Problem Set 5, Revision

This problem set is due in class on Tuesday, Nov 23, except for CVN student who should mail it by Thursday, Nov 25.

When a problem asks to give an algorithm, in your solution: (i) describe shortly and informally the main ideas in your solution; (ii) give a detailed description of the algorithm, using a style similar to (but possibly more concise than) the pseudo-code of CLR; (iii) prove the correctness of the algorithm; (iv) prove a bound on the time complexity of the algorithm.

Problem 1.  [Edge-cover]

Given a bipartite undirected graph $G = (L, R, E)$, we want to find a smallest set of edges $E' \subseteq E$ such that every vertex of $L$ is covered by at least one edge of $E'$ and every vertex of $R$ is “covered” by at least one edge of $E'$. (We say that an edge $(u, v)$ “covers” the vertices $u$ and $v$.) We assume that every vertex in $G$ has degree at least one.

(a) Show that if $M$ is a matching in $G$, then there is an edge cover for $G$ that uses $\leq |V| - |M|$ edges.

(b) Show that if $E'$ is a edge cover in $G$, then there is a matching in $G$ that has $\geq |V| - |E'|$ edges.

(c) Show how to compute a minimum edge cover by doing one max matching computation on $G$ and some simple post-processing.

Problem 2.  [A Scheduling Problem]

A set of tasks $t_1, \ldots, t_n$ is to be executed on either of two machines $A$ and $B$. The time taken by task $t_i$ to be executed on machine $A$ (respectively, $B$) is $a_i$ (respectively, $b_i$). The times $a_i$ and $b_i$ are measured in seconds and are integers ranging in the interval $1, \ldots, M$.

A solution to this scheduling problem is a partition of $\{t_1, \ldots, t_n\}$ into sets $TA$ and $TB$, where $TA$ is the set of tasks assigned to machine $A$ and $TB$ is the set of tasks assigned to machine $B$. The time needed to complete all the tasks is then $\max(\sum_{i \in TA} a_i, \sum_{i \in TB} b_i)$.

Describe an algorithm having running time polynomial in $M$ and $n$ that partitions the tasks between the two machines so that $\max(\sum_{i \in TA} a_i, \sum_{i \in TB} b_i)$ is minimized.

[Hint: use dynamic programming.]
Problem 3. [Hamiltonian paths]

A *Hamiltonian path* in a graph is a simple path that visits every vertex of the graph exactly once.

Present a polynomial time algorithm that, on input a directed acyclic graph $G = (V,E)$ and a start vertex $s \in V$, finds the longest directed simple path in $G$ that starts at $s$, where the length of a path is the number of edges in the path. (Note that there is a Hamiltonian path in $G$ that starts at $s$ iff the longest path that starts at $s$ has length $|V| - 1$.)

[Hint: first do a topological sort, then use dynamic programming.]