Communications and Transport Systems
Department of Science and Technology **Spring 2017**
Linköping University

Valentin Polishchuk
Christiane Schmidt

Design and Analysis of Algorithms Part 1 - Mathematical tools and Network problems homework 3, 10.04.2017

Problem 1 (Spanning tree):
The *tree graph* $T(G)$ of a connected graph G has a vertex for every spanning tree of G. Two of these tree vertices are adjacent if they have $|V| - 2$ edges in common. Show that $T(G)$ is connected.

Problem 2 (MSTs):
Let (G, w) be a network and v an arbitrary vertex. Show that each MST must contain an edge incident to v with the smallest weight of all edges incident to v.

Problem 3 (Shortest paths in graphs with arbitrary weights):
The algorithms by Dijkstra and Moore-Bellman-Ford for non-negative and conservative edge weights, repsectively, use Lemma 5.33.
Show that the minimum is not defined for general graphs, that is, show that there exist graphs with arbitrary real edge weights for which there is no shortest path between two vertices s and t.

Problem 4 (Longest paths and the knapsack problem):
We consider n objects. Each object has a *weight* a_j and a *value* c_j, both are positive integers. We ask for a subset of these objects, such that the sum of their weights does not exceed a bound b (the size of the knapsack) and the sum of their values is maximum (that is, we want to pack as valuable stuff as possible).
Reduce this problem to finding a longest path in an appropriate network.
Hint: Consider an acyclic network with a start vertex s, and end vertex t and $b + 1$ vertices for each object.

Problem 5 (Paths and cuts):
Let G be an undirected graph with weights $c : E(G) \to \mathbb{Z}_+$ and two vertices $s, t \in V(G)$, where t is reachable from s. Remember: for a vertex set $X \subseteq V(G)$ we call the edge set $\delta(X) = \{\{x, y\} \in E(G), x \in X, y \in V(G) \setminus X\}$ a cut in G. If $s \in X$ and $t \notin X$, $\delta(X)$ separates the vertices s and t.
Show that the minimum length of an s-t-path is equal to the maximum number of cuts that separate s and t, such that each edge e is contained in at most $c(e)$ such cuts.

(Hint: Why does it suffice to consider a graph with unit weights? Show that the maximum number of such cuts is as well an upper as a lower bound for the minimum length. For the proof of the lower bound give a set of cuts by using a BFS tree.)

Problem 6 (Independent sets):
An *independent set* (IS) or *stable set* is a set of vertices in a graph, no two of which are adjacent. An independent set that is not the subset of another independent set is called maximal. A maximal IS is a *dominating set*, that is,a subset D of V such that every vertex not in D is adjacent to at least one member of D.
Algorithm 1 computes a maximal independent set.

a) What is its runtime?

b) Show that Algorithm 1 does not compute a maximum independent set.

c) Show: If G can be vertex colored with k colors, there exists a vertex u with degree at most $\lfloor (1 - \frac{1}{k})|V| \rfloor$. (Hint: We do not know k, we only use that k is the optimal number of colors and that $k \geq 2$.)

d) Show: If G can be vertex colored with k colors, the size of the independent set found by Algorithm 1 is at least $\lceil log_k(|V|/3) \rceil$.

```
Algorithm 1: Greedy IS
Input  : Undirected graph $G = (V, E)$ with at least two vertices.
Output: A maximal independent set in $G$.
$U = \varnothing$
1.WHILE the graph is not empty
    Choose some vertex $u$ with minimum degree.
    Remove $u$ and all its neighbors from $G$.
    $U = U \cup u$.
RETURN $U$.
```

Problem 7 (Vertex coloring algorithm):
For any color, the vertices with this color form an independent set. Recall that we can find a maximal independent set in polynomial time
Consider Algorithm 2. Assume G can be colored with k colors.

a) Show that after at most $\frac{n}{1/2 log_k(n/16)}$ steps (maybe less!), at most $\frac{n}{log_k(n/16)}$ uncolored vertices remain.

b) Algorithm 2 uses at most $\frac{3n}{log_k(n/16)}$ colors.

c) How far off is Algorithm 2 from the optimum of k?

```
Algorithm 2: Greedy Vertex Coloring
Input  : Undirected graph $G = (V, E)$.
Output: A feasible vertex coloring of $G$.
1.WHILE the graph is not empty
        Determine a large independent set $U$ using Algorithm 1.
        Color all vertices in $U$ with the same color.
        Remove $U$ from $G$.
```

Problem 8 (Vertex coloring in planar graphs):
Let $G = (V, E)$ be a planar graph. Show: If G has no cycle of odd length, it is 2-colorable.

Problem 9 (Vertex coloring algorithm for planar graphs):
Consider Algorithm 3. We have to figure out that a vertex u as chosen by the algorithm, that is, a vertex of degree at most five, always exists.

a) Show: Let G be a connected planar graph, and let n, m and f denote the numbers of vertices, edges, and faces, respectively, in a plane drawing of G. Then $n - m + f = 2$.

b) Show: $m \leq 3n - 6$

c) Show: a vertex u as chosen by the algorithm always exists, that is, there is a vertex with degree at most 5.

d) How many colors does Algorithm 3 use at most?

```
Algorithm 3: Vertex Coloring for Planar graphs
Input  : Undirected planar graph $G = (V, E)$.
Output: A feasible vertex coloring of $G$.
1. IF two colors are sufficient (problem 8) DO color $G$ using two colors.
2. ELSE
        Find an uncolored vertex $u$ with degree at most 5.
        Remove $u$ and all its adjacent edges and color the remaining graph
        recursively.
        Insert $u$ and its adjacent edges back and color $u$ with a color that none of
        its neighbors has
```