Utrecht University
Faculty of Science
Department of Information and Computing Sciences

Final Exam Algorithms for Decision Support, Monday November 6, 2017, 13.30-16.30 hr.

- Switch off your smart phone, PDA and any other mobile device and put it far away.
- You are allowed to use a regular calculator.
- This exam consists of **4** questions and a bonus question.
- Answers may be provided in either Dutch or English.
- A statistical table is distributed separately and should be returned.
- All your answers should be clearly written down and provide a clear explanation. Unreadable or unclear answers may be judged as false.
- Please write down your name and student number on every exam paper that you hand in.
- The maximum score is indicated at each of the questions.

Good luck, veel succes !

Question 1: Simulation of dynamic bus station
(a: 1 pt., b: 1 pt., c: 1 pt., total: 3 points.)
We consider a dynamic bus station. We assume that we are given a timetable with expected arrival and departure times of buses at the station. We assume that the actual arrival times have a stochastic deviation from the expected arrival time.

The bus company wants to perform a discrete-event simulation study to decide on the required number of platforms. Each platform consists of a front and rear position. A bus that arrives at the station can go to any free platform position, where it has a preference to use the front position of a platform. However, a bus cannot reach the front position if the rear position of the same platform is occupied. If all reachable positions are occupied, the bus has to wait in a first-come first-served queue. The total amount of time needed to disembark the passengers and embark new passengers at the platform is subject to uncertainty; it follows a uniform distribution between 1 and 6 minutes. A bus can only leave when the disembarking and embarking is finished, but it is not allowed to leave before its scheduled departure time t^d. Clearly, a bus cannot leave a rear position, if the front position is still occupied.

(a) Describe three *different* possible actions to validate the simulation model of the dynamic bus station. The descriptions of the actions should be as specific as possible and include which aspects of the model are adressed. A general remark like: "discussion with an expert" will not result in any credit points.

(b) A careful study of the difference between the actual and scheduled arrival times revealed that

- With probability 90 % the bus arrives after the scheduled arrival time. The delay then follows an exponential distribution with average 1 minute.
- With probability 10 % the bus arrives before the scheduled arrival time. The earliness is then uniformly distributed between 0 and 2 minutes.

Describe how can we generate the actual arrival times in a program written in an imperative programming language like Java or C♯ without using **any** specific random generation libraries or functions.
Note: You do not have to give a program, but just a description or pseudo-code.

(c) Suppose we have the following 20 samples of the time (in minutes) needed to embark and disembark passengers: 1.31, 1.4, 1.74, 1.9, 2.2, 2.3, 2.7, 3.1, 3.5, 3.6, 3.85, 4.07, 4.13, 4.28, 4.53, 4.75, 5.23, 5.5, 5.61, 5.86. The hypothesis is that these number are uniformly distributed on $[1; 6]$. Test this hypothesis with a χ^2-test with $K = 5$ and confidence level $1 - \alpha$ equal to 95 %.

Question 2: Energy constrained max flow
(a: 1 pt., b: 0.5 pts. c: 1 pt total: 2.5 points.)
We consider the energy constrained max-flow problem. We are given a directed graph (V, A), where V is the set of nodes and A is the set of arcs. There is a source node $s \in V$ and a sink $t \in V$. We can send flow over the arcs. For each node, except for the source and sink, the inflow has to be equal to the outflow. Each node i has a battery with capacity E_i. Sending flow on arc (i, j) requires energy from the battery at node i, which amounts e_{ij} per unit flow. The objective is to find the maximal flow s to t (the flow from s to t equals the net outflow of the source), where the flow is required to be integral and the battery capacities are respected.

(a) Give an integer linear programming formulation for this problem. Give a clear description of the decision variables, objective and constraints.

(b) Suppose that the price of energy from battery i equals ϵ_i per unit. The objective is to maximize the net revenue, where the revenue of the flow equals α times the net outflow of the source. Extend the formulation of part (a) to this situation. Give a clear description of the decision variables, objective and constraints.

(c) Now suppose that we first have to buy an arc before we can use it to transport flow. The cost for buying arc (i, j) are f_{ij}. The objective is to maximize the net revenue, where the revenue of the flow equals α times the net outflow of the source. Extend the formulation of part (a) to this situation. You can ignore the additions from (b), so $\epsilon_i = 0$. Give a clear description of the decision variables, objective and constraints.

Question 3: Scheduling with change-over times

(a: 0.5 pts., b: 1 pt. c: 0.5 pts. d:1 pt. total: 3 points.)

