\lambda x.(\lambda f.(\lambda y.(f((xf)y)))))N))1 \\
= & (\lambda N.(\lambda f.(\lambda y.(f((Nf)y)))))1 \\
= & (\lambda N.(\lambda f.(\lambda y.(f((Nf)y)))))(\lambda w.(\lambda x.(wx))) \\
= & \lambda f.(\lambda y.(f(((\lambda w.(\lambda x.(wx)))f)y))) \\
= & \lambda f.(\lambda y.(f((\lambda x.(fx))y))) \\
= & \lambda f.(\lambda y.(f(fy))) \\
= & 2.
\end{aligned}
$$

1, p. 38: In de eigen woorden van de docent:

> De Church-Turing these is de hypothese, of het vermoeden, dat de intuïtieve notie "berekenbaar" gedefinieerd wordt door datgene wat alle berekeningsmechanismen (inclusief programmeertalen) in batch (dus in één keer, van invoer naar uitvoer) kunnen uitrekenen. De praktijk wijst namelijk uit dat alle berekeningsmechanismen even krachtig zijn.
>
> Soms wordt de CTH gezien als een *afspraak*, nl. de afspraak om die aanname maar te adopteren.

Jij hebt waarschijnlijk een andere formulering gekozen. Uit je formering moet wel duidelijk zijn dat het om een hypothese (veronderstelling) gaat. In het bijzonder is de Church-Turing these geen waarheid, of waar te bewijzen stelling.

2, p. 38: Als twee berekeningsmechanismen of programmeertalen evenveel kunnen uitrekenen noemen we ze "Turing-gelijkwaardig" of "Turing-equivalent". Als een berekeningsmechanisme of taal tenminste kan uitrekenen wat een Turing-machine kan uitrekenen, noemen we het "Turing-volledig" of "Turing-compleet". In de praktijk blijken alle programmeertalen Turing-equivalent én Turing-compleet.

3, p. 38: Een programma p dat alle priemgetallen afdrukt heeft daarvoor geen invoer nodig. Het heeft dus ariteit nul. We noteren dit als $p/0$. Als het een programma is dat alle priemgetallen tot een vooraf opgegeven getal afdrukt, zal het ariteit 1 hebben, en noteren we dit als $p/1$.

4, p. 38: Als er een algoritme $f(x, y) \to xy$ beschikbaar is voor het vermenigvuldigen van getallen, dan bestaat ook een algoritme $g(x, n) \to x^n$ voor het uitrekenen van ne machten, voor willekeurige n. Het algoritme g vermenigvuldigt x gewoon n maal met zichzelf. Is iemand daar niet van overtuigt, dan zou je eventueel nog een programmaatje in pseudo-code kunnen schrijven om dit te demonstreren:

$g(x, n)$

$z = 1$; for $i = 1$ to n do { $z = z * x$ } return z;

Maar eigenlijk volstaat het om te zeggen dat machtsverheffen met natuurlijke getallen neerkomt op herhaald vermenigvuldigen.

Geldt het omgekeerde ook? Niet dat we weten. Dan zou xy op een eenvoudige manier moeten zijn af te leiden uit x^n voor een of andere n. Direct genomen zou n dan gelijk moeten zijn aan $\log_x xy$, maar buiten het gegeven dat daar een logaritme in zit, bevat die formule ook weer een vermenigvuldiging, en die wilden we nu juist uitdrukken in termen van machten. Dus dat werkt niet. Dat het ons niet lukt om vermenigvuldigingen uit te drukken in termen van machten wil overigens niet zeggen dat het niet kan. Ons is het in ieder geval niet gelukt, en het lijkt ons ook onwaarschijnlijk.

5, p. 38: Om met de laatste hint te beginnen. Laat m een model zijn. Dan

$$\begin{aligned} & m \models (x_1 \vee y_1) \wedge (\neg y_1 \vee \neg x_2 \vee y_2) \wedge (\neg y_2 \vee \neg x_3 \vee y_3) \wedge (\neg y_3 \vee x_4 \vee y_4) \wedge (\neg y_4 \vee x_8) \\ \Leftrightarrow \quad & m \models x_1 \vee y_1 \text{ en } m \models \neg y_1 \vee \neg x_2 \vee y_2 \text{ en} \\ & m \models \neg y_2 \vee \neg x_3 \vee y_3 \text{ en } m \models \neg y_3 \vee x_4 \vee y_4 \text{ en } m \models \neg y_4 \vee x_8 \\ \Leftrightarrow \quad & m \models x_1 \vee y_1 \text{ en } m \models y_1 \rightarrow (\neg x_2 \vee y_2) \text{ en} \\ & m \models y_2 \rightarrow (\neg x_3 \vee y_3) \text{ en } m \models y_3 \rightarrow (x_4 \vee y_4) \text{ en } m \models y_4 \rightarrow x_8. \end{aligned}$$

Let op de implicaties $y_i \rightarrow \ldots$. Uit de laatste expressie kan worden afgelezen dat de y_i's "de waarheid doorgeven" aan volgende termen. Dus als $m \not\models x_1$ dan automatisch $m \models y_1$ en daarmee $m \models \neg x_2 \vee y_2$. Als $m \not\models \neg x_2$ dan automatisch $m \models y_2$ en daarmee $m \models \neg x_3 \vee y_3$. Enzovoort. Dus

$$\begin{aligned} & m \models x_1 \text{ of } m \models \neg x_2 \text{ of } m \models \neg x_3 \text{ of } m \models x_4 \text{ of } m \models x_8 \\ \Leftrightarrow \quad & m \models x_1 \vee \neg x_2 \vee \neg x_3 \vee x_4 \vee x_8. \end{aligned}$$

Omgekeerd geldt de implicatie ook. Met andere woorden, als expressie 2 op pagina 38 vervulbaar is, dan is de 3CNF 5 op pagina 39 dat ook. De 3CNF (5) is dus vervulbaarheids-equivalent met (2). Omdat deze omschrijftruc kan worden toegepast op elke term in een CNF, is elke CNF makkelijk automatisch om te schrijven naar een 3CNF. En daarmee is elke instantie van SAT makkelijk automatisch om te schrijven naar een instantie van 3SAT. Omdat SAT gereduceerd kan worden naar 3SAT, is 3SAT minstens zo moeilijk als SAT.

6, p. 39: Hier is een algoritme dat, voor willekeurige k, kan bepalen of een netwerk een onafhankelijke verzameling ter grootte k bezit.

> Laat k gegeven zijn. Gebruik het gegeven algoritme om te bepalen of het netwerk een overdekking ter grootte $N - k$ bezit, waarbij N het totaal aantal punten in het hele netwerk is. Als dat zo is, bevat het network (volgens het feit gegeven in de hint) een onafhankelijke verzameling ter grootte $N - (N - k) = k$. En als dat niet zo is, bevat het network volgens het feit gegeven in de hint ("en omgekeerd") geen onafhankelijke verzameling ter grootte $N - (N - k) = k$.

7, p. 39: Hint: stel dat er wel zo'n programma $h/1$ bestaat. We laten dan zien dat met deze aanname dan ook een test-programma (of test-procedure of test-algoritme) bestaat dat voor elk 1-plaatsig programma, $j/1$, en voor elke string, i, kan beslissen of $j(i)$ stopt—wat Turing's bevindingen zou tegenspreken.

Als volgt. Laat J/i een willekeurige programma/input-combinatie zijn waarvan moet worden bepaald of het stopt. Laat computerprogramma J' ontstaan uit J/i door de lees-statement, i.e, zo iets als

```
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
String x = br.readLine();
```

te vervangen door een expliciete toekenning van x met i, i.e., zoiets als

```
String x = i;
```

Eenvoudig is na te gaan dat J/i stopt als en slechts als J' stopt. We zouden het stop-probleem dus kunnen beslissen door J/i-combinaties automatisch te laten te omschrijven naar een inputloos computerprogramma J' waarvan we dan wel kunnen beslissen of het stopt. Dat is in tegenspraak met de onbeslisbaarheid van het stop-probleem, dus het stop-probleem voor input-loze computerprogramma's is ook onbeslisbaar.

8a, p. 39: We reduceren het invoerloze stop-probleem naar dit probleem. Laat $j/0$ een willekeurig programma zijn waarvan moet worden bepaald of het stopt.

> j': voer j uit, en onderdruk alle uitvoer, bijvoorbeeld door het naar `/dev/null` om te leiden; druk a af

Eenvoudig is na te gaan dat j' het symbool a afdrukt als en slechts als j stopt. Omdat we hebben aangenomen dat het print-probleem beslisbaar is, bestaat er een programma p dat kan beslissen of j' ooit a afdrukt, en dus of j stopt. Dat laatste is in tegenspraak met de onbeslisbaarheid van het invoerloze stop-probleem.

8b, p. 39: Stel het uniforme stop-probleem is beslisbaar. Dan bestaat dus een test-programma $h/1$ dat voor alle $j/1$ kan bepalen of het stopt op alle inputs. Maar daarmee bestaat ook een test-programma $h'/1$ voor het invoerloze stop-probleem. Het programma h' zou als volgt kunnen werken:

Accepteer $j/0$, en transformeer het automatisch naar een programma $j'/1$ dat zijn invoer i leest, maar er niets mee doet, en verder werkt als $j/0$. Duidelijk is dat j' stopt op alle inputs als en slechts als j stopt. Met h' kan nu worden getest of j' stopt op alle inputs, en dus of j stopt. Dit laatste is in strijd met de onbeslisbaarheid van het invoerloze stop-probleem.

8c, p. 39: Stel het programma-equivalentieprobleem is beslisbaar. Dan bestaat er dus een test-programma $h/2$ dat voor alle $j/0$ en $j'/0$ kan bepalen of deze zich hetzelfde gedragen. Maar daarmee bestaat ook een test-programma $h'/1$ voor het invoerloze stop-probleem. Het programma h' zou als volgt kunnen werken:

Transformeer $j/0$ automatisch naar een programma $j'/0$ dat niets afdrukt, bijvoorbeeld door alle uitvoer weg te leiden naar een bestand of `/dev/null`. Vergelijk het vervolgens met een triviaal programma k dat stopt en niets afdrukt. $h/2$ signaleert de equivalentie van j' en k als en slechts als j stopt. Dit laatste is in strijd met de onbeslisbaarheid van het invoerloze stop-probleem.

9, p. 39: Instanties $j/0$ van het invoerloze stop-probleem kunnen automatisch worden vertaald naar het programma

> $j'/0$: draai $j/0$ in quarantaine; infecteer!

Programma j' is een virus als en slechts als j stopt. Immers, als j stopt zal daarna het "infecteer!"-gedeelte worden uitgevoerd, en dus zal j' een virus zijn. Omgekeerd: als j niet stopt en maar (in quarantaine) blijft draaien, zal het infecteergedeelte nooit worden bereikt, en zal j' dus geen virus zijn.

Als nu de perfecte viruschecker $v/1$ zou bestaan, zou deze kunnen nagaan of $j'/0$ je PC infecteert en dus $j/0$ stopt. Maar het was al bekend dat het invoerloze stop-probleem onbeslisbaar is, dus zo'n perfecte viruschecker kan niet bestaan.

10a, p. 39: Ja: met bijvoorbeeld waarheidstabellen kan automatisch worden gecheckt of formulas waar (en dankzij de gezond- en volledigheidsstelling dus bewijsbaar) zijn. Met semantische tableaus kan het trouwens ook, en vaak nog iets sneller. (Vaak maar niet altijd.)

10b, p. 39: Dat bewijsbaarheid in predikatenlogica onbeslisbaar is, betekent dat we zeker weten dat er geen algoritme kan bestaan dat voor willekeurige formules uit de predikatenlogica kan uitrekenen of deze bewezen kunnen worden.

10c, p. 40: Dit betekent dat er tenminste één algoritme bestaat die bewijsbare formules kan herkennen. Maar daar heb je niet zo veel aan. Als een formule namelijk niet bewijsbaar is, is niet gegarandeerd dat zo'n algoritme stopt en het goede antwoord "onbewijsbaar" geeft.

10d, p. 40: Reduceer instanties van een bekend onbeslisbaar probleem naar bewijsbaarheidsvraagstukken in de predikatenlogica. Laat bijvoorbeeld zien dat instanties van Turing's stop-probleem netjes uitgedrukt kunnen worden in de predikatenlogica. De vraag "stopt dit programma" reduceert dan naar de vraag "is deze formule bewijsbaar?". Hoe dit idee precies uitgevoerd moet worden is vers twee, dat laten we voor een andere cursus ;-)

11, p. 40: Als een programma slechts een begrensde lees- en schrijfruimte tot z'n beschikking heeft, kan het slechts eindig veel toestanden aannemen. (Dit laatste wordt in een volgende alinea nog wat verder uitgelegd.) We kunnen ons dus een ander programma, h, indenken dat kan beslissen over het stoppen van elk willekeurig programma, j, door j uit te voeren en ondertussen alle toestanden waarin j zich bevindt bij te houden. Als j stopt, kan h dit melden. Als j niet stopt, zal het op den duur eenzelfde toestand moeten aannemen, omdat het aantal mogelijke toestanden waarin j zich kan bevinden eindig is. Het programma h zal dit herkennen en daarmee kunnen aangeven dat j in herhaling valt en niet stopt.

Nu de beloofde uitleg waarom een programma met begrensde lees- en schrijfruimte slechts eindig veel toestanden kan aannemen. In de eerste plaats is het korte-termijn geheugen ("de RAM") van een programma altijd eindig. (Ook met alle andere berekeningsmechanismen zoals de Turingmachine, de registermachine en andere programmeertalen is dat zo.) Verder is het lange-termijn geheugen ("de SSD") per aanname nu ook eindig. (Met Turing-equivalente berekeningsmechanismen zoals de Turingmachine, de registermachine, lambda-calculus, etc. dat NIET zo.) Het aantal combinaties van RAM-configuraties en SSD-configuraties is daarmee ook eindig. Immers, het Cartesisch product van twee eindige verzamelingen is ook weer eindig. Dus kan zo'n programma slechts eindig veel mogelijke toestanden aannemen.

Waarom is Turing's stop-probleem dan wel onbeslisbaar? Wel, deze versie van het stop-probleem veronderstelt een *onbegrensde* lees- en schrijfruimte. En dan gaat het mis, dat wil zeggen dat dan bewezen kan worden dat er geen programma, h, kan bestaan dat voor elk programma j kan uitmaken of het stopt. Intuïtief gesproken is het probleem dat j kan doorkladden op een oneindig grote SSD, voorbij het bevattingsvermogen van elk controleprogramma h.

12, p. 40: Het wordt inderdaad nergens geëist, maar dat hoeft ook niet. Blijkbaar bevat een oneindige opsomming van programma's onvermijdelijk elementen die, als ze worden uitgevoerd, oneindig veel schrijfruimte gebruiken. (Uiteraard zijn dit programma's die nooit stoppen.)

13, p. 40: Bijna hetzelfde argument als bij 11: stel j is het object-programma. Nu is een controle-programma h denkbaar dat programma j uitvoert en, naast alle toestanden, ook het geheugengebruik bijhoudt. Als j op een gegeven moment de nMB overschrijdt, kan h de executie van j afbreken en dit melden. Als j stopt zonder meer dan nMB te hebben verbruikt, kan h dit ook melden. Als j niet stopt en ook niet de nMB overschrijdt, zal deze zich, vanwege de eindige toestandruimte (immers, j blijft binnen de nMB), op den duur herhalen. Op het punt van herhaling kan h de executie van j afbreken en melden dat j niet meer dan nMB lees- en schrijfruimte verbruikt.

Merk op dat het er niet om gaat een een kort en/of tijds-efficiënt en/of geheugen-efficiënt controle-programma h te vinden. Het gaat ons er alleen om aan te geven dat er een programma bestáát dat de gevraagde taak kan uitvoeren. We leveren dus een zogenaamd *existentiebewijs*.

14a, p. 40: Eenvoudig is in te zien dat een programma $c/1$ te schrijven valt dat voor elke n de bijbehorende Collatzrij genereert, en stopt als die rij bij 1 aankomt. Het vermoeden van Collatz is waar als en slechts als c stopt op alle positieve gehele getallen. Vorm nu programma c' dat gelijk is aan c in die zin dat het doet wat c doet op alle positieve gehele getallen, en stopt op alle overige output. Nagaan of c' stopt op alle input kan worden opgelost als het uniforme stop-probleem beslisbaar is.

14b, p. 40: Nee, dat lukt niet. De beslisbaarheid van het uniforme stop-probleem lijkt daarmee een sterker statement dan de beslisbaarheid van Turing's stop-probleem. Echter, om te kunnen beweren dat dit daadwerkelijk zo is zouden we meer moeten uitzoeken. Dat doen we hier niet. Wel kan hier worden verklapt dat het zo is. Omgekeerd is de onbeslisbaarheid van Turing's stop-probleem daarmee dus een sterkere uitspraak dan de onbeslisbaarheid van het uniforme stop-probleem.

15a, p. 41: Amy heeft gelijk, haar argument is correct. Door T letterlijk in het controleprogramma H op te wordt H wel erg groot, maar dat geeft niet. Immers, berekenbaarheidsargumenten stellen geen grenzen aan rekenruimte. Wel moet het antwoord na eindig veel tijd gegeven kunnen worden, en aan dat laatste is voldaan.

Brian heeft ongelijk, zijn argument is niet correct. Het laat slechts zien dat q buiten de collectie $J = \{j_1, j_2, \ldots, j_n\}$ valt. Dus het laat slechts zien dat q niet te schrijven is in minder dan 1000 karakters, immers q bevat H, en die zal dan op zichzelf wel te veel karakters bevatten, dat kan dan niet anders. Herinner je dat programma's die hier worden bediscussieerd self-contained moeten zijn. In het bijzonder mogen de hier bediscussieerde programma's geen gebruik maken van externe bibliotheekmodulen waarin al het vuile werk wordt opgeknapt.

Overigens is het niet zo dat elke implementatie van q noodzakelijk een implementatie H moet herbergen, laat staan dat het noodzakelijk is dat q Amy's implementatie moet bevatten. Misschien zijn er wel slimmere (lees: kortere) manieren om q te schrijven. Maar dus nooit korter dan 1001 karakters.

15b, p. 42: Als we de beperking dat alle programma's self-contained moeten zijn laten varen, wordt het probleem meteen een stuk ingewikkelder. Brian's programma, q, is in dat geval namelijk

dan wél te schrijven in hoogstens 1000 tekens. Immers, q maakt dan gebruik van een bibliotheek, H, waarvan de aanroep in een paar tekens opgeschreven kan worden.

Aan de andere kant kan Amy met recht tegenwerpen dat het aanroepen van externe modules eigenlijk vals spelen is. Immers, wanneer q de module H aanroept, wordt als gevolg daarvan de byte-code van H op de executiestack gezet. En dán komen ineens toch erg veel bytes in het geheugen te staan, en wordt de 1000-teken grens vermoedelijk toch weer overschreden.

Het blijkt een moeilijke vraag. In feite blijkt dit probleem een variatie op een vraagstuk dat in de literatuur bekend staat als *Berry's paradox*. Deze paradox kan als volgt worden begrepen. Bekijk de uitdrukking

> "Het kleinste getal dat niet te omschrijven valt in minder dan twintig woorden."

Dit getal bestaat, immers met twintig woorden kunnen maar eindig veel getallen worden omschreven, en elke eindige verzameling bevat een kleinste element. Dus dit getal bestaat. Laten we het k noemen. Het probleem is dat k volgens de uitdrukking hierboven wel degelijk in minder dan twintig woorden kan worden omschreven, namelijk ... even tellen ... 13 woorden. Dus k is niet het kleinste getal. Maar welk getal dan wel? Probleem.

Vergelijk de uitdrukking "Deze melodie is onbeschrijfelijk mooi". Deze uitdrukking communiceert wel degelijk iets extra's over de schoonheid van de melodie, omdat deze uitdrukking voor bijna iedereen verschilt van dezelfde uitdrukking minus het woord "onbeschrijfelijk". Tja.

1, p. 43: Nederlandse samenvatting van Hoofdstuk 3 (Computability and Incomputability) van *The computational Beauty of Nature* (Flake, 1998):

3. Computers verschillen veel van elkaar. Oude computers werkten nog met ponskaarten, terwijl moderne computers bijvoorbeeld zijn terug te vinden als multicore-processor in bijvoorbeeld een telefoon. Maar als de toeters en bellen er af worden gehaald en niet gekeken wordt naar verschil in snelheid, geheugen, of opslagruimte blijkt dat computers veel gemeen hebben. Het is dan ook mogelijk de computer te abstraheren als een machine die te programmeren is en instructies kan uitvoeren.

3.1. Aan tekst strings kan een uniek nummer (een uniek natuurlijk getal) worden toegekend door gebruik te maken van de uniciteit van priemfactorisatie. Deze nummering heet Gödel nummering. Er zijn genoeg Gödelnummers omdat er oneindig veel priemgetallen bestaan. Omdat computerprogramma's in het bijzonder tekst strings zijn heeft elk computerprogramma ook een eigen Gödel nummer.

3.2. Aan het begin van de vorige eeuw werden wiskundigen het erover eens dat een aantal verschillende machines en mechanismen even veel kunnen uitrekenen. Deze machines en mechanismen zijn onder andere: algemene recursieve functies, de Turing machine (Turing), en lambda-calculus (Church). Maar er zijn meer van dergelijke, even sterke, machines en mechanismen. Deze wiskundigen stelden dat deze machines en mechanismen dan maar samen het begrip berekenbaarheid moesten vertegenwoordigen. Deze stelling heet de *Church-Turing these*. De Church-Turing these is nog steeds actueel.

3.3. Stutter is een uitgeklede versie van Lisp en is door Flake speciaal voor zijn boek bedacht als voorbeeld-taal; Lisp is bedacht als serieuze AI-taal door McCarthy en anderen in de zestiger jaren. In beide talen zijn programma's lijsten van instructies. In het hoofdstuk wordt eerst de werking van Stutter, dus Lisp, uitgelegd. Dan wordt aangetoond hoe uit Stutter, dat toch een vrij magere programmeertaal is, een heel arsenaal van begrippen, functies en commando's kan worden

opgebouwd, onder andere het begrip getal en rekenkundige functies als optellen en vermenigvuldigen. De sectie sluit af met voorbeelden die plausibel moeten maken dat Stutter tenminste even krachtig is als andere rekenformalismen en programmeertalen. (Dat minstens even krachtig zijn als andere berekeningsformalismen heet overigens *Turing-compleetheid*, maar dit begrip wordt door Flake verder niet genoemd.)

3.4. Programmeertalen zijn vanwege de *Church-Turing these* even krachtig, maar de ene taal is natuurlijk de andere niet. Meer bepaald zijn sommige talen efficiënter dan andere talen. Het aantal berekeningstappen waarmee een programma haar taak uitvoert heet de berekenings- of tijdcomplexiteit van dat programma. Dus twee programma's kunnen equivalent zijn ("hetzelfde doen") maar van een andere complexiteit zijn.

Complexiteit wordt grof gemeten. Zo maakt een lineaire factor in stappen niet uit. Een macht maakt wel uit. Bijvoorbeeld $\mathcal{O}(5x^3 + x^2)$ betekent: als de input x eenheden groot is, dan zal het programma in de orde van $5x^3 + x^2$ stappen nemen. Omdat de grootste macht geldt, is dit hetzelfde als een orde van x^3 stappen nemen. Dus $\mathcal{O}(5x^3 + x^2) = \mathcal{O}(x^3)$. Als iets in polynomiale tijd kan worden berekend i.e., $\mathcal{O}(x^n)$ met $n \geq 0$, dan wordt dat beschouwd als doenbaar. Als iets niet in polynomiale tijd kan worden berekend, bijvoorbeeld $\mathcal{O}(2^x)$, dan wordt dat beschouwd als ondoenbaar. Als de omvang van het probleem zou toenemen, dan zou het aantal stappen verdubbelen, wat als onacceptabel wordt gezien.

3.5. Het is mogelijk een Stutter-interpreter te schrijven in Stutter. Dit heet emulatie. Het is zelfs mogelijk een programma $U(x, y)$ te schrijven in Stutter dat als input twee Gödel getallen x en y krijgt, ze expandeert naar de bijbehorende unieke strings s_x en s_y, waarna de executie van programma s_x op input s_y wordt geëmuleerd. (Als s_x geen Stutter-programma is, dan geeft U uiteraard een syntax-foutmelding.) U wordt een *universele interpreter* genoemd.

Een programma dat, als het al iets afdrukt, slechts een 0 of een 1 kan afdrukken, heet een *beslisprogramma*. Ieder programma kan worden geëmuleerd door een beslisprogramma of een combinatie van beslisprogramma's. Beslisprogramma's zijn dus erg sterk. (In feite is de verzameling beslisprogramma's in een Turing-complete taal ook Turing-compleet, maar dat zegt Flake hier verder niet.)

3.6. Deze sectie bespreekt de vraag of er voor computers onoplosbare problemen bestaan. Dat antwoord luidt: ja. Een eerste aanleg wordt gegeven door te laten zien dat er tenminste één onberekenbare deelverzameling van natuurlijke getallen bestaat. Het argument dat daarbij gebruikt wordt is simpel: de verzameling van alle berekenbare deelverzamelingen van de natuurlijke getallen is aftelbaar terwijl de verzameling van alle deelverzamelingen van de natuurlijke getallen overaftelbaar is. Flake gebruikt hierbij het diagonaal-argument.

Alan Turing was de eerste die liet zien dat veel problemen onberekenbaar zijn. Hij formuleerde ook het eerste concrete onberekenbare probleem, te weten het stop-probleem. Het stop-probleem is het probleem of er een programma $M(x, y)$ te schrijven is dat van alle programma/input-combinaties (x, y) kan bepalen of de uitvoering van programma x met input y stopt. Via een diagonaal-argument liet Turing zien dat zo'n programma M niet kan bestaan. Volgens Flake kunnen computers dus blijkbaar niet aan zelf-analyse doen.

3.7. De *halting set* bestaat uit alle Gödelnummers van programma's die stoppen. Dankzij het stop-probleem kan dan worden afgeleid dat de halting set niet berekenbaar is (want als het dat wel was, dan was namelijk M hieruit te construeren).

Een verzameling getallen heet *recursief enumereerbaar* (ook wel: opsombaar, maar die term gebruikt Flake niet) als er een programma bestaat dat precies de elementen van die verzameling

(niet noodzakelijk in een bepaalde volgorde) kan afdrukken. Flake schrijft dat men kan aantonen dat de halting set recursief enumereerbaar is.

Een verzameling getallen heet *recursief* (ook wel: beslisbaar, maar die term gebruikt Flake niet) als er een programma bestaat dat voor elk getal kan uitrekenen of het een element uit die verzameling is. Een verzameling getallen kan recursief enumereerbaar zijn zonder recursief te zijn. Het is dan niet mogelijk een programma te schrijven dat precies alle getallen buiten die verzameling herkent.

Het complement van een recursief enumereerbare verzameling heet complementair recursief enumereerbaar, kortweg *co-recursief enumereerbaar*. Voor een co-recursief enumereerbare verzameling bestaat per definitie een programma dat voor elk getal kan bepalen of dat getal buiten die verzameling ligt. Volgens Flake kan een co-recursief enumereerbare verzameling het beste worden vergeleken met de achtergrond van een schilderij zonder de voorgrond. Het idee is dat dan bij opsomming van de achtergrond (getallen niet in die set) dan geleidelijk contouren van de voorgrond (getallen wel in die set) zichtbaar worden.

Een verzameling is recursief als en slechts als deze zowel recursief enumereerbaar als co-recursief enumereerbaar is.[56] Dit is in feite de stelling van Post, maar dat wordt door Flake verder niet aangegeven.

Einde samenvatting.

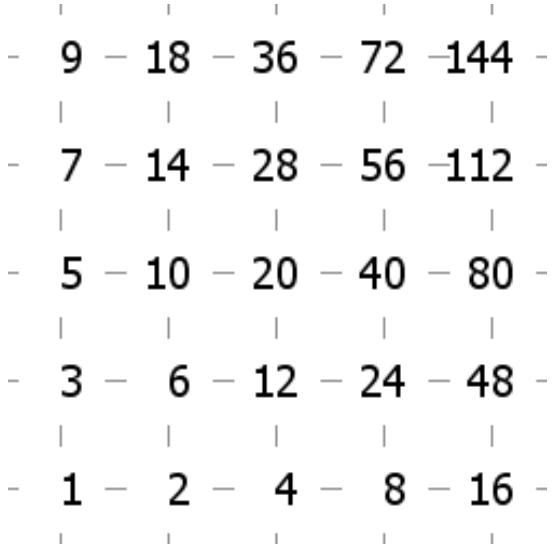
2, p. 43: **Begrippenlijst**

Hieronder volgen 41 begrippen. Het aantal te verdienen punten is gelijk aan 4. Per 10 zinnige begrippen één punt. Er zijn meerdere antwoorden mogelijk.

- 1-1 correspondentie
- $\sqrt{2}$
- aftelbaar oneindig
- algemene recursieve functies
- algoritme
- berekenbaar
- berekenbaarheid
- beslisbaar
- beslisbare verzameling
- Church-Turing these
- co-recursief enumereerbare verzameling
- complexiteitstheorie
- eindig
- emulatie
- emulator
- irrationaal getal
- Gödel nummer
- Gödel nummering
- halting problem
- halting set
- limiet
- Lisp
- lambda-calculus
- McCarthy, John
- onberekenbaar
- oneindige rij
- overaftelbaar oneindig
- priemgetal
- priemfactorisatie
- rationaal getal
- recursief enumereerbare verzameling
- recursieve verzameling
- stop-probleem
- Stutter
- Turing, Alan
- Turing-machine
- Turing-compleetheid
- tijdcomplexiteit
- wortel twee
- universele interpreter
- Zeno's paradox

[56] Het Engelse "if and only if" kan op twee manieren naar het Nederlands worden vertaald: "als en slechts als" en "dan en slechts dan". Beiden zijn prima (en courant).

3a, p. 44:



De x- en y-as lopen hier van 1 tot en met 5.

3b, p. 44: De functie f is berekenbaar, want meteen is in te zien dat een programmaatje kan worden geschreven (in eender welke taal) welke f berekent. Als mensen je niet geloven, kan je eventueel een concreet programmaatje schrijven en overleggen. Maar voor de meesten is in dit geval de mededeling dat het makkelijk voorstelbaar is een programmaatje voor f te schrijven voldoende, en in het bestek van dit vak ook een voldoende antwoord.

3c, p. 44: Elk getal is te schrijven als een 2-macht maal een oneven restfactor (haal er zoveel mogelijk 2-en uit, en wat overblijft moet dan oneven zijn). Voor surjectiviteit is verder belangrijk op te merken dat de exponent van 2 en de restfactor in (4) zo gekozen zijn dat met $x \geq 1$ en $y \geq 1$ inderdaad ook alle elementen uit $\mathbb{N}$ bereikt worden! Per definitie (van surjectiviteit) is f daarmee surjectief.

3d, p. 44: Elk getal n is maar op één manier te ontbinden in een 2-macht met een oneven restfactor (weer: haal er zoveel mogelijk 2-en uit, wat overblijft moet oneven zijn). Dus is er maar één paar (i, j) dat afbeeldt op n. Per definitie (van injectiviteit) is f daarmee injectief.

3e, p. 44: Omdat we inmiddels weten dat f injectief en surjectief dus bi-jectief is, en elke bi-jectie een inversie bezit, volgt meteen dat f een inverse heeft.

3f, p. 44: We geven een algoritme om uit een getal n het bijbehorende paar (i, j) te berekenen. Gegeven n, haal de grootste 2-macht er uit, zeg 2^k, door n steeds door 2 te delen zo lang dat kan. Laat dan $i - 1 = k$, dus $i = k + 1$. Verder is j (dus) de oplossing van $(2j - 1)2^k = n$, i.e., $j = ((n/2^k) + 1)/2$. Deze j is heel, groter dan 1 en oneven, want wat er overblijft na het uithalen van de grootste 2-macht is noodzakelijk oneven.

4a, p. 44: $2^{97}3^{46}$.

4b, p. 44: $2^{97}3^{98}5^{98}7^{97}11^{46}$.

4c, p. 44: $2^{97}3^{32}5^{99}7^{46}$.

Er bestaat overigens een ASCII-symbool met code 0, namelijk het NUL symbool. Helaas blijft dan het Gödelnummer van een string gelijk als je er NUL achteraan plakt. Dus alfa en alfa.NUL hebben hetzelfde Gödelnummer. Om het formeel helemaal mooi te doen, moet je dus bijvoorbeeld altijd een punt achter de string plakken en die bij he ontGödelen eraf halen. Maar dan is weer niet elk getal geldig als Gödelnummer.

5a, p. 44: ab.

5b, p. 44: hoi.

5c, p. 44: hallo.

6, p. 45: De verzameling X bevat in ieder geval de eerste 34 kwadraten en mogelijk meer. Het programma j kan ergens na die tien minuten nog getallen afdrukken (dan bevat X meer), of niet (en dan is X eindig). Dat is alles wat we na tien minuten weten.

7, p. 45: Voor elke eindige verzameling $X \subseteq \mathbb{N}$ is altijd een programma $j/0 \in J$ te schrijven dat X opsomt en daarna stopt: gewoon $|X|$ print-statements hard coderen.

Feit: voor bijna alle eindige deelverzamelingen van $\mathbb{N}$ is dit ook de enige manier! Voor andere eindige deelverzamelingen, typisch die met veel regelmatigheid, bestaan, ongeacht hun grootte, algoritmes die eleganter werker dan een ordinaire rij print-statements. Voorbeelden van dergelijke verzamelingen: alle even getallen onder de miljard, alle priemgetallen onder de miljard, alle Fibonaccigetallen onder de miljard.

8a, p. 45: Er zijn dus programma's j en j' die X en Y afdrukken. We kunnen ons een derde programma indenken welke de executie van j en j' vervlecht. Beide programma's krijgen dan om beurten een beetje tijd om te rekenen. Ga na dat op deze manier beide programma's onbeperkt tijd krijgen om hun hele set af te drukken.

8b, p. 45: Vervlecht de executie van j en j'. We vangen de print-statements van beide programma's op in bijvoorbeeld arrays a_j en $a_{j'}$. (De aard van het opslagmedium doet er eigenlijk niet toe.) Als een getal in a_j wordt gestopt en al in $a_{j'}$ zit drukken we het af. Omgekeerd, elke keer als een getal in $a_{j'}$ wordt gestopt en al in a_j zit, drukken we het af.

In beide gevallen kan dat getal dan uit beide arrays worden gehaald, om zo de arrays niet te hard te laten groeien. Maar in berekenbaarheidstheorie zijn efficiëntieargumenten niet belangrijk. Het gaat er alleen maar om, om te laten zien dát iets berekenbaar / beslisbaar / opsombaar is.

8c, p. 45: Laat $j/0$ de verzameling X opsommen, en laat $j'/0$ de verzameling Y opsommen. Voer j en j' pseudo-gelijktijdig uit door hun uitvoering te vervlechten. Drukt j een getal m af, druk dan alle paren

$$\{\ (m, y) \mid y \text{ reeds afgedrukt door } j'\}$$

af. Drukt j' een getal n af, druk dan alle paren

$$\{\ (x, n) \mid x \text{ reeds afgedrukt door } j\}$$

af. Op deze manier wordt $X \times Y$ opgesomd.

9a, p. 45: Met inductie: als X_1 en X_2 opsombaar zijn, dan is met het vorige onderdeel $X_1 \times X_2$ opsombaar. Als $X_1 \times X_2$ en X_3 opsombaar zijn, dan is met het vorige onderdeel $X_1 \times X_2 \times X_3$ opsombaar. Als $X_1 \times X_2 \times X_3$ en X_4 opsombaar zijn, dan is met het vorige onderdeel $X_1 \times X_2 \times X_3 \times X_4$ opsombaar. Enzovoort.

9b, p. 45: Allemaal door herhaald toepassen van het resultaat van Opgave 8 en 8c. Officieel gaat dit met volledige inductie naar het aantal gebruikte verzamelingstheoretische operatoren.

10a, p. 45: De verzameling van Java-programma's is een deelverzameling van de verzameling strings over een eindig alfabet, en de verzameling strings over een eindig alfabet is aftelbaar. Dus de verzameling van Java-programma's is aftelbaar.

10b, p. 45: Elk inputloos Java-programma kan maar één verzameling afdrukken. Het aantal mogelijke Java-programma's is aftelbaar, dus volgt het gestelde.

10c, p. 45: Volgt eigenlijk onmiddellijk uit het vorige onderdeel en de Church-Turing these. (Die stelt dat alle niet-interactieve computerprogramma's en berekeningsmechanismen even krachtig zijn.)

11a, p. 45: De verzameling van alle deelverzamelingen van de natuurlijke getallen, $2^{\mathbb{N}} = \{X \mid X \subseteq \mathbb{N}\}$, is overaftelbaar. Dit kan worden aangetoond met Cantor's diagonaalargument.

Elke *opsombare* verzameling $X \subseteq \mathbb{N}$ wordt noodzakelijkerwijs en per definitie opgesomd door een computerprogramma. De verzameling van alle computerprogramma's (in welke taal dan ook) is aftelbaar. Dus $\mathbb{N}$ bevat hoogstens aftelbaar veel opsombare deelverzamelingen.

Dus alle deelverzamelingen van $\mathbb{N}$, op aftelbaar oneindig veel na, zijn miet opsombaar.

11b, p. 45: De verzameling realisaties van zo'n experiment is overaftelbaar. Dit kan worden aangetoond met Cantor's diagonaalargument. Verder verloopt het argument als bij het vorige onderdeel.

N.B. willekeurige rijen van nullen en enen kunnen *wel* door een computer worden gerealiseerd als input van buitenaf is toegestaan (bewegingen van muis, of lichtinval, of temperatuur), of als tussentijds de computer mag worden ge-update met nieuwe software, of als echte asynchrone berekeningen mogen worden uitgevoerd. Een multicore processor op de PC voldoet overigens niet aan het laatste, want berekeningen worden daar gesynchroniseerd.)

12a, p. 46: Uit onderdeel 10c concluderen we dat "slechts" aftelbaar oneindig veel deelverzamelingen van $\mathbb{N}$ opsombaar zijn. Daarnaast leggen we het resultaat van Cantor's diagonalisatiestelling, welke zegt dat $\mathbb{N}$ overaftelbaar oneindig veel deelverzamelingen bevat. Als we deze twee feiten combineren, dan moeten we concluderen dat sommige deelverzamelingen van $\mathbb{N}$ niet opsombaar zijn.

12b, p. 46: Aftelbaar veel elementen (aftelbaar veel opsombare verzamelingen) verwijderen uit een overaftelbare verzameling (de collectie van alle deelverzamelingen van $\mathbb{N}$) levert een restant op dat nog steeds overaftelbaar is. (Stel niet: dan zou *aftelbaar* $\cup$ *aftelbaar* = *overaftelbaar*.)

13a, p. 46: Schrijf een (mogelijk gigantisch maar altijd eindig) case-statement die lidmaatschap in X verifieert.

13b, p. 46: Stel dat j_X over X kan beslissen, en j_Y over Y. Zij $n \in \mathbb{N}$ geven. Voor $j_X(n)$ en $j_Y(n)$ na elkaar uit. Dat kan, want zowel $j_X(n)$ als $j_Y(n)$ zijn beslissingsalgoritmen. Combineer na afloop beide antwoorden.

14, p. 46: Zij j een programma dat over X kan beslissen. Schrijf een programma j' als volgt

> Laat n gegeven zijn. Check of n even is. Als n even is, deel het door twee, en laat j controleren of $n/2 \in X$. Zo ja, laat j' dan 1 teruggeven, zo nee, laat j' dan 0 teruggeven. Als n oneven is, dan kan het zeker geen tweevoud zijn van een element in X, laat j' dan 0 teruggeven.

16, p. 48: – Iedere beslisbare verzameling is opsombaar: zij X beslisbaar. Vanwege beslisbaarheid is er een programma j die voor iedere $n \in \mathbb{N}$ kan bepalen of $n \in X$. Schrijf nu het volgende programma j' om X op te sommen:

> Laat j voor elk van de getallen $n = 0, 1, 2 \ldots$ achtereenvolgens bepalen of $n \in X$. Zo ja, druk dan n af. Op deze manier wordt X opgesomd.

– Als een verzameling zowel opsombaar als complementair opsombaar is, dan is die verzameling beslisbaar: laat j een programma zijn dat X opsomt, en laat j' een programma zijn dat $\mathbb{N}\backslash X$ opsomt. Schrijf nu het volgende programma k om te beslissen of $n \in X$:

> Laat n gegeven zijn. Voer j en j' pseudo-gelijktijdig uit door hun executie te vervlechten. Als j het getal n afdrukt, concludeer $n \in X$, en stop. Als j' het getal n afdrukt, concludeer dan dat $n \notin X$, en stop. Omdat
>
> $$n \in \mathbb{N} = X \cup (\mathbb{N}\backslash X)$$
>
> kunnen we er zeker van zijn dat één van beide programma's uiteindelijk n zal afdrukken.
>
> Let op:"vervlechten" betekent NIET dat het derde programma de twee eerste programma's vraagt afwisselend elementen van "hun" verzameling te laten afdrukken. Als bijvoorbeeld het eerste programma maar eindig veel elementen hoeft af te printen en het twee niet, dan zal, als het eerste programma al z'n getallen heeft geprint, het tweede programma vergeefs wachten op het moment dat het eerste programma z'n volgende getal afdrukt (dat er niet is, omdat het eerste programma alles al had afgedrukt wat er af te drukken viel). Vervlechten betekent: om beurten wat rekentijd geven.

1, p. 50: Eerst kleuren en daarna tegen-draaien (*turn'*), komt op hetzelfde neer als eerst draaien, en dan kleuren.

2a, p. 50: De eerste vijf stappen. De kolom "oriëntatie" geeft aan welke kant de mier op wijst. De kolom "draai" geeft aan in welk richting de mier draait. De kolom "stap" geeft aan in welk richting de mier stapt.

	vertrek	kleur	orientatie	draai	stap	aankomst
1.	(0, 0)	wit	N	R	E	(1, 0)
2.	(1, 0)	wit	E	R	S	(1, -1)
3.	(1, -1)	wit	S	R	W	(0, -1)
4.	(0, -1)	wit	W	R	N	(0, 0)
5.	(0, 0)	zwart	N	L	W	(-1, 0)

2b, p. 50: De eerste ongeveer $10,000$ stappen werkt de mier in een gebied binnen de rechthoek $[-25, 29] \times [-22, 22]$. Na ongeveer $10,000$ stappen wordt de rechthoek aan de linkerkant verlaten. De mier verlaat vervolgens in een semi-periodieke beweging het derde kwadrant en reist verder in zuid-westelijke richting. Het pad dat zo ontstaat wordt is ongeveer $3\sqrt{2} \approx 4.5$ eenheden breed en wordt ook wel "snelweg" ("highway") genoemd.

2c, p. 51: Vanwege het vorige antwoord kan geconcludeerd worden dat elk veld slechts een eindig aantal keren zal worden bezocht. (Het veld $(1, 1)$ blijkt het vaakst te worden bezocht, namelijk 33 keer. De velden op de "highway" blijken 1 tot en met 10 keer te worden bezocht.)

2d, p. 51: Nee, er zijn maxima voor de x- en y-coördinaten die ruim onder de 100 liggen. Om precies te zijn: de x coördinaat van het pad bedraagt maximaal 29, de y coördinaat van het pad bedraagt maximaal 23.

3a, p. 51: Forceer de mier in een zig-zag. Maak hiertoe bijvoorbeeld de diagonaal $(0, 0)$, $(1, 1)$, $(2, 2)$, $(3, 3)$, ... wit, en de daaronder liggende diagonaal $(1, 0)$, $(2, 1)$, $(3, 2)$, $(4, 3)$, ... zwart. In het algemeen is een oneindig dam- of schaakbordpatroon voldoende.

3b, p. 51: Na acht stappen op $(4, 4)$; na negen stappen op $(5, 4)$. Na N stappen op vakje

$$\begin{cases} \left(\frac{N}{2}, \frac{N}{2}\right) & \text{als } N \text{ even is,} \\ \left(\frac{N+1}{2}, \frac{N-1}{2}\right) & \text{anders.} \end{cases}$$

3c, p. 51: De snelste manier om weg te komen van de oorsprong is door zig-zaggen. De grootste afstand (hemelsbreed) wordt bereikt na elk oneven aantal stappen N. Immers (stelling van Pythagoras):

$$\sqrt{\left(\frac{N+1}{2}\right)^2 + \left(\frac{N-1}{2}\right)^2} = \sqrt{\frac{N^2+1}{2}} > \sqrt{\left(\frac{N}{2}\right)^2 + \left(\frac{N}{2}\right)^2} = \sqrt{\frac{N^2}{2}}.$$

4a, p. 51: Drie objecten kunnen op zes manieren worden gepermuteerd, zie Sectie A.1 op pagina 160. Dus behalve *turn-flip-move* is er *turn-move-flip*, *move-turn-flip*, *move-flip-turn*, *flip-move-turn*, en *flip-turn-move*. Kortweg: TFM, TMF, MFT, MTF, FMT, en FTM.

4b, p. 51: Voor TFM zijn er 6 permutaties, en voor CFM ook. Dus in totaal 12 mogelijke regimes.

4c, p. 52: Bekijk Fig. 24 op de volgende pagina. Voor elk van de 12 regimes TMF, MTF, ... laten we de mier vanaf het het startveld (groen) één stap doen, en vergelijken dan het resultaat. Als

TFM = *turn-flip-move* (het conventionele regime);
CFM = *counterturn-flip-move* = *turn'-flip-move*,

dan zien we qua gegenereerde patronen dat TFM = FCM; MTF = MFC; FTM = CFM en MFT = MCF. De onderste vier patronen zijn allemaal verschillend.

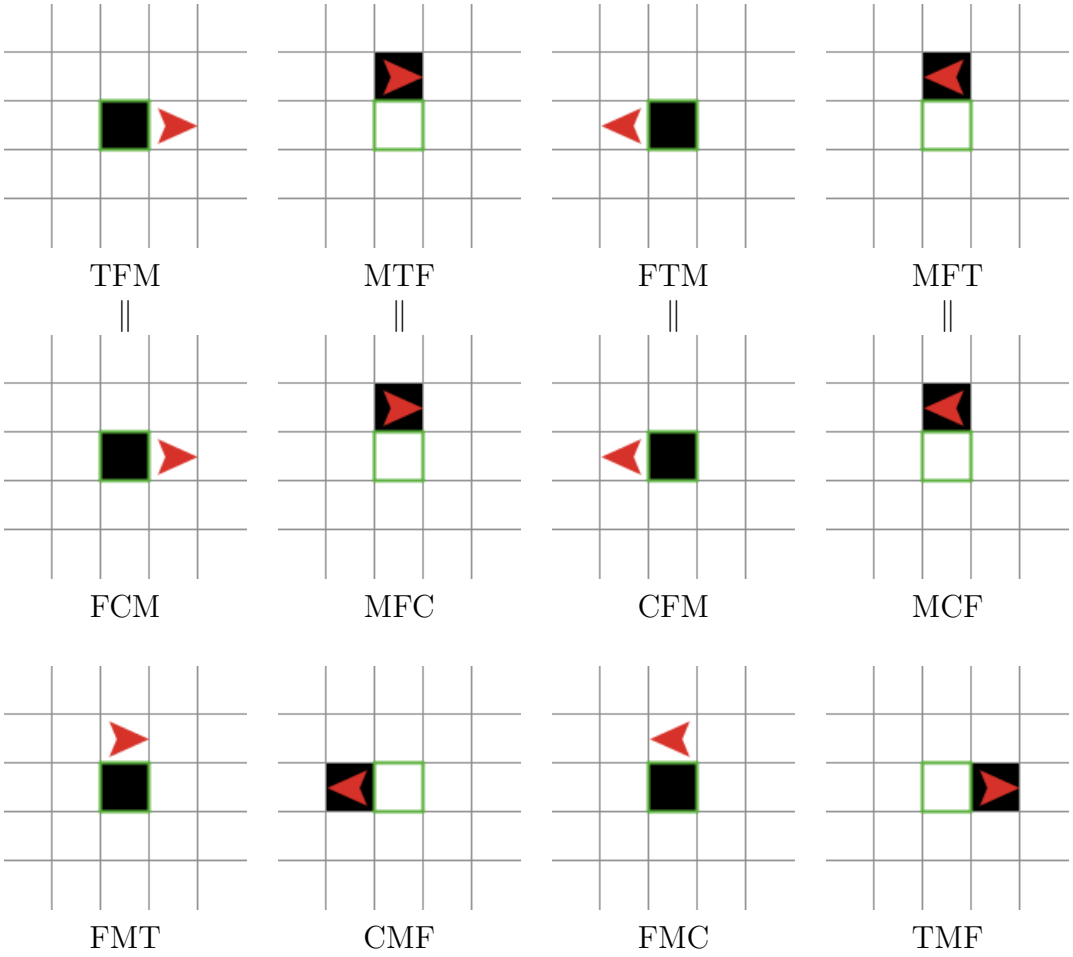


Fig. 24: Elk van de twaalf regimes na één stap.

Van de vier identieke patroon-paren moet nog wel worden onderzocht of deze *toevallig* na één stap danwel na *elk aantal* stappen identiek zijn. We bekijken het eerste paar: TFM versus FCM. Omdat in beide regimes de letter M achteraan staat, hoeven we eigenlijk alleen te kijken naar het effect van TF versus FC. Snel is in te zien dat, voor zowel witte als zwarte cellen, eerst draaien (T) en dan omkleuren (F) hetzelfde is als eerst omkleuren (F) en dan tegendraaien (C). Dus TFM $\equiv$ FCM. Voor de overige paren kan ook zo'n redenering worden gegeven, immers, voor al deze patroonparen staat M op dezelfde plek. Dus ook MTF $\equiv$ MFC; FTM $\equiv$ CFM en MFT $\equiv$ MCF.

4d, p. 52: Bekijk Fig. 25 op de pagina hierna. Deze figuur suggereert dat de resulterende patronen, op spiegeling, translatie en rotatie na, identiek zijn. Meteen is in te zien dat dit waar is, als in een regime waarin de letter de letter T voorkomt, elk voorkomen van T vervangen wordt door een C. Het gegeneerde patroon is dan immers gespiegeld in de y-as. Dus we kunnen ons beperken tot de zes regimes waarin een T voorkomt.

Het is genoeg om te laten zien dat de "kies 2 uit 6" = drie mogelijke verwisselingen FM, TM en TF dezelfde patronen opleveren modulo spiegeling, translatie en rotatie. Immers, elke permutatie kan worden opgebouwd uit successievelijke verwisselingen.

FM $\leftrightarrow$ MF Als je eerst het huidige vakje kleurt en daarna vooruit stapt, dan is dat hetzelfde als eerst vooruit stappen en daarna kleuren, behalve dat je het hele patroon een stap verschoven bekijkt. Dus deze twee regels verschillen alleen door een translatie.

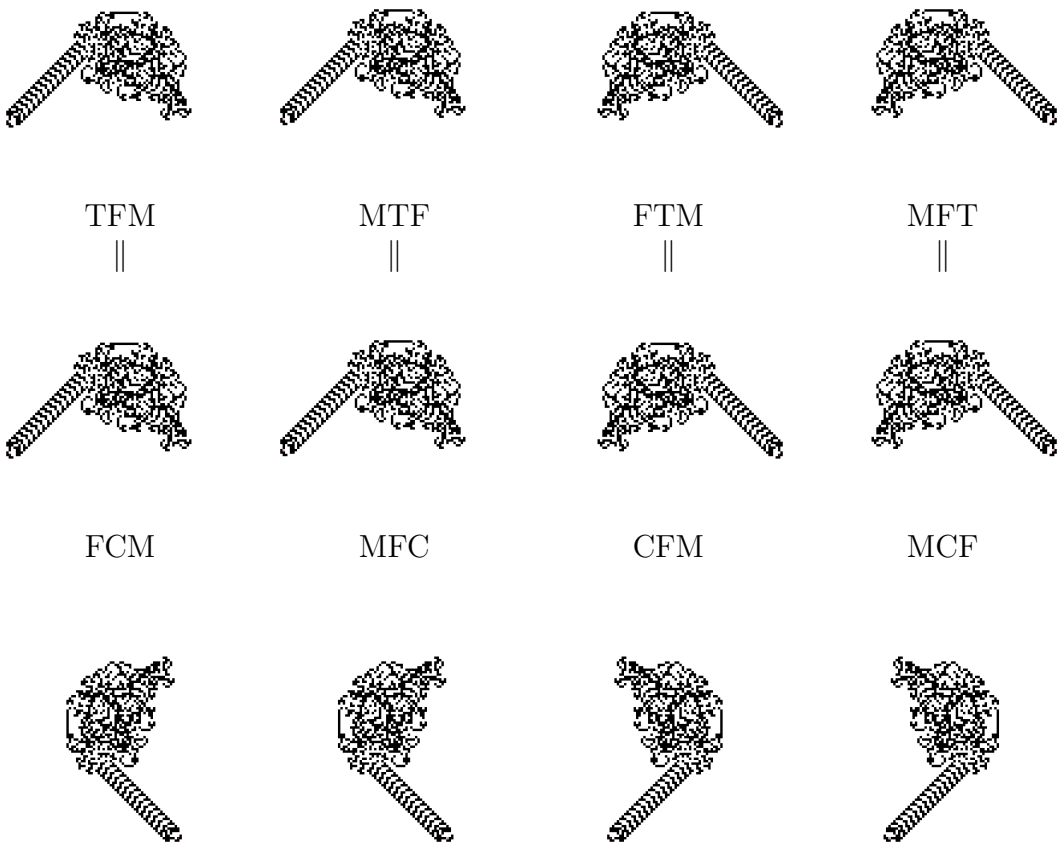


Fig. 25: Elk van de twaalf regimes na een groot aantal stappen.

TM ↔ MT Eerst draaien en dan stappen, en eerst stappen en dan draaien, leveren hetzelfde pad op als je het bord een kwartslag roteert. Dus deze twee regels zijn elkaars rotatie.

TF ↔ FT Draaien verandert enkel de oriëntatie, kleuren verandert enkel het huidige vakje. De volgorde van die twee maakt voor het globale patroon niets uit, ze zijn dus direct verwisselbaar of via een spiegel-symmetrie te herleiden.

Omdat met deze drie elementaire verwisselingen elke willekeurige volgorde van T, F en M in een andere kan worden omgezet, en elke verwisseling overeenkomt met een eenvoudige symmetrie, volgen alle zes regimes hetzelfde patroon modulo rotatie, spiegeling of translatie. En daarmee volgt de equivalentie van alle twaalf regimes.

5a, p. 52: Door achteruit te kijken kan de mier zien waar hij vandaan kwam en hoe hij stond. Dit proces kan willekeurig vaak worden herhaald.

Noot: dat in de wandeling vooruit oude paden overschreven kunnen worden door nieuwere paden maakt daarbij niet uit omdat bij achteruitgang eerst moet worden teruggelopen over de nieuwere paden. Als eenmaal is teruggelopen over nieuwere paden, komen oudere paden vanzelf weer tevoorschijn.

5b, p. 52: Zie de voorpagina als een groot raster, bijvoorbeeld 1440x900. Elk punt in het raster is zwart of wit. Plaats de mier ergens op het raster en laat deze achteruitlopen totdat de pagina

voldoende onherkenbaar is. Sla het schijnbaar willekeurige patroon op, plus de locatie en oriëntatie van de mier. Tover op elk gewenst moment de krantenpagina terug door de mier vooruit te laten lopen op het schijnbaar willekeurige patroon.

Werkt dit niet omdat de mier bepaalde delen van de krantenpagina "mijdt", voeg dan meer papierkleuren toe en/of verander de draaihoek van 90^0 naar 45^0 of een andere draaihoek—alles blijft deterministisch en reversibel. Succes gegarandeerd.

5c, p. 52: Zet die mier op een pixel met coördinaten (x, y) en laat hem N stappen achteruitlopen, N liefst een beetje groot. Stel, een plausible maximale waarden van N is 100. Omdat de mier op zijn startpunt in vier oriëntaties kan worden neergezet, is het aantal sleutels dus $X \times Y \times 4 \times 1000$. Dus een beetje hacker kan zo'n versleuteld image wel ontcijferen.

