CS 373: Combinatorial Algorithms, Summer IMCS 2000

`http://www-courses.cs.uiuc.edu/~cs373imc`

Homework 3 (due Tue Jul 4, 2000)

1. (8 pts) Hashing
   Professor Marley hypothesizes that substantial performance gains can be obtained if we modify a chaining scheme so that each list is kept in sirted order. How does the modification affect the running time for 1) successful searches, 2) unsuccessful searches, 3) insertions and 4) deletions. Gives details, justification and times for each one.

2. (4 pts) Cycle testing with an adjacency list
   Give an algorithm that determines whether or not a given undirected graph contains a cycle. Your algorithm should run in $O(V)$ time, independent of E (your time analysis should justify this).

3. (5 pts) Component graph
   Given a graph $G = (V, E)$, a *component graph* of G is a graph that has a single vertex for a strongly connected component in G, and an edge from u to v (in the component graph) if there is an edge from the component corresponding to u in G to that corresponding to v. Prove that the component graph has no cycles (i.e., is a dag).

4. (5 pts) Bottlenecked spanning tree
   Describe an efficient algorithm that determines the spanning tree of G such that the largest edge weight is minimum over all spanning trees of G.

5. (5 pts) Arbitrage
   CLR Problem 25-3 p. 546.