Lecture 31

CSE 331

Apr 17, 2020

Video mini project reminder

- Due April 20 (next Monday)
- 3 days
- Work with your teammates

- You need to submit one PDF file to Autolab.
- The only thing the PDF needs to have is the link to your video.
- Each group member must submit the exact same PDF

HW 8 out (Q3 reminder)

Homework 8

Due by **8:00am, Monday, April 27th, 2020**.

Make sure you follow all the homework policies.

! 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.

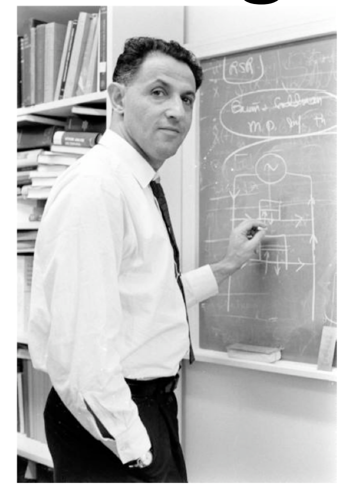
Question 1 (Finding a sink) [50 points]

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 $A[u][v] = 0$).

When to use Dynamic Programming



Richard Bellman

There are polynomially many sub-problems

$$\text{OPT}(1), \ldots, \text{OPT}(n)$$

Optimal solution can be computed from solutions to sub-problems

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

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

Today's agenda

Dynamic Program for Subset Sum problem