Exam Algorithms and Networks 2014/2015

This is the exam for part I of Algorithms and Networks.

You have two hours for the exam. You may give your answers in Dutch or in English. Write clearly. You may consult four sides of A4 with notes.

Results used in the course or exercises may be used without further proof, unless explicitly asked.

Switch of your mobile phone. Use of your mobile phone during the exam is stricktly forbidden.

Each of the five tasks: 1, 2, 3, 4, 5 counts for 2 points.

$\mathbf{Z}^+ = \{1, 2, 3, \ldots\}$.

Some parts are harder than others: use your time well, and make sure you first finish the easier parts!

If a polynomial time algorithm is asked, and your algorithm uses polynomial time, but less than (in O-notation) the optimal time, you do not score full points for the question, but will score a significant part of the points for the question.

Good luck!

Question 1: Some Short Questions. (2 points = 1+1)

1. **Explain the difference between a Hamiltonian Circuit and an Euler Tour.**

2. Consider the graph given below. We consider the minimum cost flow problem. Suppose arc (s, a) has capacity 1; arc (a, b) has capacity 2; arc (b, a) has capacity 3, and arc (b, t) has capacity 4. Suppose each arc has cost 5. Suppose we send 1 flow from s to t in this network. **Draw the residual network, and give for each arc in the residual network the cost and the capacity.**

Question 2: Nearest Neighbor (2 points = 1+1)

1. **Describe the Nearest Neighbor heuristic for TSP.**

2. Suppose we apply the Nearest Neighbor heuristic to an instance of TSP, formed by a collection of points in the plane, with Euclidean distances. **Is it possible that a tour with crossings is produced?** Explain your answer.

Question 3: Generalized matchings and degrees (2 points = 1+ 0.75 + 0.25)

Suppose we are given a bipartite graph $G = (L \cup R, E)$, such that all edges have one endpoint in L and one endpoint in R. Suppose that all vertices in L have degree $2d$, and all vertices in R have degree d.

We look for a set of edges $F \subseteq E$, such that each vertex $v \in L$, there are exactly two edges in F with v as endpoint, and for each vertex $w \in R$, there is exactly one edge in F with w as endpoint. We call such a set F a $2-1$-matching.

(i) **Give a proof that a 2-1-matching exists for G.**

(ii) **Argue that the problem, when given a bipartite graph G with degrees as described above, to find a 2-1-matching, can be solved in polynomial time.** Describe your algorithm in a clear way.

(iii) **How much time does your algorithm use as function of n and d?** Briefly explain your answer.

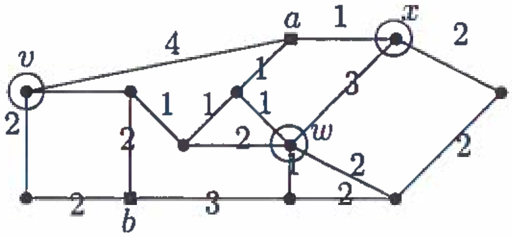
Question 4: Kitchen installers. (2 points)

The company Polynomial-Kitchen has a number of kitchen installers, and a number of clients. Each client has a house where a kitchen has to be installed. Each installer lives at a certain location. Kitchens take precisely one day to be installed. Each installer can install at most a specific number of kitchens.

- A map, represented as a graph, with lengths (in $\mathbf{Z}^+$) for the edges.
- Each client house is a vertex on the map.
- For each installer, there is a vertex on the map, with the location of the installer. Each installer has a *capacity*: a positive integer that tells what is the maximum number of kitchens he/she can install.

We want to assign each client to one installer: the installer that will install the kitchen at the clients house. This should be done such that

- For each installer, the number of clients assigned to him/her is at most his/her capacity.
- The total distance travelled by all installers is as small as possible.



Consider for instance the given map/graph, with a and b installers, and v, w, and x locations of clients. If a has capacity one, and b has capacity three, then the best solution is to assign x to a, and v and w to b: this gives a total distance of $2 * (1 + 4 + 4) = 18$. (From x to a takes 1 unit; from v to b takes 4 units; from w to b takes 4 units, and we multiply by two because the installers has to go to the location in the morning, and back home in the evening.)

(i) **Describe how this problem can be solved in polynomial time.** You may refer to results shown in the course, but must be clear in what you use.

(ii) **How much time does your algorithm use, in O-notation?**

Question 5: Stable Graduate-Job Opening Problem (2 points

We consider the problem of matching freshly graduated computer science students to job openings at different companies.

In this problem, we are given a set S of freshly graduated computer science students and a set J of job openings, with $|S| = |J| = n$, and we are looking to match the students to the job openings. Not every student can fill every job opening: some jobs require extensive algorithmic skills, other jobs require knowledge of software architecture, and again other jobs require proficiency with different programming languages. We formalise this using a set of forbidden matches $F \subseteq (S \times J)$: when $(s, j) \in F$, then student s is cannot fill job opening j.

Each student $s \in S$ has ordered all job openings that he or she can fill based on his/her preferences regarding to what job he/she would like to have. Similarly, each potential employer has ordered all the students that can fill each job opening on the preference of the employer.

An assignment A of students to job openings is a set of pairs (s, j) where student s is assigned to job opening j (so each student s and each job opening j occur at most once in a pair in A). Note that not all students and not all job openings need to be part of an assignment.

Given an assignment A of students to job openings, we call this assignment *stable* if none of the following conditions are satisfied:

1. There exists pairs (s, j) and (s', j') in A with the property that $(s, j') \notin F$ while student s prefers job j' to job j and the employer of job j' prefers student s to s' for this job.

2. There exist a pair $(s, j) \in A$ and a student s' that is not in any pair in A, with the properties that $(s', j) \notin F$ and that the employer of job opening j prefers student s' to student s for this job.

3. There exists a pair $(s, j) \in A$ and a job opening j' that is not in any pair in A, with the properties that $(s, j') \notin F$ and that student s prefers job opening j' to job opening j.

4. There exist a student s and a job opening j that both are not in any pair in A with the property $(s, j) \notin F$.

Give an algorithm that, for any given set of students S and job openings J, any given set of forbidden matches F, and any given preference list for both the students and job openings, produces a stable assignment of students to job openings.

Your algorithm must use polynomial time.