Anders wordt het als elk pixel meerdere kleuren kan aannemen, en een mier voor elke kleur een regel moet definiëren (zie Opgave 11 op pagina 54). Bij elke regel is er de keuze om die mier links- of rechts te laten afslaan én de keuze voor een volgende kleur. Het aantal sleutels neemt nu exponentieel toe.

6a, p. 52: Start de mier op een volledig blank veld, en laat deze lopen tot dat een snelweg ontstaat. Veeg dan alles van het "domein" (het alreeds betreden gebied), behalve de ontstane snelweg uit, en laat de mier vervolgens achteruit lopen. De mier zal dan de gebouwde snelweg terug afbreken, totdat bijna geen vakjes meer overblijven. Pik die configuratie er uit met de minste zwarte vakjes.

Deze methode moet worden verfijnd door bij te houden tot waar het initiële pad (die er dus uitziet als een blob) wordt gewist. Omdat de mier de snelweg bouwt in een periode van 104 stappen, zijn er ook 104 manieren om de snelweg te laten beginnen. Dus de werkelijk kleinste kiem kan worden gevonden door voor al deze mogelijke prefix-verwijderingen de mier te laten teruglopen, en voor elk van deze 104 backtraces de kleinste configuratie er uit te pikken.

6b, p. 52: Door systematisch te proberen (middels een script), kwam ik zelf uit op [t.b.a.]

7a, p. 52: Elk vakje kan twee verschillende toestanden aannemen, te weten wit en zwart. Er zijn $M \times N$ vakjes, dus we moeten $M \times N$ keer een kleur kiezen. Dat kan op 2^{MN} verschillende manieren. Dat de randen in elkaar overlopen maakt niets uit voor de berekening.

7b, p. 52: Er zijn $M \times N$ vakjes, en op elk vakje kan de mier vier posities aannemen, te weten alle vier de windrichtingen. Dus in totaal $4MN$ verschillende posities.

7c, p. 52: In totaal $4MN2^{MN} = MN2^{MN+2}$ verschillende toestanden.

7d, p. 52: De mier en het grid nemen kunnen samen "maar" eindig veel verschillende toestanden aannemen. Dus in een niet eindigende run zal de grid/mier-combinatie ooit een keer in dezelfde toestand terugkomen. Vanaf dan is er periodiek gedrag.

Let op: als je redeneert dat het voldoende is als de mier ooit een keer op hetzelfde vakje terechtkomt, dan is dat niet voldoende. De mier kan in een andere oriëntatie arriveren, en bovendien kan op zo'n moment het grid anders ingevuld zijn dan bij het eerste bezoek van dat vakje. Er is pas periodiciteit als op enig moment een integrale grid/mier-toestand wordt herhaald.

7e, p. 52: Er zijn maar $MN2^{MN+2}$ verschillende toestanden, dus dat is ook een bovengrens. Er zijn uiteraard kleinere bovengrenzen, en er zijn zelfs kleinere bovengrenzen waarvan het bestaan kan worden aangetoond, echter deze maken gebruik van complex formules die tijdens een wandeling niet veranderen. Deze formules heten *invarianten*.

8, p. 53: De fout zit hem in de redeneerstap dat, zodra de mier arriveert in een eerder geziene omgevings-toestand, deze in een cyclus van dergelijke omgevings-toestanden terecht zal komen. Maar dit is pas zo als die 5×5 omgevings-toestand ook werkelijk alle informatie bevat—en dat is niet zo: de invulling van het rooster buiten die 5×5 omgeving gaat hoogstwaarschijnlijk ook nog invloed hebben op dit toekomstige pad. Dus, het is waar dat in een run sommige omgevings-toestanden oneindig vaak voorkomen. Maar hieruit volgt niet noodzakelijk dat op een gegeven moment een cyclus van omgevings-toestanden ontstaat.

9, p. 53: Op zich is dit een goed bewijs. Helaas werd vergeten dat, in de formulering van het highway-vermoeden, een start-configuratie altijd eindig moet zijn. En Gajardo *et al.*'s ants starten op een oneindige configuratie. Bijvoorbeeld, bedradingen tussen logische circuits worden door Gajardo *et al.* gerepresenteerd door oneindige zwarte diagonalen [13]. Het bewijs houdt stand zodra men kan aantonen dat elke Turing-machine kan worden geëmuleerd op een eindige start-configuratie.

10a, p. 53: Simpel: $180 \cdot 0 - 90 = -90$, en $180 \cdot 1 - 90 = 90$.

10b, p. 54: Als $\alpha = 0$, dan vindt er geen diffusie plaats, en veranderden velden alleen van kleur door het flippen van wit naar zwart, en omgekeerd.

10c, p. 54: In mijn simulatie zie ik dat het inwendige van dit gebied convergeert naar grijswaarden dicht bij 0.5. Bijgevolg zal de mier in het inwendige rechtdoor lopen. De kleuren zullen daar ook niet zo veel veranderen. Immers, $1 -$ een grijswaarde dicht bij 0.5, resulteert wéér in een grijswaarde dicht bij 0.5. Deze grijswaarden zullen dankzij diffusie ook nog eens nivelleren naar waarden nog dichter bij 0.5.

Aan de randen "botst" de mier tegen nagenoeg witte velden aan (misschien een heel klein beetje lichtgrijs door diffusie), en zal de mier voortdurend naar rechts draaien totdat deze gericht staat naar het inwendige. Daarna wandelt de mier weer min of meer (met wat flauwe bochten) rechtdoor in het inwendige, om ongeveer aan de overkant op witte cellen te stuiten. Zal de mier zich weer in een aantal stappen omkeren naar het inwendige.

10d, p. 54: Waarschijnlijk niet, omdat het ontstaan van een snelweg vereist dat de mier periodiek gedrag gaat vertonen, wat in een continue opzet onwaarschijnlijk is. Deze laatste observatie is een plausibel vermoeden, en geen feit.

10e, p. 54: Zowel de continue als de discrete mier keren in de regel weer naar binnen als ze de rand van hun "domein" (het alreeds betreden gebied) bereiken. Dat hebben ze samen gemeen. Het rechtdoor lopen van de continue mier in het inwendige kan worden gezien als een "samenvatting" van het gedrag van de discrete mier in het inwendige. Ook de discrete mier loopt grosso modo, met slingerbewegingen, rechtdoor in het inwendige. Dus dat hebben ze ook samen gemeen.

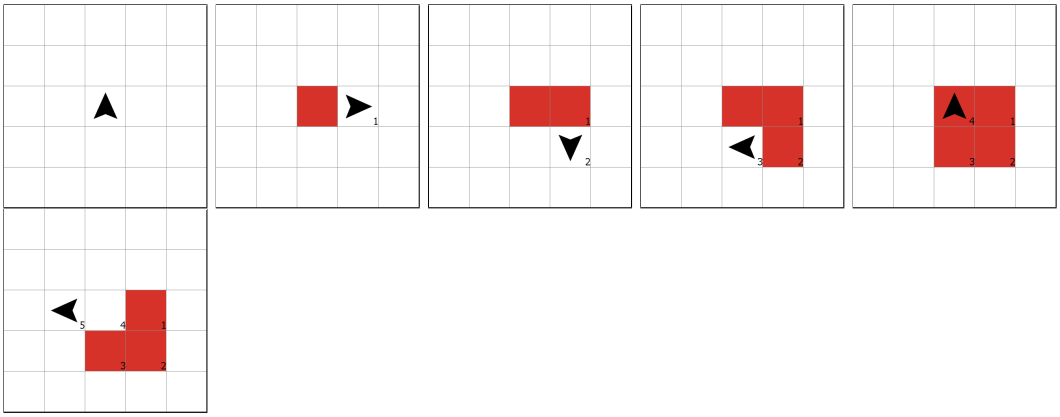
Wat de mieren hoogstwaarschijnlijk niet gemeen hebben, is het ontstaan van een snelweg: bij de discrete mier ontstaat op een blank veld een snelweg, bij de continue mier hoogstwaarschijnlijk

niet. Voor het ontstaan van een snelweg is het dus klaarblijkelijk nodig dat het systeem op enig moment in periodiek gedrag vervalt, wat in het discrete geval met minder vrijheid (i.e., alleen zwart/wit en alleen 90/−90) klaarblijkelijk makkelijker zal gebeuren dan in een continu systeem. In een continu systeem zal er waarschijnlijk te veel vrijheid zijn om periodiek gedrag te laten ontstaan en te laten voortbestaan.

11a, p. 54: Het spoor van *RL*:

	vertrek	kleur	oriëntatie	draai	stap	aankomst
1.	(0, 0)	wit	N	R	E	(1, 0)
2.	(1, 0)	wit	E	R	S	(1, -1)
3.	(1, -1)	wit	S	R	W	(0, -1)
4.	(0, -1)	wit	W	R	N	(0, 0)
5.	(0, 0)	rood	N	L	W	(-1, 0)

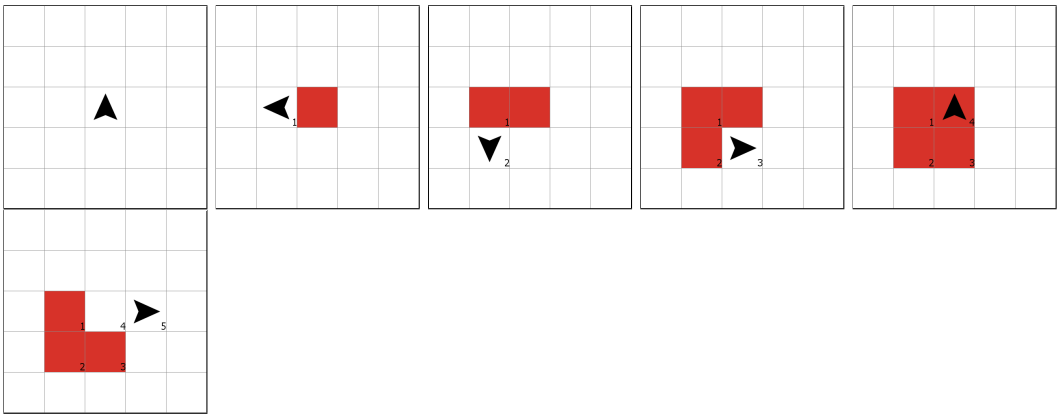
In de tijd:



11b, p. 54: Het spoor van *LR*:

	vertrek	kleur	oriëntatie	draai	stap	aankomst
1.	(0, 0)	wit	N	L	W	(-1, 0)
2.	(-1, 0)	wit	W	L	S	(-1, -1)
3.	(-1, -1)	wit	S	L	E	(0, -1)
4.	(0, -1)	wit	E	L	N	(0, 0)
5.	(0, 0)	rood	N	R	E	(1, 0)

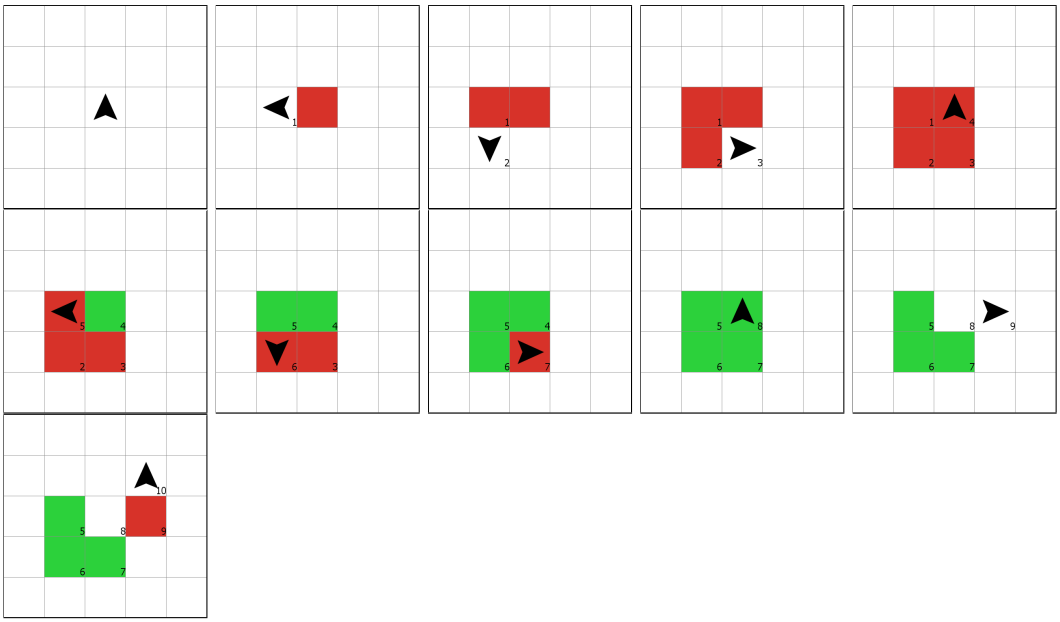
In de tijd:



11c, p. 54: Het spoor van LLR:

	vertrek	kleur	oriëntatie	draai	stap	aankomst
1.	(0, 0)	wit	N	L	W	(-1, 0)
2.	(-1, 0)	wit	W	L	S	(-1, -1)
3.	(-1, -1)	wit	S	L	E	(0, -1)
4.	(0, -1)	wit	E	L	N	(0, 0)
5.	(0, 0)	rood	N	L	W	(-1, 0)
6.	(-1, 0)	rood	W	L	S	(-1, -1)
7.	(-1, -1)	rood	S	L	E	(0, -1)
8.	(0, -1)	rood	E	L	N	(0, 0)
9.	(0, 0)	groen	N	R	E	(1, 0)
10.	(1, 0)	wit	E	L	N	(1, 1)

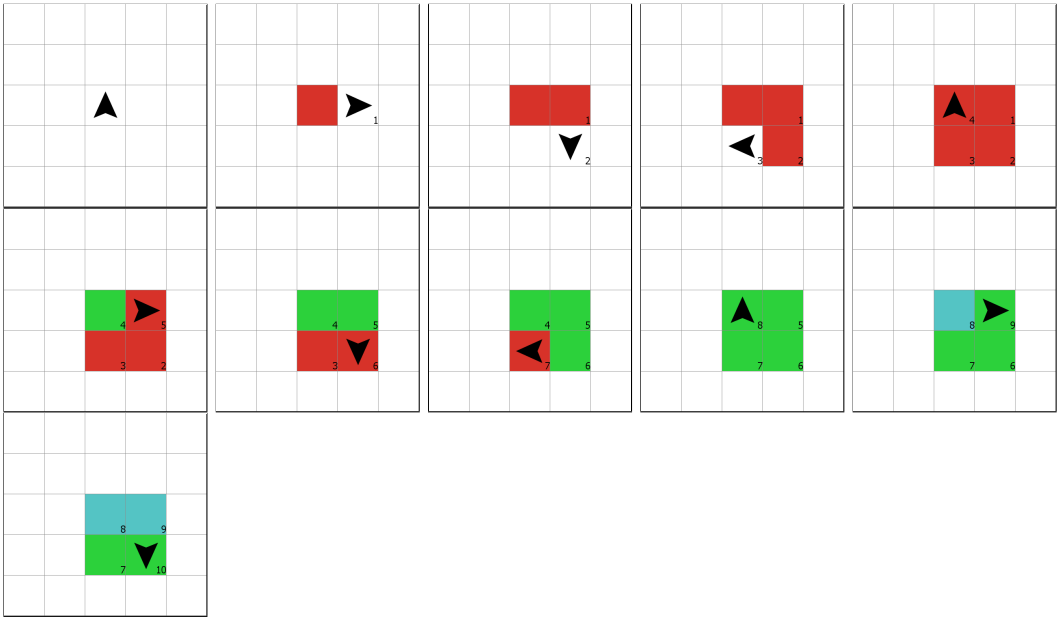
In de tijd:



11d, p. 54: Het spoor van $RRRL$:

	vertrek	kleur	oriëntatie	draai	stap	aankomst
1.	(0, 0)	wit	N	R	E	(1, 0)
2.	(1, 0)	wit	E	R	S	(1, -1)
3.	(1, -1)	wit	S	R	W	(0, -1)
4.	(0, -1)	wit	W	R	N	(0, 0)
5.	(0, 0)	rood	N	R	E	(1, 0)
6.	(1, 0)	rood	E	R	S	(1, -1)
7.	(1, -1)	rood	S	R	W	(0, -1)
8.	(0, -1)	rood	W	R	N	(0, 0)
9.	(0, 0)	groen	N	R	E	(1, 0)
10.	(1, 0)	groen	E	R	S	(1, -1)

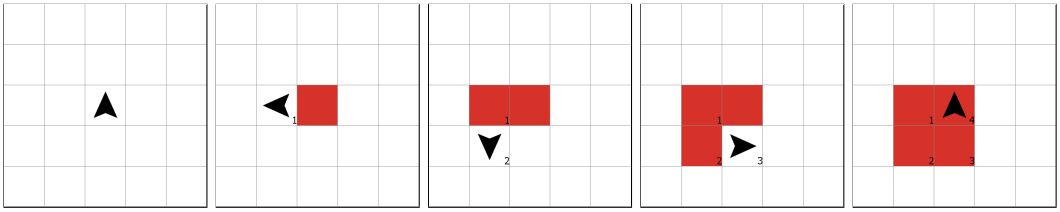
In de tijd:

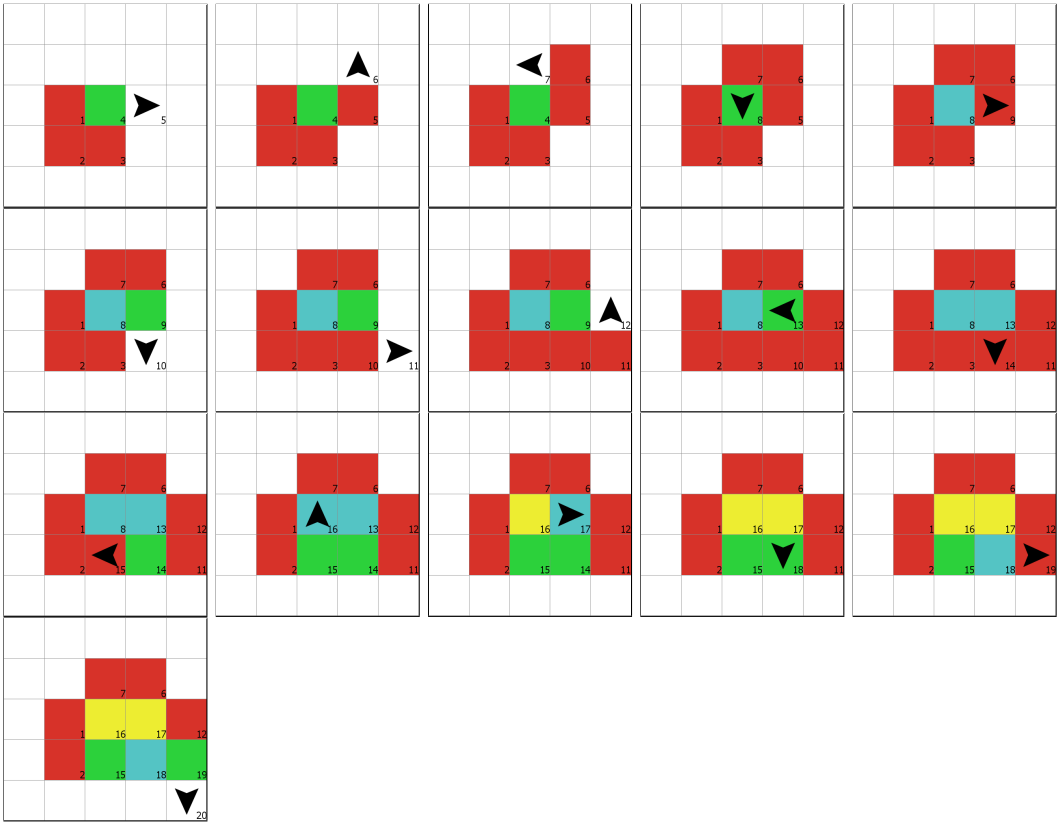


11e, p. 54: Het spoor van $LRLRLR$:

	vertrek	kleur	oriëntatie	draai	stap	aankomst
1.	(0, 0)	wit	N	L	W	(-1, 0)
2.	(-1, 0)	wit	W	L	S	(-1, -1)
3.	(-1, -1)	wit	S	L	E	(0, -1)
4.	(0, -1)	wit	E	L	N	(0, 0)
5.	(0, 0)	rood	N	R	E	(1, 0)
6.	(1, 0)	wit	E	L	N	(1, 1)
7.	(1, 1)	wit	N	L	W	(0, 1)
8.	(0, 1)	wit	W	L	S	(0, 0)
9.	(0, 0)	groen	S	L	E	(1, 0)
10.	(1, 0)	rood	E	R	S	(1, -1)
11.	(1, -1)	wit	S	L	E	(2, -1)
12.	(2, -1)	wit	E	L	N	(2, 0)
13.	(2, 0)	wit	N	L	W	(1, 0)
14.	(1, 0)	groen	W	L	S	(1, -1)
15.	(1, -1)	rood	S	R	W	(0, -1)
16.	(0, -1)	rood	W	R	N	(0, 0)
17.	(0, 0)	blauw	N	R	E	(1, 0)
18.	(1, 0)	blauw	E	R	S	(1, -1)
19.	(1, -1)	groen	S	L	E	(2, -1)
20.	(2, -1)	rood	E	R	S	(2, -2)

In de tijd:





12a, p. 54: RL betekent: naar rechts draaien als de cel-counter gelijk is aan 0, en naar links draaien als de cel-counter gelijk is aan 1. Dit is hetzelfde te zeggen: naar rechts draaien als de cel-counter even is, en naar links draaien als de cel-counter oneven is.

12b, p. 54: Draaiingen zijn modulo de lengte van de string. Dat verandert niet als de string twee, drie of N keer zo lang wordt gemaakt.

13a, p. 55: Gebruik kleuren om het aantal bezoeken te tellen:

beweging:	R	R	R	...	R	L
kleur: 1	2	3	...	N	$N+1$	

Het gaat hier inderdaad om de instructie-string $R^N L$, en niet om bijvoorbeeld $R^{N-1}L$ of $R^{N+1}L$. Immers, bij aanvang is de start-cel al meteen 1x bezocht.

13b, p. 55: Dit is blijkt moeilijk te voorspellen zonder daadwerkelijk simulaties te runnen. De mier zal veel vaker rechtsaf slaan dan linksaf.

- Voor even N zal de mier zal een "vlek" opbouwen van genummerde tegels, met in het midden waarschijnlijk de hoogst genummerde tegels. Zodra de mier van deze vlek af loopt, komt deze op een nultegel terecht en zal deze (weer) rechtsaf slaan, terug het gebied in met de genummerde tegels, waar de mier, afhankelijk van de waarde op de tegels, links- dan wel rechtsaf slaat. We zien een schijf-achtige vlek ontstaan die langzaam groeit.
- Voor oneven N zal de mier snel vervallen in het bouwen van snelwegen die ook verschillende kanten op kunnen lopen, echter altijd in hoeken van 45^0 of een veelvoud daar van.

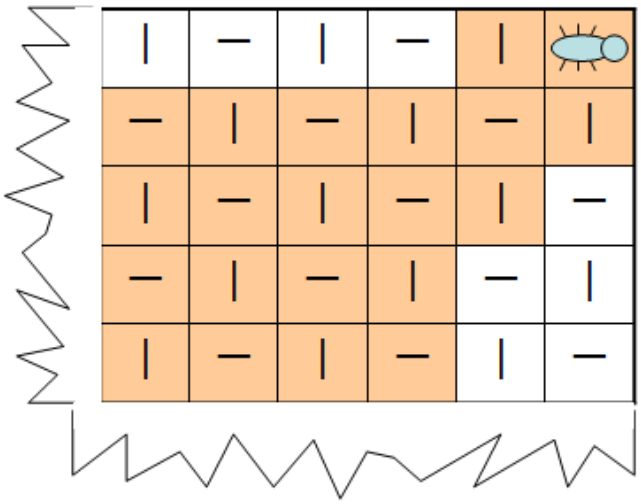
Waarom het gedrag afhankelijk is van N's pariteit is, voorzover ik kan nagaan, onbekend.

14a, p. 55: Een eindig gebied plus een eindig wandelprogramma impliceert eindig veel configuraties, en bij een oneindige run dus noodzakelijk periodiek gedrag: de mier gaat zich na verloop van tijd herhalen. Dus elke bezochte cel is, zoals dat heet, *recurrent*: is een cel eenmaal bezocht, dan wordt deze cel oneindig vaak bezocht.

Cellen vormen een schaakbord van horizontaal en verticaal bezochte cellen. Dit kun je controleren door de cellen te labellen met H en V, en vervolgens te controleren dat de mier, bij voortbewegen, zich inderdaad houdt aan de op de cellen aangegeven oriëntatie.

De meest Noord-Oost bezochte cel is z.b.d.a.[57] Een H-cel. De mier komt daar dus altijd van links binnen.

De NO cel wordt afwisselend wit en zwart gekleurd, dus de mier slaat afwisselend links- en rechtsaf. Linksaf bij de NO-cel kan niet. Tegenspraak. Dus de oorspronkelijke aanname, die van begrensdheid, is onwaar.n

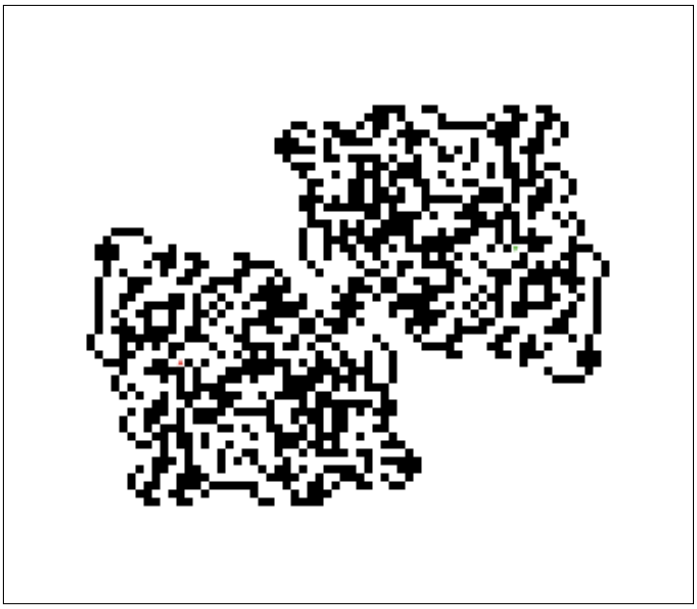


Anders gezegd, van de bezochte cellen ligt er één het meest Noord-Oost. NO moet wit zijn, want bij een zwarte cel zou de mier linksaf slaan. Dus kleurt de mier NO zwart en slaat rechtsaf. Bij de eerstvolgende doorkomst van NO, die noodzakelijk weer horizontaal en dus weer van links is, stapt de mier op de zwartgemaakte NO en zal nu wel linksaf slaan. Tegenspraak.

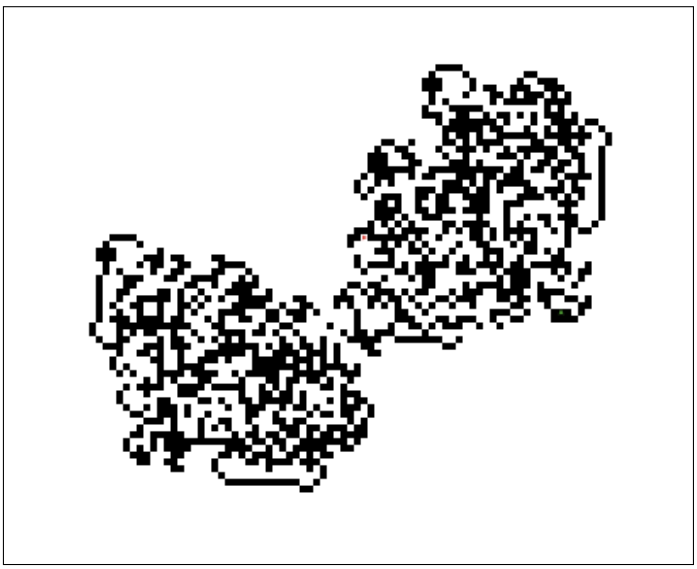
14b, p. 55: Het H/V-patroon blijft intact, dus dat kan worden hergebruikt. Als de instructie-string homogeen is, bijvoorbeeld gelijk is aan L, of LL, of RRR, in het algemeen uit L^+ of R^+, dan is het evident dat de mier evident in de rondte zal blijven cirkelen en is er dus een rechthoek waaruit de mier niet zal ontsnappen. Als de instructie-string heterogeen is, dat wil zeggen als L er R er beide in voorkomen, doet het Cohen / Kong-argument weer opgeld. Dus de veralgemeniseerde versie van Langton's mier is onbegrensd als en slechts als haar instructie-string heterogeen is.

15a, p. 55: De twee mieren heffen elkaars spoor op.

[57] Zonder beperking der algemeenheid, dat wil zeggen dat het argument niet wordt beperkt door vanaf dit moment alleen H-cellen te beschouwen. Het argument verloopt analoog voor V-cellen.



15b, p. 55: Ook in dit geval heffen de twee mieren heffen ook elkaars spoor op.



16, p. 56: Het kunnen emuleren van logische schakelingen gecombineerd met een natuurlijke klok (de stappen die de mier neemt en het daarbij veranderende grid) betekent zo veel dat Langton's ant krachtig genoeg is om willekeurige berekeningen uit te voeren. Bepaalde (maar niet alle) problemen gerelateerd aan Langton's ant zijn daar mee onbeslisbaar. Een voorbeeld van een dergelijk probleem is: "beslis, gegeven een willekeurige eindige beginconfiguratie, of de ant ooit de tegel $(100, 100)$ zal bezoeken".

Zie ook het college en de slides aangaande cellulaire automaten. Daar wordt aangetoond dat Conway's game of life Turing-volledig is. Turing-volledigheid volgt daar ook uit het feit dat in (of met) Conway's game of life ook NIET-, EN- en OF-poorten kunnen worden gebouwd.

17a, p. 57: Dit probleem is opgelost, want we kunnen gewoon de mier laten lopen, en zien dat er een snelweg ontstaat. Dit probleem is ook triviaal beslisbaar, want er hoeft maar één begin-configuratie te worden gecontroleerd, namelijk de lege begin-configuratie.

17b, p. 57: Het snelweg-vermoeden kan worden beschouwd als opgelost, als één van de volgende gebeurtenissen zich voordoet.

1. Er kan, op welke manier dan ook, worden aangetoond dat alle begin-configuraties een snelweg genereren. Daarmee is dan het vermoeden bevestigd.
2. Er wordt een tegen-voorbeeld gevonden. Er kan, met andere woorden, worden aangetoond dat er een begin-configuratie bestaat, of (sterker) zelfs aan te wijzen is, welke geen snelweg genereert. Deze begin-configuratie is dan een tegen-voorbeeld voor het snelweg-vermoeden.

17c, p. 57: Ja, het snelweg-vermoeden is beslisbaar, want het vermoeden betreft één vraag, en daarop is het antwoord altijd "ja" of "nee". Als het antwoord "ja" is, dan is het programmaatje dat (altijd) "ja" afdrukt (of teruggeeft) een (triviale maar) correcte beslisser. Analoog, als het antwoord "nee" is, dan is het programmaatje dat "nee" afdrukt een correcte beslisser.

Het snelweg-vermoeden is gelijkwaardig aan de veronderstelling dat het 'ja"-programmaatje de correcte beslisser is. De kennis dat het snelweg-vermoeden beslisbaar is, geeft ons dus geen aanvullend inzicht in de status van het snelweg-vermoeden.

17d, p. 57: Het snelweg-probleem kan worden beschouwd als opgelost, als één van de volgende gebeurtenissen zich voordoet.

1. Er kan worden aangetoond dat het probleem beslisbaar is, dat wil zeggen, er kan worden aangetoond dat er een algoritme bestaat welke, voor elke eindige begin-configuratie, kan bepalen of deze een snelweg genereert.
2. Er kan aangetoond worden dat het probleem onbeslisbaar is; bijvoorbeeld door een reductie-argument.

17e, p. 57: Het idee is Turing's oorspronkelijke stop-probleem te reduceren naar het snelweg-probleem. Dat wil zeggen, het idee is om een reductie-algoritme F te formuleren, of althans in eerste aanzet te schetsen, dat elk programma P met invoer x omzet in een eindige initiële configuratie $C_{P,x}$ zodanig dat

$$P(x) \text{ stopt} \Leftrightarrow C_{P,x} \text{ produceert een snelweg.}$$

Hoe zou zoiets werken? Wel, het idee is dat F de instructies van P codeert door elke instructie om te zetten in een blok cellen op het rooster, waarbij de volgorde van de blokken de instructievolgorde weerspiegelt en markeringen het besturings-pad aangeven. De positie en oriëntatie van de mier representeert dan de executie-pointer (de op dat moment actieve instructie). Het algoritme F draagt ook zorg voor de codering van de invoer x, dus deze krijgt ook een plek op het rooster. De beweging van de mier door deze cellen simuleert dan geheugen-bewerkingen en conditionele vertakkingen. Voor het stop-criterium wordt een speciaal halt-blok ontworpen, zodanig dat als de mier deze bereikt, een snelweg-wordt gebouwd. Omgekeerd, als $P(x)$ niet stopt, blijft de mier uit de buurt van het snelweg-blok. Kortom, $C_{P,x}$ wordt mechanisch opgebouwd door F op basis van het programma P en de invoer x, inclusief instructie-blokken, geheugen, vertakkings-paden en het halt-blok. Omdat het algoritme F voor elke programma/invoer-combinatie kan worden toegepast, zou dit een algoritmische en uniforme reductie geven. Als dit allemaal geloofwaardig wordt geformuleerd, dus als F geloofwaardig is, kan een reductie-argument worden toegepast om aan te tonen dat het snelweg-probleem onbeslisbaar is.

17f, p. 57: Hoewel de mier Turing-compleet is, is het niet evident om een uniform algoritme, F, te construeren dat elke programma/invoer-combinatie automatisch omzet in een beginconfiguratie $C_{P,x}$.

i) Ten eerste moet de simulatie van P semantisch volledig overeenkomen met P zelf: elke instructie, vertakking en geheugenbewerking moet exact worden nagebootst door de beweging van de mier, zonder kleine afwijkingen die de uitkomst zouden kunnen veranderen.

ii) Daarnaast moet er een ondubbelzinnig halting-mechanisme bestaan dat alléén geactiveerd wordt wanneer de gesimuleerde berekening echt stopt, en nooit bij andere toestanden zoals oneindige lussen of vastlopende patronen.

iii) Het computationele gebied moet geïsoleerd blijven zodat de globale effecten van de mier, die elders in het rooster cellen kan veranderen, de simulatie niet verstoren of per ongeluk een snelweg veroorzaken.

iv) Verder moet elke configuratie eindig zijn terwijl het systeem in principe onbegrensde berekeningen kan uitvoeren; dat vereist een slimme manier om potentieel oneindig geheugen te simuleren binnen een eindige startconfiguratie.

v) Ook moet de omzetting volledig mechanisch zijn: het algoritme dat de initiële configuratie genereert moet uniform werken voor elk programma en invoer zonder handmatig afgestelde *ad-hoc* patronen.

vi) Ten slotte is het zelfs denkbaar dat het snelweg-probleem zich bevindt op een complexiteitsniveau dat vergelijkbaar is met bekende onbewijsbare uitspraken, zoals het Collatz-vermoeden of het stopgedrag van bepaalde cellulaire automaten. In dat geval zou een bewijs van onbeslisbaarheid redeneringen kunnen vereisen die zelf de bewijskracht van het gebruikte formele systeem te boven gaan.

17g, p. 57: Als het snelweg-vermoeden juist blijkt, dan is hiermee het snelweg-probleem opgelost. Immers, voor elke begin-configuratie is het antwoord dan positief, en een beslis-algoritme hoeft alleen maar voor elke begin-configuratie een “ja” terug te geven.

17h, p. 57: Zonder een concreet algoritme kan niet veel worden gedaan. Een hypothetisch algoritme kan namelijk niet worden toegepast.

17i, p. 57: Met een concreet beslis-algoritme kan, voor elke eindige begin-configuratie, worden beslist of deze configuratie een snelweg genereert. Er zijn echter aftelbaar oneindig veel begin-configuraties dus hiermee kan het snelweg-vermoeden niet binnen een eindige tijd worden opgelost.

17j, p. 57: Hier is een semi-beslisalgoritme voor het snelweg-vermoeden:

> Enumereer systematisch alle begin-configuraties (bijvoorbeeld in toenemende grootte). Meteen nadat een configuratie gegenereerd is, wordt met het beslis-algoritme bepaald of deze een snelweg genereren. Er kunnen nu twee dingen gebeuren:
>
> 1. We komen een begin-configuratie tegen die geen snelweg genereert. In dat geval weten we genoeg. We stoppen, en het snelweg-vermoeden is weerlegd.

2. In het andere geval blijven we maar configuraties tegenkomen die een snelweg genereren. Dus in dat geval blijven we maar wachten, niet wetende of er ooit nog een configuratie komt die géén snelweg genereert.

De onzekerheid of er, tijdens het wachten, ooit nog uitsluitsel komt is een typische eigenschap van semi-beslisalgoritmen: als het antwoord negatief is, zal het algoritme na lang of kort rekenen uitsluitsel geven; als het antwoord positief is, zal het algoritme blijven lopen zonder ooit uitsluitsel te geven. (Meestal wordt het andersom geformuleerd: als het antwoord positief is, zal het algoritme uitsluitsel geven; als het antwoord negatief is, zal het algoritme blijven lopen zonder ooit uitsluitsel te geven.) Het probleem is dat we als buitenstaander niet weten welke scenario geldt, zodat we niet weten of we "voor niets" wachten.

18, p. 57: Het vraagstuk of een RLR mier op een blank grid een snelweg produceert is beslisbaar. De motivatie van dit antwoord loopt langs dezelfde lijnen als de uitleg van het antwoord op Vraag 17c op pagina 57.

19a, p. 57: Langton's ant kent geen inwendige toestanden, of, als je het precies wil zeggen: kent slechts één inwendige toestand. Een turmite kent typisch, net zoals een Turing-machine, meerdere inwendige toestanden.

19b, p. 57: Het enige verschil tussen Langton's ant en een Turing-machine met slechts één inwendige toestand, is dat Langton's ant over een grid mag lopen—een twee-dimensionale tape zo je wilt.

Bij een turmite wordt de richting waarin hij beweegt blijkbaar gebruikt als een soort impliciete interne toestand. Elke stap van de ant kan de richting veranderen afhankelijk van de kleur van het vakje (bijv. rechtsom bij wit, linksom bij zwart). Deze richting-veranderingen stapelen zich op in het grid, waardoor complexe patronen en een "geheugen" in het grid zelf ontstaan. De "richting + grid" fungeert klaarblijkelijk als een soort "verborgen" dynamische geheugen.

20a, p. 57: Langton's ant is Turing-compleet, dus kan wat elk ander computer-programma ook kan—dus ook een 4K bitmap met 10-bit kleurdiepte genereren.

20b, p. 57: Simpel gezegd laat de architectuur van een turmite het toe de hele afbeelding hard te coderen in het geheugen, en deze vervolgens pixel voor pixel, en regel voor regel, uit te schrijven. 4K toestanden zijn voldoende, voor de turmite is het immers vodoende te weten waar hij zich in de afbeelding bevindt. De pixelkleuren kunnen hard worden gecodeerd.

20c, p. 57: Als in het vorige antwoord, alleen zal de turmite elk gegenereerd frame moeten overschrijven met een volgend frame. Zeg dat een film bestaat uit 150,000 frames. Dan zijn 4Kx150,000 toestanden voldoende. De frame rate zal echter te wensen overlaten.

1a, p. 58: Na een overgangsfase ontstaat een stabiele toestand, namelijk een leeg vlak.

1b, p. 58: Na een overgangsfase vervalt het systeem in periodiek gedrag.

1c, p. 58: Na een overgangsfase ontstaat een stabiele toestand

1d, p. 58: Na een overgangsfase vervalt het systeem in periodiek gedrag.

1e, p. 58: Na een overgangsfase vervalt het systeem in periodiek gedrag.

1f, p. 58: In een eindige ruimte (bijvoorbeeld een torus-oppervlak) periodiek, in het oneindige platte vlak semi-periodiek.

1g, p. 58: De glider gun zelf is periodiek: de hele figuur komt na veertien periodes terug. De glider gun inclusief gliders is semi-periodiek in het oneindige platte vlak vanwege de afgegeven gliders.

2, p. 58: Bijvoorbeeld: een cel is/wordt bewoond als en slechts als het drie cellen in de omgeving aan staan, of aan is en vier cellen in de omgeving aan staan. Met omgeving wordt de Moore-omgeving bedoeld. Die omgeving bestaat uit 9 cellen, dus dat is inclusief de centrumcel.

Zo hoeft slechts één keer te worden geteld, hoeft bij drie cellen niet te worden gecheckt of de centrumcel zelf aan staat.

3a, p. 58: De volgende toestand van een cel wordt bepaald door het *aantal* levende buren—niet door de exacte positie van de levende buren.

3b, p. 58: Ja, volgens deze definitie wel.

3c, p. 58: Nee, volgens deze definitie is Conway's game of life niet totalistisch. Bij een puur totalistische regel bepaalt de som van alle cellen (centrum + buren) eenduidig de volgende toestand. In GoL leidt een totaal van 4 cellen echter tot twee verschillende uitkomsten: een levende cel overleeft, maar een dode cel blijft dood.

3d, p. 58: Ja hoor, op 30 maart 2026 kreeg ik zelf bij zoeken drie hits.

3e, p. 58: Er is geen universele standaard, de keuze hangt af meer van de context. In de formele wiskunde en de theorie van Wolfram is de strikte definitie (de som van alle cellen, inclusief de centrumcel) leidend. In de praktijk van 2D-simulaties, zoals Conway's Game of Life, is de extern-totalistische definitie (waarbij de eigen toestand niet wordt meegenomen) echter de meest gebruikte standaard. Voor de meer praktisch ingestelden is "totalistisch" simpelweg een verzamelnaam voor regels die enkel naar buren kijken.

4, p. 58: Deze bewering is waar. Een glider gun scheidt elke veertiende periode een glider af, waardoor de configuratie definitief vijf cellen groter wordt.

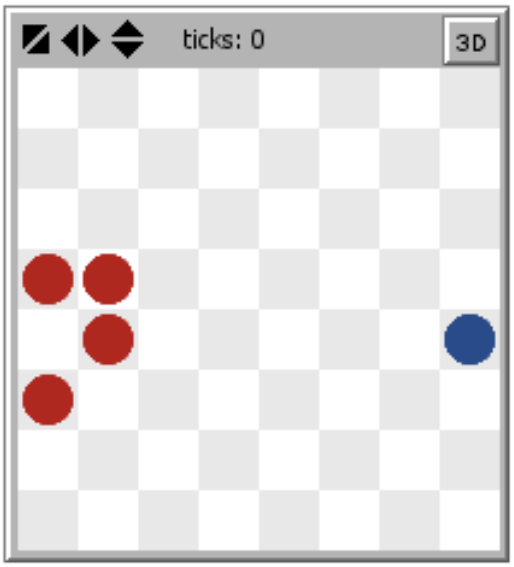
5a, p. 58: Onwaar. Als tien vierkanten van elk negen cellen elk met een "plus" ter grootte vijf cellen gevuld worden, dan zal in de volgende generatie elke levende cel verdwenen zijn. (Andere oplossingen zijn mogelijk!)

5b, p. 58: Onwaar. Maak een aaneengesloten veld van levende cellen, met aan de randen "kantelen" (als bij een kasteelmuur). Leg voldoende ver daar vandaan (dat kan!) een vierkant van vier levende cellen. Alle inwendige cellen van het grote veld zullen in de volgende generatie door overbevolking verdwenen zijn. Alle kantelen zullen de volgende generatie overleven maar geïsoleerd liggen. Het vierkant van vier blokjes zal de volgende generatie niet veranderd zijn. In de daarna volgende generatie zullen de geïsoleerde cellen (de "kantelen") uitsterven, maar het blokje zal er nog zijn. Ook in alle volgend generaties is het blokje er nog. (Andere oplossingen zijn mogelijk!)

6, p. 59: Deze vraag kan worden beantwoord door de configuratie te simuleren in een game of life simulator, bijvoorbeeld Golly. Dan vind je dat deze configuratie zich om de 30 perioden herhaald.

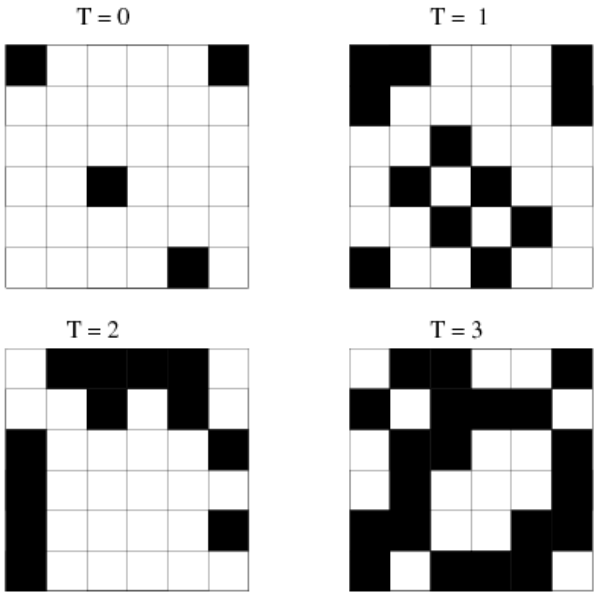
Van `http://www.conwaylife.com`: The *queen bee shuttle* (or basic shuttle) is a period 30 shuttle oscillator in which a queen bee travels back and forth between two stabilizing ends. The shuttles were originally stabilized against one another in a square of eight shuttles, though there are now a number of known ways to stabilize the ends. Some simpler methods are shown here involving blocks; for a method involving an eater 1 see buckaroo. The queen bee shuttle is the basis of all known true period 30 guns (including the famous Gosper glider gun). It was found by Bill Gosper in 1970 and was the first period 30 oscillator to be found. It is the smallest known oscillator with period greater than 15.

7, p. 60:



De makkelijkste manier om het antwoord te beredeneren is de volgende: ontwikkel acht stappen op de standaard torus, evt. met behulp van een Life implementatie. Het gedeelte van de glider dat aan de rechterkant van het bord is verdwenen en nu aan de linkerkant van het bord weer opduikt, wordt nu door de Klein-constructie gedraaid. Wentel daarom dit stuk veld 180 graden om de x-as (aangenomen dat de x-as van het veld in het midden zit). De bolletjes kom nu aan de achterkant te zitten en kleuren rood. In 2D komt de overgang naar rechts neer op: spiegelen in de x-as en van kleur veranderen.

8, p. 60: De globale toestand in de eerste drie generaties ziet er als volgt uit:



9, p. 60: Nee. Langton's ant is geen cellulaire automaat, hoewel internet vol staat met beweringen die het tegendeel suggereren. In een cellulaire automaat mogen cellen alleen hun buurcellen *lezen* om te bepalen welke toestand ze zelf in een volgend tijdstip zullen aannemen. Cellen mogen hun buren niet beïnvloeden, door ze bijvoorbeeld een volgende toestand te dicteren, of door hun toestand te overschrijven. Dit laatste staat ook wel bekend als *state push*. Dus in een cellulaire automaat vindt geen state push plaats.

Langton's ant doet wel aan state push.

Nu is het wél zo dat Langton's ant kan worden geëmuleerd door een cellulaire automaat. Maar dat is iets anders. Zie daarvoor bijvoorbeeld Opgave 19 op pagina 64.

10a, p. 60: Een eenvoudige voorbeelden is alles mappen op een constante toestand die ongelijk nul is, dus bijvoorbeeld alles mappen op toestand 1.

10b, p. 60: Bijvoorbeeld: map alle omgevingen waar geen 1 voorkomt op een 1, en alle omgevingen waar wél een 1 in voorkomt op een 2. Voor veel startconfiguraties zal de automaat dan, na een korte aanloopperiode, in een cyclus met periode 2 belanden. Bijvoorbeeld voor de configuratie 0^{31} zal dat gebeuren. Merk op dat voor deze automaat inderdaad geldt dat $\lambda = 1$.

Die, voor alle startconfiguraties, uitmondt in periodiek gedrag.

10c, p. 61: Bijvoorbeeld: map elke state op 1 als 'ie niet 1 was, en anders op 2. Elke cel kijkt dus eigenlijk alleen naar de inhoud van zichzelf om z'n volgende toestand te bepalen, en negeert de inhoud van z'n omgeving. Makkelijk is niet in te zien dat deze automaat beland in een cyclus met periode twee. Immers, alle cellen met inhoud 1 volgen de cyclus $1 \to 2 \to 1 \to 2 \to \ldots$, en alle cellen met inhoud $x \neq 1$ volgen de cyclus $x \to 1 \to 2 \to 1 \to \ldots$.

De automaat uit het vorige voorbeeld is ook periodiek, echter de periodiciteit van die automaat is moeilijker te bewijzen.

10d, p. 61: Bijvoorbeeld: laat alle cellen elke generatie één ophogen modulo $n + 1$, maar laat cellen die "dreigen" te worden afgebeeld op 0 (wat niet mag, immers $\lambda = 1$), afbeelden op 1. Alle cellen doorlopen dan een cyclus $1 \to 2 \to 3 \to \cdots \to n$. Deze cyclus bezit periode n.

Merk op dat de cycli niet noodzakelijk synchroon lopen. Merk verder op dat dit recept niet werkt voor $n = 6$, vandaar de eis $n = 2, 3, 4$ of 5.

11a, p. 61: Het aantal regels in een regelverzameling voor $K = 7$ en $R = 2$ is

$$(2R + 1)(K - 1) + 1 = 5 \times 6 + 1 = 31.$$

We moeten dus een string ter lengte 31 geven uit $\{0, 1, 2, 3, 4, 5, 6\}$ zonder nullen (immers, $\lambda = 1$) die stabiel gedrag voortbrengt. Eenvoudige voorbeelden zijn 1^{31} (31 keer een 1) of 2^{31} (31 keer een 2) of 3^{31} (31 keer een 3), etc. Immers, deze regelverzamelingen mappen alle omgevingen op één waarde tot in de lengte der oneindigheid. Dus dat is stabiel gedrag.

Andere regelverzamelingen zijn natuurlijk ook mogelijk, echter daarvan is het moeilijker te bewijzen dat deze uitmonden in stabiel gedrag.

11b, p. 61: Bijvoorbeeld: map alles op toestand 0,—behalve de omgeving met som 30, map die op toestand 1.

Voorbeeldprogressie:

```
025030003066666620250106666611133
000000000000100000000000100000000
000000000000000000000000000000000
```

Deze automaat zal gegarandeerd uitdoven omdat alle cellen in de volgende generatie toestand 0 of 1 bezitten, waardoor de omgevingssom daarna nooit meer de 30 kan halen.

Bijbehorende code: $0^{30}1$ (dertig nullen, gevolgd door een één).

12a, p. 61: Kies één cel. De vorige toestand van die cel kan 0 of 1 zijn geweest. In beide gevallen legt die keuze onmiddellijk de toestanden van de buurcellen in de vorige generatie vast (zowel links als rechts) die, op hun beurt, en samen hun opvolgers in de volgende generatie, ook weer de toestand van hun buurcellen vastleggen.

12b, p. 61: Het volstaat om te beargumenteren dat een eindige globale toestand maar kan voortkomen uit precies één mogelijke vorige eindige globale toestand.

Ook eindige globale toestanden hebben precies twee pre-images, maar één daar van valt af, omdat die uit bijna allemaal enen bestaat.

13, p. 61: Om te beginnen zien we het 3840×2160 scherm als torus en definiëren we de omgeving van elke pixel als het hele scherm. (Kan vanwege torus.) Willen we per sé een vierkante omgeving, dan kan dan ook, maar dan hebben we overlap wat minder efficiënt maar verder geen probleem is omdat we met een principe-argument bezig zijn. Omdat elke pixel 16.7 miljoen kleuren kan aannemen, definiëren we evenzoveel toestanden voor elke cel.

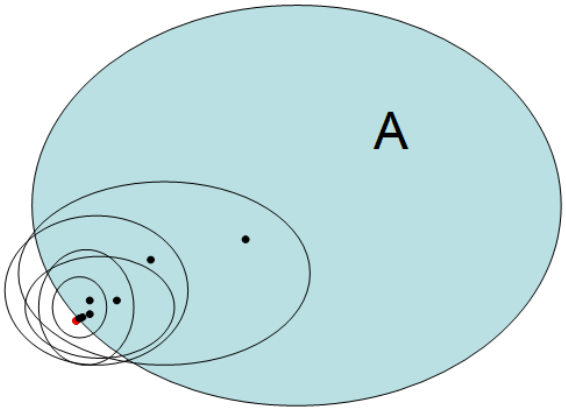
Omdat regels in een CA voor elke cel hetzelfde zijn, moeten we er voor zorgen dat een cel in het midden anders reageert op dezelfde omgeving als een cel, zeg, linksboven. Ook dat is in de regel geen probleem als er locatie-specifieke elementen in het beeld aanwezig zijn. Ondertitels zijn al voldoende, maar ook gezichten, etc. Willen we 100% garanderen dat omgevingen "locatiebewust" zijn, dan kunnen we om het beeld een zwarte rand van één pixel aanbrengen, of rechtsboven een stip of logo aanbrengen in een ongebruikelijke kleur, zoals magenta of cyaan.

Omdat een CA geen geheugen kent, mag de film geen identieke beelden bezitten, i.e., mogen er op twee verschillende tijdstippen niet exact dezelfde snapshots voorkomen. Ook dat zit met de meeste films wel goed, ook met een film als Groundhog Day waar een journalist (gespeeld door Bill Murray) steeds dezelfde saaie dag opnieuw beleefd (totdat hij ophoudt cynisch te zijn). Immers, snapshots van een film zijn zelden tot nooit exact hetzelfde.

We kunnen de regels van de CA opstellen door de film te laten lopen en elk frame de cellen (pixels) hun omgeving te laten opnemen. In feite zou je kunnen zeggen dat we tijdens het lopen van de film de CA de film laten leren.

En dan gaat het geloof ik wel goed. Interessant is—en nu verlaten we het principeargument—om na te denken over compressie. Immers, een film van 2 uur met, zeg, 60 frames per seconde waarin elke cel het volledige beeld leert, geeft een regelverzameling van astronomische grootte. We gaan hier verder niet op in :)

14a, p. 61: Limietpunt: elke omgeving van het rode punt, hoe klein ook, bevat een element uit A. Het rode punt is dus een limietpunt van A.



14b, p. 61: $\{0\}$, $\{1, -1\}$.

14b, p. 61: $\mathbb{N}$.

14c, p. 61: Bekijk een omgeving N van een limietpunt x van A. Omdat x een limietpunt van A is moet deze omgeving N een element $a_1 \in A$ bevatten, ongelijk x. Verklein N naar een omgeving N_2 van x, zó dat a_1 daar buiten valt. Omdat x een limietpunt van A is moet N_2 een element $a_2 \in A$ bevatten, ongelijk x. Zet dit proces door. Op deze manier bevat N aftelbaar oneindig veel elementen $\{a_i\}_i$ uit A.

14d, p. 61: Om te beginnen is $x = 2$ zelf geen verdichtingspunt van $A = \{2\}$: bijvoorbeeld de omgeving $(1, 3)$ bevat geen elementen uit A anders dan x zelf.

Punten buiten A zijn ook geen verdichtingspunten, omdat deze altijd een positieve afstand hebben tot 2, en er dus altijd wel een omgeving te bedenken is die het getal 2 mist. De verzameling verdichtingspunten van $\{2\}$ is dus leeg.

14e, p. 61: Geen enkel punt in $\mathbb{R}$ is een limietpunt van $\mathbb{Z}$: als $x \notin \mathbb{Z}$ dan is de afstand tot elk dichtstbijzijnde gehele getal k positief, zeg $\epsilon > 0$. Het open interval $(x - \epsilon/2, x + \epsilon/2)$ is dan een omgeving van x welke geen enkel element van $\mathbb{Z}$ bevat. In het andere geval, als $x \in \mathbb{Z}$, dan bevat bijvoorbeeld $(x - 1/2, x + 1/2)$ ook geen andere punten van $\mathbb{Z}$. De verzameling limietpunten van $\mathbb{Z}$ is dus leeg.

