The Bellman-Ford Algorithm

Space Optimization

Algorithms: Design and Analysis, Part II

Quiz

Question: How much space does the basic Bellman-Ford algorithm require? [Pick the strongest true statement.] [m = # of edges, n = # of vertices]

A) $\boxed{\Theta(n^2)}$ $\rightarrow \Theta(1)$ for each of n^2 subproblems

B) $\Theta(mn)$

C) $\Theta(n^3)$

D) $\Theta(m^2)$

Predecessor Pointers

$$A[i,v] = \min \left\{ \begin{array}{l} A[i-1,v] \\ \min_{(w,v)\in E}\{A[i-1,w] + c_{wv}\} \end{array} \right\}$$

Note: Only need the $A[i-1,v]$'s to compute the $A[i,v]$'s.

$\Rightarrow$ Only need $O(n)$ to remember the current and last rounds of subproblems [only $O(1)$ per destination!]

Concern: Without a filled-in table, how do we reconstruct the actual shortest paths?

Exercise: Find analogous optimizations for our previous DP algorithms.

Computing Predecessor Pointers

Idea: Compute a second table B, where $B[i, v] =$ 2nd-to-last vertex on a shortest $s \rightarrow v$ path with $\leq i$ edges (or NULL if no such paths exist)

("Predecessor pointers")

Reconstruction: Assume the input graph G has <u>no negative cycles</u> and we correctly compute the $B[i, v]$'s.

Then: Tracing back predecessor pointers – the $B[n-1, v]$'s (= last hop of a shortest s-v path) – from v to s yields a shortest s-v path.

[Correctness from optimal substructure of shortest paths]

Computing Predecessor Pointers

Recall:

$$A[i,v] = \min \left\{ \begin{array}{ll} (1) & A[i-1,v] \\ (2) & \min_{(w,v)\in E}\{A[i-1,w] + c_{wv}\} \end{array} \right\}$$

Base case: $B[0,v] =$NULL for all $v \in V$

To compute $B[i,v]$ with $i > 0$:

Case 1: $B[i,v] = B[i-1,v]$

Case 2: $B[i,v] =$ the vertex w achieving the minimum (i.e., the new last hop)

Correctness: Computation of $A[i,v]$ is brute-force search through the $(1+$in-deg$(v))$ possible optimal solutions, $B[i,v]$ is just caching the last hop of the winner.

To reconstruct a negative-cost cycle: Use depth-first search to check for a cycle of predecessor pointers after each round (must be a negative cost cycle). (Details omitted)