CSEP 521 – Applied Algorithm
Spring 2003
Homework 2.

Due date: 4/14/03 (see submission instructions in course web-page).

1. (30 points) Definitions:

- A *bipartite* graph is a graph in which $V = X \cup Y$, $X \cap Y = \emptyset$ and each edge has one end vertex in X and one in Y.
- *A complete bipartite* is a bipartite in which $E = X \times Y$, that is, each vertex of X is connected to all the vertices of Y and vice-versa.
- A *lace* is a tree in which the degree of all vertices is 1 or 2 (note that this implies that a lace must be of the form o—o—o—o… o—o)
- Reminder: the DFS-tree is a directed spanning tree in which $i \rightarrow j$ is an edge iff i is the first vertex to 'discover' j in the DFS execution.
- An undirected graph is *a lace producer* if for any DFS execution (that is, for any starting vertex and any possible edge selection) the resulting DFS tree is a directed lace (i.e., of the form o→o→…→o→o→o).

For each of the following cases prove or show a counter example to the statement 'G is a lace producer'.

1. G is a lace
2. G is a simple cycle
3. G is a complete bipartite with $|U|=|V|+1$
4. G is a complete bipartite with $|U|=|V|$
5. G consists of two simple cycles with a single joint vertex.

2. (25 points) G=(V,E) is an undirected graph with weights on the edges. $s,t \in V$, $e \in E$
Give efficient algorithms for each of the following problems:

a. Does e belong to *all* the shortest paths connecting s and t ?
b. Does e belong to *some* shortest path connecting s and t ?

Prove that your algorithms are correct and analyze their complexity.

3. (25 points)

a. Describe an algorithm for finding the number of shortest paths from s to t after the BFS algorithm has been performed. Each vertex v is now labeled by d(v), which is the distance of v from s. Prove that your algorithm is correct.

b. Repeat the above, after Dijkstra's algorithm has been performed. Assume c(e) >0 for every edge e. Why is this assumption necessary?
You don't need to prove your algorithm.

4. (20 points) Use Bellman-Ford algorithm to suggest how we can detect in O(|E||V|) steps if a directed graph contains a negative cycle. Explain briefly.