14f, p. 61: Elk element uit $\mathbb{R}$ is een limietpunt van $\mathbb{Q}$: elk getal uit $\mathbb{R}$ kan namelijk door een rij van breuken worden benaderd, of het nu rationaal is of niet. Bijvoorbeeld, het irrationale getal $\pi = 3.14159265359\ldots$ kan worden benaderd door de rij breuken 3, $3.1 = 31/100$, $3.14 = 314/1000$, $3.141 = 3,141/10,000$, etc. En het bijvoorbeeld het rationale getal 2.1491 kan worden benaderd door de rij breuken 2.149101, 2.1491001, 2.14910001, 2.149100001, etc. Als elk element uit X limietpunt is van $A \subseteq X$, dan zeggen we dat A *dicht ligt in* X. De verzameling $\mathbb{Q}$ ligt dus dicht in $\mathbb{R}$. Anders gezegd is er geen element uit $\mathbb{R}$ te vinden dat afgezonderd op een afstandje van $\mathbb{Q}$ ligt. Een concreet gevolg daarvan is dat computers kunnen doen alsof ze met *alle* getallen kunnen rekenen, inclusief schuine zijden van driehoeken en oppervlakten van cirkels. Makkelijk met het ontwerpen van bruggen en rotondes!

14g, p. 62: Om te beginnen is makkelijk in te zien dat 0 een verdichtingspunt is: elke omgeving van 0 bevat wel een element uit $A = \{1/n + 1/m\}_{n,m}$. Verder is er voor elke n de deelverzameling $\{1/n + 1/1, 1/n + 1/2, 1/n + 1/3, \dots\}$ waaruit makkelijk een limiet kan worden gehaald. Dus voor elke n is $1/n$ een verdichtingspunt. Als verdichtingspunten hebben we nu gevonden

$$\{0\} \cup \{\frac{1}{n}\}_n$$

Dit zijn ook de enige verdichtingspunten. Immers, oneindige deelverzamelingen uit A kunnen worden opgedeeld in drie klassen, afhangende of n en m, of beiden, doorlopen. (Beiden niet doorlopen kan niet, want dan is er geen oneindige verzameling.) Als m en n beiden doorlopen komen we op nul uit. Als n of m uiteindelijk niet meer doorloopt, komen we op een getal van de vorm $1/k$ uit. Zie zonodig stackexchange voor andere bewijzen.

15a, p. 62: Als de inhoud van alle cellen bekend is, dan is dat op zich voldoende om te bepalen hoe de automaat zich verder zal ontwikkelen als de klok weer wordt gestart. De regels veranderen niet, dus die hoeven ook niet in de toestandbeschrijving te worden opgenomen.

15b, p. 62: 1) Elk vakje kan aan of uit zijn, dus er zijn $2^{5\times 8} = 2^{40} = 1,099,511,627,776 \approx 10^{12}$ mogelijke toestanden; 2) hetzelfde antwoord; 3) het aantal vakjes van een oneindig grid is aftelbaar oneindig. Een configuratie kan worden gezien als een deelverzameling van het grid. Dus het antwoord is: de kardinaliteit van de machtsverzameling van een aftelbare verzameling. En dat is $2^{\aleph_0}$, oftewel $\mathfrak{c}$. Het antwoord "overaftelbaar oneindig" is niet goed, want kardinaliteiten hoger dan $\mathfrak{c}$ zijn dat ook.

15c, p. 62: Een glider kan twee vormen aannemen: "Italië" (L met punt) enerzijds en "persoon op punt van stoel" anderzijds. Beide vormen bestaan uit 5 cellen die aan staan. De afstand tussen C0 en overige toestanden is dus 5. Bij Vorm 1 ("Italië") zal in de eerste stap een glider 2 cellen van het origineel verschillen. Bij de tweede stap ook 2, bij de derde, vierde en vijfde stap 4, bij de zesde stap 5, bij de zevende stap 4, en tenslotte altijd 5. Dus 0, 2, 2, 4, 4, 4, 5, 4, 5, 5, 5, 5, 5,.... Bij Vorm 2 ("stoel") is de rij afstanden 0, 2, 3, 3, 4, 4, 4, 5, 5, 5,.... De afstanden 0, 2, 3, 4, 5 komen dus voor. Er is dus nooit een afstand 1. (Maar wel een afstand 3.)

15d, p. 62: Afstanden zijn altijd positieve gehele getallen. Er kan dus geen sprake zijn van steeds kleiner omgevingen. Het scenario van één glider in een verder leeg grid geeft meteen een tegenvoorbeeld: de afstanden zijn daar altijd groter dan 2 (en meestal 5).

15e, p. 62: Willekeurig kleine afstanden zijn mogelijk: laat één vakje steeds verder van de oorsprong af bewegen. Het lege grid is daarmee meteen een voorbeeld van een limietpunt. Sterker, deze rij toestanden convergeert naar het lege grid. Welke afstanden nu überhaupt mogelijk zijn is een moeilijker vraag. In ieder geval willekeurig grote en willekeurig kleine afstanden. Willekeurig kleine afstanden hebben we al gezien. Willekeurig grote afstanden zijn ook mogelijk omdat het delen door de afstand naar de oorsprong niet voldoende verkleint om de som van individuele afstanden tussen vakjes te laten convergeren.

15f, p. 63: C0 is nu wel een verdichtingspunt. Sterker, in deze metriek convergeert de rij C1, C2, C3, C4, ... zelfs rechtstreeks naar C0. De invloed van de glider wordt namelijk steeds kleiner naarmate de afstand van de glider tot de oorsprong groter wordt.

16a, p. 63: Bij de wortel kunnen niet alle deelbomen eindig zijn (immers, dan zou de boom zelf eindig zijn). Kies een oneindige deelboom uit. Voor de wortel van die deelboom kunnen we weer hetzelfde argument herhalen. Zo ontstaat een oneindige tak. Geloof je het nog niet, maak dan een tekening waarin dit proces zich afspeelt.

16b, p. 63: Z is aftelbaar, dus $Z \times Z$, dus $(Z \times Z) \times Z$. De laatste is Z^3. Laat $z_1, z_2, z_3 \ldots$ een aftelling zijn van Z^3. Een run is oneindig, dus z_1 moet tenminste één toestand $s_1 \in S$ oneindig vaak aannemen. Bekijk de globale toestanden die z_1 is s_1 brengen. Binnen die set van globale toestanden moet er een toestand $s_2 \in S$ zijn die z_2 oneindig vaak aanneemt. Bekijk de globale toestanden die (z_1, z_2) is (s_1, s_2) brengen. Binnen die oneindige set van globale toestanden moet er een toestand $s_3 \in S$ zijn die z_3 oneindig vaak aanneemt. Zo doorgaan wordt Z^3 bedekt met toestanden $s_1, s_2, \ldots$. Het verdichtingspunt dat we zoeken is de functie μ die cel z_i op toestand s_i afbeeldt. (Merk op dat König's lemma impliciet is gebruikt.)

17, p. 63: In a CA, cells get their next state by reading states of neighboring cells. It is forbidden to write states to neighboring cells. (A.k.a. "state push.") The current system of rules does not constitute a CA because the "Spread" rule influences neighboring cell, which is illegal.

It is possible to turn this system into a CA by moving the "Spread" rule to green cells, and rename it into "Take over" (from taking over the fire) or something similar.

18a, p. 64: Ja, dit kan. Een omgeving van 5×5 tegels volstaat:

	·	**c**	·	
	·	π	·	
	·	·	·	

Elke 5×5-omgeving van een lege cel waarbij π zich links, rechts, boven of onder c bevindt zal moeten beslissen of deze π zal opnemen (toelaten). We kunnen de volgende regel instellen:

> Neem de robot π op als (en slechts als) c de laagste bezoekfrequentie van π's omgeving bezit. Mocht π meerdere buur-tegels met een minimale bezoek-frequentie bezitten, dan spreken we af dat c de robot π alleen toelaat als (en alleen als) c de eerste tegel is in een van tevoren bepaalde vaste ordening van buren van π's omgeving (bijvoorbeeld $N < E < S < W$). In het geval c de robot π toelaat hoogt c de bezoekfrequentie met één op.

Met deze regel is volgende toestand van een cel uniek bepaald, en hiermee is ook het zetten-arsenaal van de robot volledig bepaald.

Bijvoorbeeld, als met de zojuist afgesproken ordening π's buren N, E, S, en W een bezoekfrequentie van respectievelijk 5, 4, 4, en 6 bezitten, dan zal π naar E wandelen, en niet naar S.

Als we voor deze regel kiezen is het noodzakelijk dat deze deterministisch is omdat de cellen N, E, S, W anders niet van elkaar kunnen "weten" welke kant de robot π opstapt. Bij een CA-simulatie zou π in dergelijke gevallen dan bij zo'n stap klonen of juist verdwijnen.

Een laatste opmerking: zojuist is een CA-regel gegeven waarmee het gewenste robot-gedrag kan worden opgeroepen. Dat wil niet zeggen dat er andere en, meer bepaald, efficiëntere regels zijn om

dit gedrag op te roepen. Het kan best zijn dat er een eenvoudiger CA-regel te bedenken is dat hetzelfde gedrag oproept. Maar daar ging het niet om.

18b, p. 64: Ja, dit kan ook. De enige complicatie is dat twee of meer robots naar eenzelfde tegel willen wandelen:

		·	·	·
		·	π_1	·
	·	**c**	·	·
	·	π_2	·	
	·	·	·	

Maar deze complicatie kan worden verholpen. Willen meerder robots naar eenzelfde vakje wandelen, dan denken wij eerst buiten de CA-metafoor om hoe dit kan worden opgelost. Dat kan door bijvoorbeeld te bepalen dat een robot voorrang krijgt als deze hoger of anders (bij gelijke hoogte) meer links staat dan zijn kompaan. Als een robot geen voorrang krijgt, dan blijft deze staan.

Vervolgens moeten wij ons er nog van vergewissen dat deze regel kan worden vertaald naar een CA-oplossing. Dat kan, want de centrale cel c kan de situatie dankzij de 5×5-omgeving volledig uitlezen en met zekerheid een volgende toestand voorspellen.

Overigens is ook hier een non-deterministische (bv. een probabilistische) oplossing niet mogelijk.

19a, p. 64: Een cel kan wit of zwart zijn (0 of 1). Een cel kan vrij zijn, of er kan een turtle op staan. Als er een turtle op staat kan de turtle oost, west, noord, zuid staan. Mogelijke toestanden: $2 \times 5 = 10$: $\{0b, 0e, 0w, 0n, 0s, 1b, 1e, 1w, 1n, 1s\}$ (b voor "blanc").

19b, p. 64: Een Von-Neumann omgeving (4 buren) volstaat.

19c, p. 64: Voor vertrekkend vanaf wit:

$$(*, *, 0n, *, *) \to 1$$
$$(*, *, 0e, *, *) \to 1$$
$$(*, *, 0s, *, *) \to 1$$
$$(*, *, 0w, *, *) \to 1$$

Voor vertrekkend vanaf zwart:

$$(*, *, 1n, *, *) \to 0$$
$$(*, *, 1e, *, *) \to 0$$
$$(*, *, 1s, *, *) \to 0$$
$$(*, *, 1w, *, *) \to 0$$

Voor inkomend vanaf wit:

$$(*, 0n, 0, *, *) \to 0e$$
$$(*, 0n, 1, *, *) \to 1e$$
$$(0e, *, 0, *, *) \to 0s$$
$$(0e, *, 1, *, *) \to 1s$$
$$(*, *, 0, 0s, *) \to 0w$$
$$(*, *, 1, 0s, *) \to 1w$$
$$(*, *, 0, *, 0w) \to 0n$$
$$(*, *, 1, *, 0w) \to 1n$$

Voor inkomend vanaf zwart:

$$(*, 1s, 0, *, *) \to 0e$$
$$(*, 1s, 1, *, *) \to 1e$$
$$(1w, *, 0, *, *) \to 0s$$
$$(1w, *, 1, *, *) \to 1s$$
$$(*, *, 0, 1n, *) \to 0w$$
$$(*, *, 1, 1n, *) \to 1w$$
$$(*, *, 0, *, 1e) \to 0n$$
$$(*, *, 1, *, 1e) \to 1n$$

Afkorten met rot-4 is helaas niet mogelijk. Immers, de oriëntaties van turtles roteren niet mee.

Er kunnen (iets) kortere en (beduidend) minder inzichtelijke regelsystemen worden opgesteld door toestanden te representeren met getallen en dan transities te formuleren door rekenregels, bijvoorbeeld $(*, *, 1, *, *, x) \to x - 4$. Wel moet dan worden aangetoond dat alle rekenregels kloppen!

19d, p. 65: Dit doen we door de CA weer terug te zien als rooster met daarop een mier. Op elk moment kan die mier maar op één manier op zijn huidige positie terecht zijn gekomen. De voorgaande configuratie rooster/mier is dus uniek. De voorgaande configuratie, opgevat als cellulaire automaat, dus ook.

19e, p. 65: Construeer reverse turtle, en herhaal 1000 keer op reverse turtle: for (i=1 ... 1000) { doe stap achteruit. Is veld zwart, draai dan naar links en maak veld wit; anders draai naar rechts en maak veld zwart. }

20a, p. 65: Spontane ontbranding is zeldzaam. De groei is tussentijds voldoende om een ontstane brand door middel van een cascade (domino-effect) over een groot oppervlak te verspreiden.

De begroeiing is dus in feite contra-productief (of zelf-regulerend, zo je wil): bij een brand zal veel groen zorgen voor een goede verspreiding van de brand en dus een effectieve vernietiging van het groen.

20b, p. 65: Deterministisch (i.e., zonder kansen) en cyclisch (de laatste conclusie mag getrokken worden door observeren).

20c, p. 65: Dit is een kunstmatig scenario. Cellen zijn altijd begroeid tenzij ze de vorige generatie verbrand zijn. Vuur zal zich met een kans die zeer dicht tegen de 0.2 aanligt verspreiden. Vuur verspreidt snel maar zal uiteindelijk doven.

20d, p. 65: Alles groen.

20e, p. 65: Uiteindelijk zal al het vuur doven omdat de verspreidingskans $1 - g = 0.2$ niet groot genoeg is om de daadwerkelijke verspreiding te bestendigen.

Meer bepaald zal elke buur van een brandende cel met kans 0.2 ontbranden. De brandende cel zelf zal zwart worden. Naar verwachting zal een (geïsoleerde) brandende cel dus 0.2×8 buren $= 1.6$ nakomelingen genereren. (Binomiaal verdeeld met $n = 8$ en $p = 0.2$.) Daar de opgebrande cel nu zelf een buur is van een brandende cel in de volgende generatie zal een volgende brandende cel naar verwachting dus 0.2×7 buren $= 1.4$ nakomelingen genereren. In de praktijk zal deze verhouding nog lager dan 1.4 liggen door overlap en zullen brandende cellen op den duur verdwijnen.

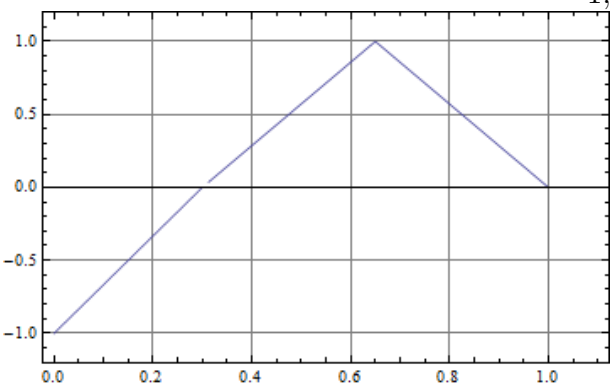
20f, p. 65: Nee, want het betreft kansen. Er is altijd een mogelijkheid, hoe klein ook, dat een deel brandende cellen zich generatie na generatie maar blijft voortplanten, zonder ooit uit te sterven.

20g, p. 66: Ja, dit kan beweerd worden. (Meteen 2 punten voor een ja-antwoord.) (Als je deze vraag goed hebt beantwoord is je intuïtie in ieder geval in orde!)

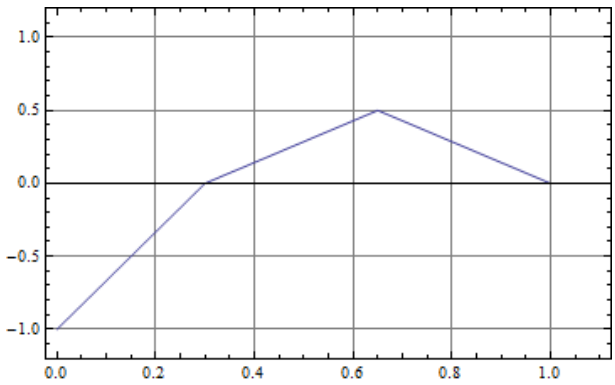
Toelichting: met kansrekening (theorie van vertakkingsprocessen) kan worden bewezen dat met kans nul een loot van brandende cellen zich generatie na generatie blijft voortplanten. De kans dat er convergentie zal plaatsvinden is dus 1. Let op: kans 1 is niet hetzelfde als absolute zekerheid: er is altijd een mogelijkheid dat een deel brandende cellen zich generatie na generatie blijft voortplanten, zonder ooit uit te sterven.

Over de theorie van vertakkingsprocessen kun je meer lezen in het boek *Introduction to Probability,* door Grinstead & Snell, GNU versie, Sectie 10.2: Branching processes (p. 376).

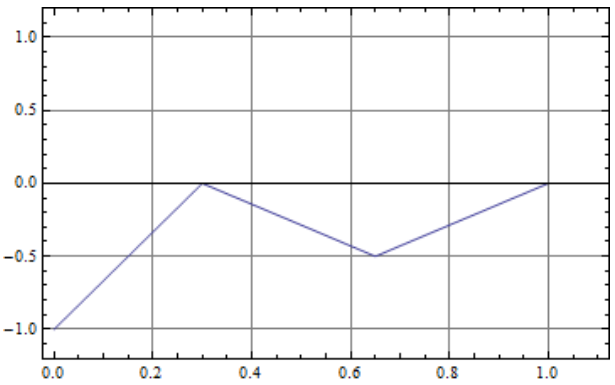
1, p. 69:



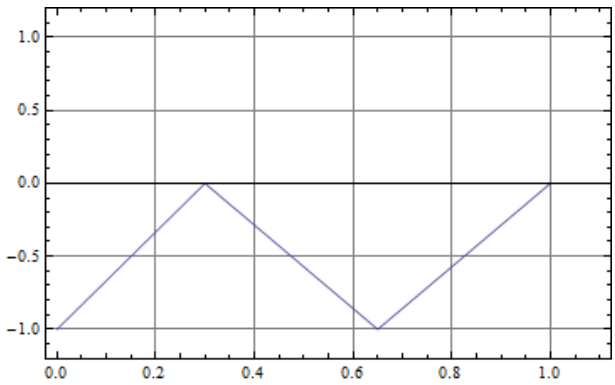
$(a, \beta) = (1, 0.3)$



$(a, \beta) = (0.5, 0.3)$



$(a, \beta) = (-0.5, 0.3)$



$(a, \beta) = (-1, 0.3)$

Toelichting:

- Als $a = 1$, is de aantrekkingskracht, a, positief. Als $r \in (0.3, 1)$ zal het deeltje worden aangetrokken. De maximale aantrekkingskracht 1.0 wordt bereikt als $r = (0.3 + 1)/2 = 0.65$. Als $r < 0.3$ zal het deeltje worden afgestoten, met een maximale afstotingskracht 1.0 voor $r = 0$.
- Als $a = 0.5$, is de aantrekkingskracht, a, positief. Voor de rest geldt zo'n beetje hetzelfde verhaal als bij $a = 1$, met de uitzondering dat de maximale aantrekkingskracht is gehalveerd.

- als $a = -1$, is de aantrekkingskracht, a, negatief. Als $r \in (0.3, 1)$ zal het deeltje worden afgestoten. De maximale afstotingskracht 1.0 wordt bereikt als $r = (0.3 + 1)/2 = 0.65$. Als $r < 0.3$ zal het deeltje ook worden afgestoten.
- Als $a = 0.5$, is de aantrekkingskracht, a, positief. Voor de rest geldt zo'n beetje hetzelfde verhaal als bij $a = -1$, met de uitzondering dat de maximale afstotingskracht is gehalveerd.

Mathematica code, waarbij b de rol van β speelt:

```
F[r_,a_,b_] = Piecewise[{
    {r/b-1, r<b},
    {a(1-Abs[2r-1-b]/(1-b)), b<r && r < 1},
    {0,r>1}
}];

Manipulate[
    Plot[F[r,a,b],{r,0,1.1},PlotRange->{-1.2,1.2},Frame->True,GridLines->Automatic],
    {{a,0},-1,1},{{b,0.3},0,1}
]
```

2a, p. 69: Een voorbeeld van een aantrekkings-matrix waarin alle deeltjes elkaar aantrekken.

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

Meest algemene vorm:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix},$$

waarbij $a_{ij} > 0$ voor alle $1 \leq i, j \leq 3$.

2b, p. 69: Een voorbeeld van een aantrekkings-matrix waarin alle deeltjes elkaar afstoten.

$$A = \begin{pmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{pmatrix}.$$

Meest algemene vorm: $a_{ij} < 0$ voor alle $1 \leq i, j \leq 3$.

2c, p. 69: Een voorbeeld van een aantrekkings-matrix waarin alleen deeltjes van hetzelfde type elkaar aantrekken

$$A = \begin{pmatrix} x & -y & -y \\ -y & x & -y \\ -y & -y & x \end{pmatrix}.$$

waarbij $0 < x, y$.

3, p. 69: Een voorbeeld van een aantrekkings-matrix met transitieve aantrekking en anti-transitieve afstoting:

$$A = \begin{pmatrix} 0 & x & 0 & -x \\ -x & 0 & x & 0 \\ 0 & -x & 0 & x \\ x & 0 & -x & 0 \end{pmatrix}.$$

waarbij $x > 0$.

4, p. 69: De aantrekkings-matrix is niet noodzakelijk symmetrisch. Een deeltje van type m kan aangetrokken worden tot deeltjes van type n, maar omgekeerd kan een deeltje van type n willen wegblijven van deeltjes van type m.

5, p. 69: Als er $N = 3$ typen deeltjes zijn, dan is de aantrekkings-matrix $N \times N = 9$ entries groot. Voor elke entry zijn er $k = 5$ keuzes, dus in deze situatie zijn er $k^{N \times N} = 5^9 = 1,953,125$ verschillende dynamieken mogelijk.

Als $N = 10$ en $a \in \{-1.0, -0.9, \ldots, -0.1, 0.0, 0.1, \ldots, 0.9, 1.0\}$, dan is $k = 21$. In dat geval zijn er $k^{N \times N} = 21^{100} \approx 1.67 \times 10^{132}$ verschillende dynamieken.

6a, p. 69: Elk type deeltje moet de aantrekking en afstoting verdelen over alle andere deeltjes, zodanig dat de som nul is. We draaien het om: als de rij-som behoorlijk positief is, zal dat type deeltje aangetrokken worden door veel andere deeltjes. Zo'n type deeltje zal zich in het algemeen bewegen in de omgeving van andere deeltjes. Als de rij-som behoorlijk negatief is, zal dat type deeltje afgestoten worden door veel andere deeltjes. Zo'n type deeltje zal zich in het algemeen geïsoleerd bewegen. Wat voor effect zoiets op de algehele dynamiek heeft is moeilijk te zeggen.

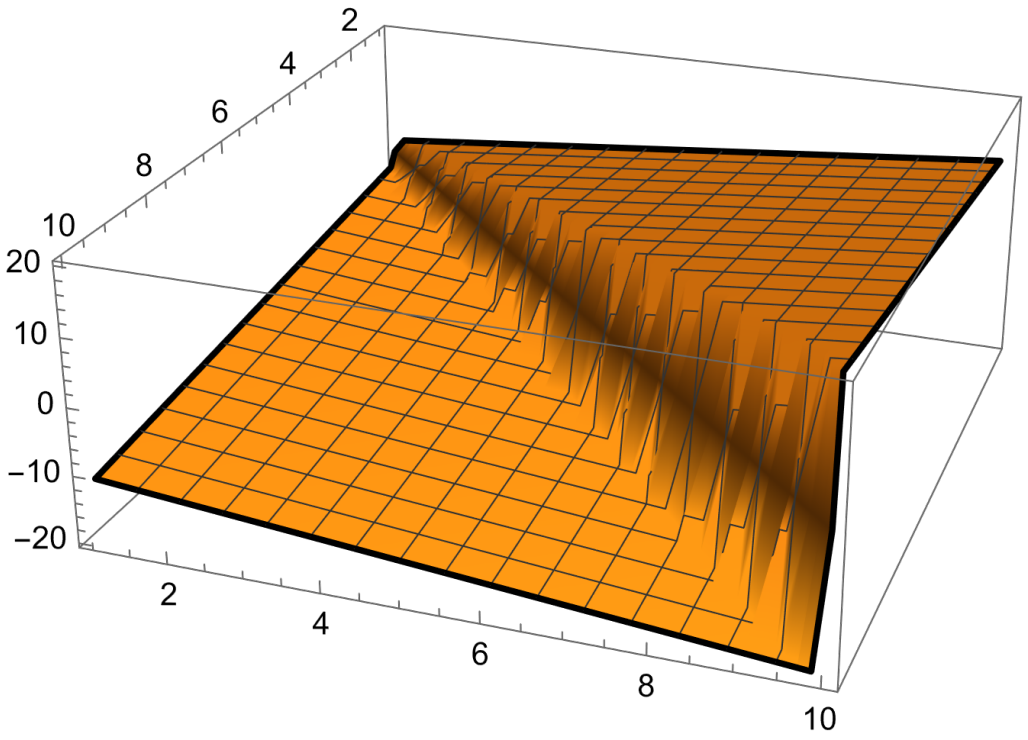
6b, p. 69: Een kolom geeft een profiel van de aantrekkings- of afstotingskracht van dat type deeltje. Als een kolom-som positief is, zal dat deeltje over het geheel genomen relatief veel andere deeltjes aantrekken; als een kolom-som negatief is, zal dat deeltje over het geheel genomen relatief veel andere deeltjes afstoten. Wat voor effect zoiets op de algehele dynamiek heeft is moeilijk te zeggen.

6c, p. 69: De dynamiek is evenwichtig, dat wil zeggen dat alle typen deeltjes geen overwegend aantrekkend of afstotend effect bezitten, Wat de globale effecten zijn is weer moeilijk te zeggen.

7a, p. 70:

l/r	1	2	3	4	5	6	7	8	9	10
1	0	3	4	5	6	7	8	9	10	11
2	−3	0	5	6	7	8	9	10	11	12
3	−4	−5	0	7	8	9	10	11	12	13
4	−5	−6	−7	0	9	10	11	12	13	14
5	−6	−7	−8	−9	0	11	12	13	14	15
6	−7	−8	−9	−10	−11	0	13	14	15	16
7	−8	−9	−10	−11	−12	−13	0	15	16	17
8	−9	−10	−11	−12	−13	−14	−15	0	17	18
9	−10	−11	−12	−13	−14	−15	−16	−17	0	19
10	−11	−12	−13	−14	−15	−16	−17	−18	−19	0

Als grafiek:



De as op de voorgrond is die van R.

7b, p. 70: In dat geval bewegen we binnen één kolom in de tabel van boven naar beneden. En in de grafiek, zoals die hierboven is afgebeeld, bewegen we van achter naar voren.

Voor kleine waarden van R is de overgang van een positieve verandering in oriëntatie naar een negatieve verandering in oriëntatie niet zo heel groot. Voor grotere waarden van R (en dus L) is de overgang van een positieve verandering in oriëntatie naar een negatieve verandering in oriëntatie erg abrupt. Bijvoorbeeld voor $R = 9$ is de overgang van $L = 8$ naar $L = 10$ een verandering van $17^0 - (-19^0) = 36^0$. (Dergelijke grote overgangen worden in de wiskunde *catastrofaal* genoemd. Er is zelfs een tak van wiskunde, genaamd *catastrofetheorie*, die zich bezighoudt met abrupte overgangen in natuurverschijnselen.)

8, p. 70: De kleur van een deeltje wordt bepaald door het aantal andere deeltjes, N_r, in een omgeving met straal r. Als $15 < N_5 \leq 35$ dan blauw; als $N_5 > 35$ dan geel; als $13 \leq N_5 \leq 15$ dan bruin; als $N_{1.3} > 15$ dan magenta; anders groen. (In deze volgorde uit te voeren, anders gaat het niet goed.)

9a, p. 70: De waarden $\Theta = 14$ en $r = 5$.

9b, p. 70: Parametercombinaties (α, β) die resulteren in een lage DHI, vormen uniform verdeelde patronen. Hoge DHI-waarden, daarentegen, geven lokale ophopingen van deeltjes aan. De laatste type configuraties zijn interessant, omdat die een zekere vorm van (zelf-) organisatie vertonen.

9c, p. 70: Rood tot donkerood.

9d, p. 71: In afnemende DHI: D, M, H, G. Namen: “cells and moving cluster”, “untidy cow pattern”, “moving structures”, en “lifelike structures”’. Deze patronen hebben gemeen dat cellen zich concentreren in clusters.

9e, p. 71: In toenemende DHI: O, K, E. Namen: “regular pattern”, “fingerprint pattern”, en “chaotic pattern”’. Deze patronen hebben gemeen dat cellen zich verspreiden over de beschikbare ruimte.

9f, p. 71: Regionen die cell- of amoebe-achtige structuren voortbrengen.

9g, p. 71: Regions of life. De notie region of life wordt in het artikel afgekort tot RoL.

9h, p. 71: Groenig. Maar omgekeerd is het niet zo dat groenige gebieden RoL’s zijn.

10, p. 71: Overgangen verlopen asynchroon: “(. . .) a population of particles moving (. . .) asynchronously in a continuous toroidal wrapped space.” p. 2. Er zal niet veel veranderen als we overgangen in het systeem synchroon laten verlopen. Problemen vinden meestal plaats als wordt toegestaan dat updates in een systeem met oorspronkelijk synchrone updates, a-synchroon mogen worden uitgevoerd. Bijvoorbeeld, er blijft niets van Conway’s game of life over als cel-updates a-synchroon mogen plaatsvinden.

11, p. 72: Non-deterministisch (ook wel: probabilistisch of stochastisch). We lezen: “(. . .) a population of particles moving deterministically (. . .) in a continuous toroidal wrapped space.” p. 2. De auteurs doelen hier op het oriëntatie-updatemechanisme; en dat is inderdaad deterministisch: er zit geen toevalselement in. Echter het feit dat updates per deeltje a-synchroon plaatsvinden zorgt er voor dat Schmickl *et al.*’s systeem grosso modo non-deterministisch is.

12a, p. 72: Het Physarum Polycephalum kent eenzelfde cyclus. Ook het Dictyostelium Discoideum (zijdelings aangestipt in het college over Physarum) kent trouwens zo’n cyclus.

12b, p. 72: De onderzoekers creëerden een scenario waarin ze typische structuren toevoegden, zoals sporen en cellen van verschillende groottes en leeftijden. Deze structuren waren vastgelegd tijdens eerdere runs. Ze observeerden hoe de omgeving van elk deeltje in de loop van de tijd veranderde door te kijken naar het aantal buren links en rechts binnen een straal van 5. Vervolgens registreerden ze hoe vaak elke combinatie van linker- en rechterburen verscheen, en gebruikten de deze gegevens om een heat map te maken. Elke combinatie van buren leidde tot een specifieke draaihoek voor de deeltjes, die werd weergegeven als een grijstint op de achtergrond. Donkere kleuren vertegenwoordigden dan linksafslaande bochten en felle kleuren vertegenwoordigden rechtsafslaande bochten. Dit creëerde de patronen die te zien zijn in de afbeelding.

13a, p. 72: Een cel deelt zich in tweeën. De twee ontstane cellen delen zelf daarna ook weer, zodat er op een gegeven moment vier cellen zijn, en zo verder.

Overigens bevat het YT filmpje een caption. Deze zegt ook al veel. Letterlijk vertaling: “In een kunstmatige chemie kunnen we onze eigen regels voor chemische reacties bedenken. Hier hebben we een aantal regels gemaakt die ervoor zorgen dat een informatiedragend molecuul in een membraan zichzelf reproduceert. De moleculen produceren een enzym (een donkerblauw paar) dat gevangen blijft in het membraan.”

13b, p. 72: De kleur geel representeert de cel-membraan, en de verschillende kleuren binnenin representeren zoiets als een genoom, dat wil zeggen, de informatiedrager van de cel.

13c, p. 72: Een beschrijving in het Nederlands:

> Hutton's kunstmatige chemie bestaat uit een groot aantal atomen in een 2D-omgeving. Ze kunnen aan elkaar worden gebonden om moleculen te vormen. Als atomen gebonden zijn, schrijft het model voor dat ze dicht bij elkaar blijven, bijvoorbeeld via een zichtbare link. In het andere geval zijn atomen vrij om lokaal te bewegen.
>
> Atomen hebben een vast type (zeg a, b, c, etc.) en een variabele toestand (zeg 1, 2, 3, etc.). Reacties kunnen optreden wanneer twee atomen botsen, en ze kunnen dan de toestand van de betrokken atomen veranderen. Ook kunnen reacties bindingen tussen atomen creëren of verbreken. Bijvoorbeeld, de reactie $a0 + a0 \rightarrow a1a1$ zal ervoor zorgen dat twee atomen met type a en toestand 0 aan elkaar worden gebonden, en toestand 1 aannemen. Zelfs met dergelijke eenvoudige regels kan geavanceerd gedrag zoals template-replicatie (i.e., patroon-duplicatie) worden geproduceerd.
>
> In het systeem wordt een reeks atomen binnen een membraan gehouden. Dit noemen we een *cel*. Het membraan is een lus van atomen van een specifieke soort. Met een geschikte set reactie-regels kan een cel zijn genetische informatie dupliceren en ervoor zorgen dat het membraan knikt en splijt op zo'n manier dat er twee dochter-cellen overblijven. Deze vorm van ongeslachtelijke celdeling wordt vaak gezien bij bacteriën, en is, volgens Hutton, één van de eenvoudigere mogelijke vormen van celdeling.
>
> De exacte details van wat een botsing vormt, wordt gespecificeerd door het physics model. Wel is het zo dat in het model bepaalde restricties zijn ingebakken, zoals dat de wereld 2-dimensionaal is, en gekruiste bindingen niet kunnen worden gevormd.

Samengevat in 100 woorden:

> Hutton's kunstmatige chemie bestaat uit atomen in een 2D-omgeving. Atomen kunnen moleculen vormen en lokaal bewegen. Hun beweging hangt af van de physics van het systeem. Atomen hebben een vast type en een variabele toestand. Bij botsingen kunnen ze reageren, hun toestand veranderen en bindingen vormen of breken. Zo kan een reactie twee atomen binden en hun toestand aanpassen. Ondanks simpele regels kan complex gedrag ontstaan, zoals het dupliceren van patronen. Atomen bevinden zich in een membraan, dat bij celdeling kan splitsen. Dit bootst ongeslachtelijke deling na, zoals bij bacteriën, en is volgens Hutton een eenvoudige vorm van celdeling.

14a, p. 72: Regels R1, R2 en R3: ten eerste veroorzaakt R1 een toestandsverandering aan de bovenkant van de genstring, waardoor het e-atoom in toestand 5 komt. Ongebonden atomen kunnen in en uit diffunderen van een omringende soep van atomen in toestand 0, en dus zal een $e0$-atoom snel botsen met het $e5$-atoom, waardoor reactie R6 in werking treedt. Reactie R2 zorgt ervoor dat de nieuwe base aan het membraan wordt gekoppeld zoals de eerste, en R3 activeert de volgende fase om door te gaan wanneer deze koppeling heeft plaatsgevonden.

14b, p. 72: Regels R4 tot en met R9. Zij zorgen voor base-duplicatie, waarbij atomen uit de omringende soep van atomen in toestand 0 worden gebonden aan hun bijpassende tegenhangers. Met behulp van dit proces kan een molecuul van elke lengte en met elke reeks basen zich dupliceren. Merk op dat het $c0$-atoom aan de andere kant van het $c5$-atoom kan binden, maar de sequentie zou op dezelfde manier doorgaan. In de vijfde stap kan het $c6$-atoom niet reageren met het verkeerde $b7$-atoom, omdat dat gekruiste bindingen zou creëren, wat niet mag/kan in de fysica van het model.

14c, p. 72: Regels R10, R11, en R13.

14d, p. 73: Het openritsen gebeurt met behulp van reacties R12 en R13. Aansluitend op de base-duplicatiesequentie, worden de twee kopieën van de genetische informatie gescheiden. Deze twee processen werden geïnspireerd door het replicatieproces van DNA-moleculen, hoewel het genoom hier bestaat uit één enkele string.

14e, p. 73: R14 en R15 initieren dit proces, en regels R16 tot en met R20 zetten dit proces voort. Dus, nadat de genstrings zijn uitgepakt, wordt de trekfase gestart. Eerst worden de uiteinden van de genstring (de f-uiteinden) aan nieuwe punten op het membraan bevestigd middels reacties R14 en R15. Vanaf dit punt wordt de trekfase voortgezet middels R16 tot en met R20.

14f, p. 73: Regels R21 tot en met R24.

14g, p. 73: Regels R25 tot en met R34.

14h, p. 73: Regel R35: om het membraan de mogelijkheid te geven om uit te zetten, laten het model spontaan atomen van of naar de omringende soep opnemen of verliezen. Reactie R35 (waarbij $i, j \in \{36, 37\}$) stuurt dit aan, waarbij drie reactanten betrokken zijn. Deze reactie kan in beide richtingen met gelijke snelheden werken. Omdat vrije atomen weg kunnen drijven als ze eenmaal zijn vrijgelaten, is de neiging dat membranen de kleinst mogelijke grootte aannemen rond de inhoud van de cel. Tegelijkertijd kunnen ze nog steeds flexibel kunnen groeien en bewegen.

15a, p. 73: Omdat de genetische streng geen fenotypische waarde had, i.e., zich niet kon profileren op performance, waardoor mutaties niet konden worden geselecteerd op gunstige eigenschappen.

15b, p. 73: In een omgeving waar geen van zijn andere functies voordelig is, selecteert natuurlijke selectie de kortste vorm die replicatie mogelijk maakt, en korte sequenties repliceren efficiënter.

15c, p. 73: Ze profiteren van de voordelen van enzymproductie zonder eraan bij te dragen. Dit leidt tot mutaties die het vermogen om nuttige enzymen te produceren, verwijderen.

15d, p. 73: Een semi-doorlaatbaar membraan maakt groei mogelijk terwijl het enzymen binnenhoudt, zodat ze de producerende cel ten goede komen in plaats van ze te verspelen aan moleculaire parasieten.

15e, p. 73: Om te testen of cellen nog steeds konden reproduceren als ze gewoon *direct* een genoom kregen dat codeerde voor het noodzakelijke enzym.

15f, p. 73: Het codeert een enzym dat de functie van reactie R3 vervangt, waardoor cellen zich kunnen reproduceren ondanks het verwijderen van R3.

15g, p. 73: Om mutaties op een hoog tempo te laten optreden terwijl natuurlijke selectie nog steeds effectief blijft werken.

15h, p. 73: Flooding verwijdert een kwart van de wereld en zorgt er voor dat de omstandigheden voor reproductie en mutatie intact blijven.

15i, p. 73: Een mutatie voegde een extra base a toe aan het begin, maar aangezien dit de enzymfunctie niet beïnvloedt, wordt het een neutrale mutatie die gewoon blijft bestaan.

15j, p. 73: Een neutrale mutatie beïnvloedt de functie van het gecodeerde enzym niet. Het is belangrijk voor evolutie omdat het toegang geeft tot nieuwe genotypische variaties zonder directe selectiedruk.

15k, p. 73: De enzymen die ze van hun vooroudercellen erven, blijven tijdelijk functioneren, waardoor ze zich korte tijd kunnen reproduceren.

15l, p. 73: Zonder enzymafbraak kunnen enzymen zich onbeperkt ophopen, wat niet overeenkomt met echte biologische systemen. Immers, daar degraderen complexe moleculen na verloop van tijd.

16, p. 73: De figuur toont een typisch resultaat van het uitvoeren van het model met een vooroudergenoom dat een noodzakelijk enzym hard codeert. In één specifieke run werd een groot aantal reproductie-gebeurtenissen geregistreerd. Ook werd een redelijk groot aantal unieke genetische soorten geobserveerd. Elke nieuwe soort die verscheen kreeg een uniek soortnummer.

De bovenste figuur toont een spreidingsdiagram van het soortnummer, afgezet tegen de tijd. Het toont de voortdurende verschijning van nieuwe soorten naarmate er mutaties optreden. In tegenstelling tot de voorouder verdwijnen deze meestal snel, omdat de enzymen die ze produceren R3 niet vervangen. Soms schaden deze enzymen zelf de cel. Na een groot aantal iteraties neemt een nieuw genoom het over.

De onderste figuur toont een diagram van reproductiesnelheden, voor elke soort, en afgezet tegen de tijd. De vette lijn toont de voorouder die de eerste helft van de simulatie domineert, voordat deze uitsterft rond $t = 2,000,000$.

17, p. 73: Het eerste experiment in de studie bevestigde dat genen die essentieel zijn voor reproductie, kunnen worden beschermd tegen mutaties. Hoewel dit resultaat bemoedigend is, is het op zichzelf niet genoeg om aan te tonen hoe complexiteit groeit in kunstmatige systemen. Om dit te begrijpen, keek Hutton naar waarom het hebben van meer genen gunstig zou kunnen zijn. Hij hoopte dat cellen zich na verloop van tijd (en met een voldoende grote populatie) zouden aanpassen en evolueren naar complexere vormen.

Een reden waarom extra complexiteit nuttig kan zijn, is als het een organisme in staat stelt om een nieuwe voedingsbron te gebruiken. In Experiment 2 introduceert Hutton een omgeving waarin cellen twee soorten grondstoffen konden gebruiken, te weten atomen in toestand 36 en atomen in toestand 0. Cellen met een extra gen konden dan beide voedselbronnen gebruiken, terwijl de oorspronkelijke cellen alleen toestand 0 konden gebruiken. Deze aanpassing stelde de complexere cellen in staat om zich efficiënter te reproduceren, ook al hadden ze meer tijd en materiaal nodig.

Experiment 2 toonde dus aan dat toenemende complexiteit gunstig kan zijn als het cellen in staat stelt om te profiteren van overvloedig voedsel. Hutton merkte verder op dat veranderingen in de omgeving, zoals de productie van een afvalproduct, nieuwe ecologische niches kunnen creëren waaraan cellen zich kunnen aanpassen. Dit proces, *niche-constructie* genoemd, is een belangrijke motor van evolutie. Hutton zegt te hopen dat zijn systeem kan helpen te onderzoeken hoe complexiteit en innovatie zich in de loop van de tijd ontwikkelen.

18, p. 73: Hutton creëert een model van kunstmatige cellen in een tweedimensionale wereld. Deze cellen planten zich voort door speciale eiwitten (enzymen) te maken op basis van hun DNA.

Omdat cellen dezelfde ruimte delen, concurreren ze om voedsel. Mutaties treden op waardoor sommige cellen in de loop van de tijd betere eigenschappen kunnen ontwikkelen.

Experimenten toonden aan dat belangrijke of nuttige genetische informatie de neiging heeft om te overleven, zelfs als mutaties optreden. Dit suggereert dat cellen in het model kunnen blijven evolueren, zelfs als hun omgeving veranderd. Hutton's werk kan wetenschappers helpen te begrijpen hoe leven zich kan ontwikkelen en aanpassen.

19, p. 74: Een mogelijke tabel:

	Particle life	Schmickl's life like system	Hutton's kunstmatige chemie
Deeltjes	N typen	Homogeen, i.e., alle deeltjes zijn hetzelfde, en gedragen zich hetzelfde	N typen. Elk type bezit een eindig aantal toestanden
Kleur	Type deeltje	Aantal deeltjes in de omgeving	Type deeltje
Aggregatie-niveau	Clusters	Clusters	Cellen
Bindingen	Nee	Nee	Ja
Aantrekking	Gedefinieerd per paar van typen	Linker- en rechter-semicirkel	Deeltjes bewegen vrij, of zijn gebonden. Verder is er geen aantrekking of afstoting
Mutatie	Nee	Nee	Door chemische reactie

1, p. 77: Drie belangrijke verschillen tussen Swarm Chemistry en Boids, geïllustreerd met voorbeelden.

1. *Heterogeniteit van gedrag.* In het Boids-model volgen alle agents dezelfde set regels en parameters (bijvoorbeeld dezelfde afstand voor separatie of dezelfde kracht voor cohesie). Dit leidt tot vrij uniforme zwermpatronen. In Swarm Chemistry daarentegen kan elk deeltje een ander *recept* dragen met eigen parameters. *Voorbeeld:* in een gemengde zwerm kunnen sommige agents sterk op cohesie gericht zijn (dicht samenklonteren), terwijl anderen meer op separatie letten (ver uit elkaar blijven). Hierdoor ontstaan subgroepen of patronen die niet in het Boids-model voorkomen.

2. *Evolutie en interactie van recepten.* Bij Boids zijn de regels en parameters statisch en veranderen ze niet door de tijd. In Swarm Chemistry kunnen recepten worden *uitgewisseld*

of gemuteerd wanneer deeltjes elkaar ontmoeten, vergelijkbaar met chemische reacties. *Voorbeeld:* als een cohesie-gericht deeltje in botsing komt met een afstotingsgericht deeltje, kan het resulterende deeltje een mengvorm van beide gedragingen vertonen, waardoor de zwermdynamiek zich gaandeweg ontwikkelt.

3. *Metafoor en doelstelling.* Het Boids-model is ontworpen met een biologisch perspectief: het doel is het nabootsen van realistisch zwermgedrag van vogels, vissen of insecten. Swarm Chemistry daarentegen gebruikt een chemisch en evolutionair perspectief: het onderzoekt hoe diversiteit en complexiteit spontaan kunnen ontstaan uit lokale interacties. *Voorbeeld:* waar Boids een vlucht vogels zou simuleren die in V-formatie vliegt, kan Swarm Chemistry leiden tot de spontane vorming van complexe, zich herorganiserende structuren die eerder doen denken aan een ecosysteem of een chemische oplossing vol reactieve moleculen.

2, p. 77: Een mogelijke tabel:

	Particle life	Schmickl's life like system	Hutton's kunstmatige chemie	Sayama's swarm chemistry
Deeltjes	N typen	Homogeen, i.e., alle deeltjes zijn hetzelfde, en gedragen zich hetzelfde	N typen. Elk type bezit een eindig aantal toestanden	N typen
Kleur	Type deeltje	Aantal deeltjes in de omgeving	Type deeltje	Type deeltje, en automatisch bijvoorbeeld middels rgb (255 * cohesion) (255 * alignment) (2.55 * separation)
Aggregatie-niveau	Clusters	Clusters	Cellen	Clusters
Bindingen	Nee	Nee	Ja	Nee
Aantrekking	Gedefinieerd per paar van typen	Linker- en rechter-semicirkel	Deeltjes bewegen vrij, of zijn gebonden. Verder is er geen aantrekking of afstoting	Middels Boid-achtige regels: coherence, alignment en separation
Mutatie	Nee	Nee	Door chemische reactie	Door menselijk ingrijpen, over populaties heen. Sayama noemt dit *interactieve evolutie*

3, p. 77: Het kunnen mergen van de updates berust op het inzicht dat de richting van v_{nieuw} in dit proces nooit veranderd—het gaat alleen om wijzigingen in de amplitude.

Oorspronkelijk eerste wijziging:

$$v_{\text{nieuw}} = \min\left\{\frac{V_{\max}}{\|v_{\text{nieuw}}\|}, 1\right\} v_{\text{nieuw}}.$$

Oorspronkelijk tweede wijziging:

$$v_{\text{nieuw}} = c_5 \frac{V_{\text{nom}}}{\|v_{\text{nieuw}}\|} v_{\text{nieuw}} + (1 - c_5) v_{\text{nieuw}}.$$

Samen:

$$v_{\text{nieuw}} = c_5 \frac{V_{\text{nom}}}{\|v_{\text{nieuw}}\|} v_{\text{nieuw}} + (1 - c_5) \min\left\{\frac{V_{\max}}{\|v_{\text{nieuw}}\|}, 1\right\} v_{\text{nieuw}}.$$

De scalar en de vector in de eerste term bezitten in veel gevallen beiden een andere waarde dan eerst. Dit is geen probleem, omdat de richtingen onaangetast blijven—allleen de lengte wordt gemanipuleerd. In zijn geheel blijft de eerste term dus onveranderd. Scalaren bij elkaar harken geeft nu

$$= \left(c_5 \frac{V_{\text{nom}}}{\|v_{\text{nieuw}}\|} + (1 - c_5) \min \left\{ \frac{V_{\text{max}}}{\|v_{\text{nieuw}}\|}, 1 \right\} \right) v_{\text{nieuw}}.$$

Het voordeel van samenbrengen is dat bijvoorbeeld de waarde van $\|v_{\text{nieuw}}\}$, nu maar één keer hoeft te worden berekend. Bij miljoenen deeltjes scheelt dit veel.

4, p. 78: Uitwerking:

a) A neemt waar. Waargenomen buren: B en C.

b) Positie van buren: $(4, 5)$ en $(5, 4)$. Snelheid van buren: $(1, 1)$ en $(1, 3)$.

c) Gemiddelde positie van buren: mean-s = $(4.50, 4.50)$. Gemiddelde snelheid van buren: mean-v = $(1.00, 2.00)$.

d) De afstotings-vector (Eng. "repulsion vector") is nu gelijk aan de som van de verschillen in positie-vectoren tussen het deeltje en zijn buren, allen gedeeld door de respectievelijke afstand in het kwadraat. Dus

$$\text{(rep-x, rep-y)} = (2 - 4, 3 - 5)/2.83^2 + (2 - 5, 3 - 4)/3.16^2 = (-0.55, -0.35).$$

e) Acceleratie in de x-richting = cohesion * (mean-sx - sx) + alignment * (mean-vx - vx) + separation * rep-x $= 0.05 * (4.50 - 2.00) + 0.15 * (1.00 - 1.00) + 0.10 * -0.55 = 0.07$.

Acceleratie in de y-richting = cohesion * (mean-sy - sy) + alignment * (mean-vy - vy) + separation * rep-y $= 0.05 * (4.50 - 3.00) + 0.15 * (2.00 - 2.00) + 0.10 * -0.35 = 0.04$.

De (ruwe) acceleratie-vector (ax, ay) is hiermee $(0.07, 0.04)$.

A's swerve parameter is nul of onvermeld, daardoor is er geen swerve. Dus de finale (gecorrigeerde) acceleratie is gelijk aan de ruwe acceleratie.

f) De (ruwe) nieuwe snelheids-vector is nu gelijk aan

$$\text{(vx-new, vy-new)} = \text{(vx + ax, vy + ay)} = (1.00 + 0.07, 2.00 + 0.04) = (1.07, 2.04).$$

Deze overschrijdt de maximale snelheid niet, i.e., $\|v\| = 2.30 \leq V_{\text{max}} = 4$, dus de snelheidsvector hoeft niet te worden afgekapt.

De paceKeeping-parameter is gelijk aan 0, dus de uiteindelijke snelheid is identiek aan laatst berekende snelheid.

g) A verplaatst zich. De nieuwe positie is nu

$$\text{(sx, sy)} = \text{(sx + vx, sy + vy)} = (2.00 + 1.07, 2.04 + 2.04) = (3.07, 5.04).$$

5, p. 78: Uitwerking:

a) A neemt waar. Waargenomen buren: B en C.

b) Positie van buren: $(4, 5)$ en $(5, 4)$. Snelheid van buren: $(1, 1)$ en $(1, 3)$.

c) Gemiddelde positie van buren: mean-s = $(4.50, 4.50)$. Gemiddelde snelheid van buren: mean-v = $(1.00, 2.00)$.

d) De afstotings-vector (Eng. "repulsion vector") is nu gelijk aan de som van de verschillen in positie-vectoren tussen het deeltje en zijn buren, allen gedeeld door de respectievelijke afstand in het kwadraat. Dus

$$\text{(rep-x, rep-y)} = (2-4, 3-5)/2.83^2 + (2-5, 3-4)/3.16^2 = (-0.55, -0.35).$$

e) Acceleratie in de x-richting = cohesion * (mean-sx - sx) + alignment * (mean-vx - vx) + separation * rep-x $= 0.05 * (4.50 - 2.00) + 0.15 * (1.00 - 1.00) + 0.10 * -0.55 = 0.07$.

Acceleratie in de y-richting = cohesion * (mean-sy - sy) + alignment * (mean-vy - vy) + separation * rep-y $= 0.05 * (4.50 - 3.00) + 0.15 * (2.00 - 2.00) + 0.10 * -0.35 = 0.04$.

De (ruwe) acceleratie-vector (ax, ay) is hiermee $(0.07, 0.04)$.

A's swerve parameter is nul of onvermeld, daardoor is er geen swerve. Dus de finale (gecorrigeerde) acceleratie is gelijk aan de ruwe acceleratie.

f) De (ruwe) nieuwe snelheids-vector is nu gelijk aan

$$\text{(vx-new, vy-new)} = \text{(vx + ax, vy + ay)} = (1.00 + 0.07, 2.00 + 0.04) = (1.07, 2.04).$$

Deze overschrijdt de maximale snelheid, i.e., $\|v\| = 2.30 > V_{\max} = 2$, en zal dus volgens voorschrift moeten worden afgekapt naar een vector met een lengte die gelijk is aan de maximale snelheid. Vermenigvuldig v daarvoor met $V_{\max}/\|v\|$:

$$\text{(vx-new, vy-new)} = \frac{2}{2.30}\text{(vx-new, vy-new)} = (0.93, 1.77).$$

Nu geldt $\|\text{(vx-new, vy-new)}\| = V_{\max}$.

Om de invloed van pace keeping te berekenen, bepalen we

$$\text{velocityFactor} = \frac{\text{nominale snelheid}}{\|v\|} = \frac{1.50}{2.00} = 0.75.$$

De finale (gecorrigeerde) snelheids-vector is nu

$$\begin{aligned}&\text{paceKeeping * velocityFactor * (vx, vy) + (1 - paceKeeping) * (vx, vy)}\\ &\quad = 0.50 * 0.75 * (0.93, 1.77) + (1.0 - 0.50) * (0.93, 1.77) = (0.81, 1.55).\end{aligned}$$

g) A verplaatst zich. De nieuwe positie is nu

$$\text{(sx, sy)} = \text{(sx + vx, sy + vy)} = (2.00 + 0.81, 1.55 + 1.55) = (2.81, 4.55).$$

6, p. 78: Uitwerking:

a) A neemt waar. Waargenomen buren: B en C.

b) Positie van buren: $(4, 5)$ en $(5, 4)$. Snelheid van buren: $(1, 2)$ en $(1, 3)$.

c) Gemiddelde positie van buren: mean-s = $(4.50, 4.50)$. Gemiddelde snelheid van buren: mean-v = $(1.00, 2.50)$.

d) De afstotings-vector (Eng. "repulsion vector") is nu gelijk aan de som van de verschillen in positie-vectoren tussen het deeltje en zijn buren, allen gedeeld door de respectievelijke afstand in het kwadraat. Dus

$$\text{(rep-x, rep-y)} = (3-4, 1-5)/4.12^2 + (3-5, 1-4)/3.61^2 = (-0.21, -0.47).$$

e) Acceleratie in de x-richting = cohesion * (mean-sx - sx) + alignment * (mean-vx - vx) + separation * rep-x $= 0.05 * (4.50 - 3.00) + 0.15 * (1.00 - 1.00) + 0.40 * -0.21 = -0.01$.

Acceleratie in de y-richting = cohesion * (mean-sy - sy) + alignment * (mean-vy - vy) + separation * rep-y $= 0.05 * (4.50 - 1.00) + 0.15 * (2.50 - 2.00) + 0.40 * -0.47 = 0.06$.

