Midterm Exam, CS 360, 23rd October, 2001

There are four questions. Each is worth 25 marks. Calculators and electronic devices are not allowed.

1. *Dynamic Programming*

Let $G = (V, E)$ be an acyclic directed graph, given in adjacency list representation: for each $u \in V$ we have a list $Adj[u] = \{v \in V : (u, v) \in E\}$. Suppose that G has a unique source s.

a). (5 marks) Give pseudo-code for an $O(|V| + |E|)$ time algorithm that computes the list $Pred[v]$ for each vertex, where $Pred[v]$ is the list of vertices u such that $(u, v) \in E$.

b). (5 marks) Let $n(v)$ denote the number of paths from s to vertex v in G. We define $n(s) = 1$. For $v \neq s$, let $u_1, u_2, \ldots, u_k$ be the vertices in the list $Pred[v]$. Give a formula for $n(v)$ in terms of $n(u_1), n(u_2), \ldots, n(u_k)$.

c). (10 marks) Let $S(v)$ be defined by the recursion

$$S(v) = \begin{cases} 0 & \text{if } v = s; \\ \sum\limits_{u \in Pred[v]} (S(u) + n(u)) & \text{otherwise.} \end{cases}$$

Give pseudo-code for an $O(|V| + |E|)$ time algorithm that computes $S(v)$ for all vertices $v \in V$. You can assume that the values $n(v)$ have already been computed.

d). **(10 bonus marks)** Prove that $S(v)$ equals the sum of the lengths of all paths from s to v, where the length of a path is taken to be the number of edges along the path.

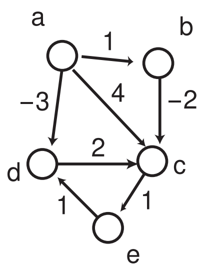
2. *Depth first search*

(25 marks) Given detailed pseudo code for an $O(|V| + |E|)$ time algorithm that uses depth first search to determine the components in an *undirected* graph G. The input to the algorithm is an adjacency list representation of the graph G. The output is a table c, where $c[v] = i$ if and only if v is in the ith component.

3. *Shortest paths*

a) (10 marks) Will Dijsktra's algorithm always compute minimum weight paths in a digraph with negative weight edges? Justify your answer with an argument or counter-example.

b) (15 marks) Apply Bellman-Ford's algorithm to the following digraph, starting at the vertex a.



At each pass through the main loop, record the value $d[a,u]$ by copying and completing the following table:

iteration	$d[a,a]$	$d[a,b]$	$d[a,c]$	$d[a,d]$	$d[a,e]$
initial	0	∞	∞	∞	∞
1					
2					
3					
4					

4. *Spanning trees*

In lectures, we proved a version of the following theorem:

Safe edge theorem
Suppose that set of edges $A \subseteq E$ is contained in some MST. Let $(S, V - S)$ be a cut and suppose that there are no edges in A going from one side of the cut to the other. If $\{u, v\}$ is an edge in E with minimum weight such that $u \in S$ and $v \in V - S$ then $A \cup \{\{u, v\}\}$ is contained in some MST.

a) (10 marks) Use the safe edge theorem to prove that if $\{u, v\}$ is an edge in $G = (V, E)$ with minimum weight then $\{u, v\}$ is in some MST for G.

Here is a version of Kruskal's algorithm for computing a minimum weight spanning tree (MST) on a weighted graph $G = (V, E)$. For each subset $A \subseteq E$ we let G_A denote the graph with vertex set V and edge set A.

$MSTKruskal(G, w)$
$A \leftarrow \emptyset$
sort E into a list $e_1, e_2, \ldots, e_m$ such that $w(e_1) \leq w(e_2) \leq \cdots \leq w(e_m)$.
for $i \leftarrow 1$ to m **do**
  if the endpoints of e_i are in different components of G_A
    $A \leftarrow A \cup \{e_i\}$
return A

b) (15 marks) Use induction to prove that after each iteration of $MSTKruskal$ the set A is contained in the set of edges of some MST of G.