Advanced Analysis of Algorithms - Final

K. Subramani
LCSEE,
West Virginia University,
Morgantown, WV
{ksmani@csee.wvu.edu}

1 Instructions

1. The final is to be turned in by $5:00$ *pm*, in class.
2. Each question is worth 4 points.
3. Attempt as many problems as you can. You will be given partial credit, as per the policy discussed in class.

2 Problems

1. Consider the following greedy algorithm for finding the Minimum Vertex Cover in an undirected, labeled graph $\mathbf{G} =< \mathbf{V}, \mathbf{E} >$: Select the vertex with the maximum degree and add it to the cover. In the event that more than one vertex has the maximum degree, choose the vertex with the smallest label. Now remove this vertex and all its incident edges from $\mathbf{G}$ (these edges are covered by the chosen vertex). Repeat this step till all edges are deleted from the graph. It is clear that this algorithm produces a cover. Will this algorithm produce a minimum size cover? Justify your answer with a proof or a counterexample.
2. In the stagewise shortest path problem, you are given a staged graph $\mathbf{G} =< \mathbf{V}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{k}, \mathbf{c} >$, where, $\mathbf{V}$ is the set of vertices, $\mathbf{E}$ is the set of edges, s is the source vertex, t is the sink vertex, k is the number of stages and $\mathbf{c}$ is a weighting function that associates a positive weight to the edges in $\mathbf{E}$. In stage 1, s is the only vertex; likewise, in stage k, t is the only vertex. All the other vertices are partitoned among the other stages, with each stage containing at least one vertex. Edges in the the graph are strictly from stage i to stage $(i+1)$, $1 \leq i \leq k-1$. Describe a linear time algorithm to determine the shortest path from s to t. You may assume that $|\mathbf{V}| = n$ and $|\mathbf{E}| = m$.
3. Consider Professor Boruvka's algorithm for determining the Minimum Spanning Tree in a weighted, undirected graph.

Function FIND-MST(**G=<V,E>**)
1: Let T be a subgraph containing only the vertices in $\mathbf{V}$.
2: **while** ($|T| \leq (n-1)$) **do**
3: **for** (each connected component C_i of T) **do**
4: Find the lightest edge $e = (u, v)$, with $u \in C_i$ and $v \notin C_i$.
5: Add e to T if it is not already in T.
6: **end for**
7: **end while**
8: **return**(T)

Algorithm 2.1: Boruvka's algorithm for Minimum Spanning Trees

Is the Professor's algorithm correct? Justify your answer.

4. In class, we showed that the HORNSAT and 2SAT problems were in P. Consider the following SAT variant called HORN$\oplus$2SAT; a CNF formula is said to be HORN$\oplus$2SAT, if every clause has exactly two literals or (inclusively) is HORN. Argue that checking the satisfiability of a HORN$\oplus$2SAT formula is NP-complete.
Hint: Reduce 3SAT to HORN$\oplus$2SAT.

5. In class, we showed that the Maximum Independent Set problem on an undirected graph $\mathbf{G} =< \mathbf{V}, \mathbf{E} >$ is NP-complete. What can you say about the complexity of the problem, if $\mathbf{G}$ does not have any cycles. Justify your answer with a polynomial time algorithm or a proof of NP-completeness.
Hint: Does Dynamic Programming work?