De (ruwe) acceleratie-vector (ax, ay) is hiermee $(-0.01, 0.06)$.

A's swerve parameter is nul of onvermeld, daardoor is er geen swerve. Dus de finale (gecorrigeerde) acceleratie is gelijk aan de ruwe acceleratie.

f) De (ruwe) nieuwe snelheids-vector is nu gelijk aan

$$\text{(vx-new, vy-new)} = \text{(vx + ax, vy + ay)} = (1.00 + -0.01, 2.00 + 0.06) = (0.99, 2.06).$$

Deze overschrijdt de maximale snelheid, i.e., $\|v\| = 2.29 > V_{\max} = 2$, en zal dus volgens voorschrift moeten worden afgekapt naar een vector met een lengte die gelijk is aan de maximale snelheid. Vermenigvuldig v daarvoor met $V_{\max}/\|v\|$:

$$\text{(vx-new, vy-new)} = \frac{2}{2.29}\text{(vx-new, vy-new)} = (0.87, 1.80).$$

Nu geldt $\|\text{(vx-new, vy-new)}\| = V_{\max}$.

Om de invloed van pace keeping te berekenen, bepalen we

$$\text{velocityFactor} = \frac{\text{nominale snelheid}}{\|v\|} = \frac{1.50}{2.00} = 0.75.$$

De finale (gecorrigeerde) snelheids-vector is nu

$$\begin{aligned}&\text{paceKeeping * velocityFactor * (vx, vy) + (1 − paceKeeping) * (vx, vy)}\\ &\quad = 0.40 * 0.75 * (0.87, 1.80) + (1.0 - 0.40) * (0.87, 1.80) = (0.78, 1.62).\end{aligned}$$

g) A verplaatst zich. De nieuwe positie is nu

$$\text{(sx, sy)} = \text{(sx + vx, sy + vy)} = (3.00 + 0.78, 1.62 + 1.62) = (3.78, 2.62).$$

7, p. 79: Uitwerking:

a) A neemt waar. Waargenomen buren: C. (Deeltjes B en D vallen buiten A's waarnemings-gebied.)

b) Positie van buur: $(5, 4)$. Snelheid van buur: $(2, 3)$.

c) Gemiddelde positie van buren: mean-s = $(5.00, 4.00)$. Gemiddelde snelheid van buren: mean-v = $(2.00, 3.00)$.

d) De afstotings-vector (Eng. "repulsion vector") is nu gelijk aan de som van de verschillen in positie-vectoren tussen het deeltje en zijn buren, allen gedeeld door de respectievelijke afstand in het kwadraat. Dus

$$(\text{rep-x}, \text{rep-y}) = (4-5, 4-4)/1.00^2 = (-1.00, 0.00).$$

e) Acceleratie in de x-richting = cohesion * (mean-sx - sx) + alignment * (mean-vx - vx) + separation * rep-x $= 0.05 * (5.00 - 4.00) + 0.15 * (2.00 - 1.00) + 0.40 * -1.00 = -0.20$.

Acceleratie in de y-richting = cohesion * (mean-sy - sy) + alignment * (mean-vy - vy) + separation * rep-y $= 0.05 * (4.00 - 4.00) + 0.15 * (3.00 - 3.00) + 0.40 * 0.00 = 0.00$.

De (ruwe) acceleratie-vector (ax, ay) is hiermee $(-0.20, 0.00)$.

A's swerve parameter is nul of onvermeld, daardoor is er geen swerve. Dus de finale (gecorrigeerde) acceleratie is gelijk aan de ruwe acceleratie.

f) De (ruwe) nieuwe snelheids-vector is nu gelijk aan

$$(\text{vx-new}, \text{vy-new}) = (\text{vx} + \text{ax}, \text{vy} + \text{ay}) = (1.00 + -0.20, 3.00 + 0.00) = (0.80, 3.00).$$

Deze overschrijdt de maximale snelheid niet, i.e., $\|v\| = 3.10 \leq V_{\max} = 5$, dus de snelheidsvector hoeft niet te worden afgekapt.

Om de invloed van pace keeping te berekenen, bepalen we

$$\text{velocityFactor} = \frac{\text{nominale snelheid}}{\|v\|} = \frac{1.50}{3.10} = 0.48.$$

De finale (gecorrigeerde) snelheids-vector is nu

$$\begin{aligned}&\text{paceKeeping} * \text{velocityFactor} * (\text{vx}, \text{vy}) + (1 - \text{paceKeeping}) * (\text{vx}, \text{vy}) \\ &\quad = 0.20 * 0.48 * (0.80, 3.00) + (1.0 - 0.20) * (0.80, 3.00) = (0.72, 2.69).\end{aligned}$$

g) *A* verplaatst zich. De nieuwe positie is nu

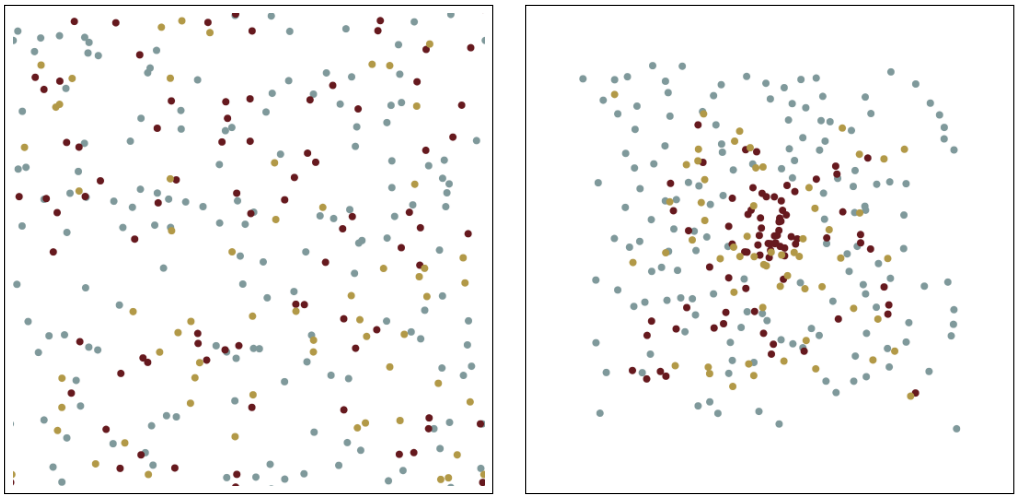
$$(\text{sx}, \text{sy}) = (\text{sx} + \text{vx}, \text{sy} + \text{vy}) = (4.00 + 0.72, 2.69 + 2.69) = (4.72, 6.69).$$

8a, p. 79: Bij mij leverde een LLM niks bruikbaars op, dus heb ik een bestaand model "Chaos Cells" van Jesse Fagan, bewerkt tot iets eenvoudigers, wat ik "Chaos Cells 2" noem:

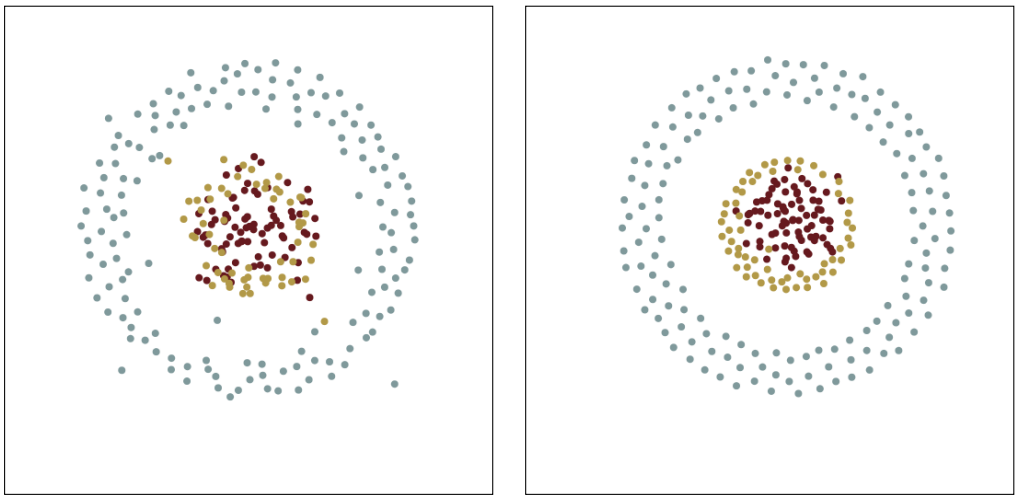
```
# aantal * (perception-radius, nominal-speed, maximum-speed, cohesion,
#                      alignment, separation, random-steering, pace-keeping)
     150 * (100, 6, 15, 0.5, 0.6, 61, 0.2, 0.2)
      80 * (100, 8, 30, 0.4, 0.1, 12, 0.5, 0.1)
      60 * ( 50, 9, 30, 0.7, 0.6, 28, 0.3, 0.4)
```

Buiten dat de parameters zijn vereenvoudigd, is de enige wijziging dat de pace keeping van het tweede type is teruggebracht naar 0.1.

8b, p. 79: Vier scherm-afbeeldingen van het resultaat. Type *A*: blauw; type *B*: roodbruin; type *C*: lichtbruin.



0 stappen 5 stappen



10 stappen 50 stappen

8c, p. 79: Zoals eerder opgemerkt is dit model een bewerking van het bestaande model genaamd “Chaos Cells” van Jesse Fagan. Doordat de pace keeping van het tweede type is teruggebracht naar 0.1 blijven deze deeltjes beter gecentreerd. Niet dat een betere centrering de norm is, maar het ziet er mooier uit. We zijn benieuwd naar jouw model!

1a, p. 81: Een beslis-probleem is NP-oplosbaar als er een algoritme bestaat dat in polynomiale tijd een oplossing voor elke instantie van dat probleem kan vinden, mits dit algoritme op keuzepunten mag forken (klonen). Uiteindelijk zal tenminste één kloon met een oplossing moeten komen.

1b, p. 81: Een beslis-probleem P is NP-volledig als het NP-oplosbaar is, en ieder ander NP-oplosbaar probleem in een tijd die polynomiaal afhangt van de lengte van de invoer, kan worden gereduceerd naar P.

1c, p. 81: Het NP-volledige Hamilton-cykel beslis-probleem moet in polynomiale tijd worden gereduceerd naar het handelsreiziger-probleem. Als we zo’n reductie kunnen vinden dan kunnen we als volgt redeneren. Als het handelsreiziger-probleem in polynomiale tijd oplosbaar zou zijn, dan zou de reductie impliceren dat het Hamilton-cykel beslis-probleem óók in polynomiale tijd oplosbaar is. Daar al bekend is dat het Hamilton-cykel probleem NP-volledig is, behoort het handelsreiziger-probleem nu ook tot de klasse van NP-volledige problemen.

2a, p. 81: Er geldt

$$\text{aantrekkelijkheid van een kant} = \frac{\text{feromoon op die kant}}{(\text{lengte van die kant})^{\beta}} = \frac{f}{d^{\beta}}.$$

In de som is $\beta = 0.5$. Dus

	A	B	C	D	E	F	G	H	I	J
C				1.123	1.025	<u>0.991</u>	<u>0.933</u>	<u>0.855</u>	1.153	1.213

Punten: 1 juiste methode, 1 juiste uitkomst modulo afrondingsfouten.

2b, p. 81: Als we, vertrekkend vanuit A, steeds de aantrekkelijkste verbinding bewandelen naar een stad die nog niet is bezocht, komen we uit bij e volgende toer.

$$A \to E \to D \to G \to B \to F \to I \to C \to J \to H \to A.$$

In dit geval is er dus maar één optimale toer. (De keuze tussen C en G vanaf I was de meest kritieke.)

Punten: 1 juiste methode, 1 juiste uitkomst.

2c, p. 81: Alle aantrekkelijkheden zijn positief, dus als we bij A starten zijn er negen mogelijke alternatieven. Na een alternatief gekozen te hebben zijn er daarna nog acht mogelijke alternatieven, etc. Dus, als er volledig wordt geëxploreerd, zijn er $9! = 362,880$ mogelijke toeren.

Punten: 1 juiste methode, 1 juiste uitkomst.

2d, p. 81: Nee. De aantrekkelijkheid van de verbindingen met onbezochte steden D, H, I en J is respectievelijk 0.95, 0.99, 1.23, en 0.91. Dus de mier zal bij exploitatie I i.p.v. J bezoeken.

Punten: 1 juiste methode, 1 juiste uitkomst.

2e, p. 81:

$$\Pr\{J \text{ wordt bezocht} \mid A, B, C, E, G \text{ al bezocht}\} = \frac{0.91}{0.95 + 0.99 + 1.23 + 0.91} = 0.223.$$

Punten: 1 juiste methode, 1 juiste uitkomst modulo afronding.

2f, p. 81: Exploitatie kiest gretig, bijziend, en kortzichtig steeds de aantrekkelijkste kant. Maar misschien kan bij exploratie door toeval een knoop via een iets minder aantrekkelijke kant worden verlaten om dan ineens een veel aantrekkelijker route te volgen. Dus een soort opoffering op de korte termijn om op de lange termijn een betere route vinden. Dus de bewering is nog niet zo heel evident, en dus misschien zelfs gewoon onwaar.

Max 2pt. voor zinvolle observaties.

3a, p. 82: Langzaam leren, maar uiteindelijk dicht bij optimum.

3b, p. 82: Snel leren, maar uiteindelijk sterke fluctuatie rond optimum.

3c, p. 82: Voorkeur voor globaal optimum (feromoon) i.p.v. lokaal optimum (kortste weg). Kleine variëteit aan routes.

3d, p. 82: Voorkeur voor locaal optimum (kortste weg) i.p.v. globaal optimum (feromoon). Grote variëteit aan routes, ook in latere generaties.

3e, p. 82: Volatiliteit notie "optimale toer" is laag, i.e., de kwaliteit van toeren zal door de generaties heen langzaam veranderen.

3f, p. 82: Volatiliteit notie "optimale toer" is hoog.

4, p. 82: Antwoord (d). Het handelsreizigerprobleem is een ja/nee vraag, bv. "kan een toer korter dan 100KM worden gevonden?". Op enig moment kan zo'n toer worden gevonden en kan er worden gestopt.

5a, p. 82: Antwoord: ja. Immers, er er is een toer aan te wijzen met lengte hoogstens 20.

5b, p. 83: Antwoord: nee. Dit kan op verschillende manier worden beredeneerd. De meest voor de hand liggende manier is om gewoon de lengte van alle toeren te bekijken:

12341: 23; 12431: 20; 13241: 21; 13421: 20; 14231: 21; 14321: 23.

Er is dus geen toer met een lengte van 19.

5c, p. 83: Minimale lengte 20 bij $1-2-4-3-1$ of, omgekeerd, $1-3-4-2-1$.

5d, p. 83: Weg $1-2: 0.5/6^{0.5} = 0.204$, weg $1-3: 0.1/2^{0.5} = 0.071$, weg $1-4: 0.3/8^{0.5} = 0.106$, weg $2-3: 0.2/4^{0.5} = 0.100$, weg $2-4: 0.4/7^{0.5} = 0.151$, weg $3-4: 0.3/5^{0.5} = 0.134$. We krijgen:

Naar:	1	2	3	4
Van 1:	–	0.204	0.071	0.106
Van 2:	0.204	–	0.100	0.151
Van 3:	0.071	0.100	–	0.134
Van 4:	0.106	0.151	0.134	–

5e, p. 83: Stad 4, want Stad 1 is al bezocht en $0.134 > 0.100$.

5f, p. 83: Z.b.d.a. beginnen we bij Stad 1. Vanuit Stad 1 is Stad 2 het aantrekkelijkst (0.204). Vanuit Stad 2 mag de mier alleen nog maar naar Stad 3 of Stad 4 want Stad 1 is al bezocht. Daarvan is de weg naar Stad 4 het aantrekkelijkst (0.151). Na Stad 4 kan de mier alleen nog naar Stad 3, want de overige steden zijn al bezocht. Om een echte toer te maken gaat de mier daarna weer terug naar Stad 1. Dus het antwoord is $1-2-4-3-1$.

5g, p. 83: Deze kans is gelijk aan $P\{\text{Stad } 2\} = \frac{0.100}{0.100+0.134} = 0.427$.

5h, p. 83: In dit geval zijn er maar twee alternatieven, en geldt
$P\{\text{stad } 4\} = 1 - P\{\text{stad } 2\} = 1 - 0.427 = 0.573$.

5i, p. 83: Dit zijn alle mogelijke toeren vanuit 1. De eerste stad ligt vast en de overige drie kunnen willekeurig worden gepermuteerd. We spreken dus over $3! = 6$ verschillende mogelijke toeren.

5j, p. 83: In ieder punt is er kans op exploratie, dus opnieuw alle zes mogelijke toeren.

5k, p. 83: De lengte van toer $T = 1-2-3-4-1$ is 23. Voor alle kanten die op T liggen is $\Delta m(i,j) = 1/23$. Voor alle overige kant is $\Delta m(i,j) = 0$.

5l, p. 83: Er geldt $m(1,2) = \rho \times 0.5 + (1-\rho) \times 1/23 = 0.9 \times 0.5 + 0.1 \times 1/23 = 0.454$, $m(2,3) = 0.9 \times 0.2 + 0.1 \times 1/23 = 0.184$, $m(3,4) = 0.9 \times 0.3 + 0.1 \times 1/23 = 0.274$, $m(4,1) = 0.9 \times 0.3 + 0.1 \times 1/23 = 0.274$. Voor de overige kanten geldt $m(x,y) = 0.9 \times m(x,y) + 0.1 \times 0$.

5m, p. 83: We gebruiken de tabel van aantrekkelijkheden van paden bij exploratie, aangemaakt bij het antwoord van onderdeel 5d.

We berekenen eerst de kans dat één mier een optimale toer aflegt. De twee optimale toeren zijn $1-2-4-3-1$ en de omkering ervan: $1-3-4-2-1$.

$P\{1-2-4-3-1\}$ is gelijk aan de kans dat er van 1 naar 2 gelopen wordt, maal de kans dat er van 2 naar 4 gelopen wordt, maal de kans dat er van 4 naar 3 gelopen wordt, maal de kans dat er van 3 naar 1 gelopen wordt

$$= \left(\alpha + (1-\alpha)\frac{0.204}{0.204+0.071+0.106}\right) \times \left(\alpha + (1-\alpha)\frac{0.151}{0.100+0.151}\right) \times 1 \times 1 = 0.615$$

De factor $\alpha + (1-\alpha)\frac{0.204}{0.204+0.071+0.106}$ wordt verklaard doordat 1-2 kan worden gelopen dankzij exploiteren (α) of exploreren ($(1-\alpha)\frac{0.204}{0.204+0.071+0.106}$). Evenzo voor de tweede factor. De laatste twee kansen zijn 1 omdat er niets meer te kiezen valt.

Analoog: $P\{1-3-4-2-1\}$

$$= (1-\alpha)\frac{0.106}{0.204+0.071+0.106} \times \left(\alpha + (1-\alpha)\frac{0.134}{0.100+0.134}\right) \times 1 \times 1 = 0.109$$

Nu:

$$P\{1-2-4-3-1 \text{ of } 1-3-4-2-1\} = P\{1-2-4-3-1\} + P\{1-3-4-2-1\} = 0.615 + 0.109 = 0.724.$$

De kans dat één mier een optimale toer aflegt is dus 0.724.

Nu gaat het snel:

- De kans dat één mier géén optimale toer aflegt is dus $1 - 0.724 = 0.276$.
- De kans dat alle mieren géén optimale toer afleggen is dus $(0.276)^{100} \approx 1e\text{-}56$. (We mogen al deze honderd kansen met elkaar vermenigvuldigen omdat aangenomen mag worden dat alle mieren onafhankelijk van elkaar lopen.)
- De kans dat tenminste één mier wél een optimale toer aflegt is dus $1 - 1e\text{-}56 \approx 1$.

6a, p. 83: In het begin ligt er $N^2 f$ feromoon. Na T tikken is daar nog

$$(1-p)^T N^2 f$$

van over. Verder komt er elke tik KD feromoon bij dankzij de mieren. (Waar dat extra feromoon terechtkomt doet er niet toe.) Dit extra feromoon verdampt meteen ook weer gedeeltelijk na elke tik. Dus na T tikken is er nog

$$\underbrace{(1-p)KD + (1-p)^2 KD + (1-p)^3 KD + \cdots + (1-p)^T KD}_{T}$$

van het mieren-feromoon over: $(1-p)KD$ van de laatste tik, $(1-p)^2 KD$ van de voorlaatste tik, $(1-p)^3 KD$ van de twee na laatste tik, enzovoort, tot en met $(1-p)^T KD$ van de eerste tik. Als $p \neq 0$ dan

$$\begin{aligned}(1-p)KD + (1-p)^2 KD + (1-p)^3 KD + \cdots + (1-p)^T KD \\ = (1-p)\left[1 + (1-p) + (1-p)^2 + \cdots + (1-p)^{T-1}\right] KD \\ = (1-p)\frac{1-(1-p)^T}{1-(1-p)} KD = (1-p)\frac{1-(1-p)^T}{p} KD.\end{aligned}$$

Dus als $p \neq 0$ dan bevindt zich na T tikken

$$(1-p)^T N^2 f + (1-p)\frac{1-(1-p)^T}{p} KD$$

feromoon op het grid. Als $p = 0$ dan bevindt zich na T tikken

$$N^2 f + T \cdot KD$$

feromoon op het grid.

– 1 punt voor formule begin-feromoon $(1-p)^T N^2 f$
– 1 punt voor formule mieren-feromoon $(1-p)KD + (1-p)^2 KD + (1-p)^3 KD + \cdots + (1-p)^T KD$
– 1 punt voor juiste combinatie en uitwerking van die formules
– 1 punt voor onderscheid $p = 0$ en $p \neq 0$

6b, p. 83: Als $p \neq 0$ dan

$$\lim_{T\to\infty} (1-p)^T N^2 f + (1-p)\frac{1-(1-p)^T}{p} KD = 0 \cdot N^2 f + (1-p)\frac{1-0}{p} KD = \frac{1-p}{p} KD.$$

Dit is ook de kleinste bovengrens. Als $p = 0$ dan is de groeisnelheid constant, te weten in totaal KD extra feromoon per tik. In het laatste geval is er ook geen bovengrens voor de totale hoeveelheid feromoon.

– max. 3 punten voor het vinden van de juiste limiet
– 1 punt voor onderscheid $p = 0$ en $p \neq 0$

7a, p. 83: Wat is de kortste toer gegeven G?

7b, p. 84: Bestaat er een toer in G met lengte hoogstens M?

7c, p. 84: M is een parameter die kan variëren maar vaststaat zodra de beslisvraag gesteld wordt.

7d, p. 84: Antwoord: ja. Immers, er zijn genoeg toeren aan te wijzen van lengte hoogstens 20.

7e, p. 84: Antwoord: nee. Immers, elke toer bevat tenminste vier kanten. Omdat er maar twee kanten lengte 2 bezitten, en de andere kanten een lengte groter dan 2 bezitten, zal het niet mogelijk zijn een toer met lengte 8 of minder af te leggen.

7f, p. 84: Minimale lengte 18 (bij $1-3-2-4-1$ of omgekeerd $1-4-2-3-1$).

7g, p. 84: Weg $1-2 : 1.1/4 = 0.275$, weg $1-3 : 0.4/16 = 0.025$, weg $1-4 : 0.2/49 = 0.0040816$, weg $2-3 : 0.8/4 = 0.2$, weg $2-4 : 0.3/25 = 0.012$, weg $3-4 : 0.1/64 = 0.0015625$.

7h, p. 84: Stad 2.

7i, p. 84: Toer $1-2-3-4-1$.

7j, p. 84: Deze kans is gelijk aan $P\{\text{stad } 2\} = (0.8/4)/(0.8/4 + 0.1/64)$.

7k, p. 84: In dit geval zijn er maar twee alternatieven, en geldt $P\{\text{stad } 4\} = 1 - P\{\text{stad } 2\}$.

7l, p. 84: Dit zijn alle mogelijke toeren vanuit 1. De eerste stad ligt vast en de overige drie kunnen willekeurig worden gepermuteerd. We spreken dus over zes verschillende mogelijke toeren.

7m, p. 84: In ieder punt is er kans op exploratie, dus opnieuw alle zes mogelijke toeren.

7n, p. 85: De lengte van toer $T = 1-2-3-4-1$ is 19. Voor alle kanten die op T liggen is $\Delta m(i,j) = 1/19$. Voor alle overige kant is $\Delta m(i,j) = 0$.

7o, p. 85: Er geldt $m(1,2) = \rho \cdot 1.1 + (1-\rho) \cdot 1/19 = 0.9 \cdot 1.1 + 0.1 \cdot 1/19$,
$m(2,3) = 0.9 \cdot 0.8 + 0.1 \cdot 1/19$, $m(3,4) = 0.9 \cdot 0.1 + 0.1 \cdot 1/19$, $m(4,1) = 0.9 \cdot 0.2 + 0.1 \cdot 1/19$. Voor de overige kanten geldt $m(x,y) = 0.9 \cdot m(x,y)$.

7p, p. 85: Het is mogelijk dat door exploratie alle 10 mieren een sub-optimaal pad lopen en uiteindelijk dus een sub-optimaal pad wordt geflitst.

Het uitrekenen van de exacte kans daarop is nog behoorlijk veel werk. Qua afstand zijn er twee optimale paden, nl. $1-3-2-4-1$ en $1-4-2-3-1$. Beide leveren een optimale toer ter lengte 18 op. Bij exploitatie van het feromoon zal een mier echter alleen het pad $1-3-2-4-1$ aflopen.

De kans dat één mier een optimaal pad loopt is gelijk aan de kans dat één mier het optimale (en door feromoonspoor aangegeven) pad $1-3-2-4-1$ loopt, plus de kans dat één mier het alternatieve (niet door feromoonspoor aangegeven) optimale pad $1-4-2-3-1$ loopt.

De kans dat één mier het optimale (en door feromoonspoor aangegeven) pad $1-3-2-4-1$ afloopt is gelijk aan

$$(\alpha+(1-\alpha)P\{\text{1-3 door expl.}\})\cdot(\alpha+(1-\alpha)P\{\text{3-2 door expl.}\})\cdot(\alpha+(1-\alpha)P\{\text{2-4 door expl.}\})$$
$$=(0.9+0.1\frac{0.025}{0.275+0.025+0.0040816})\cdot(0.9+0.1\frac{0.2}{0.2+0.0015625})\cdot 1\approx 0.91.$$

De kans dat één mier het alternatieve (niet door feromoonspoor aangegeven) optimale pad $1-4-2-3-1$ afloopt is gelijk aan

$$(\alpha+(1-\alpha)P\{\text{1-4 door expl.}\})\cdot(\alpha+(1-\alpha)P\{\text{4-2 door expl.}\})\cdot(\alpha+(1-\alpha)P\{\text{2-3 door expl.}\})$$
$$=(0.0+0.1\frac{0.0040816}{0.275+0.025+0.0040816})\cdot(0.0+0.1\frac{0.12}{0.12+0.0015625})\cdot 1\approx 0.00013.$$

Samen is die kans dus $0.91+0.00013\approx 0.91$.

De kans dat één mier een sub-optimaal pad loopt is dus ongeveer gelijk aan $1-0.91=0.09$. Dus de kans dat 10 mieren een sub-optimaal pad lopen is gelijk aan $0.09^{10}\approx 3.5\times 10^{-11}$. Die kans is dus erg klein, maar nog steeds positief.

1a, p. 88: De Japanse school tracht het gedrag van de gele slijmzwam te modelleren door netwerken aan te leggen met stromen. De wiskunde die bij deze modellen hoort is continue wiskunde. Stroomsterkten en leiding-capaciteiten zijn reëelwaardige grootheden en hun veranderingen worden gemodelleerd met differentiaalvergelijkingen. De Engelse school modelleert het gedrag van de gele slijmzwam daarentegen door een systeem van honderden tot duizenden turtles (of mieren) los the laten op een grid. De turtles deponeren een feromoon (geurstof) en laten hun loopgedrag omgekeerd ook beïnvloeden door de locatie van feromoon.

1b, p. 88: Het deeltje bestaat uit een lichaam en drie sensoren met een lengte SO (sensor offset). Eén sensor steekt recht naar voren, de andere twee steken schuin vooruit met een hoek SA (sensor angle). SO en SA zijn parameters. Typische waarden voor SO zijn $5, 8, 10, 15, \ldots$. Typische waarden voor SA zijn $45^0, 60^0, \ldots$. De sensoren hebben soms nog een bepaalde oppervlakte, bijvoorbeeld één patch (of pixel), of 4 patches.

1c, p. 89: In de sensorische fase wordt gesnoven hoeveel feromoon zich op de uiteinden van de sensoren (geursprieten) bevinden, en afhankelijk daarvan wordt er eventueel gedraaid. Laat de FL, FF, FR de feromoonwaarden van de linker-, midden-, en rechtersensor zijn.

- Als FF > FL en FF > FR, wordt er niet gedraaid.
- Anders, als FF < FL en FF < FR wordt er RA graden naar links of rechts gedraaid, waarbij RA (rotation angle) een vaste draaihoek is, meestal gelijk aan SA (sensor angle).
- Anders, als FL > FR wordt er RA graden naar links gedraaid.
- Anders, als FR > FL wordt er RA graden naar rechts gedraaid.
- In alle resterende gevallen wordt er niet gedraaid.

1d, p. 89: Na de sensorische fase wordt in de motorische fase eventueel een stap gedaan ter grootte S (step size). Typische waarden van S zijn $1, 2, 3, \ldots$. Er wordt geen stap gedaan al de doelpatch bezet is. In dat geval blijft de mier op zijn plek. Als er alleen als de mier op een nieuw veld arriveert wordt feromoon afgescheiden (bijvoorbeeld 10 eenheden). Dus als de mier blijft stilstaan wordt geen feromoon afgescheiden.

1e, p. 89: Dit is een resterend geval. De mier zal niet draaien. Daarna zal de mier indien mogelijk een stap ter lengte S vooruit doen. Als $S > 1$ en het doelveld (doelpatch) is vrij maar een tussenliggend veld bezet is, dan zal afhankelijk van de implementatie de mier wel of niet over het bezette veld heen "springen". (In de meeste modellen "springt" de mier.)

1f, p. 89: Als FL en FR beiden groter zijn dan FF en er bij toeval naar links gedraaid wordt, of als FL $\geq$ FF $> FR$. Dan word er zeker naar links gedraaid.

2a, p. 89: Het feromoon zal, nadat alle mieren hebben bewogen, verdampen en verspreiden. Het verdampen vindt plaats door bijvoorbeeld 10% van het aanwezige feromoon te laten vervliegen. Het verspreiden (Eng.: *diffuse*) vindt plaats door elke patch bijvoorbeeld 10% feromoon te laten weglekken naar de acht buren. (De buren lekken op hun beurt natuurlijk weer feromoon "terug" naar de patch in kwestie.)

In antwoord op "Zo nee, waarom niet?": diffusie is niet noodzakelijk. Maar dan moet i.h.a. de sensor-width wel groter dan 1 zijn om de mier een betrouwbaar beeld te geven van de hoeveelheden feromoon zich links, midden en rechts van de mier bevinden.

2b, p. 89: Laten we beginnen te zeggen dat aftrekken met een bepaald getalletje i.p.v. vermenigvuldigen met een bepaalde factor < 1 indruist tegen het idee van verdamping. Het idee van verdamping is dat na elke zoveel tijd $x\%$ van een stof is verdwenen. Niet dat na elke zoveel tijd een absolute hoeveelheid van een stof is verdwenen.

Maar aftrekken i.p.v. vermenigvuldigen werkt ook minder goed. Want stel je voor dat verdamping plaats vindt door aftrekken. Dat zou betekenen dat weinig gefrequenteerde patches een negatieve feromoonwaarde kunnen krijgen. Dat strookt niet erg met het idee dat feromoon een stof is die niet of in positieve hoeveelheden op een plek aanwezig is. Een ander probleem met aftrekken i.p.v. vermenigvuldigen is dat het absolute feromoonniveau op de patches onbeperkt kan stijgen of dalen. Stel dat er N mieren zijn die ieder per tik (of tijdsmoment) een verwachte hoeveelheid van $f > 0$ feromoon dumpen en er per tik van elke patch $F > 0$ feromoon wordt afgetrokken. Wat er per tik globaal aan feromoon bijkomt is dus Nf eenheden. Wat er per tik globaal aan feromoon afgaat is dus k^2F eenheden. Wat er per tik netto aan feromoon bijkomt is dus $Nf - k^2F$ eenheden. Als dit getal groter is dan nul stijgt de totale hoeveelheid feromoon na elke tik, en er is geen bovengrens. Als dit getal kleiner is dan nul daalt de totale hoeveelheid feromoon na elke tik, en er is geen ondergrens. Interessant is simulaties te bekijken waar $Nf - k^2F = 0$.

Als per tik iedere mier op de tegel onder zich een verwachte hoeveelheid van $f \geq 0$ feromoon dumpt, zal er per tik fN feromoon bijkomen. Als per tik $0 \leq d \leq 1$ feromoon verdampt, zal vervolgens $(1-d)\%$ van de dan aanwezige feromoon verdampen. Hoeveel feromoon op enig moment aanwezig kun je uitrekenen door de rij feromoon-concentraties van begin af aan op te schrijven:

$$0 \xrightarrow[\text{feromoon}]{\text{mieren leggen}} fN \xrightarrow[\text{verdampt}]{\text{feromoon}} fN(1-d) \xrightarrow[\text{feromoon}]{\text{mieren leggen}} fN(1-d) + fN \xrightarrow[\text{verdampt}]{\text{feromoon}}$$
$$(fN(1-d) + fN)(1-d) \xrightarrow[\text{feromoon}]{\text{mieren leggen}} (fN(1-d) + fN)(1-d) + fN \xrightarrow[\text{verdampt}]{\text{feromoon}}$$
$$((fN(1-d) + fN)(1-d) + fN)(1-d) \longrightarrow \ldots$$

Als we $g = fN$ en $e = 1 - d$ zetten, dan krijgen we dus de rij

$$0 \to ge \to g(e^2 + e) \to g(e^3 + e^2) \to g(e^4 + e^3 + e^2) \to \ldots$$

Aangezien de geometrische reeks $e^n + \cdots + e^2 + e + 1$ gelijk is aan $(1 - e^{n+1})/(1 - e)$ reduceert $e^n + \cdots + e$ tot $e(1 - e^n)/(1 - e)$. Dus na de n-de tik ligt er in totaal

$$ge\frac{1 - e^n}{1 - e}$$

feromoon. Als na elke slag $d\%$ feromoon verdampt en $d > 0$, dan is $e < 1$ en geldt

$$\lim_{n\to\infty} ge\frac{1 - e^n}{1 - e} = ge \lim_{n\to\infty} \frac{1 - e^n}{1 - e} = ge\frac{1}{1 - e} = g\frac{1 - d}{d}.$$

(Limiet geometrische reeks.) Dus als na elke slag $d > 0$ feromoon verdampt dan geldt dat de hoeveelheid feromoon uiteindelijk convergeert (toegroeit) naar g/d als de mieren net feromoon hebben gelegd en $g(1 - d)/d$ als er net verdamping heeft plaatsgevonden. Beiden zijn eindige hoeveelheden. Een simulatie op de computer zal daardoor niet vastlopen door een "out of bound"-error o.i.d., en het feromoonkleurings-algoritme kan gevoeglijk aannemen dat de feromoon-concentratie op elke patch tussen de 0 en de $g(1 - d)/d$ ligt, en als het feromoon redelijk verdeeld is, tussen de 0 en de $g(1 - d)/(d8k^2)$, waarbij de factor 8 weergeeft dat feromoon zich nooit op 1/8 van het grid concentreert. (Dit is een ongefundeerde maar redelijke aanname.)

2c, p. 89: Dan zullen mieren alleen nog maar letten op lokale feromoon-concentraties en geen aanleiding hebben om weg te bewegen van hun huidige lokatie. De mieren zullen rondjes gaan lopen in poeltjes van feromoon die ze daar zelf hebben neergelegd.

3a, p. 89: Er zijn tenminste twee benaderingen: 1) behoud de forward sensor, of 2) niet, en buig de forward sensor af naar een derde schuine sensor zodat je een driepoot (melkkruk) hebt. Het laatste idee is toch niet zo goed, omdat we altijd het de hoeveelheid feromoon recht voor het deeltje willen kunnen vergelijken met de hoeveelheid feromoon in andere richtingen. Er zal dus een vierde sensor bij moeten. (Immers, ga na dat de mier anders altijd aan één kant blind is.) Eén mogelijke uitbreiding is dus: één sensor recht vooruit, en drie sensoren schuin vooruit in gelijke hoeken α met de middelloodlijn van het deeltje, en gelijkelijk verdeeld over het frontale vlak (120-120-120). Dit zou kunnen worden uitgelegd in termen van pitch, yaw en roll, maar dat zou iets te ver voeren. Eigenlijk hebben we het weer over een melkkruk, maar dan met een vierde poot in het midden. Vijf of meer sensoren kunnen natuurlijk ook, maar de vraag is dan of dat iets toevoegt.

3b, p. 89: Een aantal van $k = 2$ sensoren is niet genoeg omdat sensoren zelf statisch zijn, dat wil zeggen dat sensoren niet kunnen "rollen" zoals ogen dat wel kunnen.

3c, p. 89: Van F, L, en R zijn $3! = 6$ ordeningen mogelijk. In de volgende tabel is α een vaste positieve draaihoek, en pick(A) geeft een willekeurig element uit de verzameling A terug.

Ordening	Draai
$\mathbf{F} > L > R$ $\mathbf{F} > R > L$	0
$L > \mathbf{F} > R$	α
$R > \mathbf{F} > L$	$-\alpha$
$L > R > \mathbf{F}$ $R > L > \mathbf{F}$	pick$\{\alpha, -\alpha\}$

3d, p. 90: In de 2D tabel is F een soort scheider. Laat T alle sensorvariabelen bevatten die hoger scoren dan de sensorvariable F. Als $T = \emptyset$ dan is er geen reden om te draaien. Als $T \neq \emptyset$, draai dan in de richting van een willekeurig element uit T,—ook al scoort een ander element uit T wellicht hoger. In wiskunde:

$$\text{draai} \begin{cases} 0, & \text{als } T = \emptyset, \\ \text{pick}(T) & \text{anders,} \end{cases}$$

waarbij $T = \{t \mid t > F\}$.

3e, p. 90: Dit ligt eigenlijk al besloten in het antwoord uit het vorige onderdeel: draai niet als in de richting recht vooruit het meeste feromoon ligt. Draai anders in een willekeurige richting waarin meer feromoon ligt dan in de richting recht vooruit.

3f, p. 90: Er veranderd niets. De motorische fase in 3D bevat dezelfde instructies als de motorische fase in 2D.

1, p. 91: Er geldt

$$v_P = P - s = \begin{pmatrix} 6 \\ 4 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 3 \end{pmatrix} \text{ en } v_G = G - s = \begin{pmatrix} 3 \\ 4 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

Dus

$$\begin{aligned} v_{\text{nieuw}} &= Iv + (1 - I)[A_P v_P + A_G v_G] \\ &= 0.9 \begin{pmatrix} 2 \\ -1 \end{pmatrix} + (1 - 0.9) \left[0.6 \begin{pmatrix} 5 \\ 3 \end{pmatrix} + 0.4 \begin{pmatrix} 2 \\ 3 \end{pmatrix} \right] \\ &= \begin{pmatrix} 2.18 \\ -0.60 \end{pmatrix}. \end{aligned}$$

2, p. 91: Er geldt

$$v_P = P - s = \begin{pmatrix} -7 \\ 8 \end{pmatrix} - \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 6 \\ 7 \end{pmatrix} \text{ en } v_G = G - s = \begin{pmatrix} 4 \\ -5 \end{pmatrix} - \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ -6 \end{pmatrix}.$$

Dus

$$\begin{aligned} v_{\text{nieuw}} &= Iv + (1 - I)[A_P v_P + A_G v_G] \\ &= 0.7 \begin{pmatrix} -3 \\ 2 \end{pmatrix} + (1 - 0.7) \left[0.8 \begin{pmatrix} 6 \\ 7 \end{pmatrix} + 0.2 \begin{pmatrix} 5 \\ -6 \end{pmatrix} \right] \\ &= \begin{pmatrix} -0.36 \\ -0.08 \end{pmatrix}. \end{aligned}$$

3, p. 91:

$$\begin{aligned} v_{\text{nieuw}} &= Iv + (1 - I)[A_P v_P + A_G v_G] \\ &= 0.8 \begin{pmatrix} 2 \\ -2 \end{pmatrix} + (1 - 0.8) \left[0.3 \begin{pmatrix} 1 \\ 5 \end{pmatrix} + 0.7 \begin{pmatrix} 3 \\ 7 \end{pmatrix} \right] \\ &= \begin{pmatrix} 2.08 \\ -0.32 \end{pmatrix}. \end{aligned}$$

4, p. 91:

$$\begin{aligned} v_{\text{nieuw}} &= Iv + (1-I)[r_P A_P v_P + r_G A_G v_G] \\ &= 0.8 \begin{pmatrix} 2 \\ -2 \end{pmatrix} + (1-0.8) \left[0.3 \cdot 0.3 \begin{pmatrix} 1 \\ 5 \end{pmatrix} + 0.6 \cdot 0.7 \begin{pmatrix} 3 \\ 7 \end{pmatrix} \right] \\ &= \begin{pmatrix} 1.87 \\ -0.92 \end{pmatrix}. \end{aligned}$$

5, p. 91: De inbreng van ruis kan worden gevarieerd met een parameter $\lambda \in [0,1]$, als volgt:

$$\begin{aligned} r_P &= (1-\lambda) + \lambda \cdot \text{rand}(0,1) \\ r_G &= (1-\lambda) + \lambda \cdot \text{rand}(0,1), \end{aligned}$$

waarbij $\text{rand}(0,1)$ een ter plekke gegenereerd toevalsgetal in $[0,1]$ is. In het extreme geval $\lambda = 0$ zijn r_P en r_G altijd gelijk aan 1, en is er sprake van een een ruisvrije update. In het extreme geval $\lambda = 1$ zijn r_P en r_G toevalsgetallen tussen de 0 en 1, en is er sprake van een update met volledige toevalscomponent. Daartussenin is er sprake van gedeeltelijke ruis, afhangende van λ.

6, p. 92: Wat voorbeelden:

$$\begin{aligned} v_{\text{nieuw}} &= Iv + (1-I)[A_P v_P + G A_G] + r \\ &\quad \text{of } Iv + (1-I)[A_P v_P + G A_G + r]. \end{aligned}$$

waarbij $r \in [0,1]^2$ of zelfs $r \in [0,m]^2$, met $m \in \mathbb{N}$. Op deze manier wordt de correctierichting $v_P + v_G$ in meer of mindere mate niet meer gerespecteerd, wat op zich het PSO idee niet teniet doet. Er is alleen meer exploratie ten kostte van exploitatie.

7, p. 92:
- Een populatiegrootte van 0 zet geen zoden aan de dijk: er gebeurt niets. Een populatiegrootte van 1 heeft niet zo veel zin, want er is dan geen verschil tussen een globale oplossing en een particuliere oplossing. Voor een authentiek PSO algoritme moeten er tenminste 2 deeltjes zijn. Voor serieuze PSO toepassingen lijkt een aantal van 1000 deeltjes wel voldoende, maar dat is met de spreekwoordelijke natte vinger.

- Een waarde $R = 0$ impliceert dat deeltjes bij initialisatie uniform over de zoekruimte worden verspreid. Zonder voorkennis van de zoekruimte lijkt dit het veiligst: er worden dan geen gedeeltes á priori van zoeken uitgesloten. Een maximum-waarde van R wordt eigenlijk alleen bepaald door de grenzen van de zoekruimte.

- Inertia ligt per definitie in het interval $[0,1]$. Als $I = 0$, vergeten deeltjes bij elke update hun oorspronkelijke richting en snelheid. Als gevolg bezitten ze geen 'swing" meer, in die zin dat ze elke keer compromisloos bewegen naar de vector

  $$r_P A_P v_P + r_G A_G v_G.$$

  Het gevolg daarvan is dat deeltjes makkelijker blijven steken in locale optima.

  Als $I = 1$, blijven deeltjes zich bewegen in de richting van hun oorspronkelijke snelheidsvector. De invloed van lokaal en globaal gevonden oplossingen is dan nul.

- De voorkeur voor de de tot nu toe globaal best gevonden oplossing, A_G, ligt in het interval $[0, 1]$. Als $A_G = 0$, negeren deeltjes de tot nu toe globaal best gevonden oplossing. Ze maken dan geen gebruik van elkaars informatie. Het gevolg is dat deeltjes kortzichtig en individueel met grote kans convergeren naar een particulier sub-optimum.

  Als $A_G = 1$, bezitten deeltjes geen "eigen agenda". Ze gaan in dat geval alleen af op wat er toevallig door random exploreren in de groep gevonden wordt. Er is wel convergentie, maar door het ontbreken van particulier zoeken is de snelheid van convergentie zeer laag.

- De mate van randomness, λ, ligt in het interval $[0, 1]$. Als $\lambda = 0$, is alle beweging deterministisch, en wordt exploratie volledig bepaald door v, v_P, en v_G. Daardoor kunnen, bij exploratie, interessante punten in de zoekruimte gemist worden. Met andere woorden: een beetje ruis zorgt voor variatie in het zoekproces, zodat de kans groot is dat elk punt in de zoekruimte op de lange duur een keertje bezocht wordt.

  Als $\lambda = 1$ wordt informatie uit v_P en v_G vertroebeld door ruis. Echter, bij alle positieve waarden van r_P en r_G is de correctierichting

  $$r_P A_P v_P + r_G A_G v_G$$

  nog altijd wel steeds de juiste. Zie in dat verband ook Vraag 6 op pagina 92, en het antwoord daar op.

- Praktische max-min-waarden voor de snelheidslimiet zijn 0, respectievelijk 100. Het laatste getal is met de natte vinger. Bij een lage selheidslimiet zullen deeltjes vanzelfsprekend langzamer bewegen. Bij een hoge snelheidslimiet zullen deeltjes te snel door de zoekruimte bewegen, en is het waarschijnlijker dat deeltjes over locale en globale minima heen schieten.

1, p. 94: • α bepaalt de hoeveel ruis in verplaatsingen. Een lage waarde van α, te weten $\alpha = 0$, zorgt er voor dat het systeem als geheel deterministisch is. Dat is op zich niet zo erg, maar in ongelukkige gevallen kan het gebeuren dat daardoor plekken in de zoekruimte gemist worden. Een hoge waarde van α, zorgt voor veel willekeur en langzame(re) convergentie. Het voordeel is een betere exploratie.

- β bepaalt de stapgrootte van verplaatsingen. Een minimale waarde van β, te weten $\beta = 0$, zorgt er voor dat deeltjes een pure zg. toevals-wandeling (Eng.: random walk) uitvoeren: elke tijdseenheid doen ze zomaar een stapje in welke richting dan ook. Oplossingen worden dan gevonden door puur toeval. Een hoge waarde van β zorgt er voor dat de stapgrootte naar het doel-deeltje groter wordt, en misschien wel te groot, zodat deeltjes hun doel voorbij schieten.

- γ bepaalt de de variatie van aantrekking. Een minimale waarde van γ, te weten $\gamma = 0$, zorgt er voor dat

  $$e^{-\gamma r_{21}^2} = e^{-0} = 1,$$

  zodat de aanpassing lineair is, ongeacht de afstand tussen s_1 en s_2. Een hoge waarde van γ, bv. $\gamma = 10$, zorgt er voor dat de aantrekkingskracht van ver gelegen deeltjes erg klein is, vergeleken met de aantrekkingskracht van dichtbij gelegen deeltjes. Om deze reden wordt γ vaak dicht bij nul gekozen.

2a, p. 94: De functiewaarde bij s_2 is groter dan die bij s_1, dus s_1 zal naar s_2 toe bewegen, en s_2 zal op haar plaats blijven.

$$\begin{aligned} s_1^{\text{nieuw}} &= s_1 + \beta e^{-\gamma r_{12}^2}(s_2 - s_1) + \alpha\epsilon \\ &= \begin{pmatrix}1\\3\end{pmatrix} + 1 \cdot e^{(-10^{-3}\times 5^2)} \left[\begin{pmatrix}4\\7\end{pmatrix} - \begin{pmatrix}1\\3\end{pmatrix}\right] + 1 \cdot \begin{pmatrix}-1\\0.5\end{pmatrix} \\ &= \begin{pmatrix}1\\3\end{pmatrix} + e^{-0.025}\begin{pmatrix}3\\4\end{pmatrix} + \begin{pmatrix}-1\\0.5\end{pmatrix} \\ &\approx \begin{pmatrix}1\\3\end{pmatrix} + 0.975\begin{pmatrix}3\\4\end{pmatrix} + \begin{pmatrix}-1\\0.5\end{pmatrix} \\ &= \begin{pmatrix}2.925\\7.400\end{pmatrix}. \end{aligned}$$

De y-coördinaat van de nieuwe waarde van s_1 schiet de y-coördinaat van s_2 voorbij, omdat per toeval $\epsilon_y = +0.5$. De y-coördinaat van s_2 blijft natuurlijk onveranderd.

2b, p. 95: De functiewaarde bij s_1 is groter dan die bij s_2, dus s_2 zal naar s_1 toe bewegen, en s_1 zal op haar plaats blijven.

$$\begin{aligned} s_2^{\text{nieuw}} &= s_2 + \beta e^{-\gamma r_{21}^2}(s_1 - s_2) + \alpha\epsilon \\ &= \begin{pmatrix}5\\7\end{pmatrix} + 2 \cdot e^{(-10^{-4}\times 40)} \left[\begin{pmatrix}3\\1\end{pmatrix} - \begin{pmatrix}5\\7\end{pmatrix}\right] + 2 \cdot \begin{pmatrix}0.00\\0.25\end{pmatrix} \\ &= \begin{pmatrix}5\\7\end{pmatrix} + 2e^{-0.004}\begin{pmatrix}-2\\-6\end{pmatrix} + \begin{pmatrix}0\\0.5\end{pmatrix} \\ &\approx \begin{pmatrix}5\\7\end{pmatrix} + 1.99\begin{pmatrix}-2\\-6\end{pmatrix} + \begin{pmatrix}0\\0.5\end{pmatrix} \\ &= \begin{pmatrix}1.02\\-4.44\end{pmatrix}. \end{aligned}$$

Het individu s_1 blijft op haar plaats. De nieuwe waarde van s_2 schiet de waarde van s_1 voorbij. Dus de parameterwaarde $\beta = 2$ lijkt wat aan de ruime kant voor aanpassingen, en moet naar benden worden bijgesteld.

3, p. 95: Uit de abstract:

1. Een logaritmisch spiraalpad is ontworpen om de exploitatie van de zoekruimte voor lokaal zoeken te verbeteren, wat leidt tot een versnelling van convergentie.

2. Een adaptieve schakelaar (ratio) wordt voorgesteld om de exploitatie en de exploratie tijdens het zoekproces in evenwicht te brengen. Volgens de variatie van de huidige fitnessfunctiewaarde kan elke deeltje zijn strategie aanpassen.

3. Een nieuwe adaptieve logaritmische spiraal-Lévywandeling is ontwikkeld voor een stabieler en nauwkeuriger optimaal resultaat. Een Lévywandeling is een random walk met af en toe grote, toevallige, stappen. Daarmee wordt bewerkstelligt dat een deeltje af en toe zijn lokale zoekactiviteit (als β klein en γ groot is) rigoreus verplaatst naar een ander deel in de globale zoekruimte.

4a, p. 95: Simpel: we willen weten hoe waarschijnlijk het is dat alle koppels (ongeordende paren) na M runs ook werkelijk zijn bekeken.

4b, p. 95:

$$P\{u \text{ wordt gekozen}\} = \frac{N}{S}.$$

Intuïtief is dit duidelijk, Zo niet: gebruik een een argument van symmetrie, gebruik een combinatorisch argument, of bereken stapsgewijs de som van kansen. Evenzo:

$$P\{v \neq u \text{ wordt gekozen}\} = \frac{K}{S-1}.$$

Deze twee gebeurtenissen zijn onafhankelijk, dus als je wilt berekenen wat de kans is dat ze samen optreden, kunnen de kansen gewoon vermenigvuldigd worden:

$$P\{(u,v),\, v \neq u\} = \frac{N}{S} \cdot \frac{K}{S-1} = \frac{NK}{S(S-1)}.$$

Intuïtie: er zijn $S(S-1)$ mogelijke geordende paren van verschillende elementen.NK zulke paren, dus elk specifiek paar verschijnt met kans $NK/(S(S-1))$.

4c, p. 95: De trekking van een ongeordend paar komt overeen met (de twee elkaar wederzijds uitsluitende gebeurtenissen van) de trekking van twee geordende paren:

$$P\{u,v\} = P\{(u,v),\, v \neq u\} + P\{(v,u),\, u \neq v\} = \frac{2NK}{S(S-1)}.$$

4d, p. 95: Laat

A : het ge-ordende paar (u,v) wordt getrokken;
B : het ge-ordende paar (v,u) wordt getrokken.

Dan zoeken we $P\{u,v\} = P\{A \cup B\}$. Altijd:

$$P\{A \cup B\} = P\{A\} + P\{B\} - P\{A \cap B\}.$$

$P\{A\}$ en $P\{B\}$ weten we al.

1. De kans dat zowel u als v is de eerste selectie zitten is

$$\frac{N}{S} \cdot \frac{N-1}{S-1}.$$

2. De kans dat v gekozen wordt bij u, én u bij v, is (onafhankelijk):

$$\frac{K}{S-1} \cdot \frac{K}{S-1}.$$

3. Samengenomen:

$$P\{A \cap B\} = \frac{N(N-1)}{S(S-1)} \left(\frac{K}{S-1}\right)^2.$$

4. Alles samengenomen:

$$P\{A \cup B\} = \frac{2NK}{S(S-1)} - \frac{N(N-1)K^2}{S(S-1)^3}.$$

4e, p. 95: Laat E_{uv} de gebeurtenis zijn dat paar $\{u, v\}$ minstens één keer voorkomt in één experiment (van de M experimenten). Uit het vorige onderdeel:

$$p = P\{E_{uv}\} = \frac{2NK}{S(S-1)} - \frac{N(N-1)K^2}{S(S-1)^3}.$$

Nu:

- De kans dat $\{u, v\}$ níet verschijnt in één experiment is $1 - p$ (complementaire kans).
- De kans dat $\{u, v\}$ níet verschijnt in M (onafhankelijke) experimenten is dus $(1 - p)^M$ (product van onafhankelijk gebeurtenissen).

We willen alle paren tenminste één keer. Dit is een klassiek inclusie-exclusie geval. Laat A_{uv} de gebeurtenis zijn dat paar $\{u, v\}$ nooit is verschenen na M experimenten. We zoeken

$$\bigcap_{\{u,v\}} A_{uv}^c.$$

Dus (principe van inclusie en exclusie):

$$P\{\text{alle paren gezien}\} = \sum_{j=0}^{P} (-1)^j \binom{P}{j} q_j^M,$$

waarbij P het aantal mogelijke paren is, dus "kies 2 uit S". Exact q_j bepalen wordt snel ingewikkeld, omdat paren afhankelijk zijn, maar wanneer er veel elementen zijn, zijn botsingen zeldzaam, en kan men paren bijna zien als waren ze onafhankelijk. In dat geval

$$q_j \approx (1 - p)^j$$

en dus

$$P\{\text{alle paren gezien}\} \approx \sum_{j=0}^{P} (-1)^j \binom{P}{j} (1 - p)^{jM}.$$

Dit vereenvoudigt tot

$$P\{\text{alle paren gezien}\} \approx \left(1 - (1 - p)^M\right)^P = \left(1 - (1 - p)^M\right)^{\binom{S}{2}}.$$

Daarmee hebben we ons eindresultaat.

Weetje (we bewijzen dit verder niet): als X steeds groter wordt, dan

$$P\{\text{alle paren gezien}\} \rightsquigarrow e^{\lambda}$$

voor een zekere λ. Dus voor grote verzamelingen convergeert het aantal ongeziene paren naar een Poisson-variabele met parameter λ. Voor verdere details, zie [24, Sec. 2.4.1; Ch. 5] en [2, Sec. 1.2, pp. 7-12; Sec. 4.2, pp. 65-72].

