Lecture 32

CSE 331

Apr 20, 2020

Give feedback!

note @648 stop following 64 views

Feedback on CSE 331

Hi All,

I'm asking for your feedback about 331. I prepared a form with custom questions. Please do give feedback via this anonymous form: https://forms.gle/qxp4XPfv2idLn6zj7

Filling in this form is **completely optional and anonymous**.

In particular, I would love feedback (even if it is critical). I might also disagree with your feedback but after a week or so, I'll post my response to the feedback from y'all. So at the very least, y'all would get to hear my reasoning for why certain things are the way they are in CSE 331. And then we can agree to disagree :-)

Note that this is NOT the UB's course evaluation form; I'll use the results to improve the class this semester and in future offerings.
#pin

logistics

edit · good note | 0 Updated 17 hours ago by A. Erdem Sariyuce

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

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

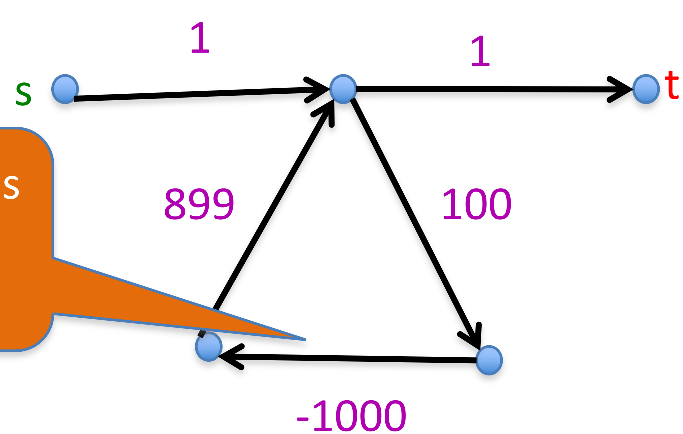
(2) $v(S)$ is maximized

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

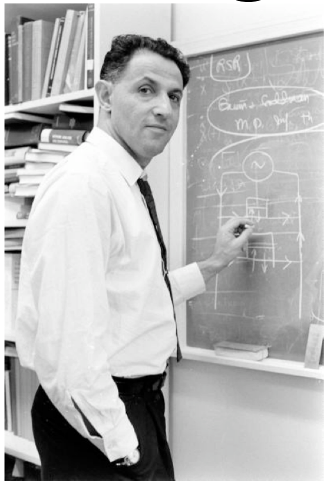
Output: Shortest path from every s to t



Shortest path has cost negative infinity

Assume that G has no negative cycle

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

Today’s agenda

Bellman-Ford algorithm