Lecture 31

CSE 331

Nov 11, 2019

HW 8 Q3 reminder

Homework 8

Due by **11:00am, Friday, November 15, 2019**.

! Note on Timeouts

For this problem the total timeout for Autolab is 480s, which is higher the the usual timeout of 180s in the earlier homeworks. So if your code takes a long time to run it'll take longer for you to get feedback on Autolab. **Please start early to avoid getting deadlocked out before the submission deadline.**

Also for this problem, `C++` and `Java` are way faster. The 480s timeout was chosen to accommodate the fact that Python is much slower than these two languages.

Question 1 (Finding a sink) [50 points]

The Problem

Given a directed graph $G = (V, E)$, a vertex $s \in V$ is called a **sink** if there are incoming edges from every other vertex to s but no outgoing edge from s, i.e. $|\{(u, s) \in E\}| = |V| - 1$ and $|\{(s, u) \in E\}| = 0$.

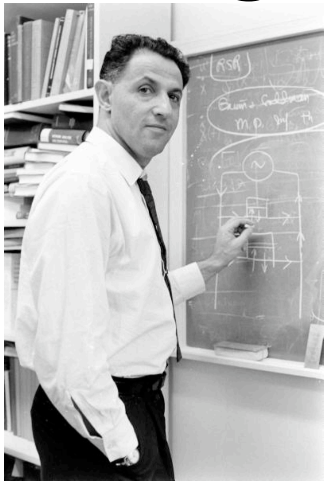
The goal of this problem is to design an algorithm to find out if G has a sink and if so, to output it. (Recall that $n = |V|$). Your algorithm is given G in its adjacency matrix A (i.e. if an ordered pair $(u, v) \in E$, then $A[u][v] = 1$ and if $(u, v) \notin E$, then $A[u][v] = 0$).

Sample Input/Output pairs

Here are two sample input/output pairs (input is the matrix, with vertex set $\{u, v, x, y, z\}$ and the rows (top to bottom) and column (from left to right) are in the order u, v, x, y, z) and the output is a vertex (if it is a sink) or `null` otherwise):

- Input:

When to use Dynamic Programming



Richard Bellman

There are polynomially many sub-problems

OPT(1), …, OPT(n)

Optimal solution can be computed from solutions to sub-problems

$$OPT(j) = \max \{ v_j + OPT(\ p(j)\ ),\ OPT(j-1) \}$$

There is an ordering among sub-problem that allows for iterative solution

OPT (j) only depends on OPT(j-1), …, OPT(1)

Scheduling to min idle cycles

n jobs, i^{th} job takes w_i cycles

You have W cycles on the cloud

What is the maximum number of cycles you can schedule?

Subset sum problem

Input: n integers $w_1, w_2, \ldots, w_n$

bound W

Output: subset S of $[n]$ such that

(1) sum of w_i for all i in S is at most W

(2) $w(S)$ is maximized

Questions?

Today's agenda

Dynamic Program for Subset Sum problem

Recursive formula

$OPT(j, B)$ = max value out of $w_1,..,w_j$ with bound B

If $w_j > W'$

$OPT(j, B) = OPT(j-1, B)$

else

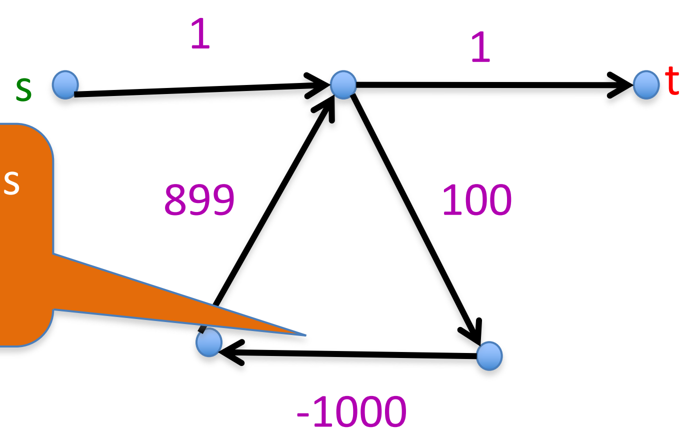
$OPT(j, B) = \max \{ OPT(j-1, B), w_j + OPT(j-1, B-w_j) \}$

Shortest Path Problem

Input: (Directed) Graph G=(V,E) and for every edge e has a cost c_e (can be <0)

t in V

Output: Shortest path from every s to t



Shortest path has cost negative infinity

Assume that G has no negative cycle

May the Bellman force be with you