1a, p. 96: Je dacht aan “Olifant”. Ja hoor:

Gewone olifanten: Deb S, Fong S, Tian Z (2015). “Elephant Search Algorithm for optimization problems.” In 2015 Tenth International Conference on Digital Information Management (ICDIM).

Clans van olifanten: Jafari M, Salajegheh E, Salajegheh J (2021). “Elephant clan optimization: A nature-inspired metaheuristic algorithm for the optimal design of structures.” Applied Soft Computing, 113, 107892. ISSN 1568-4946.

Olifantenkuddes: Wang G, Deb S, dos S. Coelho L (2015). “Elephant Herding Optimization.” In 2015 3rd International Symposium on Computational and Business Intelligence (ISCBI).

1b, p. 96: De parodie waar naar wordt verwezen is het ongepubliceerde manuscript getiteld “A Spectral Approach to Ghost Detection,” geschreven door Daniel Maturana, “Distinguished Lecturer in Parapsychology” (!) en Vology, David Fouhey, “Senior Ufologist and Ghost Hunter” [8]. In werkelijkheid is het paper geschreven door D. Fouhey and D. Maturana van het Robotics Institute aan de Carnegie Mellon University.

De parodie heeft de vorm van een wetenschappelijke publicatie, het lijkt nog het meest op een conference paper. Het product van de parodie, de paper, beargumenteert dat een groot aantal optimalisatie-algoritmen in machine learning geïnspireerd zijn door natuurlijke verschijnselen, maar dat tot nu toe echter nog geen onderzoek gedaan is naar algoritmen die zijn geïnspireerd op *boven*natuurlijke verschijnselen. In het artikel geven de auteurs een overzicht van hun zogenaamd “baanbrekende” onderzoek in deze richting, met algoritmen die zijn geïnspireerd op onder andere spoken, astrale projecties en aliens. Ze hopen onderzoekers te overtuigen van de waarde van het zich niet laten beperken van onderzoek door de realiteit.

De boodschap van de parodie zélf wordt niet echt expliciet gemaakt, maar is hoogstwaarschijnlijk dat overbodige, op de natuur gebaseerde, algoritmen belachelijk zijn, en dat mensen (wetenschappers?) eens moet ophouden het publiceren daarvan.

2a, p. 97: Dit is een interessante vraag, want de expressie “mitigating metaphors” laat zich zo één-twee-drie niet naar het Nederlands vertalen. Letterlijk betekent het: “verzachtende metaforen”. In het Nederlands zou dit waarschijnlijk worden “verhullende metaforen”, want de metaforen verhullen met prietpraat de eigenlijke algoritmen. Als we door de prietpraat heen lezen, blijkt nagenoeg altijd dat het onderliggende algoritme al veel eerder (en duidelijker!) in een ander artikel werd uitgelegd.

2b, p. 97: ABC, en dat staat voor Artificial Bee Colony Algorithm (ABC), gepubliceerd door Karaboga in 2005 [15]. In [22] wordt melding gemaakt van ongeveer 4500 citaties. Op 18 december 2024 bedroeg het aantal citaties 9205. Op 11 september 2025 bedroeg het aantal citaties 9839.

2c, p. 97: Artificial Bee Colony Algorithm (ABC), Bacterial Foraging Optimization (BFO), Bees Algorithm (BeA), Cuckoo Search (CS), en Shuffled Frog Leaping Algorithm (SFLA). (Is gewoon uit de tekst te halen.)

2d, p. 97: Grey Wolf Optimizer (GWO), Moth-Flame Optimization (MFO), en Whale Optimization Algorithm (WOA).

2e, p. 97: Artificial Bee Colony Algorithm (ABC), Bees Algorithm (BeA), Bat Algorithm (BA), Cuckoo Optimization Algorithm (COA), Cat Swarm Optimization (CSO), Charged System Search (CSS), Firefly Algorithm (FA), Fruit Fly Optimization Algorithm (FOA), Flower Pollination Algorithm (FPA), Gravitational Search Algorithm (GSA), Group Search Optimizer (GSO), Grey Wolf Optimizer (GWO), Glowworm Swarm Optimization (GwSO), Krill Herd (KH), Moth-Flame Optimization (MFO), Teacher-Learning Based Optimization (TLBO), Water Cycle Algorithm (WCA), en Whale Optimization Algorithm (WOA).

2f, p. 97: Er zijn de afgelopen twintig jaar talloze artikelen geschreven waarin nieuwe, door de natuur geïnspireerde, algoritmen worden beschreven. Helaas is het gebruikelijk geworden dat deze artikelen algoritmen beschrijven met behulp van niet-standaardterminologie. Deze terminologie is dan afgeleid van het inspiratiedomein, wat resulteert in artikelen die vaak erg moeilijk te lezen en te begrijpen zijn.

Lones doet in zijn artikel een poging om de meest geciteerde van deze algoritmen te beschrijven met behulp van standaard terminologie. De auteur hoopt daarmee dat de resulterende beschrijvingen het voor lezers gemakkelijker maakt om snel te begrijpen hoe deze algoritmen werken, zonder de oorspronkelijke artikelen te hoeven lezen. Lones' artikel doet ook een poging om de overeenkomsten tussen algoritmen te analyseren. De bestaande literatuur heeft bepaalde gevallen aangehaald waarin er een sterke gelijkenis is tussen verschillende door de natuur geïnspireerde optimalisatiealgoritmen, maar ondoorzichtige terminologie maakt het moeilijk om deze overeenkomsten in het algemeen te herkennen. De gestandaardiseerde beschrijvingen in dit artikel pogen dit proces te vergemakkelijken, en dit is aangetoond door elk van de algoritmen te relateren aan bestaande meta-heuristische concepten.

De resulterende analyse suggereert dat weinig van de algoritmen die in de afgelopen 20 jaar zijn geïntroduceerd fundamenteel nieuwe concepten introduceren; in plaats daarvan assembleren ze meestal bestaande concepten op nieuwe manieren. Omdat veel van de algoritmen gebaseerd zijn op zwermen, werd er nader gekeken naar hun overeenkomsten met deeltjeszwermoptimalisatie en de varianten daarvan. Dit onthulde weinig punten van absolute nieuwigheid, wat suggereert dat de twee gemeenschappen grotendeels dezelfde sporen hebben gevolgd. Dit artikel benadrukt ook de noodzaak om de verschillende richtingen samen te brengen, met als doel redundantie te verminderen, onderzoeksresultaten toegankelijker te maken en nieuwe benaderingen te ontwikkelen die het diverse werk dat wordt gedaan integreren.

3, p. 97: De afbeelding is afkomstig uit het artikel "Grey wolf optimizer", Advances in engineering software 69 (2014): 46-61. Auteurs: Mirjalili, Seyedali, Seyed Mohammad Mirjalili, en Andrew Lewis. Op 19 december 2024 maakte Google Scholar melding van een ontluisterend aantal van 17,348 citaties. Dit is zorgelijk want elders werd al aangetoond dat het grijze wolf optimalisatiealgoritme een eenvoudig speciaal geval is van deeltjeszwerm-optimalisatie (Eng.: particle swarm optimization, PSO) [6].

4a, p. 97: Het artikel bezit de vorm van een wetenschappelijke brief: officieel heet dit "a letter to the editor". "The editor" is dan de editor van het wetenschappelijk tijdschrift waar de brief, of het pamflet, naar wordt toegestuurd. Het maakt niet uit wie of wat dan die editor in feite is, de "a letter to the editor" is een stijlfiguur.

4b, p. 97: Wel, dat is makkelijk, want we hoeven slechts te letten op de cursieve zinsdelen waarmee alinea's geopend worden. Achtereenvolgens: (1) Nutteloze metaforen. (2) Gebrek aan

nieuwigheid. (3) Slechte experimentele validatie en vergelijking. (4) Metafoor-gebaseerde optimalisatie in toepassingsgerichte tijdschriften: het voordeel van het indienen van een manuscript bij wat alleen als een off-topic tijdschrift kan worden beschouwd, is dat redacteuren en reviewers mogelijk onvoldoende kennis bezitten ten aanzien van het specifieke veld van optimalisatie, om zo de werkelijke verdiensten van het artikel te evalueren. De auteurs merken op dat deze aanpak gelukkig meestal niet werkt.

Zelf zou ik daar nog aan willen toevoegen: verwarrende notatie, ondoorgrondelijke uitleg, het weglaten van details in beschrijvingen van algoritmen zodat je maar moet raden wat er wordt bedoeld of welke parameters moeten worden ingesteld, en schier om zaken heen draaien. Deze bezwaren worden elders ook genoemd door anderen, waaronder Michael A. Lones, de auto van het 1Mitigating Metaphors" artikel [22], zie Vraag 2.

4c, p. 97: Deze verklaring luidt, vrij vertaald: *"Dit tijdschrift zal geen artikelen publiceren die "nieuwe" metafoor-gebaseerde meta-heuristieken voorstellen, tenzij de auteurs (i) hun methode presenteren met behulp van de normale, standaard optimalisatie-terminologie; (ii) aantonen dat de nieuwe methode bruikbare en nieuwe concepten introduceert; (iii) het gebruik van de metafoor motiveren op een wetenschappelijke basis; en (iv) een eerlijke vergelijking presenteren met andere state-of-the-art algoritmen met behulp van state-of-the-art praktijken voor benchmarking."*

1a, p. 99: Het bereik van de z-as is $[0, 1]$. Er wordt immers gesproken over een kans op het accepteren van een slechtere oplossing.

1b, p. 99: Als $T = 0$ wordt er door 0 gedeeld en is er geen uitkomst.

1c, p. 99: Het berekenen van de limiet.

$$\lim_{T \downarrow 0} P\{T, \Delta U)\} = 0.$$

Dit kan formeel worden bewezen met een ϵ/δ-argument, of informeel worden ingezien.

Informeel: als in (Eq. 7 op pagina 99) de waarde van T naar 0 zakt bij vaste ΔU, dan gaat de waarde van de exponent van de e-macht naar $-\infty$. Dus dan convergeert de e-macht zelf naar 0.

Met een klassiek ϵ/δ-argument: laat $\epsilon > 0$ gegeven zijn. We moeten nu een waarde van T vinden, zó dat $e^{-\frac{\Delta U}{KT}} \leq \epsilon$. T isoleren uit deze ongelijkheid geeft

$$T \leq -\frac{K}{\Delta U} \ln(\epsilon).$$

Dus δ kan gelijk genomen worden aan

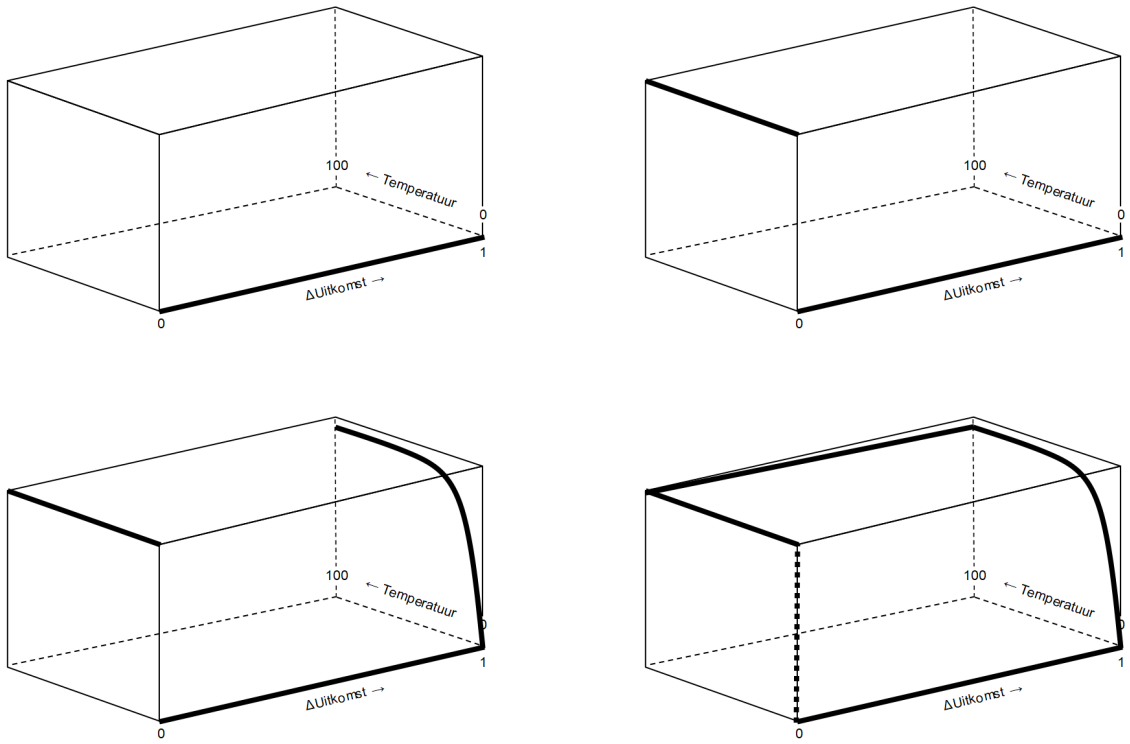
$$-\frac{K}{\Delta U} \ln(\epsilon)$$

of kleiner. In dit geval ligt het niet voor de hand een kleinere expressie te distilleren uit de rechterkant, dus laten we het maar zo.

1d, p. 100: Zie het eerste figuur in het antwoord op Vraag 1f.

1e, p. 100: Zie de tweede figuur in het antwoord op Vraag 1f.

1f, p. 100: We moet dus de functie $T \to e^{-1/T}$ tekenen, voor $T \in [0, 100]$, in het vlak $\Delta U = 1$. Dit is de vierde tekening in de figuur hieronder.



Antwoorden op dit onderdeel, en de vorige twee onderdelen.

1g, p. 100: De waarde van $P\{T, \Delta U)\}$ is dan ongedefinieerd, want het maakt uit, uit welke richting het punt $(T, \Delta U) = (0, 0)$ benaderd wordt. Wiskundig gesproken is er sprake van een discontinuïteit (informeel: "sprong") bij het punt $(0, 0)$. Deze sprong wordt in het vierde figuur van het antwoord op Vraag 1f afgebeeld als een stippellijn.

1h, p. 100: Bij $T = 0$ is deze kans ongedefinieerd, die is dus niet te gebruiken. Er is afgesproken dat bij $T = 0$ de kans op acceptatie van een slechtere oplossing gelijk is aan nul.

1i, p. 100: Ten eerste is er een discontinuïteit bij $(0, 0)$, en daar kan een lineaire functie niet mee omgaan. Ten tweede zijn zowel de temperatuur als het verschil in uitkomst onbegrensd. Een exponentiële functie kan, ook voor grotere waarden van T en ΔU, de resulterende kans voor het accepteren van een slechtere oplossing mooi in het interval $[0, 1]$ houden.

2a, p. 100: De verzamelingen A en C zijn overdekkingen, de andere niet.

2b, p. 100: Zowel A en C zijn minimale overdekkingen. (Misschien zijn er meer minimale overdekkingen, maar dat werd niet gevraagd.)

2c, p. 101: De verzameling C is een minimum overdekking, immers $|C| = 2$, de rest niet. (Misschien zijn er meer minimum overdekkingen, maar dat werd niet gevraagd.)

2d, p. 101: Niet alle minimale overdekkingen zijn minimum overdekkingen. Bijvoorbeeld overdekking A is geen minimum overdekking, omdat $|A| = 3 > 2 = |C|$. Wel zijn alle minimum overdekkingen minimaal: ieder knoop is nodig, omdat met het wegnemen van één knoop (maakt niet uit welke) de resulterende partiële overdekking strict minder elementen bevat dan een minimum overdekking, en daarmee dus geen overdekking kan zijn.

2e, p. 101: De verzameling van partiële overdekkingen van G is gelijk aan de machtsverzameling van V, als $V = \{a, b, c, d, e, f\}$. Dus het antwoord op deze vraag is 2^V of, anders genoteerd, $\mathcal{P}(V)$.

2f, p. 101: K is onze partiële overdekking. Noem een punt *niet overdekt*, of *geïsoleerd* als deze geen element van K is, en ook geen buur is van een element uit K.

Een zogenaamd gretig algoritme om een partiële overdekking K om te zetten in een echte overdekking is de volgende:

> Spoor herhaaldelijk geïsoleerde punten op, en zet die in K. Stop zodra er geen geïsoleerde punten meer te vinden zijn.

Opmerkingen:

1. Het gretige algoritme kiest elke keer willekeurig een geïsoleerd punt. Dus de uitkomst van het gretige algoritme is, na een flink aantal van dergelijke keuzes, met een grote kans elke keer weer anders.
2. De uitkomst van het gretige algoritme is altijd een minimale overdekking (immers, het algoritme stopt als er geen geïsoleerde punten meer zijn), maar niet noodzakelijk een minimum overdekking. (Ga na!)
3. De uitkomstenruimte van het gretige algoritme bestaat precies uit alle minimale overdekkingen. I.e., als een overdekking minimaal is, dan is het mogelijk dat het gretige algoritme die vindt. Omgekeerd, als het gretige algoritme een overdekking produceert, dan is deze altijd minimaal.
4. Uiteraard levert het gretige algoritme niet noodzakelijk een minimale overdekking op. Als dat zo was, dan zou het gretige algoritme een oplossing van het knopenbedekkingsprobleem zijn. (Ga na!)
5. Het gretige algoritme kán een minimum oplossing genereren. Als dat zo is, dan kunnen wij dat, als gebruiker van het algoritme, niet herkennen. Er zou een andere overdekking kunnen bestaan met nóg minder elementen.
6. Allicht zijn er andere goede algoritmen om een partiële overdekking uit te breiden naar een echte overdekking.

2g, p. 101: Er zijn verschillende mogelijkheden, waaronder de volgende:

1. Uitputtend: trek je niks aan van een bestaande oplossing, en begin elke keer met een lege partiële overdekking, om die met het gretige algoritme aan te vullen tot een echte overdekking.
2. Met vaste uithaalgrootte (Eng.: sample size) N: haal N punten uit de overdekking K, en vul de ontstane partiële overdekking aan met het gretige algoritme tot een echte overdekking.

3. Met variabele uithaalgrootte: haal N of minder punten uit de overdekking K, en vul de ontstane partiële overdekking aan met het gretige algoritme.

2h, p. 101: Laten we een functie die vanuit een bestaande oplossings-suggestie (welke altijd een minimale overdekking is) een nieuwe oplossings-suggestie genereert, "hernieuw" noemen.

Algoritme:

> Zet de temperatuur, T gelijk aan 100, en het temperatuurverloop, λ, gelijk aan 0.01. Zet de constante $K = 1$. Dit zijn waarden die proefondervindelijk worden bepaald door de programmeur.
>
> Begin met een lege partiële overdekking. Zet de huidige overdekking, K, gelijk aan de verzameling van alle knopen V, en de grootte van de kleinst gevonden overdekking, N, op plus oneindig. Herhaal nu het volgende
>
> 1. Hernieuw de bestaande partiële overdekking. Het resultaat is een minimale overdekking M.
> 2. Als $|M| < N$, zet K gelijk aan M, en N gelijk aan $|M|$.
>
>   Onthoud verder dat K een tot nu toe een kleinst gevonden overdekking is.
>
>   Als $|M| \geq N$, zet K in weerwil van het feit dat M meer dan N elementen bezit dan toch gelijk aan M, met kans
>
> $$e^{-\frac{\Delta U}{KT}},$$
>
>   waarbij
>
> $$\Delta U = |M| - N.$$
>
> 3. Stop als je tevreden bent met de gevonden oplossing, of als je niet meer verder wilt (er is bv. een antwoord nodig) of kunt (de tijd en/of geheugenruimte is bv. op).
> 4. Zet, om de temperatuur geleidelijk te laten dalen, T gelijk aan $(1 - \lambda)T$.
> 5. Ga naar Stap 1.

Opmerkingen.

- Bovenstaande stappen kunnen deels worden omgewisseld, zonder het algoritme te schaden. De stop-test, bijvoorbeeld, kan eigenlijk op elke regel worden uitgevoerd.
- K is niet noodzakelijk de kleinst gevonden overdekking. Als later een overdekking met minder elementen gevonden wordt , dan is het belangrijk deze overdekking te onthouden. Immers, door het simulated annealing effect kan K in het verdere verloop van het algoritme weer méér dan dan het minimale elementen, N, gaan bevatten.
- Er zijn meerdere goede antwoorden. Als je niet zeker bent van je zaak, laat je antwoord dan controleren door iemand anders.

1, p. 102: Eén mogelijke oplossing: we maken van de sectoren van het roulettewiel een recht stuk lijn en trekken een random getal dat op die lijn valt. Meer in detail:

- Tel van alle n individuen de fitness op. Dat wordt F.
- Bereken van elk individu de cumulatieve fitness: $c_i = \sum_{j=1}^{n} f_i$. (Hiermee is $f_n = F$.)
- Doe nu k keer het volgende:
  - Genereer een random getal r in $[0, F)$.
  - Bied a_i aan ter reproductie, kruising of mutatie als en slechts als $r \in [c_{i-1}, c_i)$. (Dit gaat goed als we $c_0 = 0$ zetten.)

Merk op:

1. Het is mogelijk meerdere keren hetzelfde individu te trekken. Voor kruising lijkt dat misschien wat raar, maar voor reproductie en mutatie is dat niet. En zelf-kruising is altijd mogelijk.
2. Normeren hoeft niet.
3. De check $r \in [c_{i-1}, c_i)$ is relatief inefficiënt: het algoritme moet immers uitvinden tot welk c-interval r behoort. Hoe zou je dit kunnen programmeren?

Er zijn ook andere oplossingen mogelijk, bijvoorbeeld door een schaduwpopulatie aan te leggen met kopieën van individuen, zodanig dat het aantal kopieën van een individu recht evenredig is met de fitness van dat individu, en dan a-select te trekken uit de schaduwpopulatie.

2a, p. 102: 2^N.

2b, p. 102: In alle gevallen kan er maar één (soort) kind worden geproduceerd. Dat kind moet gelijk aan beide ouders zijn.

2(c)i, p. 102: Laten we een locatie tussen twee bits, of voor de eerste bit, of na de laatste bit, een *snijpunt* noemen. Een snijpunt is dus een punt waarop je een string in tweeën kunt knippen. Een bitstring ter lengte N heeft dus $N + 1$ snijpunten. Laten we de twee uitwendige snijpunten *oneigenlijk* noemen, want als je daar knipt gebeurd er eigenlijk niets. De reden om oneigenlijke snijpunten toch mee te nemen in de beschouwing is dat ze handig zijn voor randgevallen.

Voor $N \leq 2$ zijn er $N - 1$ inwendige punten waarop je kunt snijden (als je helemaal vooraan of achteraan snijd, levert dat geen kruising op). Elk snijpunt levert twee verschillende kinderen op. Dat levert in totaal $2(N - 1)$ verschillende kinderen voor $N \geq 2$, en geen kinderen voor $N = 0$ of $N = 1$.

Als je wél toelaat op de buitenkanten te snijden, dan levert dat $2(N - 1) + 2 = 2N$ verschillende kinderen op. [Niet $2(N + 1)$, want de twee uitwendige snijpunten leveren beiden dezelfde kinderen. Je moet daar dus weer 2 van af trekken.] Voor de flauwerikken die lege bitstrings ook interessant vinden moeten we er volledigheidshalve nog aan toevoegen dat $N \geq 1$.

2(c)ii, p. 103: Voor $N = 8$ en $k = 3$ zie je dat er zes stukjes ter lengte 3 uit kunnen worden gehaald. Generaliseren: $(8 - 3) + 1$. Verder generaliseren: $(N - k) + 1$ stukjes. Elk stukje levert twee verschillende kinderen die, onder de aanname dat alle kinderen ook onderling weer verschillend zijn, twee keer zoveel verschillende kinderen opleveren, dus $2(N - k) + 2$ kinderen.

We gaan nu onderzoeken of alle kinderen onderling verschillend zijn. Het antwoord is: "nee". Bv. voor $N = 6$ en $k = 3$ levert snijden bij 0 en 3 dezelfde kinderen op als snijden bij 3 en 6. (Ga na door zelf te tekenen.) Om te ontdekken of er mogelijk meer strings dubbel geproduceerd worden, bekijken we twee willekeurige twee-punts kruisingen: één beginnend vanaf index l, waarbij $0 \leq l \leq N$, en één beginnend vanaf index m, waarbij $0 \leq m \leq N$. Dit levert vier kinderen op:

$$\begin{array}{ll} 0^l1^k0^{N-l-k} & 0^m1^k0^{N-m-k} \\ 1^l0^k1^{N-l-k} & 1^m0^k1^{N-m-k} \end{array}$$

De strings $0^l1^k0^{N-l-k}$ en $0^m1^k0^{N-m-k}$ zijn alleen maar identiek als $l = m$. Dit levert dus geen doublures op. De strings $0^l1^k0^{N-l-k}$ en $1^l0^k1^{N-l-k}$ kunnen voor geen enkele l gelijk zijn, want dan zou $k = l = n = 0$. Evenzo voor m. We gaan nu kijken of de string $0^l1^k0^{N-l-k}$ gelijk kan zijn aan $1^m0^k1^{N-m-k}$ voor bepaalde l en m. Dat zou op twee manieren kunnen.

1. $l = k$, $k = N - m - k$, $N - l - k = 0$, en $m = 0$. Dat levert op: $k = N/2$ en $l = N/2$.
2. $m = k$, $k = N - l - k$, $N - m - k = 0$, en $l = 0$. Dat levert op: $k = N/2$ en $m = N/2$.

Dus als N even is dan levert snijden bij 0 en $N/2$ hetzelfde op als snijden bij $N/2$ en N. (Wat we al vonden voor $N = 6$ en $k = 3$.) De andere twee strings kruislings aan elkaar gelijk stellen levert hetzelfde op. Het uiteindelijk antwoord is hiermee:

$$\begin{cases} 2(N-k) & \text{als } N \text{ even en } k = N/2, \\ 2(N-k)+2 & \text{anders.} \end{cases}$$

2(c)iii, p. 103: We kunnen twee snijpunten kiezen uit $N + 1$ plekken, want de buitenkanten tellen nu ook mee. (De buitenkanten tellen mee, want deze snijpunten leveren echt andere genotypes op.) Twee snijpunten kiezen uit $N + 1$ locaties levert $(N + 1 \;\; 2)$ "$N + 1$ boven 2" mogelijkheden op. Deze mogelijkheden produceren lang niet allemaal verschillende kinderen. Voor $N = 5$, bijvoorbeeld, levert snijden bij 0 en 2 hetzelfde op als snijden bij 2 en 5. Alleen de volgorde van de kinderen is anders. (Ga na door zelf te tekenen.) In het algemeen levert snijden bij 0 en i dezelfde kinderen op als snijden bij i en N, zolang $1 \leq i \leq N - 1$. Dat gebeurd $N - 1$ keer. Dus $2(N - 1)$ snijmogelijkheden $[\{(0, 1), (1, N), \ldots, (0, N - 1), (N - 1, N)\}]$ leveren slechts hetzelfde aantal kinderen op (en niet dubbel zo veel). De overige $(N + 1 \;\; 2) - 2(N - 1)$ snijmogelijkheden leveren wel het dubbele aantal kinderen op, i.e., $2(N + 1 \;\; 2) - 4(N - 1)$. Bij elkaar optellen levert $2(N + 1 \;\; 2) - 2(N - 1) = N^2 - N + 2$ kinderen.

2(c)iv, p. 103: Het bitmasker staat vast, dus twee kinderen.

2(d)i, p. 103: Voor de identieke plekken kunnen we sterretjes of hekjes zetten, maar in het kruisingsproces doen die er eigenlijk niet meer toe. We kunnen ze dus eigenlijk gewoon weglaten. De bitstring krijgt dan een effectieve lengte van i. Verbieden we uitwendige snijpunten dan is het antwoord: "geen" als $i < 2$ en $2(i - 1)$ anders; laten we uitwendige snijpunten toe, dan is het antwoord: geen als $i < 1$ en $2(i - 1) + 2 = 2i$ anders.

2(d)ii, p. 103: Effectieve lengte i, dus $2(N - i - k)$ mogelijke kinderen als $N - i$ even en $k = (N - i)/2$, en $2(N - i - k) + 2$ anders. (Let ook hier weer op randgevallen zoals $i \leq 1$ en $k \leq 1$. Daar geldt deze formule niet en moeten we handmatig het aantal kinderen bepalen.)

3a, p. 103:

$$\Pr\{0***\} = \frac{6+20+4+18+9+12+14+6}{200} = 0.445.$$
$$\Pr\{00**\} = \frac{6+20+4+18}{200} = 0.24.$$
$$\Pr\{000*\} = \frac{6+20}{200} = 0.13.$$
$$\Pr\{***0\} = \frac{6+9+12+13+4+14+5+8}{200} = 0.355.$$
$$\Pr\{**00\} = \frac{6+9+12+13}{200} = 0.2.$$
$$\Pr\{*000\} = \frac{6+12}{200} = 0.09.$$

3b, p. 103: De gebeurtenis dat 0000 ontstaat door een kruising is de disjuncte vereniging van drie afzonderlijke gebeurtenissen. De eerste van die drie gebeurtenissen is bijvoorbeeld dat er tussen het eerste en tweede allel gesneden wordt (kans 1/3) samen met de gebeurtenissen dat schema $0***$ en $*000$ gekozen worden (in die volgorde). Dat is 1/3 maal $\Pr\{0***\}$ maal $\Pr\{*000\}$. Evenzo voor de andere twee gebeurtenissen. Dus:

$$\begin{aligned}\Pr\{0000\} &= \Pr\{0|000\} + \Pr\{00|00\} + \Pr\{000|0\}\\ &= \Pr\{0***\}\Pr\{*000\}\frac{1}{3} + \Pr\{00**\}\Pr\{**00\}\frac{1}{3} + \Pr\{000*\}\Pr\{***0\}\frac{1}{3}\\ &= 0.445 \times 0.09\frac{1}{3} + 0.24 \times 0.2\frac{1}{3} + 0.13 \times 0.355\frac{1}{3}\\ &= 0.04473333.\end{aligned}$$

3c, p. 103: Laat X het aantal exemplaren van het genotype 0000 in de volgende generatie zijn. Dus X is het aantal successen bij 100 keer trekken (zonder teruglegging). We willen, bij 100 keer trekken, de kans berekenen op 5, 6, 7, 8, 9, of 10 successen. Waar die successen zich in die reeks van 100 bevinden, maakt niet uit. Bijvoorbeeld

..!..... ...!.!.... ....!!... ....!!.!....

waarbij een ! een succes representeert. We concluderen dat X binomiaal verdeeld is, met parameters $n = 100$ en kans op succes $p = 0.04473333$. Immers,

$$\Pr\{5 \le X \le 10\} = \Pr\{X \le 10\} - \Pr\{X \le 4\} = 0.99481 - 0.53512 = 0.45969.$$

4a, p. 103:

$$p_s(x) = \frac{f(x)}{\sum X} = \frac{f(x)}{|X|f(X)} = \frac{1}{|X|}\frac{f(x)}{f(X)}.$$

4b, p. 103:

$$\sum_{x\in X} p_s(x) = \sum_{x\in X} \frac{1}{|X|}\frac{f(x)}{f(X)} = \frac{1}{f(X)}\cdot\frac{1}{|X|}\sum_{x\in X} f(x) = \frac{1}{f(X)}\cdot f(X) = 1.$$

4c, p. 103: Als $f(x) = a\cdot f(y)$, dan

$$p_s(x) = \frac{1}{|X|}\frac{f(x)}{f(X)} = \frac{1}{|X|}\frac{af(y)}{f(X)} = a\cdot\frac{1}{|X|}\frac{f(y)}{f(X)} = a\cdot p_s(y).$$

De identiteit geldt ook voor $a = 0$, immers de herschrijving hierboven blijft gelden. Er wordt niet gedeeld door $a = 0$ o.i.d.

4d, p. 104:

$$p_s(H) = \frac{\sum H}{\sum X} = \frac{|H|f(H)}{|X|f(X)} = \frac{|H|}{|X|}\frac{f(H)}{f(X)}.$$

4e, p. 104: Met het vorige onderdeel geldt

$$p_s(H) = \frac{|H|}{|X|}\frac{f(H)}{f(X)}. \tag{17}$$

De kansen ϵ_m en ϵ_c liggen in $[0, 1]$, dus het product $(1 - \epsilon_m)(1 - \epsilon_c)$ ligt ook in $[0, 1]$.

Voor willekeurige getallen $x, y \geq 0$ geldt in het algemeen:

$$y = x \quad \text{en} \quad p \in [0, 1] \quad \Rightarrow \quad y \geq px.$$

Immers, p verkleint x. Met $p = (1 - \epsilon_m)(1 - \epsilon_c)$ toegepast op (17) geeft dit meteen het gevraagde.

Hiermee kan meteen worden gezien dat Holland's schemastelling weinig voorstelt. De betekenis van de kansen ϵ_m en ϵ_c blijken helemaal niet belangrijk! Het enige dat wordt gebruikt is $(1 - \epsilon_m)(1 - \epsilon_c) \in [0, 1]$. Wat Holland's schemastelling eigenlijk doet is een op zich triviale gelijkheid (17) verzwakken tot een minder informatieve ongelijkheid ...

5a, p. 104: Als de toernooi-groep uit meer dan één element bestaat, dan verliest g_1 altijd! Dus:

$$P\{\text{element } g_1 \text{ komt als winnaar uit de bus in één toernooi}\} = \begin{cases} 1/n & \text{als } k \leq 1, \\ 0 & \text{anders.} \end{cases}$$

5b, p. 104: Reken met complementaire kansen:

$$P\{g_1 \text{ komt NIET als winnaar uit de bus in één toernooi}\} = \begin{cases} 1 - 1/n & \text{als } k \leq 1, \\ 1 & \text{anders.} \end{cases}$$

Dus

$$P\{g_1 \text{ komt niet de mating pool terecht}\} = \begin{cases} (1 - 1/n)^N & \text{als } k \leq 1, \\ 1 & \text{anders.} \end{cases}$$

5c, p. 104: Er zijn $\binom{n}{k}$ mogelijke toernooien. Als $i < k$ dan wordt element g_i in geen enkel toernooi gekozen: de toernooi-groep bevat dan altijd wel een element met een hogere fitness. Als $i \geq k$ dan wordt element g_i in

$$1\cdot\binom{i-1}{k-1}$$

mogelijke toernooien gekozen. Immers, $1\times$ kiezen voor g_i zelf, en dan kunnen de resterende $k - 1$ elementen uit $i - 1$ genotypen met een lagere fitness worden gekozen. Dus

$$P\{g_i \text{ komt na één toernooi als winnaar uit de bus}\} = \begin{cases} \frac{\binom{i-1}{k-1}}{\binom{n}{k}} & \text{als } i \geq k, \\ 0 & \text{anders.} \end{cases}$$

5d, p. 104: Laat $i \geq k$ en

$$P\{g_i \text{ komt na één toernooi als winnaar uit de bus}\} = p_i.$$

De kans dat g_i na n toernooien precies m keer in de mating pool terecht is gekomen is dan binomiaal verdeeld:

$$\begin{aligned} P\{g_i \text{ precies } m \text{ keer in mating pool}\} &= \binom{n}{m} p_i^m (1-p_i)^{n-m} \\ &= \binom{n}{m} \left(\frac{\binom{i-1}{k-1}}{\binom{n}{k}} \right)^m \left(1 - \frac{\binom{i-1}{k-1}}{\binom{n}{k}} \right)^{n-m}. \end{aligned}$$

Als $i < k$ dan uiteraard $P\{g_i \text{ precies } m \text{ keer in mating pool}\} = 0$: de mating pool is dan groter dan het genotype waar het om gaat.

5e, p. 104: De verwachting van een binomiale verdeling met parameters n en p_i is gelijk aan np_i. In de mating pool kunnen dus

$$n \frac{\binom{i-1}{k-1}}{\binom{n}{k}}$$

exemplaren van g_i worden verwacht als $i \geq k$, en nul exemplaren anders. De verwachte gemiddelde fitness van de mating pool is dus gelijk aan

$$E[\bar{f}] = E[\frac{1}{n} \sum_{i=0}^{n} f_i] = \frac{1}{n} \sum_{i=0}^{n} E[f_i] = \frac{1}{n} \left(\sum_{i=1}^{k-1} 0 + \sum_{i=k}^{n} n \frac{\binom{i-1}{k-1}}{\binom{n}{k}} f_i \right) = \sum_{i=k}^{n} \frac{\binom{i-1}{k-1}}{\binom{n}{k}} f_i.$$

6a, p. 105: Alle antwoorden hebben betrekking op generatie t.

1. De totale fitness, T, is fN.
2. De kans dat, bij één selectiemoment, dus bij één keer spinnen van het gewogen roulettewiel, een specifieke winnaar, bv. winnaar nr. 1 wordt gekozen is gelijk aan
$$p = \frac{F}{fN}.$$
3. De kans dat, bij één selectiemoment een willekeurige winnaar wordt gekozen is gelijk aan wp.
4. De kans dat, bij één selectiemoment, geen winnaar wordt gekozen is gelijk aan $1 - wp$.
5. De kans dat alle N selectiemomenten geen winnaar wordt gekozen is gelijk aan $(1 - wp)^N$.
6. De kans dat, na N selectiemomenten, tenminste één winnaar werd gekozen is gelijk aan $1 - (1 - wp)^N$. Uitgedrukt in f, F, w en N is dat
$$1 - (1 - w\frac{F}{fN})^N.$$

6b, p. 105: Daar

$$\lim_{n\to\infty}(1+x/n)^n = e^x,$$

geldt

$$\lim_{N\to\infty} 1-(1-w\frac{F}{fN})^N = 1-e^{-\frac{wF}{f}}.$$

Daar $F \geq f$ en $w \geq 1$ geldt dat

$$\frac{wF}{f} \geq 1.$$

Omdat de functie $x \mapsto 1-e^{-x}$ stijgt geldt nu

$$1-e^{-\frac{wF}{f}} \geq 1-e^{-1} \approx 0.63.$$

6c, p. 105: *a*) Te snelle convergentie naar locale optima.

b) Weinig selectiedruk als fitness-waarden dicht bij elkaar liggen.

c) Verandering van selectiedruk bij eenvoudige uniforme veranderingen van de fitness functie.

d) Relatief moeilijk implementeerbaar.

e) Intensieve berekening bij grote populaties.

7a, p. 105: (Antwoord Marco Wiering:) Let erop dat het oplossen van een doolhof betekent dat de agent weet hoe hij naar het doel moet vanuit elke willekeurige startpositie (deze was immers niet gegeven) de agent weet wel zijn positie, dat om problemen met partial observable environments te voorkomen.

Representatie: In elk veld moet de agent een actie selecteren. Dit wordt dus in elk geval gerepresenteerd in een string van lengte N waarin elke allele 4 waarden kan hebben. Om alvast rekening te houden met het evolutie proces, willen we dat individuen die veel lijken op de optimale oplossing een hoge fitness verkrijgen. Dus als een agent 1 stapje voor de doeltoestand een fout maakt, moet hij toch een hoge fitness hebben. Dit kunnen we doen door een extra allele toe te voegen welke de kans op het selecteren van een random actie (in plaats van de actie gerepresenteerd in de string) bepaalt. Hiermee bereiken we dus altijd het doel en hebben we geen problemen om afstanden etc. tot het doel te bepalen. Dus de zoekruimte is in principe oneindig als we continue parameters toelaten voor de random actie kans.

7b, p. 105: (MW:) Simpel: voor de velden muteer een actie door deze door een andere actie te vervangen. Voor de kans kunnen we hier Gaussiaan verdeelde ruis toevoegen met de eis dat deze kans nooit kleiner dan 0 wordt.

7c, p. 105: (MW:) Om een oplossing niet te veel te breken kunnen we 1-point crossover gebruiken. De kans op een random actie kunnen we middelen over de 2 ouders.

7d, p. 105: (MW:) Aangezien we met de kans altijd het doel bereiken, kunnen we het aantal stappen makkelijk gebruiken als fitness functie. We testen over alle mogelijke begintoestanden en berekenen het gemiddelde aantal stappen om vanuit een willekeurige begintoestand naar de doeltoestand te gaan.

7e, p. 105: (MW:) Het GA is niet heel erg efficient. De zoekruimte is erg groot, en de mutatie en crossover operatoren niet erg doelgericht. Verder gooien we veel individuen weg hetgeen betekent dat die extra stappen voor niks gemaakt zijn. Reinforcement leren is een efficiënter algoritme voor dit probleem, tenzij er partial observability is (de agent weet dan niet altijd waar hij is).

8a, p. 105: Er wordt een eerst een vaste toernooigrootte $k > 2$ bepaald. Groter dan 2, want er moet iets zijn te vergelijken. Als een volgende generatie van N elementen moet worden gemaakt, herhaal dan N keer het volgende. Kies k genotypen uit de huidige generatie, en plaats één van de winnaars in de volgende generatie.

8b, p. 105: Uit het dictaat van Marco Wiering:

a) Weinig nadeel van de absolute grootte van fitness-waarden (bv. verandering van selectiedruk bij triviale wijzigingen van de fitness-functie).

b) Het berekenen van absolute fitness-waarden is niet echt nodig. Het voldoende te weten hoe individuen zich relatief t.o.v. elkaar vergelijken.

c) Makkelijk te implementeren.

d) Sneller te berekenen.

8c, p. 105: De kans dat het meest fitte individu NIET voorkomt in een aselecte greep zonder teruglegging van k individuen uit een populatie van N individuen is

$$\frac{N-1}{N}\frac{N-2}{N-1}\cdots\frac{N-k}{N-k+1} = \frac{N-k}{N}.$$

Immers, de eerste greep uit N moet het meest fitte individu mijden, en de tweede greep uit $N-1$ moet het meest fitte individu mijden, de derde greep uit $N-2$ moet het meest fitte individu mijden, en de vierde greep uit $N-4$ moet het meest fitte individu mijden, ..., en de ke greep uit $N-k+1$ moet het meest fitte individu mijden.

Wat ook kan is het aantal gunstige uitkomsten delen door het aantal mogelijke uitkomsten:

$$\frac{\binom{N-1}{k}}{\binom{N}{k}} = \frac{\frac{(N-1)!}{(N-1-k)!k!}}{\frac{N!}{(N-k)!k!}} = (\text{wegstrepen})\frac{N-k}{N}.$$

De kans dat het meest fitte individu bij toernooi-selectie NIET in de mating pool terecht komt is dus N keer achter elkaar dit meest fitte individu bij trekking missen:

$$\left(\frac{N-k}{N}\right)^N.$$

De kans dat het meest fitte individu bij toernooi-selectie uiteindelijk WEL in de mating pool terecht komt is dus

$$1-\left(\frac{N-k}{N}\right)^N.$$

8d, p. 105:

$$\lim_{N\to\infty} 1-(\frac{N-k}{N})^N = 1-\lim_{N\to\infty}(\frac{N-k}{N})^N = 1-\lim_{N\to\infty}(1+\frac{-k}{N})^N = 1-e^{-k}.$$

9a, p. 106:

$$\frac{|H_t|}{|X_t|}\frac{f(H_t)}{f(X_t)}.$$

9b, p. 106: K^L.

9c, p. 106: $(K+1)^L$.

9d, p. 106: 2^L.

9e, p. 106: Orde: aantal niet-sterren. Definiërende lengte: afstand tussen eerste en laatste niet-ster. Bv. orde "1*10*1" is vier en definiërende lengte is vijf.

9f, p. 106: 2^m.

9g, p. 106: $p_m = o(H)/L$.

9h, p. 106: Als $|K| > 1$ dan $\epsilon_m \leq p_m$, omdat een mutatie in sommige gevallen hetzelfde allel weer kan terugzetten.

9i, p. 106: Als $|K| > 1$ dan is deze kans groter dan $1 - \epsilon_m$, dus groter dan $1 - o(H)/L$.

9j, p. 106: $p_c = \delta(H)/(L-1)$. Immers, altijd $\delta(H) \leq L - 1$.

9k, p. 106: Als $|K| > 1$ dan $\epsilon_c \leq p_c$, omdat een kruising in sommige gevallen identieke kinderen kan opleveren.

9l, p. 106: Als $|K| > 1$ dan is deze kans groter dan $1 - \epsilon_c$, dus groter dan $1 - \delta(H)/(L-1)$.

9m, p. 106:

$$p(H_{t+1}) \geq \frac{|H_t|}{|X_t|}\frac{f(H_t)}{f(X_t)}(1-\epsilon_c)(1-\epsilon_m)$$

waarbij de proceskansen voor selectie op kruising, mutatie en reproductie buiten beschouwing zijn gelaten.

10a, p. 107: $(1+1)$-ES: Op elk moment bestaat de populatie uit één individu. Elke generatie wordt een kind uit dat individu geproduceerd, door iets op de ouder te variëren. Als individuen bijvoorbeeld een lengte en een gewicht bezitten, dan zal het kind een iets andere lengte en een iets ander gewicht krijgen, door toeval bepaald. De variatie van dit toeval wordt typisch bepaald door een normale of uniforme verdeling. Welke verdeling wordt gebruikt hangt af van de voorkeur van de programmeur. Vervolgens wordt de performance van het kind en de ouder met elkaar vergeleken. Als het kind beter presteert, komt dát individu in de volgende generatie. In het andere geval blijft de ouder. De selectiepoel is dus: { 1 ouder, 1 kind } en de sterkste mag door.

Opmerkingen.

1. Dit algoritme komt dus eigenlijk neer op hill climbing: verbeter steeds één punt, totdat een stop-criterium bereikt wordt.

2. In sommige beschrijvingen van $(1 + 1)$-ES is het niet nodig dat het kind beter presteert dan de ouder. Het mag dan al de plaats van de ouder innemen als het **net zo goed** presteert als de ouder. Het zoekproces is dan dynamischer: door het voorkomend inwisselen van op zich even goede oplossingen wordt er meer de door de zoekruimte bewogen.

3. In principe is het aantal iteraties eindeloos. Mogelijke stop-criteria zijn: geen verbetering meer, de rekentijd is op, of er moet op enig moment een antwoord komen.

10b, p. 107: $(1 + \lambda)$-ES: dit is hetzelfde als $(1 + 1)$-ES, alleen worden nu λ kinderen uit één individu gegenereerd. Er is dus een betrekkelijk grotere kans dat de ouder wordt vervangen. De selectiepoel is daarmee: { 1 ouder, λ kinderen } en de sterkste mag door.

10c, p. 107: $(1, \lambda)$-ES: dit is hetzelfde als $(1 + \lambda)$-ES, alleen de selectiepoel is nu { λ kinderen }, en de sterkste mag door.

Dit is hetzelfde als $(1 + \lambda)$-ES, met het verschil dat het ouder-individu niet in de selectiepoel geplaatst wordt. Het kan echter toch in de volgende generatie terugkomen als een gemuteerd kind identiek is aan dat ouder-individu, dat kind een fitness bezit welke minstens zo hoog is als de fitness van de overige gegenereerde kinderen, en dat kind geselecteerd wordt voor de volgende populatie met $\mu = 1$.

10d, p. 107: Eén ouder weggooien zonder het te laten concurreren met het geproduceerde kind, dus onvoorwaardelijk het geproduceerde kind accepteren, resulteert slechts in een toevalswandeling door de zoekruimte. Alhoewel een toevalswandeling strict genomen ook een zoekproces genoemd mag worden, is het een *ontaard* zoekproces. Er wordt immers niet echt gezocht. De kennis die wordt opgedaan door rond te wandelen (exploratie) wordt nooit verder uitgebuit in het zoeken (exploitatie). Korter: 100% exploratie en 0% exploitatie.

Opmerking: in [11, Sec. 6.2, p. 101] komt de volgende passage voor: “An alternative scheme, denoted as $(1, 1)$-ES, (pronounce: one comma one ES) always replaces the parent by the offspring, thus forgetting the previous solutions by definition”. Ik (GV) snap niet helemaal waarom dit schema daar überhaupt genoemd wordt. De auteurs gaan er ook verder niet op in.

10e, p. 107: $(\mu + \lambda)$-ES: op elk moment bestaat de populatie uit μ individuen. Per generatie worden λ kinderen uit die μ ouders geproduceerd, bijvoorbeeld door fitness-proportionele selectie of toernooi-selectie. De μ best presterende kinderen vormen de volgende generatie. De selectiepoel is (dus) { μ ouders, λ kinderen }, waarvan de μ sterksten door mogen.

10f, p. 107: (μ, λ)-ES: op elk moment bestaat de populatie uit μ individuen. Per generatie worden λ kinderen uit die μ ouders geproduceerd, bijvoorbeeld door fitness-proportionele selectie of door toernooi-selectie. De μ best presterende kinderen vormen de volgende generatie. De selectiepoel is { λ kinderen } waarvan de μ sterksten door mogen.

Opmerking: voor dit algoritme moet gelden dat $\lambda \geq \mu$, anders zijn er niet genoeg kinderen om uit te selecteren.

10g, p. 107: Er geldt $\rho \leq \mu$. De selectiepoel bestaat uit { μ ouders + λ kinderen }.

10h, p. 107: $(\mu/\rho, \lambda)$-ES: hetzelfde als $(\mu/\rho + \lambda)$-ES, alleen doen de ouders niet meer mee in het uiteindelijke selectieproces. De selectiepoel bestaat derhalve uit $\{\ \lambda$ kinderen $\}$ waarvan de μ sterksten door mogen. Ook hier moet gelden dat $\lambda \geq \mu$.

11, p. 108: Bij $(\mu + \lambda)$-ES worden fitte genotypen uit oudere populaties meegesleept. Voor deze exemplaren kan de geregistreerde fitness inmiddels achterhaald zijn. Dit probleem zou kunnen worden ondervangen door voor deze exemplaren de fitness opnieuw uit te rekenen.

12a, p. 108: We nemen de beste 25 van 100, dus de verwachte gemiddelde fitness van individuen in de volgende generatie is gelijk aan

$$\begin{aligned} E\left[\frac{X_{(76)} + \cdots + X_{(100)}}{25}\right] &= \frac{E[X_{(76)}] + \cdots + E[X_{(100)}]}{25} \\ &= \frac{E[100U_{(76)}] + \cdots + E[100U_{(100)}]}{25} \\ &= \frac{100E[U_{(76)}] + \cdots + 100E[U_{(100)}]}{25} \\ &= 100\frac{\frac{76}{101} + \cdots + \frac{100}{101}}{25} \\ &= 87.1287. \end{aligned}$$

12b, p. 109: We nemen de beste 25 van 200, dus de verwachte gemiddelde fitness van individuen in de volgende generatie is gelijk aan

$$\begin{aligned} \frac{E[X_{(176)}] + \cdots + E[X_{(200)}]}{25} &= \frac{100 \times \frac{176}{201} + \cdots + 100 \times \frac{200}{201}}{25} \\ &= 100\frac{176 + \cdots + 200}{25 \times 201} \\ &= 93.5323. \end{aligned}$$

12c, p. 109: We nemen de beste 25 van λ, dus de verwachte gemiddelde fitness van individuen in de volgende generatie is gelijk aan

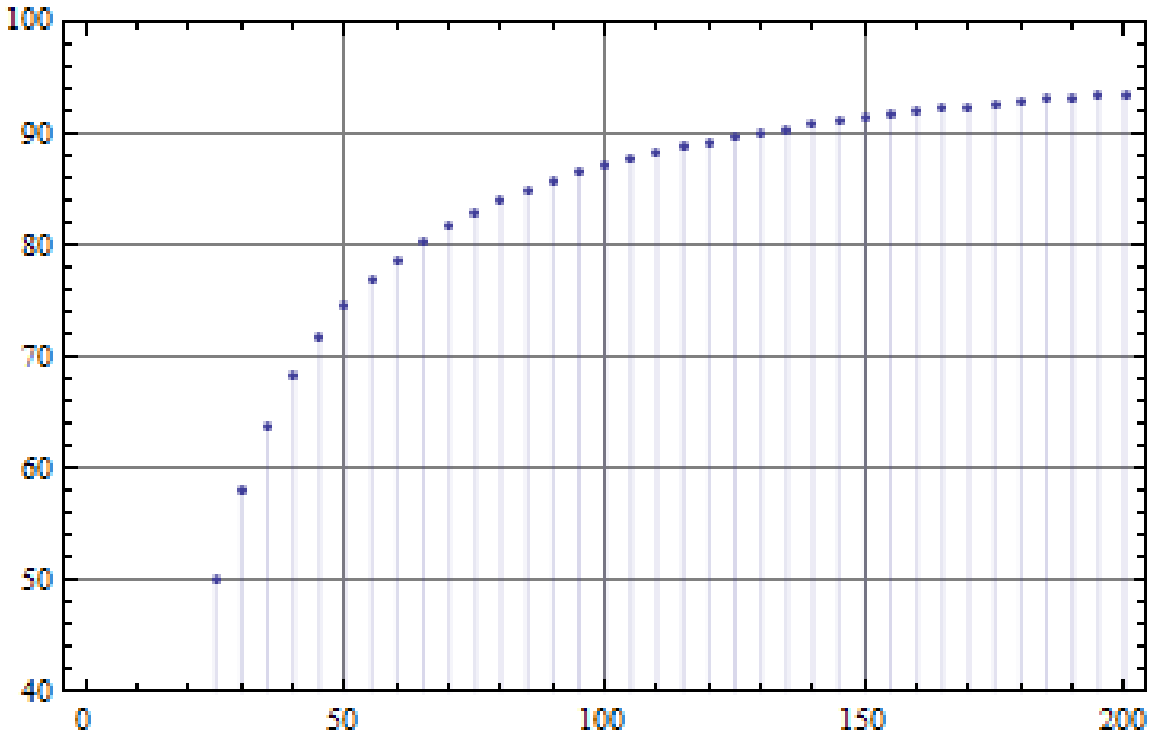
$$\begin{aligned} \frac{E[X_{(\lambda-25+1)}] + \cdots + E[X_{(\lambda)}]}{25} &= 100\frac{(\lambda - 25 + 1) + (\lambda - 24 + 1) + \cdots + \lambda}{25(\lambda + 1)} \\ &= 100\frac{25\lambda - (1 + 2 + \cdots + 24)}{25(\lambda + 1)} \\ &= 100\frac{25\lambda - 300}{25(\lambda + 1)} \\ &= 100\frac{\lambda - 12}{\lambda + 1}. \end{aligned}$$

Voor $1 + 2 + \cdots + 24$ gebruikten we de formule

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}. \tag{18}$$

12d, p. 109: De grafiek van

$$\mathbb{N} \cap [25, \to) \to [0, 100] : \lambda \mapsto 100\frac{\lambda - 12}{\lambda + 1}$$



alleen getekend voor veelvouden van 5. (Je mag geen continue lijn tekenen, λ is immers een geheel getal.) Voor $\lambda = 25$ is de verwachte gemiddelde fitness

$$100\frac{25 - 12}{25 + 1} = 50.$$

Dat is ook wel logisch, immers, we zijn in dat geval verplicht alle 25 getrokken individuen te accepteren—er mag niets worden weggegooid. De grafiek stijgt monotoon, omdat een top 25 kiezen uit grotere samples natuurlijk betere individuen oplevert. De grafiek bezit limiet 100 omdat bij extreem grote samples de top 25 bestaat uit elementen met een fitness dicht bij 100.

13a, p. 109: In Ruby:

```
#!/usr/bin/ruby - w

class Array

   def mean # computes the average
      inject{ |sum, x| sum + x } / length.to_f
      # coerce length into float to avoid integer division
   end

   def pick # selects a random element
      at rand(length)
```

```
      # rand(n) returns a random integer in the range 0, ..., n-1
   end

end

def Ek_in_sim(list, r, n, number_of_trials)
   average = 0.0 # running average
   number_of_trials.times { |trial|
      # create a sample of n random elements
      sample  = Array.new(n) { list.pick }
      # sort the sample, then compute the mean of the elements r, r+1, ..., n
      mean_l  = sample.sort.slice(r-1,n-r+1).mean
      # slice works like: [0, 1, 2, 3, 4, 5].slice(2,3) -> [2, 3, 4]
      # finally, update the running average
      average = (trial*average + mean_l) / (trial + 1).to_f
   }
   average
end

p Ek_in_sim([1, 1, 2, 3, 3, 3, 5, 5, 8, 8, 8, 9], 19, 30, 100_000)
```

Na 100,000 experimenten krijgen we $\kappa = 7.5953525$. Om een betrouwbaarheidsinterval te krijgen zouden we elk van deze miljoen experimenten bijvoorbeeld 30 keer kunnen herhalen. Zo ver gaan we niet.

