Utrecht University
Faculty of Science
Department of Information and Computing Sciences

Final Exam Algorithms for Decision Support, Monday November 5, 2018, 13.30-16.30 hr.

- Switch off your smart phone, PDA and any other mobile device and put it far away.
- A calculator is not necessary and you are **not** allowed to use one.
- This exam consists of **4** questions and a bonus question.
- Answers may be provided in either Dutch or English.
- 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.
- If you want to go to the bathroom, you have to hand in your smart phone.

Good luck, veel succes !

Question 1: The home care company

(a: 1 pt., b: 0.5 pt., c: 0.75 pt., d: 0.75 pt, total: 3 points.)

We consider a home care company in the center of the Netherlands. It employs 10 nurses and 20 assistants. There are n customers requiring care from the company. The care for a customer is performed in two steps:

1. Service from an assistant. The required time equals the maximum of 10 minutes and an amount following a normal distribution with an average of p_i. The average assistant service time equals 20 minutes and the variance equals v_i^2.
2. Service from a nurse. This takes an amount of time which equals the maximum of 5 minutes and an amount following a normal distribution with an average of q_i. The average nurse service time equals 10 minutes and the variance equals w_i^2.

The steps have to be performed consecutively, so a nurse can only start after the assistant has finished.

You may assume that the travelling time from one customer to the next follows a gamma distribution with an average of 15 minutes and a shape parameter $\alpha = 3$.

Furthermore, customers are assigned to assistants in increasing order of their index i. This implies that, if an assistant has finished her/his work, she/he is going to travel to the customer with lowest index which has not been assigned to an assistant yet. The company wants that the nurse arrives shortly after the assistant has finished. Therefore, a nurse is requested to start travelling 5 minutes after the start of the work of the assistant. If, at this point in time, there is no nurse available, the request is put into a FIFO queue.

The company wants to perform a simulation study to determine the *makespan*, i.e., the total time required to serve the complete set of the customers. Moreover, they want to determine the average service time of the customers.

(a) Which events are included in the event-scheduling model for this problem? Draw an event graph and for each arc give the corresponding time delay.

(b) Describe two *different* possible actions to validate the simulation model of the home care company. 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.

(c) Describe how we can generate the travelling times of the nurses and assistants 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.

(d) Suppose we have the following 5 samples of the service time of the nurses: 6, 7, 8, 10, 13. A consultant claims that the service times are uniformly distributed on $[5; 15]$. We want to test this claim with a Q-Q plot. What are the points in the Q-Q plot? Draw the Q-Q plot. Does the Q-Q plot support the claim or not? Explain your answer.
Observation: the number of samples is chosen artifically small for educational reasons.

Question 2: Beun de Haas *(a: 1 pt., b: 1 pt total: 2 points.)*
Beun de Haas is an independent entrepeneur, who performs small jobs for clients. We consider the planning for the next T days. On day t ($t = 1, \ldots, T$) Beun can spend at most Q_t time units on jobs for clients. Given is a set of n possible jobs where job j ($j = 1, \ldots, n$) has a reward of c_j when completed and takes a_j time units. If Beun decides to perform a job, he has to do the complete job on a single day. Because of the agenda of the clients, a job can only be performed on a given subset of days. We denote by J_t, the set of jobs that are available on day t. The objective is to make a workplan for Beun for T days that maximizes his reward.

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

(b) For some combinations of jobs, it is more efficient if they are planned on the same day. Let S be a set of job pairs which require less time if they are planned on the same day. For $(i, j) \in S$, we have that if job i and j are planned on the same day, this saves an amount of time equal to s_{ij}. Extend the model of part (a) to an integer linear programming model including this option. Clearly describe the decision variables, objective, and constraints.

Question 3: Load balancing

(a: 0.5 pt., b: 1 pt. c: 0.5 pt. d:1 pt. e: 0.5 pt. total: 3.5 points.)

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

We consider the following LOAD BALANCING problem. We are given n computation jobs. Job j has processing time p_j ($j = 1, \ldots, n$). These jobs have to be divided over m processors. The goal is to assign the jobs to the processors in such a way that $L_{\max}$ is minimized, where $L_{\max}$ equals the maximum workload of a processor. The workload of a processor is defined as the sum of the processing times of the jobs on the processor.

(a) Formulate the decision problem corresponding to LOAD BALANCING.

(b) Prove that the decision variant of LOAD BALANCING is $\mathcal{N}P$-complete.

(c) Describe two **construction** heuristics for LOAD BALANCING.

(d) Give an integer linear programming formulation for LOAD BALANCING.

(e) Now we are allowed to buy two additional processors. The first additional processor costs Q_1 and the second processor Q_2, where $Q_2 < Q_1$. The second additional processor can only be used in combination with the first. Extend the model of part (d) with the possibility of extra processors. The goal is to minimize the sum of $L_{\max}$ and the cost of the extra processors.

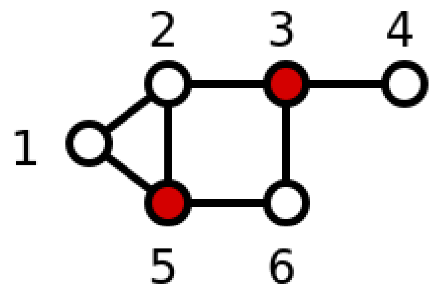

Figure 1: An example of a dominating set

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: PARTITION, SUBSET SUM, CLIQUE, INDEPENDENT SET,VERTEX COVER, HAMILTONIAN PATH, HAMILTONIAN CYCLE, TRAVELLING SALESMAN PROBLEM
We consider the problem DOMINATING SET. We are given a graph $G = (V, E)$. A subset 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 subset of the vertices such that each edge has one end point in the subset. 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.

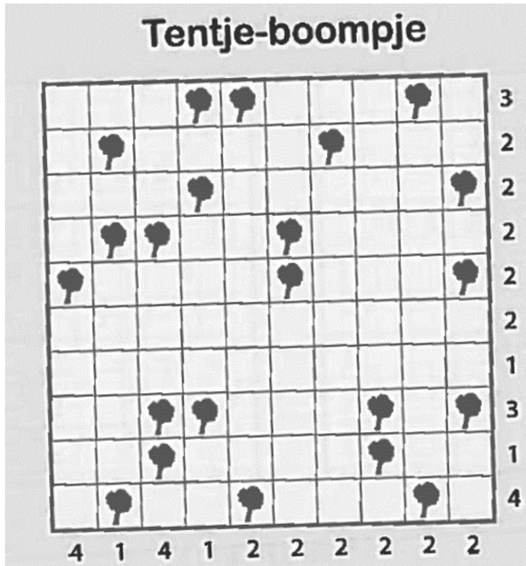

Figure 2: An example of Camp ground

Bonus question 5: Campground
(1 point)
We consider the puzzle called 'Campground', also known as *'Tentje-boompje'*. We have to find the positions of tents on a campground. The campground is modelled by a grid of m rows and n columns. When you want to stay at the campground, the owner directs you to 'your' tree and you must put your tent directly next to this tree (not diagonally). There can be at most one tent on a position. Different tents are not allowed to be on horizontally, vertically or diagonally adjacent positions. We are given numbers r_i and c_j, which denote the number of tents in row i and column j respectively. Formulate an Integer Linear Programming problem that solves the puzzle.