Lecture 32

CSE 331

Nov 16, 2022

HW 7 out

Homework 7

Due by **11:30pm, Tuesday, November 29, 2022**.

Make sure you follow all the homework policies.

All submissions should be done via Autolab.

Question 1 (Ex 1 in Chap 6) [50 points]

The Problem

Exercise 1 in Chapter 6. The part `(a)` and `(b)` for this problem correspond to the part (**(a)**+**(b)**) and part **(c)** in Exercise 1 in Chapter 6 in the textbook (respectively).

Sample Input/Output

See the textbook for a sample input and the corresponding optimal output solution.

! 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.

CSE 331 final exam conflict

note @432

stop following | 1 view

Actions

CSE 331 final exam conflict

If you do not have a final exam conflict with CSE 331, you can ignore this post.

However, if

- You do have a final exam conflict with CSE 331 AND
- You would like to request to move your CSE 331 final exam

please **request moving your CSE 331 exam by email to me by Monday, November 28**. *Any requests after that I can give you no guarantees.*

Also note that the alternate CSE 331 exams will be **before** the in-class exam-- i.e. on Th or Fri in last week of class.

final

Edit | good note | 0

Updated 11 seconds ago by Atri Rudra

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

Recursive formula

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

If $w_j > B$

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

else

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

Questions?

Algo run on the board…

Recursive formula

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

Can compute final S with recursion/ backtracking

If $w_j > B$

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

else

j not in OPT

j in OPT

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

Knapsack problem

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

n pairs $(w_1, v_1), \ldots, (w_n, v_n)$,

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

(2) $v(S)$ is maximized

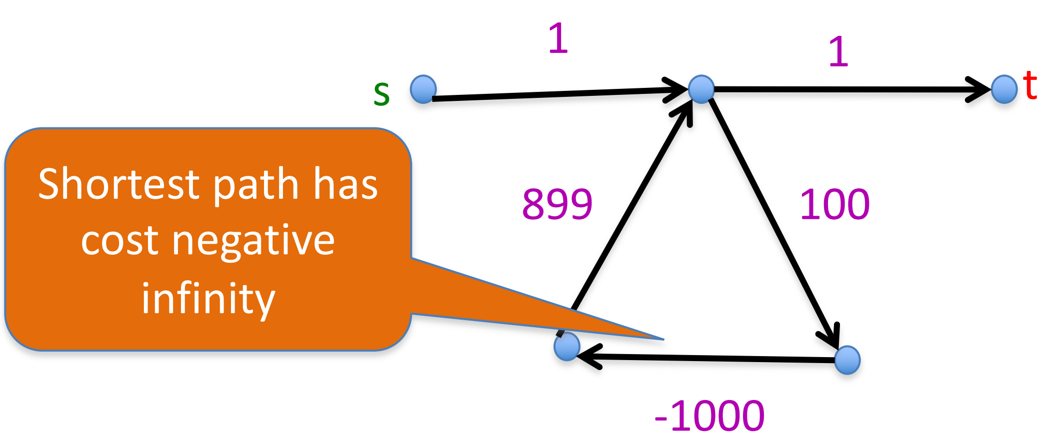
Questions?

Shortest Path Problem

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

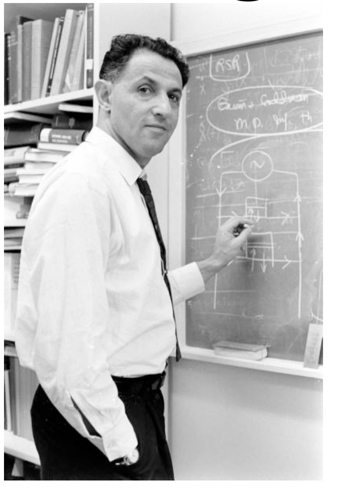
t in V

Output: Shortest path from every s to t



Assume that G has no negative cycle

When to use Dynamic Programming



Richard Bellman

There are polynomially many sub-problems

Optimal solution can be computed from solutions to sub-problems

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

Rest of today’s agenda

Bellman-Ford algorithm