13b, p. 109: De populatie van genotypen heeft fitness 1, 1, 2, 3, 3, 3, 5, 5, 8, 8, 8, en 9, dus de kansdichtheid-functie van deze populatie is gelijk aan

$$f_X(x) = \begin{cases} 1/6 & \text{als } x = 1 \\ 1/12 & \text{als } x = 2, \\ 1/4 & \text{als } x = 3, \\ 1/6 & \text{als } x = 5, \\ 1/4 & \text{als } x = 8, \\ 1/12 & \text{als } x = 9, \\ 0 & \text{anders.} \end{cases}$$

De cumulatieve verdelingsfunctie F_X, en de complementaire cumulatieve verdelingsfunctie S_X zijn dan per definitie gelijk aan respectievelijk[58]

$$F_X(x) = \sum_{i=1}^{x} f_X(i) \qquad \text{en} \qquad S_X(x) = \sum_{i=x}^{9} f_X(i).$$

We trekken 30 keer. Dit levert ongeordende waarden $X_1, X_2, \ldots, X_{30}$. We ordenen $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(30)}$ en selecteren de beste 12. We zijn geïnteresseerd in het gemiddelde van

$$X_{(19)}, X_{(20)}, X_{(21)}, \ldots, X_{(28)}, X_{(29)}, X_{(30)}.$$

[58] De functienaam S_X staat voor het Engelse survival function. Voor discrete verdelingen geldt $F_X(x)+S_X(x+1) = 1$.

Volgens [12, Sec. 2.2., p. 23] geldt dat de kansdichtheid-functie $f_{X_{(r)}}$ van $X_{(r)}$ gelijk is aan

$$f_{X_{(r)}}(x) = \sum_{u=0}^{r-1} \sum_{w=0}^{n-r} \binom{n}{u, n-u-w, w} F_X^u(x-1) f_X^{n-u-w}(x) S_X^w(x+1).$$

Per definitie van verwachting geldt

$$E[X_{(r)}] = \sum_{i=1}^{9} i f_{X(r)}(i).$$

(We zitten inmiddels op drie keer sommeren.) Dus

$$\kappa = \frac{E[X_{(19)}] + E[X_{(20)}] + \cdots + E[X_{(30)}]}{12}.$$

(We zitten inmiddels op vier keer sommeren.) Allemaal uitrekenen, met de hand, of met een wiskundig softwarepakket, geeft $\kappa = 7.59571989163302$.

1, p. 110: 1. **Aantal Runs en Hiërarchie**: Het algemene diagram bevat een overkoepelende lus voor meerdere runs (Run := 0 tot Run = N), waarbij elke run een nieuwe populatie initialiseert. Het Koza-diagram start direct met een eenmalige generatie-initialisatie (Gen := 0).

2. **Gedetailleerde Fitness-evaluatie**: In het algemene diagram wordt de fitness-meting expliciet weergegeven als een iteratief proces per individu ($i := 0$ tot $i = M$). Het Koza-diagram vat deze stap samen in één procesblok: "Evaluate fitness of each individual in population".

3. **Variatie in Genetische Operaties**: Het algemene diagram specificeert vier operaties: reproductie (P_r), crossover (P_c), mutatie (P_m) en architectuur-wijzigende operaties (P_a). Het Koza-diagram toont in de visuele flow enkel reproductie (P_r) en crossover (P_c).

4. **Architectuur-wijzigingen**: Het algemene diagram bevat een specifiek pad voor "Architecture Altering Operations", inclusief een selectie-stap op basis van waarschijnlijkheid. Deze ontbreekt in het Koza-diagram.

5. **Crossover Output**: Het Koza-diagram is specifieker over het resultaat van crossover; het vermeldt expliciet het invoegen van twee nakomelingen ("Insert Two Offspring") in de nieuwe populatie. Het algemene diagram spreekt over het invoegen van "Offspring"zonder een specifiek aantal te noemen.

6. **Terminatie-afhandeling**: Bij Koza leidt het voldoen aan het terminatie-criterium direct naar de definitieve "End"-status. In het algemene diagram is dit slechts de afsluiting van één specifieke run, waarna de teller Run := Run + 1 wordt opgehoogd om te controleren of er nog meer runs nodig zijn.

2, p. 111: 1. **Het probleem (Koza I)**: In het eerste boek bestond een individu uit één enkele boomstructuur. Bij complexe problemen leidde dit tot de *curse of dimensionality*: de zoekruimte werd exponentieel groter en de bomen werden te diep en ineffici ent ("bloat", waarover we in het college spraken), waardoor het algoritme soms vastliep voordat een oplossing werd gevonden.

2. **De oplossing (Koza II)**: *Automatically Defined Functions* (ADFs) introduceerden modulariteit. In plaats van één boom, heeft een individu nu een hoofdprogramma en één of meer sub-bomen. Het algoritme leert tijdens de evolutie niet alleen het hoofdprogramma, maar definieert en optimaliseert ook zelfstandig de herbruikbare sub-routines.

3. **Hi erarchische structuren**: ADFs maken *hierarchical decomposition* mogelijk. Net zoals een menselijke programmeur functies schrijft voor terugkerende taken, kan GP met ADF's patronen herkennen, deze in een functie vastleggen en ze op meerdere plekken in het hoofdprogramma aanroepen. Dit is essentieel omdat het de zoekruimte verkleint: het algoritme hoeft een complex patroon niet telkens opnieuw uit te vinden, maar hoeft alleen de aanroep naar de eerder ge evolueerde ADF te optimaliseren. Dit maakte de sprong mogelijk van simpele wiskundige functies naar het ontwerpen van complexe circuits in Koza III en IV.

3, p. 111: FORTRAN (FORmula TRANslator), Cobol (COmmon Business Oriented Language), en Algol (Algorithmic Language). LISP (LISt Processor) bezit een veel eenvoudiger syntax: alles is een lijst. Of we "LISP" of "LISP" moeten schrijven is niet duidelijk. Het lijkt er op alsof in de periode 1960-1980 vaker LISP all caps werd geschreven, en het daarna niet zo veel meer uitmaakte.

Jon McCarthy (Stanford U.) beschreef LISP aan het eind van de jaren vijftig. Gebaseerd op de lambda-calculus, werd het al snel de programmeertaal bij uitstek voor AI-toepassingen na de publicatie ervan in 1960. Nog steeds is LISP belangrijk met name vanwege haar uiterst eenvoudige syntax. In geen enkele andere taal ligt het zo voor de hand om programma's te zien als data welke kan worden gemanipuleerd, bijvoorbeeld door LISP zelf.

Edsger Dijkstra: "LISP has been jokingly described as 'the most intelligent way to misuse a computer'. I think that description is a great compliment because it transmits the full flavor of liberation: it has assisted a number of our most gifted fellow humans in thinking previously impossible thoughts." Lees vooral Paul Graham's site voor meer quotes. Deze karakteriseren als geen ander de taal en cultuur van LISP.

LISP is nu geen serieuze industrietaal meer (Google "is LISP dead?"), en is ingehaald door scripttalen als Perl (de Cadillac onder de scripttalen), Python (de Daf-Volvo onder de scripttalen, ugh!), en Ruby (de Toyota Supra GR onder de scripttalen).

4a, p. 111: Ja, ja, nee, ja, ja, nee.

4b, p. 111: Ja, ja, nee, ja, ja, ja. Het laatste antwoord doet wellicht vreemd aan. LISP ziet echter geen verschil tussen `NIL` en de lege lijst `()`. Dat komt omdat beide expressies in LISP dezelfde interne representatie hebben. In LISP heeft NIL ook drie betekenissen: de lege lijst, false, en de nul-waarde (zoals null in PHP.) Een zuiverder versie van LISP, Scheme, is wat dat betreft duidelijker. Daar bestaat `NIL` niet, maar heb je `#T` en `#F` die true en false representeren en beide naar zichzelf evalueren.

4c, p. 111: Ja, want elke lijst is een s-expressie; ja, want een lijst van vier lege lijsten is een lijst; ja; nee. We zouden `(A, B, C)` kunnen lezen als een lijst met vijf atomen (waarvan twee komma's) maar de komma is niet toegestaan als atoom in LISP. Deze wordt gebruikt in de definitie van zg. macro's. Ja; nee.

4d, p. 111: S-expressies: `(- X 5)`, `(+ X (+ Y Z))`, `(+ X Y Z)` [toegestaan in LISP], `(SQRT(- X 5))`, `(/ (SQRT (- X 5)) 4)`, `(SQRT (- (/ X 4) 5))`, `(/ (SQRT (+ 4 (- X 5))) (- Y (- W)))`.

4e, p. 111: S-expressies: `(IF (FOOD) (MOVE) (TURN))`; `(IF (FOOD) (IF (TASTY) (EAT) (TURN)) (TURN))`; `(IF (FOOD) (IF (TASTY) (IF (SCARCE) (SAVE) (EAT)) (TURN)) (TURN))`.

4f, p. 111: De string `NIL` evalueert naar zichzelf; de string `T` ook; de string `112` is een getal, dus ook; de string `(112)` is een lijst dus een s-expressie, maar evaluatie daarvan resulteert in een fout, want LISP "denkt" dat, omdat `112` als eerste element in een lijst staat, dat de functie `112` moet worden aangeroepen. Maar `112` is een getal, en getallen kunnen in LISP niet fungeren als functiesymbolen. De string `(* 5 (- 2 3))` evalueert naar `-5`; de string `(+ (- 9 5) (- 2 3))` evalueert naar `3`; de string `(-2 3)` is een s-expressie maar evaluatie resulteert in een fout, want `-2` is geen functiesymbool; de string `(* 5)` is een s-expressie en zal naar `5` evalueren omdat de vermenigvuldigingsfunctie in LISP geen vaste ariteit bezit. LISP begint met `1` en vermenigvuldigt dit gewoon met alle getallen die daar achter staan. De string `(* 2 3 4)` evalueert naar `24`.

6a, p. 112: `(+ 1 (* 2 3))`

6b, p. 112: `7`

6c, p. 112: `(+ 1 '(* 2 3))`

6d, p. 112: `Error.`

7, p. 112: Het eerste programmaatje bevat een echte if-statement, met een test `(FOOD-AHEAD)`, een then-gedeelte en een else-gedeelte. Het tweede programmaatje heeft geen if-statement, daar is de test hard gecodeerd ("ingebakken") in een instructie.

Typen eerste programma naar volgorde openingshaken: actie, test, actie actie. Typen tweede programma naar volgorde openingshaken: actie, actie, actie.

8a, p. 112: Deze expressies bevat twee Boolse sub-expressies (i.e., tests), en negen numerieke sub-expressies, te weten `(if ..)`, `x`, `3`, `(if ..)`, `y`, `5`, `x`, `7`, en `z`. (De uitkomst van if-expressies is hier ook numeriek, en een hoofd-expressie is ook een sub-expressie.) Elke Boolse expressie uit Ouder 1 kan kruisen met twee Boolse expressies uit Ouder 2. Kruisen op Boolse expressies geeft dus $2 \times 2 = 4$ mogelijkheden. Elke numerieke expressie uit Ouder 1 kan kruisen met negen numerieke expressies uit Ouder 2. Kruisen op numerieke expressies geeft dus $9 \times 9 = 81$ mogelijkheden. In totaal geeft dit 85 mogelijkheden.

8b, p. 112: Expressie 1: twee tests, te weten `(wall-ahead)` en `(NOT (wall-ahead))` en drie instructies, te weten `(IF ..)`, `(move)` en `(turn)`. Expressie 2: twee tests en drie instructies. In totaal $2 \times 2 + 3 \times 3 = 13$ mogelijke kruisingen.

8c, p. 112: De eerste expressie bevat twee numerieke sub-expressies waarvan er één een lijst is. Er mag alleen op lijst-niveau gekruist worden.

Eerste expressie: 5 statements, 2 tests, 1 numerieke expressie `(rand)`. Tweede expressie: 2 statements, 1 test, 0 numerieke expressies. Totaal $5 \times 2 + 2 \times 1 + 1 \times 0 = 12$ mogelijke kruisingen.

8d, p. 113: Expressie 1 bevat één Boolese expressie (i.e., test), te weten `(FOOD-AHEAD)`. Expressie 2 bevat twee Boolese expressies, te weten `(NOT (FOOD-AHEAD))` en `(FOOD-AHEAD)`. Dit geeft $1 \times 2 = 2$ mogelijke kruisingen.

Expressie 1 bevat drie instructies, te weten `(IF ..)`, `(RIGHT)` en `(LEFT)`. Expressie 2 bevat evenzo drie instructies. Dit geeft $3 \times 3 = 9$ mogelijke kruisingen. Samen geeft dit $2 + 9 = 11$ mogelijke kruisingen.

8e, p. 113: De s-expressie $E =$ `(/ (SQRT (+ 4 (- X 5))) (- Y (- W)))` is homogeen, immers alle sub-expressies zijn numeriek. E bezit 11 sub-expressies, namelijk zes lijsten, te weten `(/ ..)`, `(SQRT ..)`, `(+ ..)`, en drie keer `(- ..)`, en vijf bladeren te weten `4`, `X`, `5`, `Y`, en `W`.

Bij twee ouders is elke sub-expressie een mogelijk punt van kruising. Deze kruispunten moeten wel verschillend zijn, anders wordt E weer verkregen. Bij Ouder 1 zijn is het kruispunt vrij te kiezen en zijn er dus 11 keuzes. Bij Ouder 2 zijn er voor elk van die 11 keuzes dan nog maar $11 - 1 = 10$ keuzes. Dat geeft $11 \times 10 = 110$ mogelijke kruisingen.

9a, p. 113: Knopen worden gevormd door if-then-else statements, if-then statements, tests, en acties. Dus respectievelijk $4 + 4 + 4 = 12$ knopen. Bladeren worden gevormd door tests en acties, niet door if-then-else statements of if-then statements, want die zijn samengesteld. Dus respectievelijk $4 + 4 = 8$ bladeren.

9b, p. 113: Er zijn vier testen en acht statements, waarvan drie if-then-else statements, en één if statement zonder else. Alle testen kunnen met elkaar worden gekruist en alle statements kunnen met elkaar worden gekruist. In totaal geeft dat $4^2 + 8^2 = 80$ kruising-mogelijkheden.

9c, p. 113: Neem een willekeurig atomair statement in de boom van bovenstaand programma, en vervang dat atomaire statement door het programma zelf. In alle gevallen krijgen we een kind met $12 - 1 + 12 = 23$ instructies. Er zijn vier van dergelijke elementaire statements, namelijk alle `(action-*)` statements, dus er zijn vier van dergelijke kruisingen, of acht, als je net zo telt als in de vorige opgave, namelijk boom 1 in boom 2 en dan vervolgens boom 2 in boom 1.

10a, p. 114: Alle eerste elementen van een lijst vormen een instructie—ook de lijsten die beginnen met `IF-*`. Immers, het eerste element van bv. `(IF-FOOD-HERE <blah1> <blah2>)` is een twee-weg instructie met een voorgebakken test. Als we goed tellen zijn er 47 identifiers. Dat levert dus $47^2 = 2209$ mogelijke kruisingen op.

10b, p. 114: Neem een willekeurig blad in de boom van bovenstaand programma, en vervang dat blad door het programma zelf. In alle gevallen krijgen we een kind met $47 - 1 + 47 = 93$ instructies. Er zijn 18 bladeren, dus 18 kruisingen, dus ook 18 kinderen met 93 instructies. (Niet 36 kinderen.)

10c, p. 114: Diepte 0, 1, 2, 3, 0, 0 (want lege lijst is `NIL` en `NIL` is een atoom), 1, 1, 2.

10d, p. 114: Een blad op niveau 10 vervangen door de hele boom levert een structuur met diepte $2 * 10$ op. [Niet $10 + (10 - 1)$ o.i.d. Probeer zo nodig uit met `(A (B C) D)`.]

11, p. 114: Neem de grotere functie-set $\{\neg, \wedge, \rightarrow, \vee, \equiv\}$. Daarmee kun je het GP-algoritme korte formules laten maken. Korte formules zijn overzichtelijker dan lange formules met alleen $\neg$ en $\wedge$ en verdienen dus de voorkeur.

13a, p. 115: Om 26 verschillende letters te kunnen representeren heb je vijf bits nodig. Deze geven je 31 plekken. (Je houd dus nog $31 - 26$ vijf codons over.)

13e, p. 115: Non-terminal: alle expressies die nog herschreven kunnen worden: `<prog>`, `<action>`, `<else>`, etc. Terminals: alle expressies die niet herschreven kunnen worden `;`, `if`, `else`, `food-here?`, `bug-here?`, `left`, `right`, `move`, `eq`, `lt`, `gt`, `+`, `-`, `*`, `/`, `x`, `y`, `z`, `0.5`, `1`, `5`, `10`, `50`, `100`, `500`, en `1000`. Tokens: alle zinvolle kleinste elementen in een programma. Bv. `if` is een zinvol kleinste element, de afzonderlijke letters `i` en `f` zijn dat niet. Tokens en terminals vallen hier samen.

13f, p. 115: 23.

13g, p. 115: Zie linkerkolom: 10.

13i, p. 115: `if bug-here? right else right` (2x), `left`, en een onaf programma.

13j, p. 115: 112, 10210122100, 1021012210111, ...

13k, p. 115: Voor bv. `move` zijn er $2 \times 4 \times 4$ keuzemogelijkheden.

13l, p. 115: Drie bits.

13m, p. 115: Dan kunnen bij sommige herschrijfregels niet alle alternatieven worden bereikt.

15a, p. 116: First we should decide if one individual (pacman) may respawn during one episode (test-periode for one generation). Let us choose to do not so. Further we suppose and individual is alloted a fixed number of steps, or a fixed amount of time. If the individual is still alive after this number of steps, its run end nevertheless.

Given these assumptions, fitness should take into account a number of aspects. The following might seem relevant. Number of ticks survived (t), number of points scored by eating pills and blue ghosts (p).

A possible fitness function could then be $f(i) = \alpha t + \beta p$, where $\alpha, \beta 0$ are constants > 0.

15b, p. 116: A solution (there are many alternatives):

Grammar:

```
<command> ::= (do-nothing) | (left) | (right) | (up) | (down)
            | ( <ifelse> <command> <command> )
<ifelse>  ::= ifelse-dots-left? | ifelse-dots-right?
            | ifelse-dots-up?   | ifelse-dots-down?
            | ifelse-ghosts-left? | ifelse-ghosts-right?
            | ifelse-ghosts-up?   | ifelse-ghosts-down?
            | ifelse-ghosts-blue?
```

(E.g., “ifelse-dots-in-neighbourhood?” would not make much sense because then pacman does not know how to orientate).

Non-terminals: `<command>`. Terminals: parentheses, do-nothing, left, right, up, down, ifelse-dots-left?, ifelse-dots-right?, ifelse-dots-up?, ifelse-dots-down?, ifelse-ghosts-left?, ifelse-ghosts-right?, ifelse-ghosts-up?, ifelse-ghosts-down?.

15c, p. 116: Bijvoorbeeld

```
(ifelse-ghosts-left? (ifelse-ghosts-blue? (left) (right)) (up))
```

15d, p. 116: Intelligent primitives will do much of the forework that is needed for intelligent manoeuvering. Hence, such primitives will increase the chance that short successful programs will be found relatively quickly.

16b, p. 116: Bijvoorbeeld:

```
<program> ::= <program> <cmd> | <cmd>
<cmd>     ::= <act> | <move> | <ifstat>
<ifstat>  ::= if <test> [ <program> ]
            | ifelse <test> [ <program> ] [ <program> ]
<test>    ::= food-here? | not-pheromone-here? | turtle-in-my-sight?
<move>    ::= face <loc> | ft <step> | rt <degree>
<act>     ::= drop-pheromone | set haste <haste>
<loc>     ::= closest-food | closest-turtle
<step>    ::= <amp> * haste
<amp>     ::= -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5
<degree>  ::= -90 | 0 | 1 | 5 | 90 | 180 ; for instance
<haste>   ::= 0 | 1 | 2 | 3 ; for instance
```

16c, p. 117: Non-terminal `<amp>` possesses eleven alternatives, hence we need $\lceil log_2(11) \rceil = 4$ bits to be able to address all alternatives.

16d, p. 117: The bitstring `0000` will continue to expand `<program>` into `<program> <cmd>`.

17a, p. 117: S-expressies: `(- (SQRT (- 12 X)))`, `(SQRT (SQRT (- 4)))`.

17b, p. 117: S-expressies: `(IF (FOOD) (TURN) (MOVE))`; `(IF (FOOD) (IF (TASTY) (EAT) (IF (SCARCE) (SAVE) (EAT))) (IF (SCARCE) (EAT) (TURN)))`.

17c, p. 117: De string `889` evalueert naar 889, de string `(889)` geeft een runtime error, de string `(- (4 5) 1)` geeft een runtime error, de string `(+ (- 9 5) (- 3))` evalueert naar 1, de string `(-7 3)` evalueert naar 1, de string `(* 5)` geeft een runtime error, de string `(- 2 3 4)` evalueert naar -5.

18a, p. 117: De tweede s-expressie is homogeen, meer i.h.b. opgebouwd uit alleen instructies.

18b, p. 117: Elf. Eerste: twee instructies (while en eat) en één test (food-ahead). Tweede: vier instructies (if, right, while, eat) en drie testen (not, food-ahead, can-eat). Dus `(NOT (FOOD-AHEAD))` en `(FOOD-AHEAD)` zijn tests. Samen $2 \times 4 + 1 \times 3 = 11$ mogelijkheden.

18c, p. 117: De eerste expressie bezit twee instructies die onderling kunnen kruisen ($+2 \times 2$) en één test die onderling (zonder resultaat) kan kruisen ($+1 \times 1$). De tweede expressie bezit vier instructies die onderling kunnen kruisen ($+4 \times 4$) en drie testen die onderling kunnen kruisen ($+3 \times 3$). Samen zijn er nu $4 + 1 + 16 + 9 = 30$ mogelijke onderlinge kruisingen. Plus 11 mogelijke wederzijdse kruisingen (zie onderdeel 18b) geeft 41 mogelijke kruisingen in totaal.

1, p. 121: "rather than evolve individuals against a fixed objective metric, we engage individuals in the task of improving their performance against other evolving individuals."

1, p. 121: "for many machine learning domains, a suitable objective metric of performance is simply not available"

1, p. 121: "a) Providing a target that is 'hittable'–gradient." en "b) A target that is relevant–focusing. " en "c) A moving target–open-endedness."

1, p. 121: "a) Loss of gradient. " en "b) Focusing on the wrong things." en "c) Relativism."

1, p. 121: Aanpak: "we introduce in this paper a minimal substrate in which co-evolutionary concepts, dynamics, and problems can be investigated". Motivatie: "This substrate enables us to illustrate some important concepts that may be underlying the problems we introduced above.". De onderzoekers zeggen dus te kiezen voor een minimale aanpak. Voor de aanpak wordt gekozen omdat ze verwachten dat dan de essentiële problemen die optreden bij kunstmatige co-evolutie beter bestudeerd kunnen worden.

2, p. 122: Het succes van een speerwerper kan worden afgemeten aan wat hij (zij) gemiddeld werpt. Daar heeft hij of zij niemand anders voor nodig. Het succes van een schermer hangt af van anderen.

3a, p. 123: De totale fitness van populatie A is gelijk aan 34. Dus $f(a_1) = 3/34$, $f(a_2) = 3/34$, $f(a_3) = 8/34$, $f(a_3) = 1/34$, $f(a_4) = 1/34$, ... De totale fitness van populatie B is gelijk aan 43. Dus $f(b_1) = 3/43$, $f(b_2) = 5/43$, $f(b_3) = 2/43$, $f(b_3) = 7/43$, $f(b_4) = 2/43$, ...

3b, p. 124:

1. $f(a_8, \{b_3\}) = 1$
2. $f(a_8, \{b_7\}) = 0$
3. $f(a_8, \{b_9\}) = 1$
4. $f(a_8, \{b_{10}\}) = 1$
5. $f(a_8, \{b_2\}) = 0$
6. $f(a_8, \{b_2, b_3, b_7, b_9, b_{10}\}) = 3$
7. $f(a_8, \{b_1, b_3, b_5, b_6, b_9, b_{10}\}) = 6$
8. $f(a_8, \{b_2, b_4, b_7, b_8\}) = 0$
9. $f(a_8, \{b_1, b_4, b_5, b_6, b_7\}) = 3$
10. $f(a_8, \emptyset) = 0$
11. $f(a_8, B) = 6$
12. $f(a_3, B) = 9$

3c, p. 124: De relatieve fitness voor elk element van A is gelijk aan resp. 1, 1, 2, 0, 0, 0, 1, 2, 2, 1.

3d, p. 124: $S = \{b_6, b_8\}$ bevat één pertinente winnaar met fitness 9 en één pertinente verliezer met fitness 1. Bijna ieder element uit A scoort dus hetzelfde op B. De relatieve scores zijn resp. 1, 1, 1, 0, 0, 0, 1, 1, 1, 1.

De elementen $\{a_4, a_5, a_6\}$ onderscheiden zich hierdoor niet meer van elkaar, deze bezitten allemaal een relatieve fitness van 0. De rest, $A - \{a_4, a_5, a_6\}$, onderscheidt zich ook niet meer van elkaar, zij bezitten allemaal een relatieve fitness van 1.

3e, p. 124: De relatieve fitness voor elk element van A ten opzichte van S is gelijk aan resp. 1, 1, 2, 0, 0, 0, 1, 2, 2, 1. De grootte van de vergelijkingsset, S is gelijk aan 2, i.e. $|S| = 2$. De proportioneel relatieve fitness voor elk element van A ten opzichte van S is dus gelijk aan resp. 1/2, 1/2, 2/2, 0/2, 0/2, 0/2, 1/2, 2/2, 2/2, 1/2.

3f, p. 124: De proportioneel relatieve fitness voor elk element van A ten opzichte van S is gelijk aan resp. 1/2, 1/2, 2/2, 0/2, 0/2, 0/2, 1/2, 2/2, 2/2, 1/2. In totaal is dit gelijk aan 5. De genormeerde relatieve fitness voor elk element van A ten opzichte van S is dus gelijk aan resp. 1/10, 1/10, 2/10, 0/10, 0/10, 0/10, 1/10, 2/10, 2/10, 1/10.

Waarschuwing: omdat S hier constant is geldt

$$\bar{f}(a, S) = \frac{f(a, S)}{\text{totale relatieve fitness}}.$$

Echter, zodra de grootte van S gaat variëren, gaat deze gelijkheid niet meer op.

4a, p. 124: Elementen in B krijgen door $S = \{a_4, a_5, a_6\}$ een relatieve fitness van resp. 0, 1, 0, 3, 0, 0, 2, 3, 1, en 1.

Klasse 1 (fitness 0): $\{b_1, b_3, b_5, b_6\}$;
Klasse 2 (fitness 1): $\{b_2, b_9, b_{10}\}$;
Klasse 3 (fitness 2): $\{b_7\}$;
Klasse 4 (fitness 3): $\{b_4, b_8\}$.

4b, p. 125: Het komt er op neer dat we moeten laten zien dat, als b en b' dezelfde relatieve fitness ten opzichte van S' bezitten, zij ook dezelfde fitness ten opzichte van S bezitten. We moeten dus laten zien dat het verkleinen van S' naar S geen nieuwe verschillen in fitness introduceert.

Laat $S' = \{s_1, \ldots, s_n\}$. Omdat b en b' dezelfde relatieve fitness ten opzichte van S' bezitten zal er een $1 \leq k \leq n$ moeten bestaan zo dat

$$s_1 \leq \cdots \leq s_k < b, b' \leq s_{k+1} \leq \cdots \leq s_n.$$

In dit geval bezitten b en b' beiden dus fitness k. Bij weghalen van een element uit S' (om S te krijgen) blijven b en b' dezelfde relatieve fitness behouden. Immers, als er een s_i wordt weggehaald met $1 \leq i \leq k$ (i.e., onder de b's) dan zal de relatieve fitness van b en b' met 1 zakken naar $k - 1$. Als er een s_i wordt weggehaald met $k + 1 \leq i \leq n$ (i.e., boven de b's) dan zal de relatieve fitness van b en b' niet veranderen, en gelijk aan k blijven.

5a, p. 125: Gemiddeld genomen zal a_8 zes van de tien keer winnen als het tegen één individu uit B wordt opgezet. De verwachte relatieve fitness van a_8 bij steekproef-grootte k is dus gelijk aan $6k/10$.

5b, p. 125: Dat is $6k/10$ gedeeld door de steekproef-grootte k, dat is dus $6/10$.

5c, p. 125: Hoe groter de sample, hoe representatiever de relatieve fitness. De variantie zal dus afnemen.

6, p. 125: Absolute fitness van een individu: intrinsieke score van een individu. Dat moet een reëel getal zijn. Genormeerde absolute fitness van een individu: absolute fitness van dat individu gedeeld door de som van de absolute fitness van alle individuen. Relatieve fitness van een individu a: hangt van het groepje S af waarmee a wordt vergeleken. In dat geval is relatieve fitness het aantal elementen in S dat door a gedomineerd wordt. Genormeerde relatieve fitness: relatieve fitness van dat individu gedeeld door de som van de relatieve fitness van alle individuen. Let wel:

1. Relatieve fitness wordt altijd bepaald in een configuratie $(a_1, S_1), \ldots, (a_n, S_n)$. Dus om de relatieve fitness te bepalen wordt elke a_i vergeleken met een bepaald groepje S_i. Het kan zijn dat $S_i = S_j$ voor alle $1 \leq i, j \leq n$, i.e., het kan zijn dat alle groepjes hetzelfde zijn, maar dat wordt niet op voorhand geëist.
2. Om een eerlijke relatieve fitness te krijgen, moet $|\, S_i \,| = |\, S_j \,|$ voor alle $1 \leq i, j \leq n$. M.a.w. elke a_i moet tegen een even groot groepje strijden om evenveel scoringskansen te hebben.

7a, p. 125: Nee. Tenzij je iets hebt gedefinieerd wat een punt in R^2 reduceert tot een reëel getal. (Bv. som van geboortejaar en het getal dat wordt gevormd door de eerste drie cijfers van ons collegekaartnummer.)

8a, p. 126: Merk eerst op dat het aantal winsten op een lege verzameling van concurrenten altijd nul is. Voor onderdelen (8(a)i-8(a)x) krijgen we dan:

(a): $0 < 0 \Rightarrow false$ (b): $0 < 1 \Rightarrow true$ (c): $1 < 2 \Rightarrow true$ (d): $1 < 2 \Rightarrow true$
(e): $2 < 2 \Rightarrow false$ (f): $1 < 1 \Rightarrow false$ (g): $0 < 0 \Rightarrow false$ (h): $0 < 0 \Rightarrow false$
(i): $3 < 0 \Rightarrow false$ (j): $3 < 3 \Rightarrow false$

8b, p. 126: Objectieve preferentie: vergelijk twee individuen op hun absolute fitness. Het meest fitte individu wordt geprefereerd. Subjectieve preferentie: vergelijk twee individuen op hun relatieve fitness. Het meest fitte individu wordt geprefereerd.

8c, p. 126: Vele voorbeelden zijn mogelijk.

8d, p. 126: Die is de ontkenning van het antwoord op Vraag (8e). Kijk daar dus even. Uiteindelijk komen we uit op die combinaties van a, b, A en B zodanig dat $|(\leftarrow, a) \cap B| \geq |(\leftarrow, b) \cap A|$.

8e, p. 126: De vraag is in welke omstandigheden $P_{subj}^{B', A'}(a, b) = P_{obj}(a, b)$. De vraag is dus voor welke a, b, A en B geldt dat

$$a < b \Leftrightarrow f(B, a) < f(A, b).$$

Stel $a < b$. Voor welke A en B is het dan zo dat $f(B, a) < f(A, b)$? Dat is het geval als $|(\leftarrow, a) \cap B| < |(\leftarrow, b) \cap A|$. Dus als er meer elementen uit A vóór b liggen dan dat er elementen uit B vóór a liggen. (Als $A = B$ valt er meer te zeggen, en is eigenlijk een interessanter vraagstuk, maar dat werd helaas niet gevraagd.)

9a, p. 126: Er bestaan 2^{100} bit strings ter lengte 100. Voor elke bit kunnen namelijk twee keuzes worden gemaakt: zetten we er een 0 of een 1 neer?

Er geldt: $\log(2^{100}) \approx 30$ dus $2^{100} \approx 10^{30}$. Dus een 1 met dertig nullen.

9b, p. 126: Er bestaat 1 representatie van het getal 0, dat is de string met 100 nullen. Er bestaan 100 representaties van het getal 1: er zijn immers 100 plekken waar die 1 kan worden neergezet. De rest wordt 0. Er bestaan

$$\binom{100}{50}$$

representaties van het getal 50: we moeten 50 uit 100 plekken kiezen om een 1 neer te zetten. Er bestaan

$$\binom{100}{n}$$

representaties van het getal n: we moeten n uit 100 plekken kiezen om een 1 neer te zetten.

9c, p. 126: Deze kans is binomiaal verdeeld met $n = 100$ en $p = 0.5$. Dus

$$P\{\text{fitness is } n\} = \binom{100}{n} \left(\frac{1}{2}\right)^n \left(\frac{1}{2}\right)^{100-n}.$$

9d, p. 127: De kans dat er minder dan 45 enen in een bit string zitten is $\Pr(X \leq 44)$, waarbij Pr de binomiale verdeling is met $n = 100$ en $p = 0.5$. Opzoeken levert $\Pr(X \leq 44) = 0.1356$. De kans dat er meer dan 75 enen in een bit string zitten, is natuurlijk even groot. (Want gelijk aan de kans op minder dan 25 nullen.) De kans dat er tussen de 45 en 55 enen in een willekeurige bit string van honderd zitten is daarmee $1 - 2 \cdot 0.1356 \approx 0.7288$.

9e, p. 127: Als er k enen zijn, zijn er evenzoveel kwetsbare plekken voor mutatie. De kans dat één van die plekken getroffen wordt is $k/100$. De kans dat vervolgens die 1 in een 0 veranderd is een $1/2$. Het antwoord is dus $k/200$.

9f, p. 127: Laat X gelijk zijn aan de fitness van een willekeurig element uit B. Inmiddels is bekend dat X binomiaal verdeeld is met $n = 100$ en $p = 0.5$. De kans dat a één keer van een element uit B wint is dus gelijk aan

$$P\{X < f\} = P\{X \leq f - 1\} = \sum_{i=0}^{f-1} \binom{100}{i} \left(\frac{1}{2}\right)^i \left(\frac{1}{2}\right)^{(1-i)}$$

Kort deze kans af met p_f. De kans dat a nu precies k keer wint in 5 vergelijkingen is ook weer binomiaal verdeeld, dit keer met parameters $n = 5$ en $p = p_f$. Dus

$$P\{\text{relatieve fitness } = k\} = \binom{5}{k} p_f^k (1 - p_f)^{5-k}.$$

Deze kans is verder moeilijk te vereenvoudigen, ook niet met hulp van bekende benaderingen van de binomiale verdeling (benadering met de normale verdeling dan wel een benadering met de Poissonverdeling.

10, p. 127: Op het eerste gezicht lijkt `score3` puur instrumenteel: het lijkt slechts te zijn gefabriceerd met als enig doel een intransitieve ordening te produceren. Dit blijkt uit Watson en Pollack's argumenten:

> *"A simple way to modify our game to incorporate intransitive superiority is to modify Equation 2 so that the dimension that determines the outcome of a game is the dimension in which the players are most similar (instead of most different)."*
>
> Iets verder:
>
> *"Note that this game still has the desirable property that a player that is maximal in both dimensions beats all other players. Again, we assert that the objective fitness of an individual in this coevolutionary game is the sum of all dimensions."* [32, p. 704]

De auteurs willen dus een gecontroleerde omgeving waarin cyclische dominantie—het "rots-schaar-papier"-fenomeen—aantoonbaar optreedt en bestudeerd kan worden, los van de complicaties van een concreet toepassingsdomein. In die zin is `score3` dus een bewust ontworpen test-instrument.

Desondanks heeft de onderliggende logica van `score3` wel degelijk wortels in de realiteit:

1. *Meerdimensionaal vermogen.* De auteurs wijzen erop dat de bekwaamheid van een speler in veel reële domeinen—schermen, schaken, ruimtelijke robotica—niet tot één scalaire waarde te reduceren is. Welke dimensie de uitkomst bepaalt, hangt af van de tegenstander. Dit is geen gefabriceerde aanname, maar een eigenschap die Watson & Pollack expliciet ontlenen aan de literatuur over co-evolutie. Watson *et al.* beroepen zich in dit verband op [7, 31].

2. *Intransitiviteit in de natuur.* Circulaire dominantie-relaties zijn empirisch geobserveerd in ecologische systemen (bijvoorbeeld steen-schaar-papier-dynamiek bij hagedissen van de soort *Uta stansburiana*) en in speltheorie (niet-transitieve voorkeuren). Het concept is dan ook een erkend verschijnsel.

3. *De specifieke regel is vereenvoudigd maar gemotiveerd.* De keuze om de *meest gelijkende* dimensie te laten bepalen is een vereenvoudiging ten opzichte van werkelijke competitie, maar weerspiegelt een realistisch principe: wanneer twee spelers in één eigenschap sterk op elkaar lijken, is juist die eigenschap beslissend. Denk aan een schaakpartij waarbij twee spelers in tactiek even sterk zijn maar in positioneel spel van elkaar afwijken: dan is de tactische dimensie doorslaggevend.

De `score3`-metriek is dus *zowel* instrumenteel als geworteld in de werkelijkheid. Het is instrumenteel in de zin dat het expliciet ontworpen is om intransitiviteit te induceren en zo coëvolutionaire mislukkingen te demonstreren; tegelijkertijd is het niet willekeurig: de onderliggende structuur komt gewoon voor in sport, ecologie en speltheorie. De auteurs benadrukken dan ook dat de problemen die `score3` blootlegt een waarschuwing is voor het niet werken van co-evolutie in complexere en realistische toepassingen.

11a, p. 127: Mutatie: één willekeurig bit wordt veranderd in een willekeurig ander bit. Merk op dat met kans $1/2$ er dus geen verandering plaatsvindt! De kans dat op verbetering is de kans dat één van de $100 - m$ nullen gekozen wordt voor mutatie maal de aks dat zo'n mutatie de geslecteerd 0 in een 1 veranderd. Deze kans is dus gelijk aan $(100 - m)/100 \times 1/2 = (100 - m)/200$.

11b, p. 127: In latere stadia van evolutie is het waarschijnlijk dat mutaties een negatieve invloed hebben op de fitness.

11c, p. 127: Sec. 3.1, einde middelste alinea:

> *"In the later stages of evolution it is likely to be the case that nearly all changes to an individual will be detrimental. We will call this situation a negative mutation bias. These basic observations have theoretic underpinnings in the simple models used by Fisher (1930)."*

Volgens Watson en Pollack gebruikte Fisher modellen waarbij mutatie een aantoonbaar negatieve bias had op fitte individuen. Fisher's argument bestond dus uit het tonen van deze modellen, en te wijzen op het gedrag van deze modellen ten aanzien van de neiging tot negatieve mutaties.

11d, p. 127: Relatieve fitness is geen eenvoudige optelling (of gewogen som) van de sterkte van individuele attributen. Het hangt meer af van overwinningen t.o.v. opponenten, en dan telkens met betrekking tot één specifiek attribuut. De ene keer wint (of verliest) een individu van zijn (haar) opponent omdat de opponent zwakker (sterker) is op attribuut x; de andere keer wint (of verliest) een individu van zijn (haar) opponent omdat deze zwakker (sterker) is op een ander attribuut y.

11e, p. 127: Cliff en Miller schrijven het volgende:

> *"For efficiency, we use the same technique as Sims [18], where each individual's fitness is evaluated only in trials against the elite (i.e. highest-scoring) individual from the previous generation of the opponent population: we refer to this technique as LEO (Last Elite Opponent) evaluation. At the end of each trial, the individual under evaluation is given a score."* [7, p. 203]

Verder:

> *"Cycling between strategies is possible if there is an intransitive dominance relationship between strategies. For example, suppose one attempted to co-evolve two populations that compete by playing each other at the children's game 'rock-paper-scissors' where each individual in the population is limited to one genetically determined choice which it uses in all contests in its lifetime (i.e., in game-theoretic terms, only pure strategies are allowed). (...) LEO contests would tend to make cycles especially likely."* [7, p. 208]

In de aanpak van Cliff en Miller is LEO (Last Elite Opponent) dus een evolutionair systeem met een natuurlijke intransitieve ordening.

12a, p. 128: Twee concurrerende populatie-typen. Eén populatie bestaat uit cellulaire automaten, de andere uit te classificeren initiële configuraties. Deze populaties proberen het elkaar zo moeilijk mogelijk te maken.

- De meest voor de hand liggende manier om een initiële configuratie van een 1-dimensionale cellulaire automaat met n cellen te representeren is door middel van een bit string ter lengte n.

- De meest voor de hand liggende manier om een 1-dim. cellulaire automaat te representeren is door middel van een bit string ter lengte k^{2R+1}, in dit geval ter lengte 2^{2R+1}. Het eerste bit geeft de toestand van de nieuwe cel aan bij omgeving 00000 (even aannemende dat $R = 2$), de tweede bit geeft de toestand van de nieuwe cel aan bij omgeving 00001, de derde bit geeft de toestand van de nieuwe cel aan bij omgeving 00010, ..., bit nr. 2^{2R+1} geeft de toestand van de nieuwe cel aan bij omgeving 11111.

De populaties zijn, zeg, elk 100 individuen groot. Vullen mag ad-random gebeuren. Individuen kunnen elke keer tegen, zeg, 15 opponenten hun prestatie meten. Een CA scoort 1 a.e.s.a. het een initiële configuratie correct classificeert, anders 0. Bij initiële configuraties is de score net omgekeerd. Fitness is dan de gemiddelde prestatie over de laatste, zeg, 20 ronden. Mutatie: standaard. Kruising is niet echt nodig (maar mag wel).

12b, p. 128: Door natuurlijke selectie overleven initiële configuraties die de meest zwakke plekken (lees: configuraties) van de op dat moment aanwezige CA's weten te vinden. Deze populatie van initiële configuraties is in het algemeen niet representatief voor *alle mogelijke* initiële configuraties.

Beide populaties kunnen binnen een aantal generaties er compleet anders uitzien. Zo stabiliseert het proces nooit.

12c, p. 128: Selecteer initiële configuraties *niet* op fitness. Op deze manier blijft de populatie van te classificeren initiële configuraties voldoende representatief en divers om CA-s een voldoende brede basis te geven om alle initiële configuraties te leren classificeren.

Zie verder Paredis (1997) "Coevolving cellular automata: Be aware of the red queen" in *Proc. of the 3rd Int. Conf. on Parallel Problems Solving from Nature*, 1997.

1, p. 130: Als communicatie over het spel is toegestaan kun je afspreken dat je gaat samenwerken. Als je elkaar vertrouwt kan die afspraak vervolgens worden verzilverd. De laatste ronde zou je kunnen verzaken—het is immers toch niet meer van invloed op je reputatie. Als je opponent ook op dat idee komt ben je natuurlijk weer slechter af.

2a, p. 130:

$$\begin{array}{c} \\ \mathrm{W} \\ \mathrm{L} \end{array} \begin{array}{c} \begin{array}{cc} \mathrm{W} & \mathrm{L} \end{array} \\ \begin{pmatrix} 2,2 & -1,-1 \\ -1,-1 & 1,1 \end{pmatrix} \end{array}$$

Nash equilibria: two pure: $(0,0)$ en $(1,1)$, one mixed: $(2/5, 2/5)$.

2b, p. 130: Let the government be the row player and the unemployed civilian the column player. The bimatrix game is

$$\begin{array}{r} \\ \text{Aid} \\ \text{not Aid} \end{array} \begin{array}{c} \begin{array}{cc} \text{Try} & \text{not Try} \end{array} \\ \begin{pmatrix} 3,2 & -1,3 \\ -1,1 & 0,0 \end{pmatrix} \end{array}$$

Nash equilibria: one mixed: $(1/5, 1/2)$.

2c, p. 130:

$$\begin{array}{c} \\ \mathrm{C1} \\ \mathrm{C2} \end{array} \begin{array}{c} \begin{array}{cc} \mathrm{C1} & \mathrm{C2} \end{array} \\ \begin{pmatrix} w_1/2, w_1/2 & w_1, w_2 \\ w_2, w_1 & w_2/2, w_2/2 \end{pmatrix} \end{array}$$

This game has two pure strategy Nash equilibria and one other (mixed strategy) Nash equilibrium.

2d, p. 131: 1% is gelijk aan 10,000 Euro. Wanneer een bedrijf ervoor kiest te adverteren dan neemt zijn marktaandeel toe met 20%, en hij betaalt 10,000 voor de adverteren. Standaard is zijn marktaandeel 50%: $50 * 10.000 = 500,000$. Die neemt toe met 20%: $500,000 * 1.2 = 600,000$. Hier moet nog de prijs die hij betaalt voor adverteren vanaf: $600.000 - 10.000 = 590.000$. Die van het andere bedrijf blijft wel gewoon 400,000.

Stel dat ze beiden adverteren, dan blijft hun marktaandeel 50%, maar ze betalen wel allebei voor adverteren. Dit resulteert in de volgende payoff matrix.

$$\begin{array}{rc} & \begin{array}{cc} \text{A} & \text{not A} \end{array} \\ \begin{array}{r} \text{A} \\ \text{not A} \end{array} & \left(\begin{array}{cc} 49E4, 49E4 & 59E4, 40E4 \\ 40E4, 59E4 & 50E4, 50E4 \end{array} \right) \end{array}$$

Eén evenwicht: $(1, 1)$. Dit evenwicht is puur.

3a, p. 131: De JS (joint strategy) $D(C)$ geeft een payoff 4(2) die elke andere payoff domineert. Dit is dus het enige Pareto optimum. (Voor de volledigheid: we zeggen dat payoff (a', b') payoff (a, b) domineert als tenminste één speler echt beter wordt van de verandering van (a, b) naar (a', b'), terwijl de rest van de spelers er niet echt slechter van wordt.)

3d, p. 131: We zeggen dat een joint strategy een Nash-equilibrium is als geen enkele speler er beter van wordt om vanuit deze joint strategy eenzijdig zijn strategie te veranderen. Dit zijn de joint strategies $C(D)$ en $D(C)$.

3f, p. 131: Door af en toe eens te exploreren. Bv. door met kans $\epsilon > 0$ een random move te doen. Als A en B door exploreren ooit $D(C)$ spelen, zullen ze hier, ook na verder exploreren, blijven. (Exploreren levert iets op omdat hiermee ook niet-Nash gesanctioneerde strategie-wijzigingen uitgeprobeerd worden.)

3g, p. 131: A kan in het vervolg beter samenwerken als en slechts als

$$\text{Payoff}_A(C \mid \Pr_B(C) = q_0) > \text{Payoff}_A(D \mid \Pr_B(C) = q_0)$$

$$\begin{aligned} \text{Payoff}_A(C \mid \Pr_B(C) = q_0) &= \text{Payoff}_A(C, C)q_0 + \text{Payoff}_A(C, D)(1 - q_0) \\ &= 1 \cdot q_0 + 3(1 - q_0) \\ &= -2q_0 + 3 \end{aligned}$$

$$\begin{aligned} \text{Payoff}_A(D \mid \Pr_B(C) = q_0) &= \text{Payoff}_A(D, C)q_0 + \text{Payoff}_A(D, D)(1 - q_0) \\ &= 4 \cdot q_0 + 0 \cdot (1 - q_0) \\ &= 4q_0 \end{aligned}$$

A kan dus in het vervolg beter samenwerken als en slechts als $-2q_0 + 3 > 4q_0$, oftewel a.e.s.a. $q_0 < 1/2$. Als $q_0 > 1/2$, dan kan A beter ophouden met samenwerken. Als $q_0 = 1/2$, dan maakt het niet uit. Al deze beweringen zijn alleen geldig als B inderdaad ook C blíjft spelen met frequentie q_0.

3h, p. 131:

$$\begin{aligned}\text{Payoff}_A &= p(q\text{Payoff}_A(C,C) + (1-q)\text{Payoff}_A(C,D)) + \\ &\quad (1-p)(q\text{Payoff}_A(C,C) + (1-q)\text{Payoff}_A(C,D)) \\ &= pq\cdot 1 + p(1-q)\cdot 3 + (1-p)q\cdot 4 + (1-p)(1-q)\cdot 0 \\ &= -6pq + 3p + 4q\end{aligned}$$

3i, p. 131:

$$\begin{aligned}\text{Payoff}_B &= p(q\text{Payoff}_B(C,C) + (1-q)\text{Payoff}_B(C,D)) + \\ &\quad (1-p)(q\text{Payoff}_B(C,C) + (1-q)\text{Payoff}_B(C,D)) \\ &= pq\cdot -1 + p(1-q)\cdot 0 + (1-p)q\cdot 2 + (1-p)(1-q)\cdot -1 \\ &= -4pq + p + 3q - 1\end{aligned}$$

3j, p. 131: $\partial\text{Payoff}_A(p,q)/\partial p = \partial(-6pq + 3p + 4q)/\partial p = -6q + 3$. Voor hoekpunten met $q = 0$ is $\partial\text{Payoff}_A/\partial p$ positief, dus in $(0,0)$ en $(1,0)$ is A gemotiveerd om p te laten stijgen. Voor $(1,0)$ kan dat niet meer, dus A heeft geen reden om zijn strategie te wijzigen in $(1,0)$. Voor $(0,0)$ kan dat nog wel, dus voor dat punt heeft A reden om zijn strategie te wijzigen. Evenzo vinden we dat A voor $(1,1)$ reden heeft om van strategie te veranderen en voor $(0,1)$ niet. Rustpunten voor A zijn dus $(1,0)$ en $(0,1)$.

3k, p. 131: $\partial\text{Payoff}_B(p,q)/\partial q = \partial(-4pq + p + 3q - 1)/\partial q = -4p + 3$. Rustpunten voor B zijn dus toevallig ook gelijk aan $(1,0)$ en $(0,1)$.

3l, p. 131: Om te beginnen zijn dat natuurlijk $(1,0)$ en $(0,1)$, omdat zowel A als B daar niet gemotiveerd zijn om van strategie te veranderen. Maar als er een punt (of JS) is waarbij de afgeleiden voor A en B van de verwachtte opbrengst gelijktijdig nul zijn, dan zijn in dat punt beide partijen ook niet gemotiveerd af te wijken van hun strategie.

$$\begin{aligned}&\begin{cases}\partial\text{Payoff}_A/\partial p = 0 \\ \partial\text{Payoff}_B/\partial q = 0\end{cases} \\ \Leftrightarrow\ &\begin{cases}-6q + 3 = 0 \\ -4p + 3 = 0\end{cases} \\ \Leftrightarrow\ &(p,q) = (3/4, 1/2)\end{aligned}$$

De verzameling van alle Nash-equilibria is dus gelijk aan $\{(0,1),(1,0),(3/4,1/2)\}$.

p_1	p_2	q_1	q_2	$\partial\text{Payoff}_A/\partial p_1$	$\partial\text{Payoff}_A/\partial p_2$	motief om te veranderen
0	0	0	0	-2	-2	**nee**
0	0	0	1	-1	2	ja, via p_2
0	0	1	0	-2	-1	**nee**
0	0	1	1	-1	3	ja, via p_2
0	1	0	0	-2	-2	ja, via p_2
0	1	0	1	-1	2	**nee**
0	1	1	0	-2	-1	ja, via p_2
0	1	1	1	-1	3	**nee**
1	0	0	0	-2	-2	ja, via p_1
1	0	0	1	-1	2	ja, via p_1 of p_2
1	0	1	0	-2	-1	ja, via p_1
1	0	1	1	-1	3	ja, via p_1 of p_2
1	1	0	0	-2	-2	ja, via p_1 of p_2
1	1	0	1	-1	2	ja, via p_1
1	1	1	0	-2	-1	ja, via p_1 of p_2
1	1	1	1	-1	3	ja, via p_1

Stabiele punten voor A.

4d, p. 132: De gemiddelde opbrengst van zowel a_3 als b_3 is voor beide spelers, bij uniforme exploratie, het hoogst, namelijk 3. Deze strategie zal dus op den duur het meest profijtelijk blijken. Het is wel belangrijk dat (a_3, b_3) een Nash-equilibrium is, anders hebben spelers op den duur door exploratie weer de neiging uit dit punt te willen weglopen. Met de huidige payoff-matrix levert exploratie buiten (a_3, b_3) voor beide partijen niets op, i.i.g. niets extras.

4e, p. 132: Het kan zijn dat spelers op een afwijkende, niet-uniforme, manier exploreren. Bv. als A en B in die 10 keer door stom toeval alleen acties $\{a_1, a_2, b_1, b_2\}$ en $\{a_3, b_3\}$ tegen elkaar uitspelen zullen er een andere gemiddelden ontstaan (te weten 4, 8 en 2), als gevolg waarvan beide spelers andere acties als optimaal zullen gaan beschouwen (te weten a_2 en b_2).

4f, p. 132: Vanwege hum kennis van de payoff matrix zijn ze niet zo onverstandig te starten in de "put" (a_3, b_3). Het is waarschijnlijk dat ze, om risico te vermijden, zullen starten in (a_2, b_2) en daar ook zullen blijven.

4g, p. 132: Als er zelf-vertrouwen is, en dat mag je toch verwachten van een computer-programma, op (a_1, b_1).

4h, p. 132: A hanteert strategie (p_1, p_2), waarbij p_1 en p_2 kansen zijn voor a_1 en a_2 en $1 - p_1 - p_2$ de kans is voor a_3. Randvoorwaarden: $0 \leq p_1, p_2, 1 - p_1 - p_2 \leq 1$ (dus $p_1 + p_2 \leq 1$).

"$(q1, q2) = (-6, 2)$ should be $(p1, p2) = (2, -6)$, I think. Although because of the symmetry of the matrix $(q1, q2) = (-6, 2)$ is probably also correct, but just not the answer that should be there in this context."

$$\begin{aligned} \text{Payoff}_A &= p_1(4q_1) + p_2(5q_1 + 3q_2) + (1 - p_1 - p_2)(6q_1 + q_2 + 2(1 - q_1 - q_2)) \\ &= p_1(q_2 - 2) + p_2(q_1 + 4q_2 - 2) + \text{nog wat termen zonder } p_1 \text{ en } p_2 \end{aligned}$$

Evenzo:

$$\text{Payoff}_B \;=\; q_1(p_2-2)+q_2(p_1+4p_2-2)+\text{nog wat termen zonder } q_1 \text{ en } q_2$$

We concentreren ons verder op de uitbetaling voor A. Speler A is indifferent als

$$\begin{aligned} & \begin{cases} \partial\text{Payoff}_A/\partial p_1 = 0 \\ \partial\text{Payoff}_A/\partial p_2 = 0 \end{cases} \\ \Leftrightarrow\; & \begin{cases} q_2 - 2 = 0 \\ q_1 + 4q_2 - 2 = 0 \end{cases} \\ \Leftrightarrow\; & (q_1, q_2) = (-6, 2) \end{aligned}$$

Dit zijn geen kansen, dus A bezit geen niet-extreem vlak punt. Omdat het spel symmetrisch is, B ook niet.

Gaan we kijken in de extrema. Voor een JS (p_1, p_2, q_1, q_2) zijn er daar $2^4 = 16$ van: $(0,0,0,0)$, $(0,0,0,1)$,... $(1,1,1,1)$.

Kijken we naar bovenstaande tabel, dan bezit A vier punten waar hij geen reden heeft om van strategie te veranderen:

$(0,0,0,0)$, $(0,0,1,0)$, $(0,1,0,1)$, en $(0,1,1,1)$.

