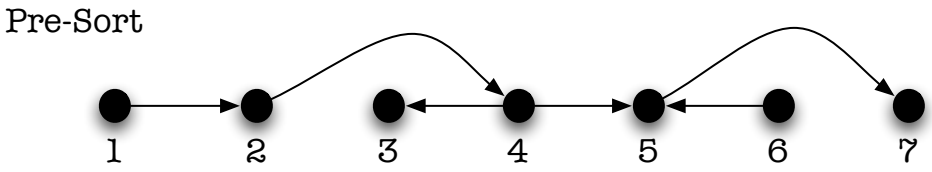
Nima Haghpanah, Tom Hayden, Michael Lucas

Randomized Algs Assignment 1

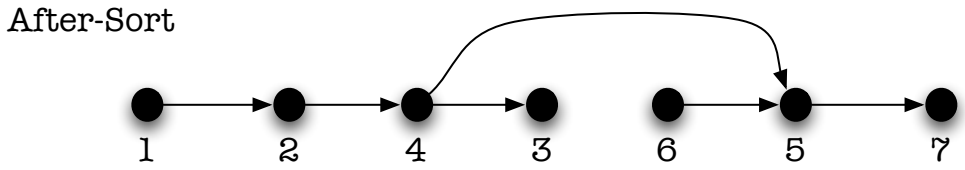
February 1, 2010

3. (a) We first need to topologically sort the vertices of the DAG. That is, order the nodes of the graph from left to right so that all the edges go from left to right. Since the graph is acyclic, there is a node with no incoming edges (otherwise we can follow the edges in the reverse order and find a cycle). We can number this vertex 1, and then recursively order the graph induced from removing 1 and all its edges.




**Claim 0.1** *We can do this efficiently in time $O(m)$, which is linear in the number of edges $m$.*

First, we find the degree of all nodes in time $O(m)$. Now let $S$ be the set of vertices with in-degree 0. Select any one of them to be the first node, and for each edge incident to it update the in-degree of the other end, and add that vertex to $A$ if its degree is 0. The number of updates is equal to the number of edges, so the algorithm takes time $O(m)$.




Now we show how to use the sorted graph to find the longest path. Let $\ell(i)$ show the longest path starting from vertex $i$. For any $j$ so that $(i,j) \in E$, we have $\ell[i] \geq 1 + \ell[j]$, since going to $j$ and then continuing from there is a valid path from $i$.

Now, Let the next vertex in the shortest path starting from $i$ be $j$. We can also see that $\ell[i] = 1 + \ell[j]$. This is because the same path minus the starting vertex is a valid path for $j$. So we have $\ell[i] = 1 + \max_{j:(i,j)\in E} \ell[j]$. So we can start from the rightmost vertex, and update the entries in $\ell$ vector. From the rightmore vertex, we can move leftwards considering the value of each $\ell$ long the way, checking the edges going out of that vertex. So the number of moves is equal to the number of edges. So the algorithm is $O(m)$.

(b) **Claim 0.2** *If $G$ has a simple path of length $k$, then the resulting DAG has a directed path of length at least $k$ with probability at least $\frac{2}{(k+1)!}$*

A path of length $k$ has $k+1$ vertices. Let the path be $i_1, i_2, \ldots, i_{k+1}$. There are two valid orderings of these vertices; One is $i_1, \ldots, i_{k+1}$, and the other is $i_{k+1}, \ldots, i_1$. There are $(k+1)!$ ways to order the vertices, and since all the orderings happen with equal probability (and the ordering of the other vertices does not matter), the probability of having the right ordering is $2/(k+1)!$.

(c) As seen in the previous part, the probability of making a mistake is $2/(k+1)!$ (drop the one and assume that it is $2/k!$). Now if we repeat the algorithm $k!$ times, the probability of error will be $(1-2/k!)^{k!} = (1/e)^2$. So we only need to check that $k!$ is polynomial in $n$ for $k = \log n / \log\log n$. This is true because:

$$\frac{\log n}{\log\log n}! \quad = \quad e^{\log \frac{\log n}{\log\log n}!} \simeq e^{\frac{\log n}{\log\log n} \log \frac{\log n}{\log\log n}} \leq e^{\frac{\log n}{\log\log n} \log\log n} = n$$

So we can have a constant error in polynomial time. Repeating this polynomial times, we get an exponential error in polynomial time.

(d) Add a new vertex $s$ to the graph and connect it to any other vertex with a new color. The new graph has a colorful path of lenght $k+1$ iff the old graph has one of length $k$. So we can assume that our problem is to find a colorful path (now assume with length $k$) from a certain vertex. Let $A(\ell, v)$ denote the set of colors of size $\ell$ from $s$ to $v$. That is, each $S \in A(\ell, v)$ is a subset of $1, \ldots, k$ of size $\ell$, which shows that there is a colorful path of length $\ell$ from $s$ to $v$ with colors $S$. Obviously we want to see if there is a vertex $v$ with $A(k, v) \neq \emptyset$. Now we will show how to compute the the vector $A$ in desired time. Assume the we have computed paths of length up to $\ell$. There exists a colorful path of length $\ell+1$ from $s$ to some vertex $v$, if and only if there exists a colorful path of length $\ell$ from $s$ to some other vertex $u$ so that there is an edge from $u$ to $v$ and the path from $s$ to $u$ does not use the color of vertex $v$. So we can write the recursive relation $A(\ell, u) = \bigcup_{v:(v,u)\in E, \exists S \in A(\ell-1,v), color(u)\notin S} (A(\ell-1, v) \cup color(u))$. In order to find all the paths of length $\ell$ using paths of length $\ell-1$, for each edge $(v, u)$ we have to check all the sets in $A(\ell-1, v)$. So it takes time $m2^k$ to update from $\ell-1$ to $\ell$. So the overall process takes time $m2^k k$. Assuming $k = O(\log n)$, this will be $O(mn \log n)$, which is polynomial.

In order to bound the probability of error, consider any path of length $k$. There are $(k+1)^{k+1}$ ways to color all the vertices, and $(k+1)!$ of these colorings are valid. So the probability of success is $(k+1)!/(k+1)^{k+1}$, which, using the power series expansion from Sterling's formula is approximately $(1/e)^{k+1}$. So the probability of error, if we repeat the process $e^k$ times, is equal to $(1-(1/e)^k)^{e^k} = 1/e$. The number of repetitions is $k = O(\log n)$ is $e^{O(\log n)} = O(n)$. So again we can have exponentially small error by repeating the algorithm polynomial number of times.