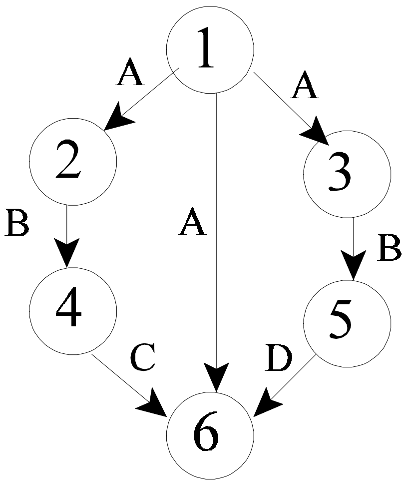
Edward D. Baum
CS 280A
11/2/00

Homework #2 Problem #5 Solution

Part a:

We find such a path recursively. Call a path on edgws σ_1 to σ_i ending on vertex v P(i,v) and an edge from u to v with label σ_k $\sigma_k(u,v)$. Then $P(i,v) = P(i-1,u) + \sigma_i(u,v)$ for some u. Thus we can find the whole path by finding first the path s by finding a path from our distinguished vertex without the final edge(sound) in s leading to some node from which there is the final edge in this path. We can do this in a dynamic programming manner by finding all paths from the distinguished vertex of length one whose edge is the first edge in s, then seeing if we can find the second edge in s leading from the ends of these paths to some node, then the third from these paths, etc. The table we will use to do this has one column for each node, and one row per edge in s. To fill out the first row we mark with the symbol for the distinguished vertex each column in which there is an appropriate edge from the distinguished vertex to the vertex represented with that row, all other columns are marked with NO_PATH (0). For subsequent rows (say i) we mark each column with the label of a vertex from which there is an appropriate edge (σ_i) which leads from that vertex to the one this column represents. If there is none NO_PATH (0) is marked. In the end we check if there are any non-NO_PATH entries in the last row. If there aren't then no path exists. Otherwise we can fins the path corresponding to s by tracing back. Here is an example.



Consider s =<A,B,C> and 1 as the distinguished vertex, then the first row of the table consists of 0,1,1,0,0,1 as vertices 2, 3, and 6 match the first symbol of s and are of course reached from the distinguished vertex. The second row is 0,0,0,2,3,0 as vertex 4 can continue the path with the label B from vertex 2, similarly for 5 and 3. The last row, corresponding to C, is 0,0,0,0,0,4 as this is the only continuation of a path. So by following back we can find the path is 6,4,2,1 reversed, so 1,2,4,6.

The running time for this algorithm is $O(n^2k)$ as for each element in the table we must scan a previous row of length n, we must do this n times per row, and we have k rows.

Part b:

The extension is simple. We merely keep track of the probability of the path so far in each cell along with the vertex that we came from. If there is a choice ofvarious methods to get to a vertex we choose the one which will have the highest probability to that point. At the end we merely find the entry in the last row with the highest probability.