Om reden van symmetrie bezit B ook vier punten waar hij geen reden heeft om van strategie te veranderen:

$(0,0,0,0)$, $(1,0,0,0)$, $(0,1,0,1)$, en $(1,1,0,1)$.

Daarvan zijn twee punten gemeenschappelijk, te weten $(0,0,0,0)$ en $(0,1,0,1)$. Deze twee punten corresponderen met de al eerder gevonden Nash-equilibria voor pure strategieën, waarin beide partijen gezamenlijk Actie 2 of gezamenlijk Actie 3 spelen. (Niet echt verrassend.)

5a, p. 132: Twee pure Nash-evenwichten dienen zich onmiddellijk aan, dat zijn $(p,q) = (1,0)$ en $(p,q) = (0,1)$. Een eventueel gemengd Nash-evenwicht is te vinden door het nut dat beide spelers vangen bij strategieprofiel (p,q) uit te rekenen.

$$\begin{aligned} U_1(p,q) &= 0pq + 2p(1-q) + 5(1-p)q + 1(1-p)(1-q) = 1 + p + 4q - 6pq \\ U_2(p,q) &= 0pq + 5p(1-q) + 2(1-p)q + 1(1-p)(1-q) = 1 + 4p + q - 6pq. \end{aligned}$$

Speler 1 (de rijspeler) kan zijn opbrengst alleen maar via p beïnvloeden. Evenzo voor Speler 2 en q. Bereken voor die grootheden dus de partiële afgeleiden.

$$\begin{aligned} \partial U_1(p,q)/\partial p &= 1 - 6q \\ \partial U_2(p,q)/\partial q &= 1 - 6p. \end{aligned}$$

Beide spelers zijn indifferent bij profiel $(p,q) = (1/6, 1/6)$. Daar is dus een gemengd evenwicht.

5b, p. 132: Twee pure Nash-evenwichten dienen zich onmiddellijk aan, dat zijn $(p,q) = (1,0)$ en $(p,q) = (0,1)$. Een eventueel gemengd Nash-evenwicht is te vinden door het nut dat beide spelers vangen bij strategieprofiel (p,q) uit te rekenen.

$$\begin{aligned} U_1(p,q) &= 0pq + -1p(1-q) + 1(1-p)q + -3(1-p)(1-q) = -3 + 2p + 4q - 3pq \\ U_2(p,q) &= 0pq + 1p(1-q) + -1(1-p)q + -3(1-p)(1-q) = -3 + 4p + 2q - 3pq. \end{aligned}$$

Speler 1 (de rijspeler) kan zijn opbrengst alleen maar via p beïnvloeden. Evenzo voor Speler 2 en q. Bereken voor die grootheden dus de partiële afgeleiden.

$$\partial U_1(p,q)/\partial p = 2 - 3q$$
$$\partial U_2(p,q)/\partial q = 2 - 3p.$$

Beide spelers zijn indifferent bij profiel $(p,q) = (2/3, 2/3)$. Daar is dus een gemengd evenwicht.

5c, p. 132: Twee pure Nash-evenwichten dienen zich onmiddellijk aan, dat zijn $(p,q) = (1,0)$ en $(p,q) = (0,1)$. Een eventueel gemengd Nash-evenwicht is te vinden door het nut dat beide spelers vangen bij strategieprofiel (p,q) uit te rekenen.

$$U_1(p,q) = -1pq + 2p(1-q) + 2(1-p)q + 0(1-p)(1-q) = 1 + p + q - 4pq$$
$$U_2(p,q) = 1pq + 2p(1-q) + 0(1-p)q + 0(1-p)(1-q) = 2p - pq.$$

Speler 1 (de rijspeler) kan zijn opbrengst alleen maar via p beïnvloeden. Evenzo voor Speler 2 en q. Bereken voor die grootheden dus de partiële afgeleiden.

$$\partial U_1(p,q)/\partial p = 1 - 4q$$
$$\partial U_2(p,q)/\partial q = -p.$$

Beide spelers zijn indifferent bij profiel $(p,q) = (0, 1/4)$. Daar is dus een gemengd evenwicht.

5d, p. 132: Eén puur Nash-evenwicht dient zich onmiddellijk aan, dat is $(p,q) = (0,1)$. Een eventueel gemengd Nash-evenwicht is te vinden door het nut dat beide spelers vangen bij strategieprofiel (p,q) uit te rekenen.

$$U_1(p,q) = -1pq + 0p(1-q) + 0(1-p)q + 2(1-p)(1-q) = 2 - 2p - 2q + pq$$
$$U_2(p,q) = -1pq + 1p(1-q) + 1(1-p)q + -3(1-p)(1-q) = -3 + 4p + 4q - 6pq.$$

Speler 1 (de rijspeler) kan zijn opbrengst alleen maar via p beïnvloeden. Evenzo voor Speler 2 en q. Bereken voor die grootheden dus de partiële afgeleiden.

$$\partial U_1(p,q)/\partial p = -2 + q$$
$$\partial U_2(p,q)/\partial q = 4 - 6p.$$

Beide spelers zouden indifferent zijn bij profiel $(p,q) = (2/3, 2)$, maar dit laatste is geen profile, want ligt buiten het eenheidsvierkant. Dus geen gemengde evenwichten.

5e, p. 132: Er zijn geen pure evenwichten. Een eventueel gemengd Nash-evenwicht is te vinden door het nut dat beide spelers vangen bij strategieprofiel (p,q) uit te rekenen.

$$U_1(p,q) = 0pq + 6p(1-q) + 2(1-p)q + 4(1-p)(1-q) = 4 + 2p - 2q - 4pq$$
$$U_2(p,q) = 3pq + 2p(1-q) + 0(1-p)q + 1(1-p)(1-q) = 1 + p - q + 2pq.$$

Speler 1 (de rijspeler) kan zijn opbrengst alleen maar via p beïnvloeden. Evenzo voor Speler 2 en q. Bereken voor die grootheden dus de partiële afgeleiden.

$$\partial U_1(p,q)/\partial p = 2 - 4q$$
$$\partial U_2(p,q)/\partial q = 2p - 1.$$

Beide spelers zijn indifferent bij profiel $(p,q) = (1/2, 1/2)$. Daar is dus een gemengd evenwicht.

5f, p. 132: Drie pure Nash-evenwichten dienen zich onmiddellijk aan, dat zijn $(p,q)=(0,0)$, $(p,q)=(0,1)$, $(p,q)=(1,1)$. Een eventueel gemengd Nash-evenwicht is te vinden door het nut dat beide spelers vangen bij strategieprofiel (p,q) uit te rekenen.

$$U_1(p,q) = pq + p(1-q) + (1-p)q + (1-p)(1-q) = 0$$
$$U_2(p,q) = pq + p(1-q) + (1-p)q + (1-p)(1-q) = p - pq.$$

Speler 1 (de rijspeler) kan zijn opbrengst alleen maar via p beïnvloeden. Evenzo voor Speler 2 en q. Bereken voor die grootheden dus de partiële afgeleiden.

$$\partial U_1(p,q)/\partial p = 0$$
$$\partial U_2(p,q)/\partial q = -p.$$

Dus $p=0$ en andere eisen zijn er niet. Beide spelers zijn dus indifferent op de verzameling $\{0\}\times[0,1]$. Daar zijn dus allemaal gemengde evenwichten.

6, p. 133: Voor $0\le p,q\le 1$ geldt:

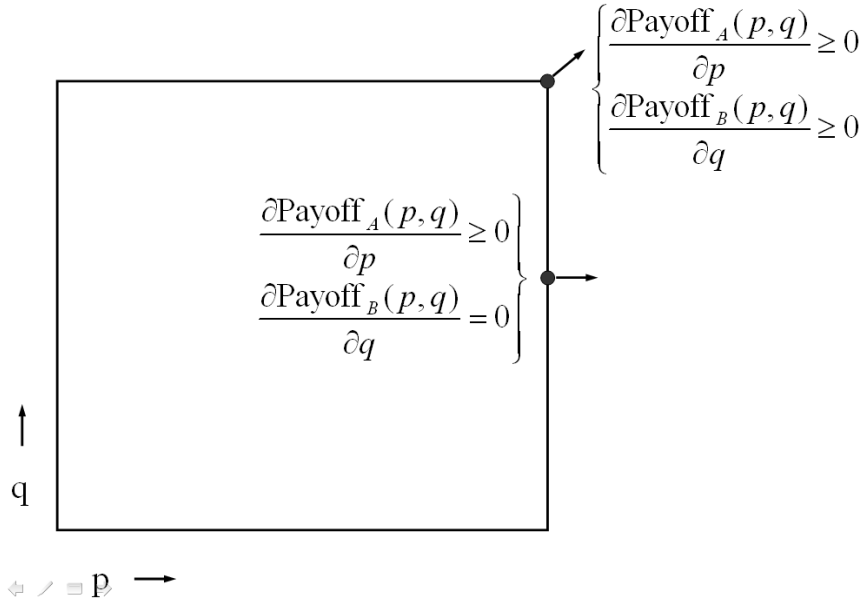
$$\begin{cases} \text{Payoff}_A = p(qR+(1-q)S)+(1-p)(qT+(1-q)P) \\ \text{Payoff}_B = q(pr+(1-p)s)+(1-q)(pt+(1-p)v) \end{cases}$$

Voor $0\le p,q\le 1$ geldt dus:

$$\begin{cases} \partial\text{Payoff}_A/\partial p = (R-S+P-T)q+S-P \\ \partial\text{Payoff}_B/\partial q = (r-s+v-t)p+s-v \end{cases}$$

Als expressies als $R-S+P-T$, etc. afgekort worden tot α_1, etc., dan besparen we onszelf een hoop schrijfwerk:

$$\begin{cases} \partial\text{Payoff}_A/\partial p = \alpha_1 q+\alpha_2 \\ \partial\text{Payoff}_B/\partial q = \beta_1 p+\beta_2 \end{cases}$$



Oneigenlijke equilibria op randen en hoeken.

Voor $0 \leq p, q \leq 1$ wordt een puur Nash-equilibrium verkregen als en slechts als

$$\begin{array}{rl} & \left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p = 0 \\ \partial \text{Payoff}_B / \partial q = 0 \end{array} \right. \\ \Leftrightarrow & \left\{ \begin{array}{l} \alpha_1 q + \alpha_2 = 0 \\ \beta_1 p + \beta_2 = 0 \end{array} \right. \\ \Leftrightarrow & (p, q) = (-\alpha_2/\alpha_1, -\beta_2/\beta_1) \end{array}$$

mits de uitgerekende waarden van p en q en beiden in $[0, 1]$ liggen en gedefinieerd zijn.
Dan oneigenlijke equilibria op randen en hoeken (cf. bovenstaande figuur). Eerst op hoeken.

Voor $(p, q) = (0, 0)$ wordt een hoekmaximum bereikt als

$$\left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p \leq 0 \\ \partial \text{Payoff}_B / \partial q \leq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot 0 + \alpha_2 \leq 0 \\ \beta_1 \cdot 0 + \beta_2 \leq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_2 \leq 0 \\ \beta_2 \leq 0 \end{array} \right.$$

Voor $(p, q) = (0, 1)$ wordt een hoekmaximum bereikt als

$$\left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p \leq 0 \\ \partial \text{Payoff}_B / \partial q \geq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot 1 + \alpha_2 \leq 0 \\ \beta_1 \cdot 0 + \beta_2 \geq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 + \alpha_2 \leq 0 \\ \beta_2 \geq 0 \end{array} \right.$$

Voor $(p, q) = (1, 0)$ wordt een hoekmaximum bereikt als

$$\left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p \geq 0 \\ \partial \text{Payoff}_B / \partial q \leq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot 0 + \alpha_2 \geq 0 \\ \beta_1 \cdot 1 + \beta_2 \leq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_2 \geq 0 \\ \beta_1 + \beta_2 \leq 0 \end{array} \right.$$

Voor $(p, q) = (1, 1)$ wordt een hoekmaximum bereikt als

$$\left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p \geq 0 \\ \partial \text{Payoff}_B / \partial q \geq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot 1 + \alpha_2 \geq 0 \\ \beta_1 \cdot 1 + \beta_2 \geq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 + \alpha_2 \geq 0 \\ \beta_1 + \beta_2 \geq 0 \end{array} \right.$$

Dan op randen:

Voor $p = 0$ wordt een randevenwicht bereikt als

$$\begin{array}{rl} & \left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p \leq 0 \\ \partial \text{Payoff}_B / \partial q = 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot q + \alpha_2 \leq 0 \\ \beta_1 \cdot 0 + \beta_2 = 0 \end{array} \right. \\ \Leftrightarrow & \beta_2 = 0 \text{ en } \left\{ \begin{array}{ll} q \leq -\alpha_2/\alpha_1 & \text{als } \alpha_1 > 0, \\ q \geq -\alpha_2/\alpha_1 & \text{als } \alpha_1 < 0, \\ q \in [0, 1] & \text{als } \alpha_1 = 0 \text{ en } \alpha_2 \leq 0 \end{array} \right. \end{array}$$

Voor $p = 1$ wordt een randevenwicht bereikt als

$$\begin{array}{rl} & \left\{ \begin{array}{l} \partial \text{Payoff}_A / \partial p \geq 0 \\ \partial \text{Payoff}_B / \partial q = 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot q + \alpha_2 \geq 0 \\ \beta_1 \cdot 1 + \beta_2 = 0 \end{array} \right. \\ \Leftrightarrow & \beta_1 + \beta_2 = 0 \text{ en } \left\{ \begin{array}{ll} q \geq -\alpha_2/\alpha_1 & \text{als } \alpha_1 > 0, \\ q \leq -\alpha_2/\alpha_1 & \text{als } \alpha_1 < 0, \\ q \in [0, 1] & \text{als } \alpha_1 = 0 \text{ en } \alpha_2 \geq 0 \end{array} \right. \end{array}$$

Voor $q = 0$ wordt een randevenwicht bereikt als

$$\begin{array}{rl} & \left\{ \begin{array}{l} \partial\text{Payoff}_A/\partial p = 0 \\ \partial\text{Payoff}_B/\partial q \leq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot 0 + \alpha_2 = 0 \\ \beta_1 \cdot p + \beta_2 \leq 0 \end{array} \right. \\ \Leftrightarrow & \alpha_2 = 0 \text{ en } \left\{ \begin{array}{ll} p \geq -\beta_2/\beta_1 & \text{als } \beta_1 > 0, \\ p \leq -\beta_2/\beta_1 & \text{als } \beta_1 < 0, \\ p \in [0,1] & \text{als } \beta_1 = 0 \text{ en } \beta_2 \leq 0 \end{array} \right. \end{array}$$

Voor $q = 1$ wordt een randevenwicht bereikt als

$$\begin{array}{rl} & \left\{ \begin{array}{l} \partial\text{Payoff}_A/\partial p = 0 \\ \partial\text{Payoff}_B/\partial q \geq 0 \end{array} \right. \Leftrightarrow \left\{ \begin{array}{l} \alpha_1 \cdot 1 + \alpha_2 = 0 \\ \beta_1 \cdot p + \beta_2 \geq 0 \end{array} \right. \\ \Leftrightarrow & \alpha_1 + \alpha_2 = 0 \text{ en } \left\{ \begin{array}{ll} p \leq -\beta_2/\beta_1 & \text{als } \beta_1 > 0, \\ p \geq -\beta_2/\beta_1 & \text{als } \beta_1 < 0, \\ p \in [0,1] & \text{als } \beta_1 = 0 \text{ en } \beta_2 \geq 0 \end{array} \right. \end{array}$$

Mits intervallen goed gedefinieerd en noemers ongelijk nul zijn.

De zin van deze berekeningen is dat we nu voor *alle* 2-persoons asymmetrische niet-nulsom spelen onmiddellijk kunnen aangeven waar de equilibria liggen, en dus onmiddellijk de optimale gemixte strategieën voor beide spelers kunnen geven. Dit is gedaan op

`https://ics.uu.nl/docs/vakken/ias/main.php?page=nash`.

7, p. 133: Row players' utility (or payoff, or profit) from playing this game when he plays T with probability p and column player plays L with probability q, is

$$U = pq \cdot x + p(1-q) \cdot -1 + (1-p)q \cdot -1 + (1-p)(1-q) \cdot 1.$$

The row player can influence his payoff only by varying his own strategy, p. He cannot change the strategy of his opponent, q. To see in which direction p must move to obtain a higher payoff, compute U's derivative with respect to p:

$$\begin{aligned} \frac{\partial U}{\partial p} &= qx - (1-q) + q + q - 1 \\ &= (x+3)q - 2. \end{aligned}$$

A saddle point occurs where U is stationary with respect to p, i.e., when

$$\begin{aligned} & \frac{\partial U}{\partial p} = 0 \\ \Leftrightarrow\ & (x+3)q - 2 = 0 \\ \Leftrightarrow\ & q = \frac{2}{x+3},\ x \neq -3. \end{aligned}$$

(Don't forget the "$x \neq -3$".) This stationary point q must be in $(0,1)$. For this to happen, x must satisfy

$$\begin{aligned} & 0 < \frac{2}{x+3} < 1,\ x \neq -3 \\ \Leftrightarrow\ & \left\{ \begin{array}{ll} 0 < 2 < x+3, & x > -3 \\ 0 > 2 > x+3, & x < -3 \quad \text{(impossible)} \end{array} \right. \\ \Leftrightarrow\ & -3 < -1 < x,\ x > -3 \\ \Leftrightarrow\ & -1 < x. \end{aligned}$$

Thus, there are saddlepoint(s) if and only if $x > -1$.

By the way, if $x = 1$, we obtain the normal coin game (cf. Flake) with $q = 1/2$.

8a, p. 133: Pareto-optimale strategie-profielen zijn

$$\{\ (a_1, b_1),\ (a_3, b_1),\ (a_1, b_3)\ \}$$

(Let op dat je strategie-profielen vermeld en geen uitbetalings-profielen.) De resterende strategie-profielen worden gedomineerd door deze drie Pareto-optimale profielen.

8b, p. 133: Pure Nash-equilibria zijn

$$\{\ (a_2, b_2),\ (a_3, b_3)\ \}$$

(Let weer op dat je strategie-profielen vermeld en geen uitbetalings-profielen.)

8c, p. 133: Alle uitbetalingen van a_1 worden, onafhankelijk van wat B doet, gedomineerd door uitbetalingen van a_2 (of uitbetalingen van a_3). Analoog worden alle uitbetalingen van b_1, onafhankelijk van wat A doet, gedomineerd door uitbetalingen van b_2 (of uitbetalingen van b_3).

8d, p. 133:

		Speler B	
		b_2	b_3
Speler A	a_2	3(3)	0(1)
	a_3	1(0)	2(2)

Laten we veronderstellen dat Speler A actie a_2 speelt met kans p en actie a_3 speelt met kans $1 - p$, en laten we op dezelfde manier veronderstellen dat Speler B actie b_2 speelt met kans q en actie b_3 speelt met kans $1 - q$. Er geldt:

$$\begin{cases} \text{Payoff}_A = p(qR + (1-q)S) + (1-p)(qT + (1-q)P) \\ \text{Payoff}_B = q(pr + (1-p)s) + (1-q)(pt + (1-p)v) \end{cases}$$

Voor $0 \leq p, q \leq 1$ geldt dus:

$$\begin{cases} \partial\text{Payoff}_A/\partial p = (R - S + P - T)q + S - P = 4q - 2 \\ \partial\text{Payoff}_B/\partial q = (r - s + v - t)p + s - v = 4p - 2 \end{cases}$$

De gradient (de richting waarin beide speler hun kansen willen aanpassen) is hier dus

$$(4q - 2, 4p - 2).$$

Deze gradient is alleen nul als $p = 0.5$ en $q = 0.5$. Dit is dus het enige gemixte equilibrium van de 2×2-matrix. Van de oorspronkelijke 3×3-matrix is het corresponderende gemixte equilibrium dus dat A met kans 0 actie a_1 speelt, met kans 0.5 actie a_2 speelt, en met kans 0.5 actie a_3 speelt, en dat tegelijkertijd B met kans 0 actie b_1 speelt, met kans 0.5 actie b_2 speelt, en met kans 0.5 actie b_3 speelt.

8e, p. 133: Geen enkel Nash-equilibrium is Pareto-optimaal.

9a, p. 133: Het uitbetalingsprofiel $(1,1)$ wordt gedomineerd door zowel $(2,5)$ als $(4,4)$. Het uitbetalingsprofiel $(3,-1)$ wordt gedomineerd door $(4,4)$. De uitbetalingsprofielen $(2,5)$ en $(4,4)$ worden niet gedomineerd door andere uitbetalingsprofielen. Het Pareto-front is dus gelijk aan $\{(2,5),(4,4)\}$.

9b, p. 133: Pure Nash evenwichten: $\{(0,0),(1,1)\}$. Dus beiden spelen hun eerste actie of beiden spelen hun tweede actie. In deze scenario's hebben beiden partijen geen behoefte om hun actie te wijzigen.

9c, p. 133: Nash evenwicht: $\{(0,0),(1/3,1/2),(1,1)\}$. (Afgeleiden payoffs nul stellen en oplossen.)

9d, p. 133: De uitbetalingsprofielen $(1,1)$ en $(3,-1)$ worden gedomineerd door $(4,4)$. Het uitbetalingsprofiel $(x,5)$ wordt niet gedomineerd door het uitbetalingsprofiel $(4,4)$. (Daar zorgt de "5" wel voor.) Het uitbetalingsprofiel $(4,4)$ wordt gedomineerd door $(x,5)$ als en slechts als $x > 4$.

Het Pareto-front is dus gelijk aan:

$$\begin{cases} \{(x,5),(4,4)\} & \text{als } x \leq 4, \\ \{(x,5)\} & \text{anders.} \end{cases}$$

9e, p. 134:

$$\begin{aligned} U_{\text{rij}} \quad (p,q) &= xpq + 3p(1-q) + (1-p)q + 4(1-p)(1-q), \\ U_{\text{kolom}}\,(p,q) &= 5pq - 1p(1-q) + (1-p)q + 4(1-p)(1-q). \end{aligned}$$

Dus

$$\begin{aligned} \tfrac{1}{\partial p} U_{\text{rij}} \quad (p,q) &= qx - 1, \\ \tfrac{1}{\partial q} U_{\text{kolom}}\,(p,q) &= 9p - 3. \end{aligned}$$

Er is dus een gemengd evenwicht

$$\left(\frac{1}{x}, \frac{1}{3}\right)$$

als en slechts als $x \geq 1$. Als $x < 1$, dan is alleen $(p,q) = (0,0)$ een evenwicht, en dit evenwicht is puur. Merk op dat $(p,q) = (0,0)$ correspondeert met het actieprofiel (a_3, b_3), dus met de laatste twee acties. De kans dat de rij-speler de eerste actie speelt is immers $p = 0$. Evenzo voor de kolom-speler met $q = 0$.

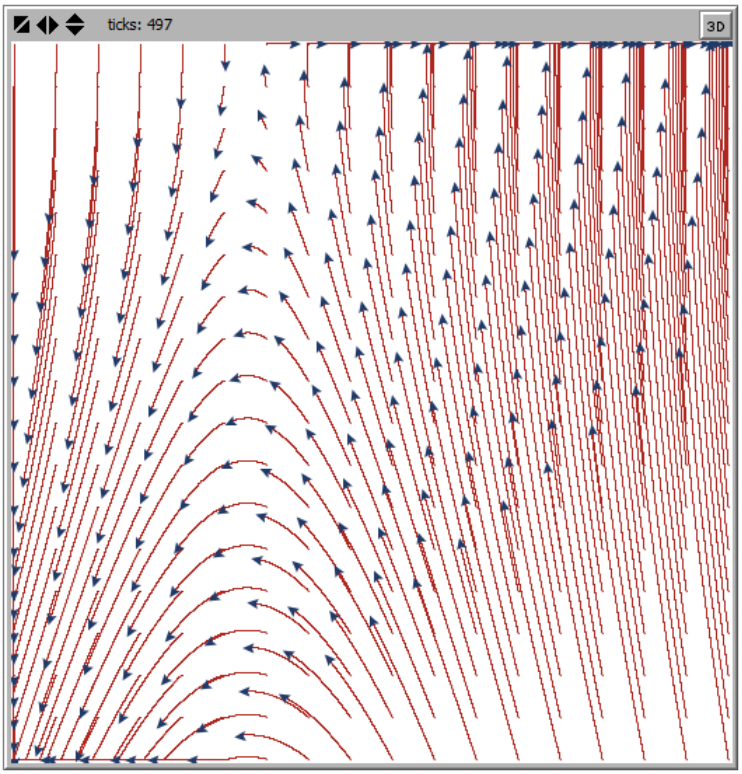
Antwoord:

$$\begin{cases} \{(0,0),(1/3,1/x),(1,1)\} & \text{als } x \geq 1, \\ \{(0,0)\} & \text{anders.} \end{cases}$$

9f, p. 134:

$$\begin{cases} \partial \text{Payoff}_A / \partial p = q - 1 \\ \partial \text{Payoff}_B / \partial q = 9p - 3 \end{cases}$$

9g, p. 134: Gradient is alleen nul bij $(1/3, 1)$. Aan hoeken en randen is de gradient i.h.b. ongelijk aan de nulvector.



10a, p. 134: Ja. De som van de uitbetalingen zijn voor elk actieprofiel nul.

10b, p. 134: Omdat A dan voorspelbaar is en B dat zal ontdekken en vervolgens de hele tijd "munt" op zal gooien.

10c, p. 134: Omdat B dan voorspelbaar is en A dat zal ontdekken en vervolgens de hele tijd ook "kop" zal gooien.

10d, p. 134: Speler B kan in de volgende ronde het beste kiezen voor actie "kop". Zo heeft hij 95% kans op een uitbetaling van 1.

10e, p. 134: Speler A zal vaker "kop" gaan kiezen.

10g, p. 134: Speler A bezit $7 - 13 = -6$ nutseenheden. Speler B bezit (dus) $13 - 7 = 6$ nutseenheden.

10h, p. 135: Speler A heeft precies 10 van de 20 keer voor "kop" gekozen, dus B zal een willekeurige actie spelen.

10i, p. 135: Als A in ronde $n+1$ "munt" speelt, dan

$$p = \frac{h_A^n}{h_A^n + t_A^n + 1} = \frac{np + 0}{n+1}.$$

10j, p. 135: Als A in ronde $n+1$ "kop" speelt, dan

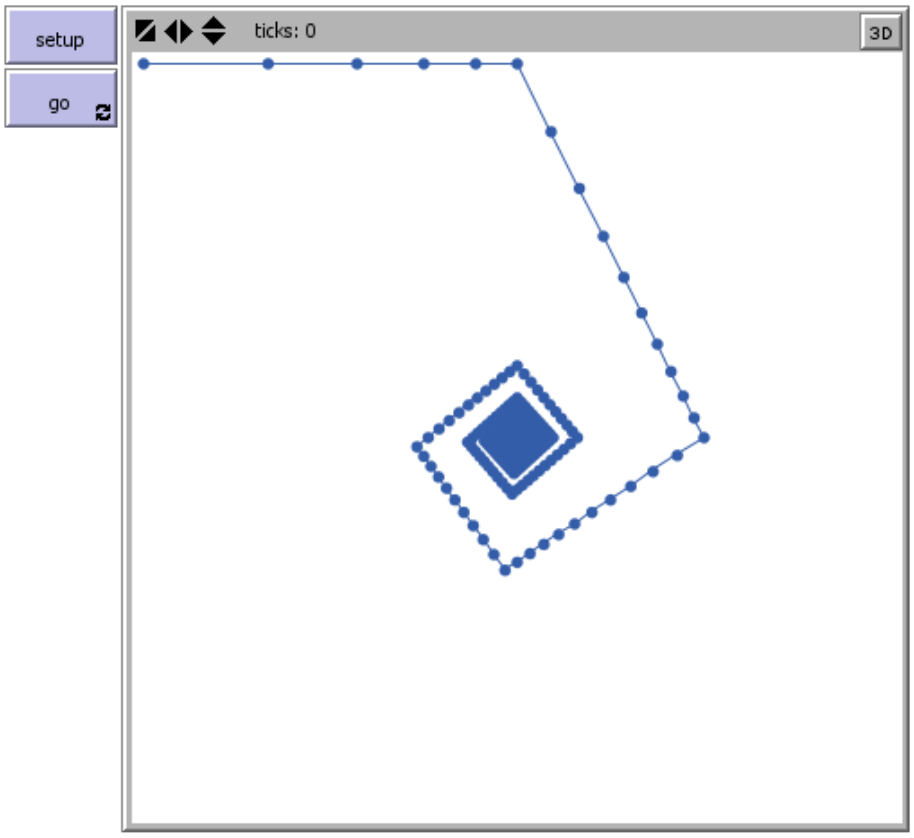
$$p = \frac{h_A^n + 1}{h_A^n + t_A^n + 1} = \frac{np + 1}{n+1}.$$

10k, p. 135: Als (p, q) in zó is dat $p > 1/2$ en $q > 1/2$, dan hebben A en B in de afgelopen n ronden overwegend "kop" gespeeld. In de volgende ronde zullen A en B dus respectievelijk "kop" en "munt" spelen. De nieuwe waarde van p wordt dan $(np + 1)/(n + 1) > p$. De nieuwe waarde van q wordt $nq/(n + 1) < q$.

10l, p. 135: Richting van pijlen:

- $p > 1/2$ en $q > 1/2$: een pijl met richting tussen 270^0 en 360^0.
- $p > 1/2$ en $q < 1/2$: een pijl met richting tussen 180^0 en 270^0.
- $p < 1/2$ en $q < 1/2$: een pijl met richting tussen 90^0 en 180^0.
- $p > 1/2$ en $q < 1/2$: een pijl met richting tussen 0^0 en 90^0.

10m, p. 135: In ieder geval cirkel-vormig of spiraal-vormig met de klok mee. Bij wat beter doorrekenen blijkt het spoor van strategie-profielen spiraal-vormig met de klok mee te zijn, zie onderstaande figuur. Dit is een screendump van een Netlogo-simulatie van 20 regels groot. Deze simulatie is gestart met $n = 5$, $h_A = 0$ en $h_B = 5$.



De dynamiek van strategieën bij matching pennies.

Hier is de code van deze simulatie.

```
turtles-own [ a-plays-h b-plays-h rounds ]

to setup
```

```
  ca
  ask patches [ set pcolor white ]
  crt 1 [
    set color blue set shape "circle" set size 0.5
    set a-plays-h 0 set b-plays-h 5 set rounds 5
    placeme pd
  ]
end

to go
  ask turtle 0 [
    if (a-plays-h / rounds) < 0.5 [ set b-plays-h b-plays-h + 1 ]
    if (b-plays-h / rounds) > 0.5 [ set a-plays-h a-plays-h + 1 ]
    set rounds rounds + 1
    placeme
  ]
end

to placeme
  setxy (a-plays-h / rounds) * max-pxcor (b-plays-h / rounds) * max-pycor
  hatch 1
end
```

1, p. 136: Ontwikkeling:

	5	1	1	1	...	1
All-D	D	D	D	D	...	D
Unforgiving	C	D	D	D	...	D
	0	1	1	1	...	1

In die 100 ronden verdient All-D $5 + 99 \times 1 = 104$, en Unforgiving $0 + 99 \times 1 = 99$. De gemiddelde opbrengst is dus respectievelijk $1004/100 = 1.04$ en $99/100 = 0.99$.

2, p. 136: Drie equivalente definities van Pavlov:

1. Wissel van strategie a.e.s.a. je tegenstander in de vorige ronde verzaakte.
2. Wissel van strategie a.e.s.a. jij de vorige ronde S (Sucker) of P (Punishment) was.
3. Werk samen a.e.s.a. zowel jij als de tegenstander in de vorige ronde dezelfde strategie hanteerden.

Dus

	0	1	0	1	...	0	1
Pavlov	C	D	C	D	...	C	D
All-D	D	D	D	D	...	D	D
	5	1	5	1	...	5	1

In die 100 ronden verdient Pavlov $50 \times (0+1) = 50$, en All-D $50 \times (5+1) = 300$. De gemiddelde opbrengst is dus respectievelijk $50/100 = 1/2$ en $300/100 = 3$.

3, p. 136: Dat is 2.2425. Elke ronde speelt RAND actie C met kans 1/2. Deze acties worden ge-echoot door TFT in elke volgende ronde. In elke ronde zijn de acties van RAND en TFT onafhankelijk. Immers, RAND genereert elke volgende een nieuwe actie. Tegelijkertijd weten we dat de acties van TFT, als echo van RAND, evenredig verdeeld zijn over {C,D}. Dus elk van de vier actieprofielen komt door de bank genomen even vaak voor. De enige uitzondering is de eerste ronde, omdat TFT daar samenwerkt. Dus daar is de verwachte opbrengst $(3+0)/2 = 1.5$. Voor de overige 99 ronden is de totaal verwachte opbrengst voor TFT $99 \times (0+1+3+5)/4$.

De verwachte gemiddelde opbrengst voor TFT na 100 ronden is dus

$$\frac{1\frac{1}{2} + 99 \times \frac{0+1+3+5}{4}}{100} = 2.2425.$$

4a, p. 136: Pavlov werkt samen als (en slechts als) beide spelers in de vorige ronde dezelfde actie speelden; TFT echoot de tegenstander. We krijgen dan:

1.	D	C	D	D	...
	1	0	5	1	...
2.	D	D	C	D	...
	1	5	0	1	...

Dus na drie ronden begin alles zich te herhalen. In een cyclus verdienen beide partijen zes punten. Er gaan $\lfloor 1000/3 \rfloor = 333$ cycli in 1000 ronden en in ronde 1000 wordt dan (D, D) gespeeld, met opbrengst 1 voor beiden. Voor beiden is de totale opbrengst dus $333 \times 6 + 1 = 1999$.

4b, p. 136: Naar verwachting wordt elk actieprofiel even vaak gespeeld, dus naar verwachting is de gemiddelde opbrengst per ronde $(0+1+3+5)/4 = 2.25$. Dus ook na 1000 ronden. Dus voor beiden is de verwachte totale opbrengst na 1000 ronden gelijk aan $1000 \times 2.25 = 2250$.

4c, p. 136: Na elke dubbel-D reageert TF2T met een D. Wat is de kans op een dubbel-D in een random-rij met door de bank genomen evenveel C's als D's?. Wel, aan het begin van zo'n random rij is die kans i.i.g. nul. Laten we op een willekeurig ander punt staan. Daar is die kans 1/4. Immers, eerst moet dat punt zelf een D zijn, maar de voorgaande actie moet ook een D zijn. Dus is het verwachte aantal dubbel D's in een rij van 1000 gelijk aan $1 \cdot 0 + 999 \cdot 1/4$. Leuk om te weten, en het helpt zeker ook bij de verdere beantwoording van dit onderdeel, maar naar dat getal zijn we niet op zoek. Immers TF2T kan alleen reageren in de 3e t/m de 1000e ronde. (Ga na!)

Laten we daarom maar eerst de eerste twee ronden bekijken, en daar het aantal verwachte voorkomens van de actieprofielen (C, C), (C, D), (D, C), en (D, C) bepalen, dat is het makkelijkst. Meteen is in te zien dat, in die twee eerste ronden, de profielen met een D in de tweede coördinaat niet voorkomen, omdat Speler 2, die TF2T uitvoert, dan altijd samenwerkt. De kans dat (C, C) in Ronde 1 gespeeld wordt is gelijk aan 1/2, omdat Speler 1 in elke situatie 50% RANDOM speelt. Dit geldt ook voor Ronde 2. Dus het aantal verwachte voorkomens van profiel (C, C) in de eerste twee ronden is $2 \cdot 1/2$. Analoog voor profiel (D, C), het aantal verwachte voorkomens van dat profiel in de eerste twee ronden is ook $2 \cdot 1/2$. Deze gegevens zijn opgenomen in de tabel hieronder.

Laten we nu naar de resterende 998 ronden kijken. De kans dat RANDOM-50% daarin een dubbel-D speelt is, vanwege de redenering in de eerste alinea, gelijk aan 1/4. De kans dat TF2T

met D reageert is dus ook 1/4. In die situatie doet RANDOM-50% gewoon weer z'n eigen ding. Met kans 1/2 speelt RANDOM-50% dus C. Dus is de kans op profiel (C, D) in elk van de resterende 998 ronden gelijk aan $1/2 \cdot 1/4 = 1/8$. Analoog voor profiel (D, D), de kans dat dat profiel gespeeld wordt in elk van de resterende 998 ronden is om dezelfde reden ook 1/8. Dus het aantal verwachte voorkomens van de actieprofielen (C, D) en (D, D) in de resterende 998 ronden is voor beide profielen gelijk aan $998 \cdot 1/8$. Dezelfde redenering gaat op voor de profielen (C, C) en (D, C), zij het dat de kans dat TF2T een C speelt gelijk aan 3/4 is. Daarmee is het aantal verwachte voorkomens van de actieprofielen (C, C) en (D, C) in de resterende 998 ronden gelijk aan $998 \cdot 3/8$. Deze gegevens zijn opgenomen in de tabel hieronder.

Rest alle verwachte frequenties te vermenigvuldigen met de uitbetalingsprofielen en de resultaten bij elkaar op te tellen:

actieprofiel	*verwachte frequentie a.p.*	*uitbetalingsprofiel*	*verwachte uitbetaling*	
(C, C)	$2 \cdot \frac{1}{2} + 998 \cdot \frac{3}{8} = 375.25$	$(3, 3)$	$(1125.75, \quad 1125.75)$	
(C, D)	$2 \cdot 0 + 998 \cdot \frac{1}{8} = 124.75$	$(0, 5)$	$(0 \quad , \quad 623.75)$	
(D, C)	$2 \cdot \frac{1}{2} + 998 \cdot \frac{3}{8} = 375.25$	$(5, 0)$	$(1876.25, \quad 0 \quad)$	
(D, D)	$2 \cdot 0 + 998 \cdot \frac{1}{8} = 124.75$	$(1, 1)$	$(124.75, \quad 124.75)$	+
			$(3126.75, \quad 1874.25)$	

Voor random 50% is de verwachte totale opbrengst na 1000 ronden dus gelijk aan 3126.75, en voor TF2T is de verwachte totale opbrengst na 1000 ronden dus gelijk aan 1874.25.

5a, p. 136: De uitbetalingsmatrix van het gevangen-dilemma is gelijk aan.

	C	D
C	(3, 3)	(0, 5)
D	(5, 0)	(1, 1)

NE $= \{(D, D)\}$, PO $= \{(C, C), (C, D), (D, C)\}$.

De uitbetalingsmatrix voor het SIEPD is gelijk aan

	C	D
C	(1, 1)	(0, α)
D	(α, 0)	(0, 0)

Wil D een aantrekkelijk alternatief zijn, dan moet in ieder geval $\alpha > 1$. De Pareto-optima vallen dan samen met de Pareto-optima van het gevangen-dilemma. Echter, door de nullen in de uitbetalingsmatrix is (D, D) niet het enige NE meer, maar zijn (C, D) en (D, C) dat ook. (Ga na!) Dus voor geen enkele $\alpha \geq 0$ is de uitbetalingsmatrix voor het SIEPD qua Pareto-optima en Nash-equilibria gelijk aan de uitbetalingsmatrix van het gevangenen-dilemma.

Een half punt per item:

- Opmerken dat, wil D een aantrekkelijk alternatief zijn, dan moet in ieder geval $\alpha > 1$.
- Het identificeren van de Pareto-optima in de *alpha*-matrix.
- Het identificeren van de Nash-evenwichten in de *alpha*-matrix als $\alpha > 1$.
- Het constateren dat, voor elke $\alpha \geq 0$, de Nash-equilibria in de uitbetalingsmatrix voor het SIEPD niet samenvallen met de Nash-equilibria in de uitbetalingsmatrix van het gevangen-dilemma.

5b, p. 137: De opbrengst voor de centrumcel is $6\alpha + 2 \cdot 0 = 6 \cdot 1.2 = 7.2$. Het paar $U_{\max}$ is gelijk aan $(C, 7.3)$. Er geldt $7.3 > 7.2$ dus de centrumcel gaat meewerken in de volgende generatie.

De opbrengst voor de tweede centrumcel is $5 \cdot 1 + 3 \cdot 0 = 5$. Het paar $U_{\max}$ is gelijk aan $(D, 7)$. Er geldt $7 > 5$ dus de centrumcel gaat verzaken in de volgende generatie.

6, p. 137: We maken de volgende tabel:

	1 IPD	2 IEPD	4 SIEPD	8 SSIEPD	16 CSIEPD	Σ	Σ′
handig een grand table te hebben		√		√		10	
TFT is een mogelijke strategie	√	√		√		11	
het aantal deelnemers is eindig	√		♢	♢	♢	29	(1)
deelnemers bezitten een omgeving			√	√	√	28	(29)
deeln. kunnen toestanden aannemen	$		√	√	√	28	
deeln. in vier mogelijke toestanden			†			0	(4)
de populatie is homogeen	‡	√				2	(3)
oneindig veel mogelijke strategieën					√	16	

♢ Zowel eindig (implementatie) als oneindig (theorie) wordt goed gerekend.

$ In het IPD hoeven deelnemers geen toestanden te bezitten: ze lezen de situatie af uit een extern bijgehouden gezamenlijke geschiedenis van zetten, in ronde 7 bijvoorbeeld

	1	2	3	4	5	6	
Speler 1	C	D	C	C	D	C	
Speler 2	C	C	C	D	C	C	.

Theoretisch gesproken kunnen deelnemers deze geschiedenis ook zelf, i.e. intern, bijhouden. In dat geval is het aantal toestanden dat deelnemers kunnen aannemen oneindig. Maar deze theoretische representatie is wat mij betreft een beetje ver gezocht.

† Strict genomen twee toestanden: C of D. Aan de andere kant kan een kleur, als representatie van een vorige en een huidige toestand, ook als toestand worden aangemerkt. Er zijn vier kleuren, te weten rood, groen, geel en blauw, dus dat zou betekenen dat er vier toestanden zijn.

‡ Van tegenstanders in het IPD kan in zekere zin ook worden beweerd dat ze met iedereen interacteren. Aan de andere kant wordt hier niet van een populatie gesproken.

Rechtvaardigingen per rij:

Rij 1: een grand table is alleen handig als de payoffs tussen meerdere strategieën zoals All-C, All-D, TFT, TF2T, en Pavlov moeten worden uitgerekend. Dat is alleen het geval bij de replicator dynamic (IEPD) en bij de CA-versie daarvan (SSIEPD).

Rij 2: bij het SIEPD is er alleen een toestand met waarde C of D, bij het CSIEPD is zijn er oneindig veel toestanden $I \geq 0$—maar geen strategeieën zoals TFT.

Rij 3: de replicatorvergelijking werkt met proporties (percentages) i.p.v. met individuen.

Rij 4: uiteraard alleen het geval bij cellulaire automaatversies van het gevangenendilemma.

Rij 5: een toestand is een staat van zijn, d.w.z. een configuratie van iets of iemand op een bepaald tijdstip. Bij het IPD hoeven deelnemers niet te onthouden wat er is gebeurd, i.e., ze hoeven de geschiedenis te onthouden. Deze is extern en ligt in het spelverloop. Bij het IEPD zijn er überhaupt geen individueel onderscheidbare deelnemers. IN CA-versies nemen deelnemers altijd toestanden aan. Bij SIEPD C of D, bij SSIEPD één uit, zeg, All-C, All-D, TFT, TF2T, en Pavlov, bij CSIEPD een reëel getal $I \geq 0$.

Rij 6: Zie †. Een blank wordt ook goedgekeurd bij SIEPD?

Rij 7: Zie ‡. Bij de CA-varianten is het duidelijk dat iedereen slechts met acht anderen, en dus niet alle anderen, acteert.

Rij 8: Alleen voor de continue variant van SIEPD kan een deelnemer kiezen uit oneindig veel mogelijke strategieën: één voor elke $I \geq 0$. Bij de overige varianten is dat niet zo, omdat daar de strategieën vaststaan (IPD) of uit een eindig arsenaal worden gekozen, typisch uit All-C, All-D, TFT, TF2T, Pavlov, etc.

1a, p. 138: Als $P_i = 0$, dan ook $P_i S_i = 0$. Dus ook $P_i{}^{\text{nieuw}} = P_i S_i / \cdots = 0$.

1b, p. 138: Als alle $P_j > 0$, dan $S_i = \sum_{j=1}^{n} P_j R_{ij} > 0$ en dus $P_i S_i > 0$ en dus $P_i{}^{\text{nieuw}} > 0$.

1c, p. 138: Relatieve score matrix en initiële proporties:

$$R = \begin{pmatrix} 0 & 1 & 3 \\ 2 & 0 & 1 \\ 1 & 3 & 0 \end{pmatrix}, \qquad P = \begin{pmatrix} 0.2 \\ 0.5 \\ 0.3 \end{pmatrix}.$$

Score per soort:

$$R \cdot P = \begin{pmatrix} 0 & 1 & 3 \\ 2 & 0 & 1 \\ 1 & 3 & 0 \end{pmatrix} \begin{pmatrix} 0.2 \\ 0.5 \\ 0.3 \end{pmatrix} = \begin{pmatrix} 1.4 \\ 0.7 \\ 1.7 \end{pmatrix}.$$

Ongenormaliseerde nieuwe proporties:

$$Q = \begin{pmatrix} 0.2 \times 1.4 \\ 0.5 \times 0.7 \\ 0.3 \times 1.7 \end{pmatrix} = \begin{pmatrix} 0.28 \\ 0.35 \\ 0.51 \end{pmatrix}.$$

Genormaliseerde nieuwe proporties:

$$P = \frac{1}{\sum Q} Q = \frac{1}{0.28 + 0.35 + 0.51} \begin{pmatrix} 0.28 \\ 0.35 \\ 0.51 \end{pmatrix} = \begin{pmatrix} 0.246 \\ 0.307 \\ 0.447 \end{pmatrix}.$$

1d, p. 139: Relatieve score matrix en initiële proporties:

$$R = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 0 & 1 \\ 3 & 4 & 0 \end{pmatrix}, \qquad P = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.7 \end{pmatrix}.$$

Score per soort:

$$R \cdot P = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 0 & 1 \\ 3 & 4 & 0 \end{pmatrix} \begin{pmatrix} 0.1 \\ 0.2 \\ 0.7 \end{pmatrix} = \begin{pmatrix} 2.6 \\ 1.0 \\ 1.1 \end{pmatrix}.$$

Ongenormaliseerde nieuwe proporties:

$$Q = \begin{pmatrix} 0.1 \times 2.6 \\ 0.2 \times 1.0 \\ 0.7 \times 1.1 \end{pmatrix} = \begin{pmatrix} 0.26 \\ 0.20 \\ 0.77 \end{pmatrix}.$$

Genormaliseerde nieuwe proporties:

$$P = \frac{1}{\sum Q} Q = \frac{1}{0.26 + 0.20 + 0.77} \begin{pmatrix} 0.26 \\ 0.20 \\ 0.77 \end{pmatrix} = \begin{pmatrix} 0.211 \\ 0.163 \\ 0.626 \end{pmatrix}.$$

2a, p. 139: Relatieve score matrix en initiële proporties:

$$R = \begin{pmatrix} 0 & 1 & 2 \\ 2 & 1 & 1 \\ 1 & 2 & 0 \end{pmatrix}, \qquad P = \begin{pmatrix} 0.8 \\ 0.1 \\ 0.1 \end{pmatrix}.$$

Score per soort:

$$R \cdot P = \begin{pmatrix} 0 & 1 & 2 \\ 2 & 1 & 1 \\ 1 & 2 & 0 \end{pmatrix} \begin{pmatrix} 0.8 \\ 0.1 \\ 0.1 \end{pmatrix} = \begin{pmatrix} 0.3 \\ 1.8 \\ 1.0 \end{pmatrix}.$$

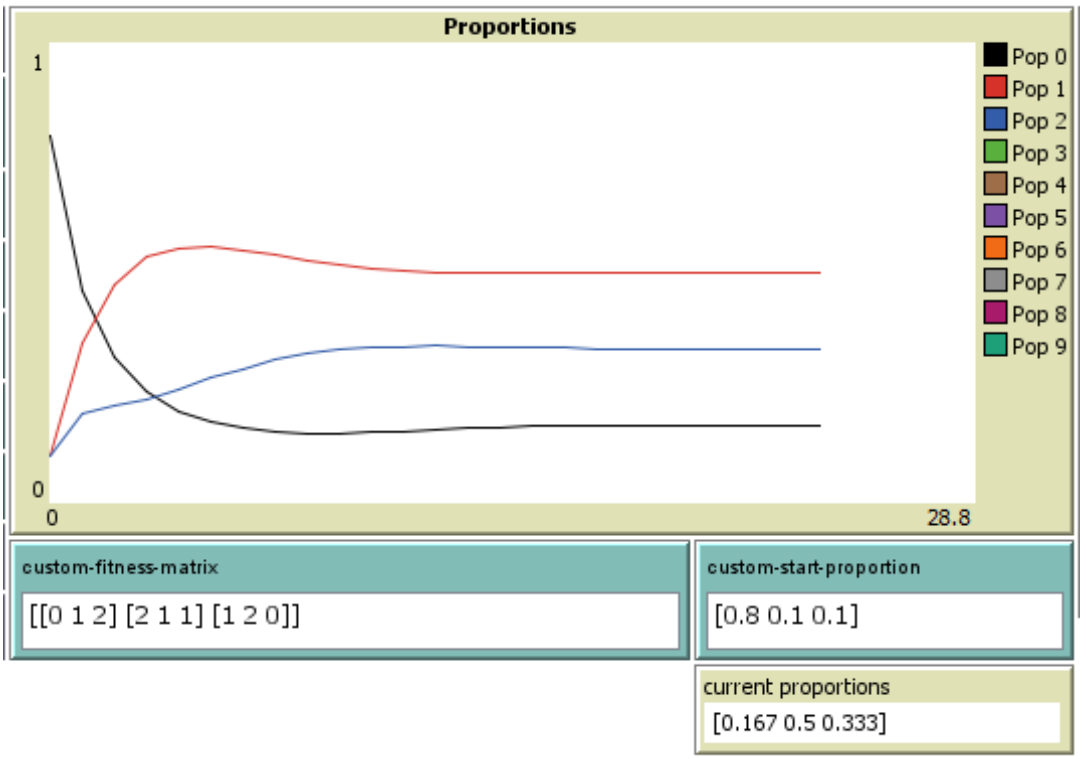
Ongenormaliseerde nieuwe proporties:

$$Q = \begin{pmatrix} 0.8 \times 0.3 \\ 0.1 \times 1.8 \\ 0.1 \times 1.0 \end{pmatrix} = \begin{pmatrix} 0.24 \\ 0.18 \\ 0.10 \end{pmatrix}.$$

Genormaliseerde nieuwe proporties:

$$P = \frac{1}{\sum Q} Q = \frac{1}{0.24 + 0.18 + 0.10} \begin{pmatrix} 0.24 \\ 0.18 \\ 0.10 \end{pmatrix} = \begin{pmatrix} 0.46 \\ 0.35 \\ 0.19 \end{pmatrix}.$$

2b, p. 139: $P^* = (1/6, 1/2, 1/3)$:



3b, p. 139: Voor soort 1:

$$0.1 \times R_{11} + 0.4 \times R_{12} + 0.5 \times R_{13} = 0.1 \times 1 + 0.4 \times 3 + 0.5 \times 1 = 0.1 + 1.2 + 0.5 = 1.8.$$

Voor soort 2:

$$0.1 \times R_{21} + 0.4 \times R_{22} + 0.5 \times R_{23} = 0.1 \times 1 + 0.4 \times 2 + 0.5 \times 3 = 0.1 + 0.8 + 1.5 = 2.4.$$

Voor soort 3:

$$0.1 \times R_{31} + 0.4 \times R_{32} + 0.5 \times R_{33} = 0.1 \times 4 + 0.4 \times 1 + 0.5 \times 3 = 0.4 + 0.4 + 1.5 = 2.3.$$

3c, p. 140:

$$\begin{aligned} E\{\text{opbrengst}_i\} = \\ E\{\text{opbrengst}_i \mid \text{treft } 1\}P\{\text{treft } 1\} + \cdots + E\{\text{opbrengst}_i \mid \text{treft } n\}P\{\text{treft } n\} \\ = R_{i1}p_1 + \cdots + R_{in}p_n = \sum_{j=1}^{n} p_j R_{ij}. \end{aligned}$$

3d, p. 140:

$$\begin{cases} f_1 = p_1R_{11} + p_2R_{12} + p_3R_{13} \\ f_2 = p_1R_{21} + p_2R_{22} + p_3R_{23} \\ f_3 = p_1R_{31} + p_2R_{32} + p_3R_{33} \end{cases} \quad \Rightarrow \quad \begin{cases} f_1 = p_1 + 3p_2 + p_3 \\ f_2 = p_1 + 2p_2 + 3p_3 \\ f_3 = 4p_1 + p_2 + 3p_3 \end{cases}$$

Je kunt zelfs matrixrekening gebruiken:

$$f = Rp = \begin{pmatrix} R_{11} & \dots & R_{1n} \\ \vdots & \ddots & \vdots \\ R_{n1} & \dots & R_{nn} \end{pmatrix} \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix} = \begin{pmatrix} 1 & 3 & 1 \\ 1 & 2 & 3 \\ 4 & 1 & 3 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = \begin{pmatrix} p_1 + 3p_2 + p_3 \\ p_1 + 2p_2 + 3p_3 \\ 4p_1 + p_2 + 3p_3 \end{pmatrix}.$$

3e, p. 140: Als $f_i(t) = -1 - \beta$ dan

$$q_i(t+1) = q_i(t)[1 + \beta + f_i(t)] = q_i(t)[1 + \beta - 1 - \beta] = q_i(t) \times 0 = 0.$$

Stel omgekeerd dat $q_i(t+1) = 0$ terwijl $q_i(t) > 0$. Dan moet de factor $1 + \beta + f_i(t)$ gelijk nul zijn en dus $f_i(t) = -1 - \beta$. Dus het is echt "als en slechts als".

3f, p. 140:

$$\begin{aligned}\Delta q_i(t) &= q_i(t+1) - q_i(t) \\ &= q_i(t)[1 + \beta + f_i(t)] - q_i(t) \\ &= q_i(t)[\beta + f_i(t)].\end{aligned}$$

3g, p. 140:

$$\bar{f}(t) = p_1 f_1(t) + \cdots + p_n f_n(t) = 0.1 \times 1.8 + 0.4 \times 2.4 + 0.5 \times 2.3 = 2.29.$$

3h, p. 140: In kleine stapjes:

$$\begin{aligned}p_i(t+1) &= \frac{q_i(t+1)}{\sum_{j=1}^n q_j(t+1)} \\ &= \frac{q_i(t)[1 + \beta + f_i(t)]}{\sum_{j=1}^n q_j(t)[1 + \beta + f_j(t)]} \\ &= \frac{\frac{1}{q(t)} q_i(t)[1 + \beta + f_i(t)]}{\frac{1}{q(t)} \sum_{j=1}^n q_j(t)[1 + \beta + f_j(t)]} \\ &= \frac{p_i(t)[1 + \beta + f_i(t)]}{\sum_{j=1}^n p_j(t)[1 + \beta + f_j(t)]} \\ &= \frac{p_i(t)[1 + \beta + f_i(t)]}{\sum_{j=1}^n p_j(t) + \beta \sum_{j=1}^n p_j(t) + \sum_{j=1}^n p_j(t) f_j(t)]} \\ &= \frac{p_i(t)[1 + \beta + f_i(t)]}{1 + \beta + \bar{f}(t)} \\ &= p_i(t) \frac{1 + \beta + f_i(t)}{1 + \beta + \bar{f}(t)}.\end{aligned}$$

3i, p. 140:

$$\begin{aligned}& p_i(t+1) > p_i(t) \\ \Leftrightarrow \quad & p_i(t) \frac{1 + \beta + f_i(t)}{1 + \beta + \bar{f}(t)} > p_i(t) \\ \Leftrightarrow \quad & 1 + \beta + f_i(t) > 1 + \beta + \bar{f}(t), && p_i(t) > 0 \\ \Leftrightarrow \quad & f_i(t) > \bar{f}(t), && p_i(t) > 0\end{aligned}$$

3j, p. 140: Als β erg groot is dan zijn de verschillen in groei tussen de soorten kleiner, en is de dynamiek trager.

