École Polytechnique Fédérale de Lausanne
Prof. Eisenbrand

May 3, 2018

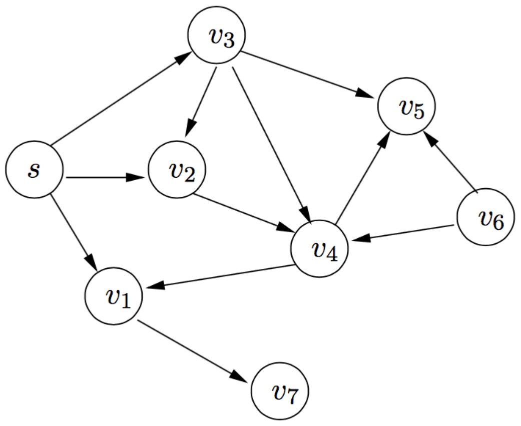
Discrete Optimization (Spring 2018)

Assignment 10

Problem 3 can be **submitted** until May 11, 12:00 noon, into the box in front of MA C1 563. You are allowed to submit your solutions in groups of at most three students.

Problem 1

In an unweighted graph (i.e., where all edges are of the unit weight) the shortest path from s to all other vertices can be computed by using the breadth-first search (BFS) algorithm. Apply it to the graph bellow.



Consider each iteration of the BFS (i.e. each time when a new vertex is taken from the head of the queue). Note which vertex has been currently processed, the distance labels and the snapshot of the queue at the end of the iteration.

Problem 2

There are n types of animals, and you want to assign them to two stables. Unfortunately, some animals would eat other animals when left unattended. Therefore you need to assign the animals carefully. There are m relations of the form "u eats v", where u and v are animals.
Find an $O(n + m)$ algorithm that decides whether there is an assignment of animals to the two stables such that no animal eats another one of the same stable, and outputs a feasible assignment.

Hint: Observe that the problem is equivalent to checking if the underlying (undirected) graph $G = (V, E)$ is bipartite, i.e., can V be partitioned into sets A and B so that there is no edge $\{u, v\} \in E$ such that $\{u, v\} \subseteq A$ or $\{u, v\} \subseteq B$? Modify the BFS algorithm seen in class so that it can be run even if G has several connected components (i.e. there is no path between each two nodes in G) and if G is undirected. Use this modified BFS to check if G is bipartite. Recall that a graph is bipartite if and only if it does not contain an odd cycle.

Problem 3 ($\star$)

Consider a directed graph $D = (V, A)$ with n vertices and m arcs. A toplogical sort of the vertices is a total ordering on V such that there is no arc $(u, v) \in D$ such that $u > v$, i.e, such that u is placed after v in the ordering.
Formulate an $O(m + n)$ algorithm that finds a topological sort of the vertices or decides that there is a directed cycle in G.

Problem 4

Let $D = (V, A)$ be a directed *acyclic* graph, i.e., there exists no directed cycle in D, and let $w : A \to \mathbb{R}$ be arc weights. Assume that you are given a topological sort of the vertices. Show how the above ordering can be used to compute single-source shortest paths, with respect to w, in $O(m)$ arithmetic operations.

Problem 5

Let $D = (V, A)$ be a directed graph, $w : A \to \mathbb{R}$ be arc weights and $s \in V$. Suppose that there exists a path from s to each other node of V.
Consider the folowing linear program:

$$\begin{aligned} \max \quad & \sum_{v \in V \setminus \{s\}} x_v \\ \text{s.t.} \quad & x_v - x_u \leq w(u, v), \quad \forall (u, v) \in A \\ & x_s \leq 0. \end{aligned} \tag{1}$$

Show the following:

a) This LP is feasible if and only if D has no negative cycle;

b) If D has no negative cycle, then (1) has a unique optimal solution.

Problem 6

Given n numbers $a_1, \ldots, a_n$ find indices i and j, $1 \leq i \leq j \leq n$, such that $\sum_{k=i}^{j} a_k$ is minimized. We will develop two algorithms for this problem that run in linear time, *i.e.*, the number of (arithmetic) operations is linear in n.

(a) Solve the problem using Bellman-Ford as a subroutine. In particular, construct a graph such that a shortest path in this graph yields the optimal solution to the above problem. Show that the graph can be generated in linear time and that Bellman-Ford can be implemented to run in linear time on this graph.

(b) Define $d(j) = \min_{1 \leq i \leq j} \sum_{k=i}^{j} a_k$. Conclude that the above problem is equivalent to computing $\min_{1 \leq j \leq n} d(j)$. Show how this can be done in linear time.