Technische Universität München, Zentrum Mathematik
Lehrstuhl für Operations Research

Advanced Graph Algorithms (MA5227)

Dr. Jannik Matuschke | M.Sc. Marinus Gottschau

Problem Set 1

Bonus system: *At the beginning of each tutorial class a list is handed out in which students may mark which homework exercises they have prepared and are willing to present their solution in class. Solutions for each homework exercise are then supposed to be presented by a randomly chosen student who marked that exercise. Obvious fraud, i.e. if the chosen student has not prepared the solution of the exercise, may lead to exclusion from the bonus. Each student that marks at least* 2/3 *of all homework exercises achieves a grade bonus, which improves the grade of a passed exam by one grade (e.g. a 4.0 becomes a 3.7, or a 3.7 becomes a 3.3, etc. Note, a 1.0 cannot be improved). The bonus applies to both, the final exam and the retake exam.*

Homework Exercise 1.1 (MaximumWeightBranchingProblem)
Consider the MaximumWeightBranchingProblem discussed in class. Prove that one cannot employ similar greedy ideas like adding edges in Kruskal's algorithm (i.e. add edges by increasing weight if they do not close a cycle) or like adding edges in Prim's algorithm (i.e. iteratively add an edge e from the cut* $\delta_G(S)$ of minimum weight) to solve the MaximumWeightBranchingProblem.
*Recall that a cut in a graph $G = (V, E)$ is an edge set of type $\delta_G(S) := \{\{u, v\} \in E : u \in X, v \in V \setminus X\}$ for some $\emptyset \neq X \subsetneq V$.

Homework Exercise 1.2 (BottleneckPathProblem)
Modify Dijkstra's algorithm in order to solve the BottleneckPathProblem: Given a digraph $D = (V, A)$, $w : A \to \mathbb{R}$, and two vertices $s, t \in V$, find a s-t-path P that minimizes the maximum weight on that path.
Also, prove correctness of your algorithm.

Homework Exercise 1.3 (NextToShortestPathProblem)
Given a digraph $D = (V, A)$, $l : A \to \mathbb{R}_+$ and two vertices $s, t \in V$.

a) Suppose there is a unique shortest s-t-path P. Can one find the next shorter s-t-path or decide that none exists in polynomial time?

Bonus*:

b) Assume $l(a) = 1$ for all $a \in A$. Can one compute a s-t-path that is longer than a shortest s-t-path or decide that none exists in polynomial time?

*You do not need to solve this exercise. Be aware that this is probably a hard exercise!

Please turn over.

Tutorial Exercise 1.4 (Another algorithm for the MinimumSpanningTreeProblem)
We consider an algorithm that colors a connected graph $G = (V, E)$ with weights $w : E \to \mathbb{R}$. Let $E = U \dot\cup B \dot\cup R$, where U is the set of uncolored edges, B is the set of blue edges, and R is the set of red edges.
The algorithm starts with the uncolored graph, i.e. with $U = V$, and iteratively applies in arbitrary order one of the following two rules and colors edges in red or blue until every edge is colored.
Blue rule: Select a cut $\delta(S)$ containing no blue edge and color $e' \in \delta(S) \cap U$ that minimizes $w(e')$ in blue, i.e. remove e' from U and add e' to B
Red rule: Select a cycle C containing no red edge and color $e' \in C \cap U$ that maximizes $w(e')$ in red, i.e. remove e' from U and add e' to R.

a) Prove that one of the two rules is always applicable as long as there are uncolored edges left.

b) Next, show that after each iteration there exists a minimum spanning tree containing all blue edges but no red edge and deduce that the algorithm computes a minimum spanning tree.

c) Finally, show that Kruskal's and Prim's algorithms are special cases of this algorithm.

Tutorial Exercise 1.5 (Floyd-Warshall algorithm for the AllPairsShortestPathsProblem)
Prove correctness of Floyd-Warshall algorithm for the AllPairsShortestPathProblem.

Algorithm 1: Floyd-Warshall algorithm

Input: A digraph $D = (V, A)$ with $V = \{1, \ldots, n\}$ and conservative weights $w : A \to \mathbb{R}$
Output: Matrices $(l_{ij})_{1 \le i,j \le n}$ and $(p_{ij})_{1 \le i,j \le n}$ where l_{ij} is the length of a shortest i-j-path, and (p_{ij}, j) is the final arc of such a path.

```
1 Set $l_{ij} := \begin{cases} 0 & \text{for } i = j \\ w((i,j)) & \text{for all } (i,j) \in A \\ \infty & \text{for all } (i,j) \in (V \times V) \setminus A \quad \text{with } i \neq j \end{cases}$
2 Set $p_{ij} := i$ for all $i, j \in V$
3 for $i \in \{1, \ldots, n\}$ do
4     for $j \in \{1, \ldots, n\} \setminus \{i\}$ do
5         for $k \in \{1, \ldots, n\} \setminus \{i\}$ do
6             if $l_{jk} > l_{ji} + l_{ik}$ then set $l_{jk} := l_{ji} + l_{ik}$ and $p_{jk} := p_{ik}$
```

This problem set will be discussed in the tutorials on November 21st, 2017.