4a, p. 141: De gemiddelde fitness van de populatie is:

$$\hat{f(\vec{x})} = \sum_{i=1}^{n} x_i f_i(\vec{x}) = 6$$

4b, p. 141: De replicator vergelijking met aanpassingssnelheid α:

$$\Delta x_i = \alpha x_i (f_i(\vec{x}) - \hat{f(\vec{x})})$$

Als we nu over alle delta's (veranderingen) sommeren krijgen we

$$\begin{aligned}
\sum_i \Delta x_i &= \sum_i \alpha x_i (f_i(\vec{x}) - \hat{f}(\vec{x})) && (19)\\
&= \sum_i \alpha x_i f_i(\vec{x}) - \sum_i \alpha x_i \hat{f}(\vec{x})) && (20)\\
&= \alpha \hat{f}(\vec{x}) - \alpha 1 \hat{f}(\vec{x}) && (21)\\
&= 0 && (22)
\end{aligned}$$

4c, p. 141: In voorbeeld:

$$\Delta x_1 = 1.0\frac{1}{3}(9-6) = 1.$$

Dus in de volgende generatie zou $x_1 = 1 + 1/3$. Dat kan niet, want x_1 stelt een proportie voor.

4d, p. 141: Iteratie 1: $\hat{f}(\vec{x}) = 6$

$$\begin{aligned}
x_1 &= \frac{1}{3} + 0.1\frac{1}{3}(9-6) = 0.433 && (23)\\
x_2 &= \frac{1}{3} + 0.1\frac{1}{3}(6-6) = 0.333 && (24)\\
x_3 &= \frac{1}{3} + 0.1\frac{1}{3}(3-6) = 0.233 && (25)
\end{aligned}$$

Iteratie 2: $\hat{f}(\vec{x}) = 6.6$

$$\begin{aligned}
x_1 &= 0.433 + 0.10.433(9-6.6) = 0.54 && (26)\\
x_2 &= \frac{1}{3} + 0.1\frac{1}{3}(6-6.6) = 0.31 && (27)\\
x_3 &= 0.233 + 0.10.233(3-6.6) = 0.15 && (28)
\end{aligned}$$

1a, p. 142: De totale populatiegrootte is gelijk aan $H(t) + S(t) + I(t)$.

1b, p. 142: $I(t+1) = H(t) + S(t) + I(t) - (H(t+1) + S(t+1))$.

1c, p. 142: $I(t+1) = I(t) + bS(t)$.

1d, p. 142: Personen die immuun zijn kunnen niet ziek worden. (En hersteld zijn ze al.)

1e, p. 142: Parameter $a = 0$: niemand wordt ziek, en eventuele zieken herstellen. Als a klein is neemt het aantal zieken langzaam toe.

1f, p. 142: Als a groot is dan wordt iedereen snel ziek (maar de zieke populatie herstelt gegarandeerd).

1g, p. 142: Parameter $b = 0$: niemand herstelt. Als b klein is herstellen zieken langzaam.

1h, p. 142: Als b groot is dan herstelt iedereen snel—wellicht zó snel dat niet iedereen de kans krijgt om ziek te worden.

3a, p. 142: Ja, want $a \leq 3$. (Voor $a = 3$ is er ook nog convergentie.)

3b, p. 142: Nee, want als $a \in (3, 1 + \sqrt{6}]$ oscilleert (10) tussen twee waarden.

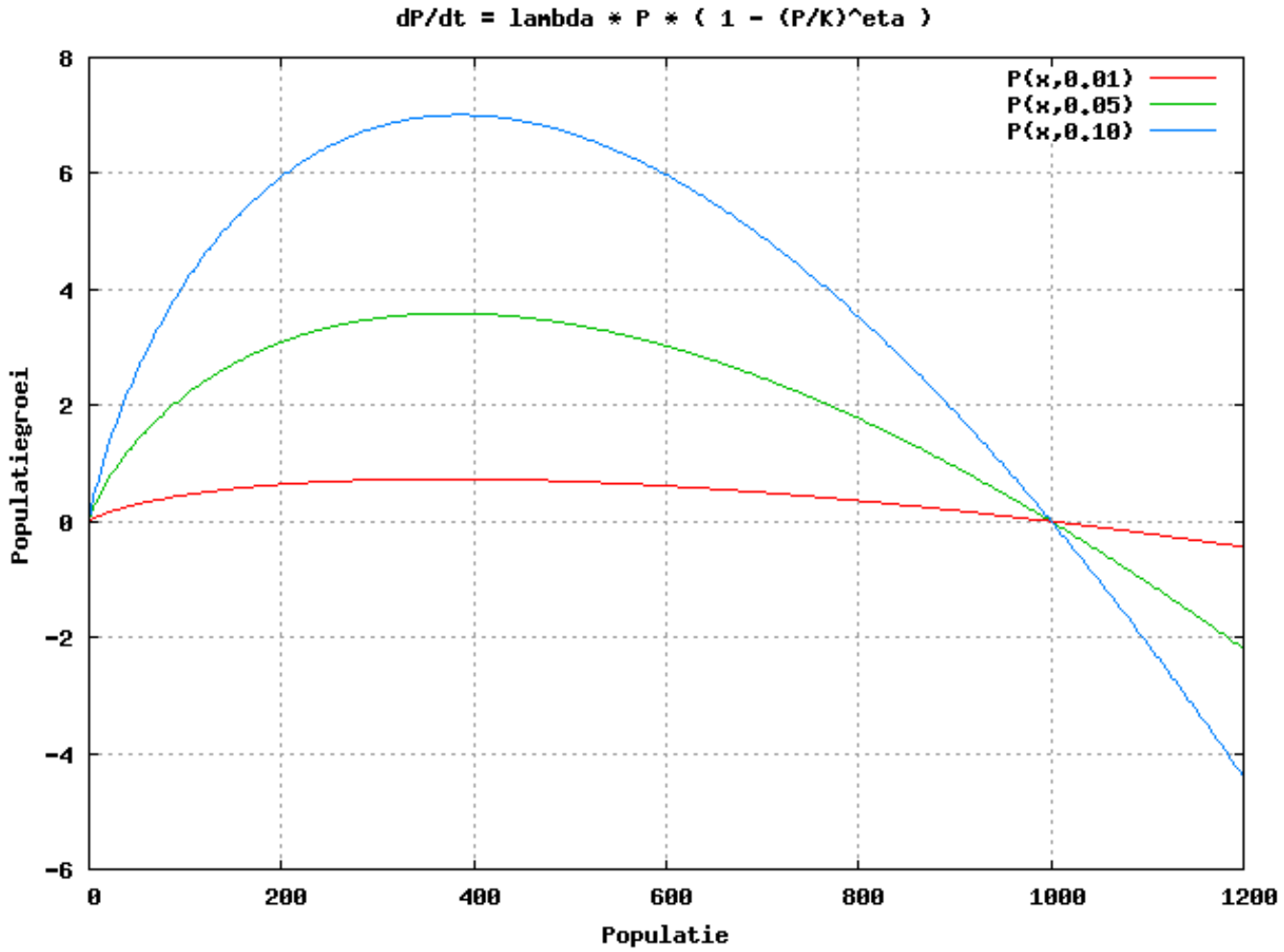
3c, p. 142: Omdat voor grotere a je niet kunt zeggen of er periodiciteit, semi-periodiciteit of chaos is.

3d, p. 142: Heel snel al: voor 3.57 verdubbelen de perioden naar $4, 8, 16, 32, \ldots$. Je zou dan evenveel keer op de repeat van je calculator moeten drukken om na dat aantal keer te kunnen constateren dat er een periode van 2^k is.

4a, p. 142: Eq. (11) is een differentiaalvergelijking: de groei van P hangt af van P zélf, i.p.v. t. Laat je hier verder niet door storen.

Een evenwichtspunt is een punt in de tijd waarop er geen groei is. Daarvoor moeten we de afgeleide van de groeiformule gelijk aan nul stellen. De afgeleide is reeds gegeven, dat is namelijk (11). Gelijk aan nul stellen levert op: $P = 0$ en $P = K$. Dus de groei is nul als de populatiegrootte nul, of K, is. (We nemen aan dat $\eta > 0$, anders wordt de populatiegrootte vanaf nul negatief, en dat kan niet.) Zijn dit stabiele punten? Nul is geen stabiel punt, want als P klein beetje groter wordt dan nul dan wordt ook de groei iets positief. P divergeert bij 0. K is een stabiel punt: als P iets kleiner is dan K dan is de groei iets positief. Als P iets over K heen komt, dan wordt de groei iets negatief. De populatiegrootte P convergeert voor alle positieve η naar K, wat dus blijkbaar de maximaal haalbare populatiegrootte is.

4b, p. 143: Ziehier een grafiek van de populatiegroei. Voor kleine η zien we dat de groei minder heftig verloopt. Merk op dat op de x-as niet t (tijd), maar P (populatie-omvang) staat.



Populatiegroei voor $\eta = 0.01$ (rood), $\eta = 0.05$ (groen), en $\eta = 0.10$ (blauw).

4c, p. 143: Laat P_t staan voor absolute populatiegrootte op tijdstip t. Dan kan de logistieke functie geschreven worden als

$$\frac{P_{t+1}}{K} = a\frac{P_t}{K}\left(1 - \frac{P_t}{K}\right)$$

Ofwel:

$$P_{t+1} = aP_t\left(1 - \frac{P_t}{K}\right) \tag{29}$$

Populatiegroei voor (29) is dan

$$\begin{aligned}\Delta P_t = P_{t+1} - P_t &= aP_t\left(1 - \frac{P_t}{K}\right) - P_t \\ &= aP_t - \frac{P_t^2}{K} - P_t \\ &= aP_t\left(\frac{a-1}{a} - \frac{P_t}{K}\right)\end{aligned}$$

4d, p. 143: De logistieke map:

$$P_{t+1} = aP_t(1 - P_t)$$

De logistieke functie:

$$\frac{dP}{dt} = \lambda P\left(1 - \left(\frac{P}{K}\right)^{\eta}\right)$$

a. De logistieke map beschrijft groei als een discreet proces, met onderscheidbare stapjes in de tijd (i.e., $P_0, P_1, P_2, \ldots$, etc). De logistieke functie beschrijft groei als een continu proces dat op elk moment afhangt van de populatiegrootte.

b. De logistieke map convergeert niet meer vanaf $a = 3/4$.De logistieke functie convergeert altijd (althans, voor positieve η).

c. De logistieke map gaat over relatieve populatiegrootte (variërend van 0 tot 1). De logistieke functie gaat over absolute populatiegrootte (variërend van 0 tot K en een beetje verder).

5a, p. 143: Er een dekpunt als en slechts als er stilstand in de populatiedynamiek is. Dus als en slechts als $dx/dt = 0$ en $dx/dt = 0$. Dit is het geval als ($x = 0$ of $y = a/b$) en ($y = 0$ of $x = c/d$). Deze vier mogelijkheden leveren twee dekpunten op, te weten $(0, 0)$ en $(c/d, a/b)$. Het eerste dekpunt is triviaal, en staat voor uitsterving, terwijl het tweede niet-triviale dekpunt een dynamisch evenwicht representeert. Er is dus één niet-trivial dekpunt.

5b, p. 143: Er zijn geen konijnen en vossen sterven langzaam uit.

$$\begin{cases} dx/dt = 0 \\ dy/dt = -cy \end{cases}$$

Oplossing: $(x_t, y_t) = (0, y_0 e^{-ct})$.

5c, p. 143: Er zijn geen vossen en konijnen groeien exponentieel:

$$\begin{cases} dx/dt = ax \\ dy/dt = 0 \end{cases}$$

Oplossing: $(x_t, y_t) = (x_0 e^{at}, 0)$.

5d, p. 144: Om de dynamiek rond beide dekpunten $(0, 0)$ en $(c/d, a/b)$ te bepalen, moeten we naar de afgeleide van (12) kijken. In R^2 is dat de Jacobiaan. De Jacobiaan geeft in de omgeving van elk punt een lineaire benadering van de dynamiek in dat punt. (Cf. The Comp. Beauty of Nature, Sec. 13.2, p. 207). De Jacobiaan van (12) is

$$J(x, y) = \begin{pmatrix} \delta x(a - by)/\delta x & \delta x(a - by)/\delta y \\ \delta y(-c + dx)/\delta x & \delta y(-c + dx)/\delta y \end{pmatrix} = \begin{pmatrix} a - by & -bx \\ dy & dy - c \end{pmatrix}$$

Om te bepalen of $(0, 0)$ een aantrekkings- of afstotingspunt is, bepalen we de eigenwaarden van J in $(0, 0)$. (Cf. The Comp. Beauty of Nature, Sec. 13.4, p. 209.) We vinden $\lambda \in \{a, -c\}$. De eerste eigenwaarde, a, zegt: aantrekken; de andere eigenwaarde, $-c$, zegt: in andere as afstoten. We hebben te maken met een zadelpunt. Kleine populaties (x en y dicht bij 0) worden dus niet automatisch toegezogen naar uitsterving, maar doorlopen óók netjes een cyclus.

Om te bepalen of $(c/d, a/b)$ een aantrekkings- of afstotingspunt is, bepalen we de eigenwaarden van J in $(c/d, a/b)$. We vinden $\lambda \in \{i\sqrt{ac}, -i\sqrt{ac}\}$. Beide waarden zijn imaginair maar tegengesteld. Hieruit kunnen we geen conclusies trekken, maar gewoon kijken naar het fase-diagram (The Comp. Beauty of Nature, Fig. 12.1, p. 185) laat ons zien dat het niet-triviale stabiele punt omgeven wordt door concentrische cycli.

5e, p. 144: Tussentijdse zelf-doorsnijding is niet mogelijk, omdat het traject volledig wordt bepaald door de locatie (x, y, z). Eenzelfde locatie betekent dus dezelfde richting in dezelfde snelheid. Tussentijdse zelf-doorsnijding kan worden mogelijk gemaakt door het inbrengen van een variabele in de rechterkant van de differentiaalvergelijkingen die niet gerelateerd is aan de locatie, bijvoorbeeld de tijd, t, of een toevalsfactor r, die bij elke aanroep een klein random getal produceert.

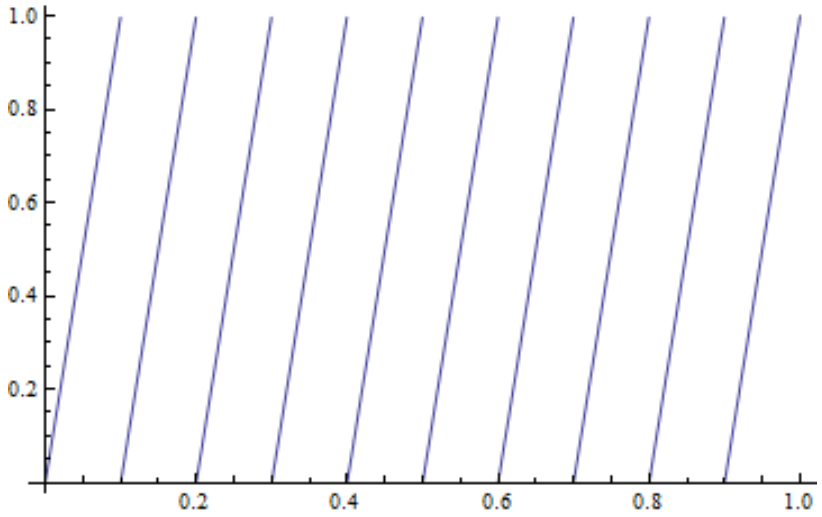
6, p. 145: Antwoord (a).

1a, p. 147: Uit het functievoorschrift volgt meteen dat elk getal $x \in [0, 1]$ door f wordt afgebeeld op steeds dezelfde functiewaarde $f(x)$.

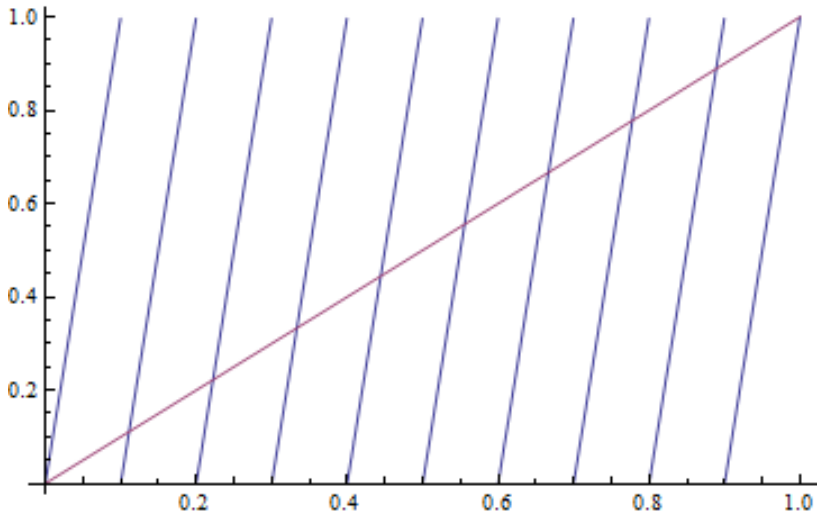
1b, p. 147: Alle decimalen schuiven één positie naar links. Als een decimaal daardoor vóór de decimale punt komt te staan, verdwijnt deze decimaal, i.e., deze verandert in een nul.

1c, p. 147: $f(0.012345) = f(0.812345) = 0.12345 \neq f(0.12345) = 0.2345$. De functie f is niet injectief.

1d, p. 147: Een grafiek van de decimale schuif-afbeelding:



1e, p. 147:



Er zijn tenminste twee manieren om aan dekpunten te komen. Door te redeneren en door te rekenen. Beredeneren: een getal mag niet veranderen als de decimalen één positie naar links verschuiven. Dus moeten alle decimalen van dat getal hetzelfde zijn, en moet er voor de komma al een nul staan.

Dat zijn dus $0.000\ldots$, $0.111\ldots$, $0.222\ldots$, $0.333\ldots$, $0.444\ldots$, en 0.888.

Het getal $0.999 = 1$ is een twijfelgeval. Bij nadere inspectie is 0.999 geen dekpunt omdat $0.999 = 1$ en $1 \pmod 1 = 0$. Kijk ook naar de grafiek. Elke schuine streep begint met een dicht bolletje en eindigt met een open bolletje.

Anderzijds is het mogelijk alle dekpunten direct uit te rekenen:

$$\begin{aligned} x = 10x \pmod 1 &\Leftrightarrow x = 10x + n,\ n \in \mathbb{Z},\ x \in [0,1) \\ &\Leftrightarrow 9x = n,\ n \in \mathbb{Z},\ x \in [0,1) \\ &\Leftrightarrow x = \frac{n}{9},\ 0 \leq n < 9 \\ &\Leftrightarrow x \in \left\{0, \frac{1}{9}, \frac{2}{9}, \ldots, \frac{8}{9}\right\}. \end{aligned}$$

1f, p. 147: Wil je een punt x met precies periode k, neem dan de eerste k decimalen van

$$C = 0.0123456789\,10\,11\,12\,13\,14\,15\,16\,17\,18\,19\,20\,21\,22\,23\,24\,25\,26\,27\,28\,29\,30\,31\,32\,\ldots \tag{30}$$

en herhaal dit groepje decimalen. Bijvoorbeeld voor $k = 11$ is dit groepje 01234567891 en word

$$x = 0.01234567891\,01234567891\,01234567891\,\ldots$$

Getallen zo geconstrueerd kennen geen perioden kleiner dan k omdat de aftelling $0, \ldots, k$ uniek is. De constante C uit 30 heeft overigens een naam. Deze wordt de constante van Champernowne genoemd.

Je kunt het ook uitrekenen:

$$\begin{aligned} x = 10^k x \pmod 1 &\Leftrightarrow x = 10^k x + n,\ n \in \mathbb{Z},\ x \in [0,1) \\ &\Leftrightarrow (10^k - 1)x = n,\ n \in \mathbb{Z},\ x \in [0,1) \\ &\Leftrightarrow x = \frac{n}{10^k - 1},\ 0 \leq n < 10^k - 1 \\ &\Leftrightarrow x \in \left\{0, \frac{1}{10^k - 1}, \frac{2}{10^k - 1}, \ldots, \frac{10^k - 2}{10^k - 1}\right\}. \end{aligned}$$

1g, p. 147: Er zijn $10^k - 1$ punten met periode k. Bijvoorbeeld, er zijn 999 punten met periode $k = 3$:

$$x \in \left\{0, \frac{1}{999}, \frac{2}{999}, \ldots, \frac{998}{999}\right\} = \{0, 0.001001\ldots, 0.002002\ldots,\ \ldots\}.$$

Hier kunnen ook punten met periode kleiner dan k in voorkomen.

1h, p. 148: Het antwoord volgt eigenlijk meteen uit beide hints. $\mathbb{R} - \mathbb{Q}$ bevat precies alle getallen met een a-periodieke decimale ontwikkeling. Een a-periodieke decimale ontwikkeling zorgt direct

voor een a-periodieke ontwikkeling van de reeks x, $f(x)$, $f^2(x)$, ... (Ga na!) $\mathbb{Q}$ is aftelbaar, dus blijven er overaftelbaar veel punten in $[0,1]$ over die a-periodiek zijn.

Het antwoord kan ook direct worden berekend. Immers, uit de berekening van het vorige antwoord volgt dat alle periodieke punten breuken zijn. $\mathbb{Q}$ is aftelbaar, dus blijven er overaftelbaar veel punten in $[0,1]$ over die a-periodiek zijn.

1i, p. 148: Laat $y \in [0,1]$ het getal zijn wat je wil benaderen met nauwkeurigheid $\epsilon > 0$. Laten we zeggen dat je y wilt benaderen met n decimalen achter de komma. (Dus $0 < \epsilon \leq 10^{-n}$.) Neem als startpunt een rijk reëel getal x. In zo'n getal komt elke eindige rij decimalen wel een keer voor. Dus ook de eerste n decimalen van y. Nu de decimalen van x een aantal van k keren naar links schuiven met f, totdat eerste n decimalen van y tevoorschijn komen.

Niet van elk a-periodiek punt ligt de ontwikkeling dicht in $[0,1]$. Kijk bijvoorbeeld naar de ontwikkeling van

$$0.101001\underbrace{000}_{3}1\underbrace{0000}_{4}1\underbrace{00000}_{5}1\ldots$$

Dit punt is a-periodiek maar zal nooit in de buurt van bijvoorbeeld 0.9 komen.

1j, p. 148: Gegeven is een willekeurig dichte afstand 10^{-n}, $n \in \mathbb{N}$. Simpel gezegd komt het antwoord er nu op neer dat we altijd twee getallen $x, y \in [0,1]$ kunnen kiezen die tot aan de n-e decimaal aan elkaar gelijk zijn, waarna de decimalen van x verder allemaal nullen zijn, en de decimalen van y verder allemaal negens zijn (met af en toe een decimaal $\neq 9$ om te voorkomen dat $f^n(y) = 1 = 0 = f^n(x) \pmod 1$). Het zal nu hopelijk duidelijk zijn dat na de ne iteratie de beelden van x en y oneindig vaak minstens 0.9 uit elkaar liggen.

Andere voorbeelden kunnen zeker ook goed zijn. Het gaat er om aan te tonen dat bij minimale start-verschillen de beelden al snel uit elkaar liggen.

1k, p. 148: We moeten een punt vinden dat, als het geïtereerd wordt, praktisch gezien overal in de toestandruimte komt. Preciezer gezegd: waarvan het spoor (= alle geïtereerde waarden) dicht ligt in $[0,1]$. Hiervoor kan weer de constante van Champernowne worden gebruikt (30 op de vorige pagina).

2a, p. 148: De dynamiek convergeert naar een stabiel vast punt voor $1 < r < 3$. Voor $r < 1$ convergeert het systeem altijd naar 0 (uitsterven). Tussen 1 en 3 stabiliseert de populatie zich op een waarde van $1 - (1/r)$. Dat laatste hoef je zelf niet te kunnen afleiden, maar wordt meegegeven als feit.

2b, p. 148: De dynamiek vervalt in een cyclus van periode 2 (het springen tussen twee waarden) zodra r groter wordt dan 3. Dit bereik loopt van $3 < r < 3.449\ldots$. Het wordt niet van je verlangd de exacte waarden te geven; aflezen van de grafiek volstaat.

2c, p. 148: Een verdere toename van r leidt tot een periode-verdubbeling. De dynamiek vertoont een periode 4 voor waarden van ongeveer $3.449 < r < 3.544$. Ook hier weer verlangen we niet van je dat je de exacte waardengeeft; aflezen van de grafiek volstaat.

2d, p. 148: Periode 8 treedt op in een nog kleiner interval, beginnend bij $r \approx 3.544$ tot ongeveer 3.564. Hierna volgen periode 16, 32, enzovoort, steeds sneller achter elkaar (de Feigenbaum-constante beschrijft deze versnelling).

2e, p. 148: Ja, er zijn waarden voor r waarbij periode 3 voorkomt. Dit gebeurt in een "venster van orde"' binnen het chaotische gebied, rond $r \approx 3.828$. Interessant feitje: volgens de zogenaamde stelling van Sharkovskii impliceert het bestaan van periode 3 dat er cycli van alle andere mogelijke periodes bestaan. Dit feit hoef je niet te kennen.

2f, p. 148: Ja, er zijn ook waarden voor periode 5. Een bekend venster voor een stabiele periode 5 bevindt zich rond $r \approx 3.738$ en ook vlakbij $r \approx 3.905$.

2g, p. 148: Bij $r = 4$ is de dynamiek volledig chaotisch. De baan van x_0 (de startwaarde) vult het hele interval $(0, 1)$ en is extreem gevoelig voor beginvoorwaarden (het vlindereffect). Er is geen sprake meer van een stabiele cyclus; het spinnenweb-diagram vult de ruimte op een schijnbaar willekeurige, maar deterministische wijze.

2h, p. 148: Ja, chaos begint eigenlijk al direct na de accumulatie van alle periode-verdubbelingen, vanaf de zogenaamde Feigenbaum-waarde $r \approx 3.5699\ldots$. Tussen deze waarde en $r = 4$ wisselen gebieden van chaos en eilanden van orde (zoals de genoemde periode 3 en 5 vensters) elkaar af.

3, p. 148: Dit kan: 0.75 invullen geeft direct weer 0.75. Dus de baan van 0.75 is daarmee $0.75, 0.75, 0.75, \ldots$

4, p. 150: Antwoord (d).

5a, p. 150: Alle $3.0000 < r \leq 3.5462$ gegeven een stabiele cyclus met periode $p = 2$. Omdat het bifurcatiediagram geen punten boven of onder de hoofdlijnen laat zien, lijkt elke startwaarde goed. Pas echter wel op dat je geen startwaarde $x_0 = (r - 1)/r$ neemt, daar dit een dekpunt is van $f_r(x)$.

5b, p. 150: Voor $3.5462 \leq r \leq 3.5660$. Als startwaarden mogen geen dekpunten van f_r^2 worden gekozen. (Deze dekpunten zijn verder wat moeilijk expliciet uit te drukken, dus we laten het bij deze aanduiding.)

5c, p. 150: Voor $3.8274 \leq r \leq 3.8418$. Alle startwaarden uit eerdere bifurcaties mogen niet worden gekozen. Dat zijn er aftelbaar veel. De kans dat een willekeurig gekozen startwaarde daar bij zit is nul, omdat $[0, 1]$ overaftelbaar is.

5d, p. 150: Voor 5: $3.7377 \leq r \leq 3.7414$. Voor 6: direct na periode 3. Voor 7: $3.7013 \leq r \leq 3.7031$. Voor 8: direct na periode 4. Voor 9: $3.6873 \leq r \leq 3.6884$.

6a, p. 150:

$$f_r(x) = x \Leftrightarrow rx(1-x) = x \Leftrightarrow \begin{cases} x = 0 \text{ of} \\ x \neq 0 \text{ en } x = \dfrac{r-1}{r}, \ r \in [1, 4]. \end{cases}$$

Dus als $0 \leq r < 1$ bezit f_r één dekpunt, in andere gevallen twee.

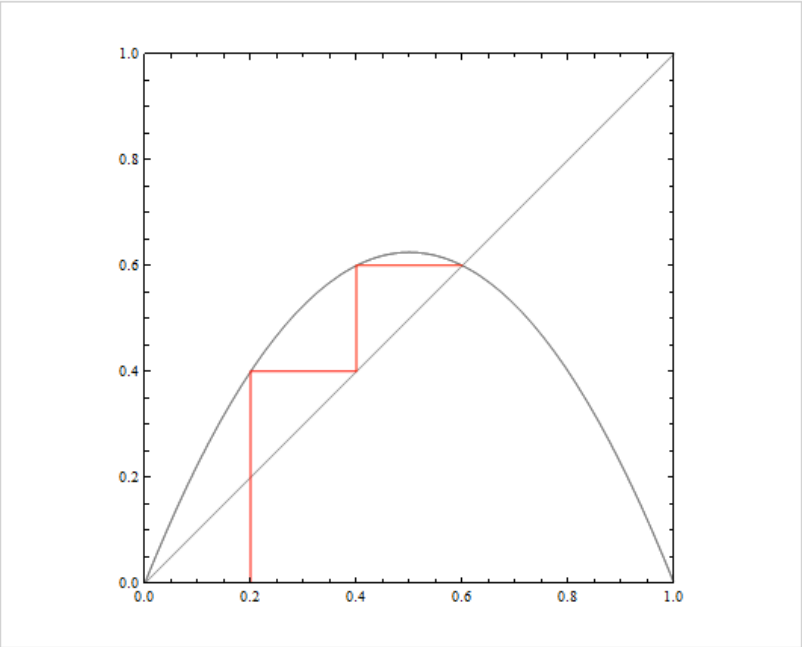
6b, p. 151:

$$f_r'(x) = r - 2rx.$$

Dekpunt 0 is stabiel $\Leftrightarrow |f_r'(0)| < 1 \Leftrightarrow |r - 2r\cdot 0| < 1 \Leftrightarrow 0 \leq r < 1$. Dekpunt 0 is labiel $\Leftrightarrow$ $1 < r \leq 4$. Als $r = 1$ kan op basis van f_r' niets gezegd worden. Enig experimenteren levert op dat voor $r = 1$ het dekpunt $(r-1)/r = 0$ aantrekt noch afstoot.

Dekpunt $(r-1)/r$ is stabiel $\Leftrightarrow |f_r'((r-1)/r)| < 1 \Leftrightarrow |r - 2r\cdot r| < 1 \Leftrightarrow 1 < r \leq 3$. Dekpunt $(r-1)/r$ is labiel $\Leftrightarrow 3 < r \leq 4$. Als $r = 3$ kan op basis van f_r' niets gezegd worden. Enig experimenteren levert op dat voor $r = 3$ het dekpunt $(r-1)/r2/3$ aantrekt noch afstoot.

Convergentie in een cobweb (spinnenweb) diagram voor $r = 2.5$ en startwaarde $x_0 = 0.2$:



6d, p. 151:

$$\begin{aligned} f_r^2[x] &= x && \Leftrightarrow \\ [rx(1-x)]^2 &= x && \Leftrightarrow \end{aligned}$$

Dekpunt $x = 0$ er uit halen:

$$(x-0)\left[x(x-1)^2r^2 - 1\right] = 0 \qquad \Leftrightarrow$$

Dekpunt $(r-1)/r$ er uit halen:

$$(x-0)\left(x - \frac{r-1}{r}\right)\left(r^2x^2 - (r^2+r)x + 1\right) = 0.$$

Vervolgens $r^2x^2 - (r^2+r)x + 1 = 0$ oplossen met de abc-formule, onder de voorwaarde dat $r \in [0, 4]$. Dit geeft:

$$x = \frac{r + 1 \pm \sqrt{(r-3)(r+1)}}{2r}.$$

mits discriminant > 0 dus een periode twee bestaat voor $r \in [3, 4]$.

6e, p. 151: Laat x_1^*, x_2^* de punten zijn waartussen f_r met periode 2 alterneert. Met de kettingregel:

$$(f^2)'(x) = [f(f(x))]' = f'(f(x))f'(x).$$

Omdat $f(x_2^*) = x_1^*$,

$$(f^2)'(x_2^*) = f'(f(x_2^*))f'(x_2^*) = f'(x_1^*)f'(x_2^*).$$

Dus de 2-cykel is stabiel a.e.s.a. $|f'(x_1^*)f'(x_2^*)| < 1$.

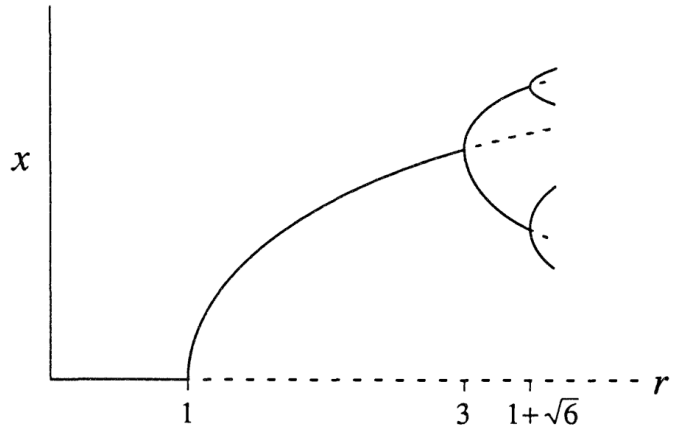
Nu is

$$\begin{aligned} f'(x_1^*)f'(x_2^*) &= r(1-2x_1^*)r(1-2x_2^*) \\ &= r^2\left(1-2(x_1^*+x_2^*)+4x_1^*x_2^*)\right) \\ &= r^2\left(1-2(r+1)/r+4(r+1)/r^2\right) \\ &= -r^2+2r+4, \end{aligned}$$

zodat

$$|f'(x_1^*)f'(x_2^*)| < 1 \Leftrightarrow |-r^2+2r+4| < 1 \Leftrightarrow 3 < r < 1+\sqrt{6}.$$

Dus tot $1+\sqrt{6}$ zijn de 2-cykels stabiel, daarna niet meer.



7a, p. 152: x_{n+1} uit de eerste vergelijking invullen in de tweede vergelijking met indices n weglaten geeft

$$\begin{aligned} arx(1-x)+b &= (ax+b)^2+c \\ \Leftrightarrow -arx^2+arx+b &= a^2x^2+2abx+b^2+c. \end{aligned}$$

Omdat dit voor alle x moet gelden, moet

$$\begin{cases} -ar &= a^2 \\ ar &= 2ab \\ b &= b^2+c \end{cases}$$

Oplossen geeft $a = -r$, $b = r/2$, $c = r/2(1-r/2)$. Dus $z = -rx + r/2 = r(1/2 - x)$.

7b, p. 152: Er geldt $r \in [0,4]$. Voor deze waarden van r is $c = r/2(1-r/2)$ een bergparabool met bereik $[-2, 1/4]$. Dit zijn dan ook de toegestane waarden van c.

Omdat $z = r(1/2 - x)$ en $x \in [0, 1]$ moet $-r/2 \leq z \leq r/2$. Omdat $c = r/2(1 - r/2)$ en dus $r = 1 + \sqrt{1 - 4c}$, geldt

$$-\frac{1 + \sqrt{1 - 4c}}{2} \leq z \leq \frac{1 + \sqrt{1 - 4c}}{2}$$

Functievoorschrift als logistieke afbeelding:

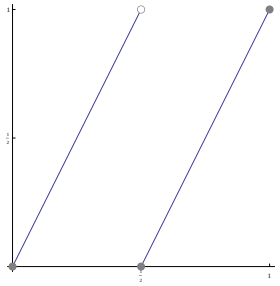
$$g_c : \left[-\frac{1 + \sqrt{1 - 4c}}{2},\ \frac{1 + \sqrt{1 - 4c}}{2}\right] : z \mapsto z^2 + c, \qquad c \in \left[-2, \frac{1}{4}\right].$$

Verloop van f_3 met startwaarde $x = 1/2$: $1/2 \to r/4 \to 1/4(1 - r/4)r^2$. Omdat $r = 3$ levert dit 0.5625 op.

Omdat $r = 3$ moet $c = 3/2(1 - 3/2) = -3/4$. Verloop van $g_{-3/4}$ met startwaarde $z = r(1/2 - x) = 3 \cdot 0 = 0$: $0 \to -3/4 \to -0.187500$. De uitkomst -0.1875 met $x = 1/2 - z/r$ terugvertalen naar x geeft inderdaad $x = 1/2 - z/r = 0.5 - (-0.1875)/3 = 0.5625$.

Voor f_r begint de weg naar chaos bij $r \approx 3.56994567\ldots$. Omdat $c = r/2(1 - r/2)$ begint bij g_c de chaos zo'n beetje in de buurt van $c = -1.40116\ldots$

8a, p. 152: De grafiek van de dyadische transformatie:



8b, p. 152:

$$(10011)_2 = \frac{1}{2} + \frac{1}{16} + \frac{1}{32} = \frac{19}{32} = 0.59375$$

$$0.90625 = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{32} = (0.11101)_2$$

8c, p. 152: Binaire representaties van getallen achter de komma herhalen zich vrijwel altijd in groepjes. Dat is altijd het geval als getal achter de komma geen machten van twee zijn. Herhalende groepjes kunnen met een overline worden genoteerd:

$$\begin{aligned}
1/3 &= (0.\overline{01})_2 \\
2/3 &= (0.\overline{10})_2 \\
1/5 &= (0.\overline{0011})_2 \\
2/5 &= (0.0\overline{1100})_2 \\
3/5 &= (0.1\overline{0011})_2 \\
4/5 &= (0.\overline{1100})_2
\end{aligned}$$

8d, p. 152:

$$\begin{aligned} D(x) &= 2x \\ &= 2(0.0b_2b_3b_4\ldots)_2 \\ &= 2\left(0 \times \frac{1}{2} + b_2\frac{1}{4} + b_3\frac{1}{8} + b_4\frac{1}{16} + \ldots\right) \\ &= b_2\frac{1}{2} + b_3\frac{1}{4} + b_4\frac{1}{8} + \ldots \\ &= (0.b_2b_3b_4\ldots)_2 \end{aligned}$$

Alle bits schuiven dus één naar links.

8e, p. 152:

$$\begin{aligned} D(x) &= 2x - 1 \\ &= 2(0.1b_2b_3b_4\ldots)_2 - 1 \\ &= 2\left(\frac{1}{2} + b_2\frac{1}{4} + b_3\frac{1}{8} + b_4\frac{1}{16} + \ldots\right) - 1 \\ &= 2 \times \frac{1}{2} + b_2\frac{1}{2} + b_3\frac{1}{4} + b_4\frac{1}{8} + \cdots - 1 \\ &= (0.b_2b_3b_4\ldots)_2 \end{aligned}$$

Weer schuiven alle bits één naar links. De dyadische transformatie wordt daarom ook wel de *bit shift map* genoemd.

8f, p. 153: Laat k gegeven zijn. Dan is gemakkelijk een rijtje bits B ter lengte k te bedenken zó dat geen enkele prefix van dat rijtje zich in dat rijtje herhaalt. Voorbeeld: $10110011100011110000\ldots$ Dan bezit $x = (0.BBBBB\ldots)_2 = (0.\overline{B})_2$ periode k en geen periode m met $m < k$.

8g, p. 153: Laat $x \in [0,1]$ en $\epsilon > 0$ gegeven zijn. We moeten een element $p \in P$ vinden, zó dat p niet meer dan ϵ van x af ligt. Welnu, als $x = (0.b_1b_2b_3b_4\ldots)_2$, laat p dan zo lang de binaire expansie van x volgen totdat deze dicht genoeg bij x ligt. Zeg dat p getal x volgt tot en met bit n. Laat B een eindige bitstring ter lengte k zijn zoals in het vorige onderdeel. Dan is

$$p = 0.b_1b_2\ldots b_nBBB\ldots$$

een punt uit P dat aan het gevraagde voldoet. Immers, na het opschuiven van de eerste n bits vervalt p dankzij de bit-blokken $BBB\ldots$ in een periode ter lengte k.

8h, p. 153: Alle rationale getallen worden gerepresenteerd door expansies die eindigen of zich herhalen. Op precies deze getallen stabiliseert D of herhaalt D zich, respectievelijk. Op de rest van de getallen d.w.z. getallen met een a-periodieke binaire expansie, zal D zich uiteindelijk ook a-periodiek gaan gedragen. “De rest” van een aftelbare verzameling in een overaftelbare verzameling is overaftelbaar, dus A is overaftelbaar.

8i, p. 153: De verzameling eindige bit strings is aftelbaar en kan dus achter elkaar worden gezet. Vorm hiermee een binair getal:

$$0.0\ 1\ 00\ 01\ 10\ 11\ 000\ 001\ 010\ 011\ 100\ 101\ 110\ 111\ 0000\ 0001\ \ldots$$

Makkelijk is na te gaan dat iedere eindige groep bits in dit getal voorkomt. (Sterker: iedere eindige groep bits in dit getal komt oneindig vaak voor. Dat laatste hebben we niet nodig.) Laat $y = (b_1b_2b_3b_4\ldots)_2$ gegeven zijn, en stel dat we y willen benaderen tot het ne bit. D kan dan net zo vaak worden toegepast op x, i.e., de bits van x kunnen net zo lang naar links worden opgeschoven, totdat het groepje $b_1b_2b_3b_4\ldots b_n$ verschijnt. (Op dat moment geldt $|x-y| \leq 1/2^n$.) Omdat n willekeurig groot kan worden gemaakt, ligt het spoor van x, i.e. ligt de verzameling $\{x, D(x), D(D(x)), \ldots\}$, dicht in $[0,1]$. Dus x mixt in $[0,1]$.

8j, p. 153: Laat $x \in [0,1]$ en $\epsilon > 0$ gegeven zijn. We moeten een element $m \in M$ vinden, zó dat m niet meer dan ϵ van x af ligt. Dit gaat op dezelfde manier als in onderdeel 8g: als $x = (0.b_1b_2b_3b_4\ldots)_2$, laat m dan zo lang de binaire expansie van x volgen totdat deze dicht genoeg bij x ligt. Pas vervolgens de constructie uit het vorige onderdeel toe zodat het spoor van m zelf óók dicht ligt in $[0,1]$.

8k, p. 153: Laat $\epsilon > 0$ gegeven zijn. Laat x en y tot op het ne bit overeenkomen z.d.d. $|x-y| \leq \epsilon$, en vervolgens verder gaan in nullen resp. enen. Na n iteraties liggen de beelden zo ver mogelijk, d.w.z. een afstand van 1 uit elkaar.

9, p. 153: Antwoord (a): elk punt p wordt willekeurig dicht benaderd door het f-spoor dat begint in x_0. (Met het laatste wordt de rij x_0, $f(x_0)$, $f^2(x_0)$, $f^3(x_0)$, ... bedoeld.)

In wiskunde: voor elk punt $p \in [0,1]$ en elke vooraf opgegeven nauwkeurigheid $\epsilon > 0$ is er een aantal iteraties $n > 0$, zó dat $f^n(x_0)$ en dat punt p niet verder dan ϵ van elkaar af liggen.

10a, p. 153: Het bereik van f is $[0,1]*$. Dit is een begrensde verzameling.

10b, p. 153: Laat x_0 gegeven zijn. Om continuïteit in x_0 aan te tonen moeten we laten zien dat er voor elke $\epsilon > 0$ een $\delta > 0$ bestaat zo dat, als x met $|x - x_0(\text{mod } 1)| \leq \delta$, dan $|f(x) - f(x_0)| \leq \epsilon$. Wel, dit geldt al voor $\delta = \epsilon$. Immers,
$|f(x) - f(x_0)| = |x + r \ (\text{mod } 1) - (x_0 + r \ (\text{mod } 1))| = |x - x_0(\text{mod } 1)|$.

10c, p. 153: $f^n(x) = x + nr \ (\text{mod } 1)$.

10d, p. 153: Stel wel. Dan zou er een x_0 moeten zijn zo dat $f(x_0) = x_0$. Dus $x_0 + r \ (\text{mod } 1) = x_0$. Dus $x_0 + r = x_0 + k\cdot 1$. Beide kanten x_0 aftrekken levert $r = k$ wat zou impliceren dat $r \in Q$. Dat laatste hadden we juist uitgesloten. Dus f kan geen dekpunt hebben.

10e, p. 153: Stel wel. Dan zou er een x_0 moeten zijn en een positief geheel getal n zo dat $f^n(x_0) = x_0$. Dus $x_0 + nr \ (\text{mod } 1) = x_0$. Dus $x_0 + nr = x_0 + k\cdot 1$. Beide kanten x_0 aftrekken levert $r = k/n$ wat zou impliceren dat $r \in Q$. Dat laatste hadden we juist uitgesloten. Dus f kan niet periodiek zijn.

10f, p. 153: Laat ϵ een klein positief of negatief getal dicht bij nul zijn.
$f^n(x+\epsilon) = x + \epsilon + nr \pmod 1$. We zien dat als de startwaarde ϵ verandert, dat dan de functiewaarde na n iteraties ook maar ϵ is veranderd.

10g, p. 153: De functie f voldoet aan de volgende eigenschappen: geen dekpunt, begrensd, niet periodiek, niet gevoelig voor kleine veranderingen in startwaarden.

11a, p. 154: Laat $\epsilon > 0$. Elke keer als P een afstand ϵ vooruit rijdt, zal P een vierkant met een oppervlakte ϵ^2 bezetten waar P niet meer mag komen. Dit kan P maximaal $\lceil 1/\epsilon^2 \rceil$ keer doen vooraleer deze het hele eenheidsvierkant heeft afgelegd en vastloopt. (Waarschijnlijk zal P eerder vastlopen, maar dat maakt ons verder niet uit. Het ging er om te bewijzen dat P *op den duur* moet vastlopen.)

11b, p. 155: Het is mogelijk dat P blijft doorrijden, bijvoorbeeld in een hoekige spiraal waarvan de banen steeds dichter bij elkaar komen te liggen. Omdat hoeken van Tron-sporen niet differentieerbaar zijn (het zijn scherpe hoeken van 90^0), kan in dergelijke gevallen de stelling van Poincaré en Bendixson niet meteen worden toegepast. Deze vereist immers dat het spoor overal differentieerbaar is (i.e., nergens "knikt"). Gelukkig bestaan er versies van Poincaré en Bendixson;s stelling waarbij differentieerbaarheid wordt omzeild (Hajek, 1968), of niet wordt gebruikt (Gutierrez, 1986).[59]

Met Gutierrez en/of Hajek is de stelling van Poincaré en Bendixson nu dus ook van toepassing op Tron-sporen. De limiet-verzameling van een oneindig Tron-spoor kan derhalve drie gedaantes aannemen: een dekpunt, een samenhangende verzameling van dekpunten, of een cyclus (periodieke baan). Bijvoorbeeld, als het spoor een (hoekige) spiraal is die naar binnen draait zal de limiet-verzameling een dekpunt zijn; als het spoor een (hoekige) spiraal is die naar buiten draait zal de limiet-verzameling topologisch equivalent zijn aan een cirkel.

11c, p. 155: Deze bewering klopt niet omdat per keer (per "run") maar één specifieke Peano-kromme kan worden afgelegd, bijvoorbeeld P_3 of P_{42} of $P_{9959746}$. Elke specifieke Peano-kromme bezit nog steeds dimensie 1. Alleen de *limiet* van een oneindige reeks steeds fijnere Peano-krommen bezit dimensie 2.

12a, p. 156: Het spoor zal oneindig vaak slagen van boven naar beneden (en van beneden naar boven) maken. Daardoor komt het spoor steeds dichter bij de rand van het eenheids-vierkant. De limiet-verzameling is dus

$$L = \partial[0,1]^2 = \{(x,y) \in \mathbb{R}^2 \mid x \in \{0,1\} \text{ en } y \in \{0,1\}\}.$$

12b, p. 156: De eerste verticale slag zal omlaag gaan en op $1/4$ van de rand van $[0,1]^2$ plaatsvinden, te weten langs de lijn $x = 1/4$. De tweede verticale slag zal omhoog gaan en op $1/8$ van de rand plaatsvinden, te weten langs de lijn $x = 1 - 1/8$. De n^{e} verticale slag zal op $1/2^{n+1}$

[59] Gutierrez, bijvoorbeeld, bewees in 1986 dat elk continu (maar niet noodzakelijk differentieerbaar) spoor op elke compacte continu-differentieerbare variëteit (dus i.h.b. op $[0,1]^2$), topologisch equivalent is met een differentieerbaar spoor op diezelfde variëteit. (C. Gutierrez, 1986. "Smoothing continuous flows on two-manifolds and recurrences," *Ergodic Theory and Dynamical Systems*, **6**, pp. 17-44.) Ook bestaan er bewijzen van Poincaré en Bendixson's stelling waarbij de differentieerbaarheid van het spoor niet wordt gebruikt. (Hajek, 1968. *Dynamical systems in the plane*, Academic Press, London etc., Sec. VII.1; Beck, 1984. *Continuous flows in the plane*, Springer-Verlag, Berlin, Ch. 2.)

van de rand plaatsvinden, en omhoog gaan als en slechts als n even is. Bij 42 negens zal de 41^{e} slag omlaag gaan en zich langs de lijn $x = 1/2^{42}$ bewegen. De 42^{e} en laatste slag zal omhoog gaan en zich langs de lijn $x = 1 - 1/2^{43}$ bewegen. Vervolgens zal het spoor boven blijven en zich naar het lijnstuk

$$L = \{(x, 1) \in \mathbb{R}^2 \mid 1/2^{42} \leq x \leq 1 - 1/2^{43}\}$$

toe bewegen. L is de limiet-verzameling.

13a, p. 156: Er zijn verschillende mogelijkheden om de tent-afbeelding te formuleren. Hier zijn er een paar.

1. $T : [0,1] \to [0,1] : x \mapsto c \min\{x, 1-x\},\ 0 \leq c \leq 2$
2. $T : [0,1] \to [0,1] : x \mapsto c(1/2 - |1/2 - x|),\ 0 \leq c \leq 2$
3. $T : [0,1] \to [0,1] : x \mapsto \begin{cases} cx, & x \leq 1/2, \\ c(1-x), & \text{anders.} \end{cases},\ 0 \leq c \leq 2$

13b, p. 156: Nee. Het punt $(c, x) = (1.2, 0.53)$ ligt in "het oog" van het bifurcatiediagram.

13c, p. 156: Als $x \leq 1/2$ dan geldt vanwege $c = 1$ dat $T(x) = x$ en is er meteen een stabiel punt. Als $x > 1/2$ dan geldt $T(x) = 1 - x < 1/2$. Bij de volgende iteratie zal $1 - x$ bij itereren niet bijmeer veranderen.

13d, p. 156: Als $x \leq 1/2$ dan geldt vanwege $c < 1$ dat $T(x) = cx < x$. Dus $T(x), T(T(x)), T(T(T(x))), \ldots$ vormt een strikt dalende naar beneden begrensde rij en heeft volgens een elementaire stelling uit de calculus een limiet.

Als $x > 1/2$ dan geldt vanwege $c < 1$ dat $T(x) = c(1-x) < 1 - x$. Daar $1 - x < 1/2$ geldt nu $T(x) < 1/2$ zodat dit resultaat bij verder itereren ook een strikt dalende naar beneden begrensde rij vormt.

13e, p. 156: Laat $x \in [0,1]$. We bekijken het verschil tussen de beelden van x en $x + \epsilon$. Merk op dat, ongeacht x, we de waarde van $\epsilon > 0$ zó klein kunnen kiezen dat x en $x + \epsilon < 1/2$ of x en $x + \epsilon \geq 1/2$.

$$\begin{aligned} |T(x) - T(x+\epsilon)| &= \begin{cases} |cx - c(x+\epsilon)|, & x \text{ en } x+\epsilon < 1/2, \\ |c(1-x) - c(1-(x+\epsilon))|, & \text{anders.} \end{cases} \\ &= \begin{cases} |-c\epsilon|, & x \text{ en } x+\epsilon < 1/2, \\ |-c\epsilon|, & \text{anders.} \end{cases} \\ &= |-c\epsilon| \\ &= c\epsilon. \end{aligned}$$

In woorden: als $x, y \in [0,1]$ dicht genoeg bij elkaar liggen, dan zullen de beelden van x en y een vaste factor $c > 1$ verder uit elkaar liggen. Zolang x en y aan dezelfde kant van $1/2$ blijven groeit de afstand tussen x en y dus exponentieel, totdat x en y elk aan een andere kant van $1/2$ liggen. Stel $x < y$. Spiegel op dat moment het origineel van y in de lijn $x = 1/2$ en we zien dat de beelden vanaf dan weer exponentieel hard uit elkaar groeien. Dit laatste is een kenmerk van chaotisch gedrag: een kleine fout in het begin groeit zeer snel uit naar een grote fout verderop in het proces.

14a, p. 157: MIT's Beer Game laat zien dat supply-demand ketens met redelijk en voorspelbaar gedrag aan de ultieme vraagkant tot chaotisch gedrag kan leiden.

14b, p. 157: Chaotisch gedrag.

14c, p. 157: Uit documentatie: "Demand Style: This is simply a way to try customer demand other than the usual step function. Sine and Square vary the demand over a 52 week period, min of 0 and max of 8, thus can be thought of as seasonal variation. Random simply chooses random numbers between 0 and 8."

Uit Netlogo code:

```
to-report calculate-Step    report orderN  end
to-report calculate-Square  report orderN * (floor ((ticks - tickN) / 26) mod 2) end
to-report calculate-Sine    report round ((orderN / 2) * (1 + sin (360 * (ticks - tickN) /
```

Step: 4 orders per week, later 8 orders per week. Sine, Square: 8 orders per week, maar variëren op tijd. Random: random tussen 0 en 8. Allen komen in aanmerking, behalve random, want random is non-deterministisch.

14e, p. 157: Het Bullwhip effect verwijst naar het probleem dat kan ontstaan door de fluctuerende vraag van orders binnen een voorraadketen. Door de fluctuerende vraag wordt het voorspellen van de productie bemoeilijkt en kunnen leveranciers met te grote voorraden blijven zitten.

14f, p. 157: Kleine voorraden aanhouden en goederen (halffabrikaten e.d.) pas bestellen bij de leverancier als eigen afnemer een bestelling plaatst. Voordeel: lage voorraadkosten. Nadeel: trage levering. Just in time mechanismen hebben succes door korte communicatie-ketens en snelle levertijden.

14g, p. 157: Twee methoden: De OGY (Ott, Grebogi and Yorke) methode: sturen met discrete "bursts"; Pyragas' continue controle: correctiekracht hangt af van verschil met labiele cyclus.

14h, p. 157: Door het controleren van chaos kan de bestelketen geduwd worden naar een stabiele toestand waar iedereen gemiddeld 8 orders per week plaatst.

14i, p. 157: Voorraadketenbeheer (Eng.: supply chain management) is de wetenschap van het slim beheren van voorraden van de eigen onderneming of die van een klant. Sleutelbegrippen: Voorraadbeheer. Ketenautomatisering. Ware house management. Logistiek.

15a, p. 157: In Netlogo:

```
to draw
  no-display
  clear-all-plots
  set-plot-x-range min-s max-s
  let delta-s (max-s - min-s) / steps-s
  let s min-s
  while [s <= max-s] [
```

```
    let x seed-x
    let i 0
    repeat trans-x + finish-x  [
      if i >= trans-x [ plotxy s x ]
      set x s * x * (1 - x)
      set i i + 1
    ]
    set s s + delta-s
  ]
  display
end
```

15b, p. 158: In Netlogo:

```
to draw
  no-display
  clear-all-plots
  set-plot-x-range XStart XStop
  set-plot-y-range 0 1
  let R XStart
  let XIncrement (XStop - XStart) / XSteps
  while [R <= XStop] [
    let x Seed
    ; set x S / 2 * (1 - (abs x) ^ T) - 1 ; http://demonstrations.wolfram.com/FiniteLyapun
    repeat startTick [ set x R / 4 * (1 - (abs (2 * x - 1)) ^ beta) ]
    repeat YSamples  [ set x R / 4 * (1 - (abs (2 * x - 1)) ^ beta) plotxy R x ]
    set R R + XIncrement
  ]
  display
end
```