In this question you may use the fact that the following problems are $\mathcal{N}P$-complete: PARTITION, SUBSET SUM, HAMILTONIAN PATH, HAMILTONIAN CYCLE, TRAVELLING SALESMAN PROBLEM.

We consider the problem SCHEDULING WITH CHANGE-OVER TIMES. We are given a single machine on which we have to process n jobs denoted by $J_1, J_2, \ldots, J_n$. The machine is continuously available from time zero onwards and can process at most one job at a time. Each job requires an uninterrupted processing time of length p. However, before we can start a new job we have to adapt the configuration of the machine. This means that if we want to start job J_j after J_i we need a change-over time equal to s_{ij}. You may assume that the first job can be started at time 0. The objective is to mimimize the makespan, i.e. the completion time of the last job.

(a) Formulate the decision problem corresponding to SCHEDULING WITH CHANGE-OVER TIMES.

(b) Prove that SCHEDULING WITH CHANGE-OVER TIMES is $\mathcal{N}P$-complete.

(c) Describe two **construction** heuristics for the problem SCHEDULING WITH CHANGE-OVER TIMES.

(d) A manufacturer of soft drinks encounters the problem SCHEDULING WITH CHANGE-OVER TIMES in his weekly production planning and is looking for an appropriate solution algorithm. Explain the available options.

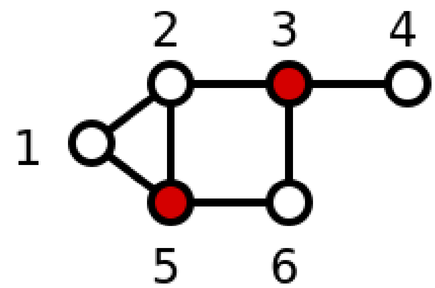

Figure 1: An example of a dominating set

Question 4: Dominating set
(total: 1.5 points)
In this question you may use the fact that the following problems are $\mathcal{N}P$-complete: CLIQUE, INDEPENDENT SET,VERTEX COVER, HAMILTONIAN PATH, HAMILTONIAN CYCLE.
We consider the problem DOMINATING SET. We are given a graph $G = (V, E)$. A subet D of the vertices is called a dominating set if each vertex not in D is connected to at least one vertex in D. Note that this is different from a vertex cover, which is a subet of the vertices such that each edge has one end point in the subet. In the example $\{3, 5\}$ is a dominating set, but not a vertex cover. Prove that the decision variant of DOMINATING SET is $\mathcal{N}P$-complete.

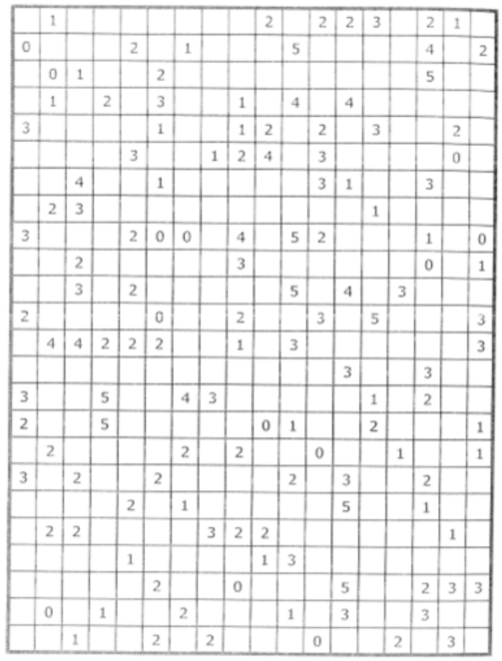

Figure 2: An example of treasure island

Bonus question 5: Treasure island
(1 point)
We consider the puzzle called 'Treasure island'. On this island some diamonds are buried, and we have to find them. The island is modelled by a grid of m rows and n columns. In each grid point (i, j) at most one diamond may be buried. For a subset G of the positions, we are given a number g_{ij} which equals the total number of diamonds buried in the positions directly surrounding (i, j) (note that for a point in the interior of the grid these are 8 positions). There are no diamonds on the positions in G. An example is depicted above. We consider Treasure island with pitfall. In this case *exactly one* of the numbers g_{ij} is wrong. Formulate an Integer Linear Programming problem that solves this variant of the puzzle. Note that this is in fact a feasibility problem, i.e. we want to decide if there is a feasible solution, so that we can choose the objective to be constant.