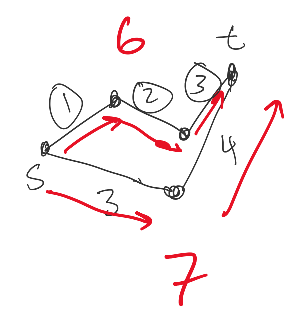
10s: Bellman-Ford (Shortest Path)
Thursday, April 15, 2021
4:52 PM

Shortest Path Problem

Input: Graph $G=(V,E)$ $w: E \to \mathbb{Z}$ $s,t \in V$
Output: Shortest path from s to t in G
⇓
sum edge weights on path to get length



3 approaches

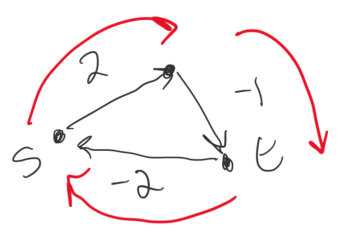
- Dynamic Programming
- Greedy
- Brute Force

Bellman-Ford: used for graphs with
- directed or undirected graphs
- positive or negative weights
- no negative cycles
- global or distributed description of G

adj mat.
adj. list

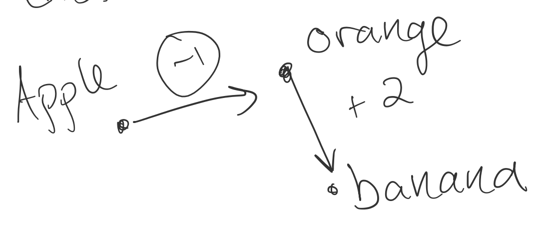
internet

Negative Cycle:



Applications

- Data routing in distributed networks
- Bartering

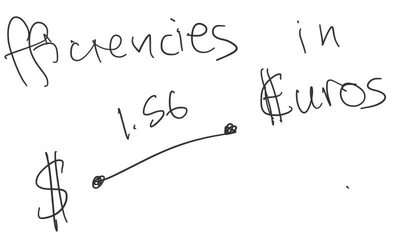


Beneficial
- Efficient
- Faster
- Share paths?
- Priorities?
- Traffic jams?

Map routing
- Some streets might not want or appropriate traffic

Who has access algorithm? Unf advantage.

- Arbitrage → finding inefficiencies in currency trading markets



Goals:
- Design a dynamic programming algorithm for shortest path

Questions:
- Can any question become Yes-No? Yes! Sort?
- How are weights decided?

Announcements:
- Rising Seniors: Fill out 701 survey:

https://docs.google.com/forms/d/e/1FAIpQLSe8olsX2IAB5YdYxnx2feQJ6Qz1sROqxFtqEIH94_rLXMX-8w/viewform?usp=sf_link

Can you sort this list in T steps?
Does it provide this (a, c, d, e, f) order?
Does the output strictly increase? ←
Is there an ordering where ordering is strictly increasing?
Is "a" the 2nd elt of the sorted list?
Is the n^{th} bit of the sorted output 1?

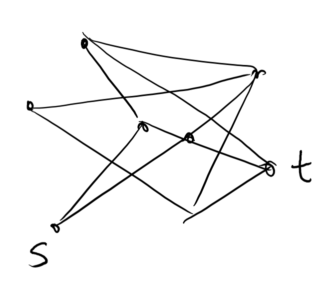
1001 . . .

Knapsack ∈ NP
What y could I check to convince me that x is a yes instance. $(c_1, c_2, \ldots c_n, v_1, \ldots v_n, C, V)$ ← Yes?
y = set of items that fit in knapsack + have value greater than V.

[Harm/Benefit? + Something you are looking forward to doing in warmer weather]
- Knapsack ∈ NP

Designing D.P.

Think about solution as a sequence of choices. What is final choice? Create a recurrence in terms of smaller subproblems.



Final Choice?

	Option	Only option if
$P_{s,t} = P_{s,u} + (u,t)$ ←	• u was last before t	$(u,t) \in E$
$P_{s,t} = P_{s,v} + (v,t)$	• v " " t	$(v,t) \in E$
	⋮	⋮
$P_{s,t} = P_{s,s} + (s,t)$ (with $P_{s,s}$ crossed out, ∅)	• s was last before t	$(s,t) \in E$

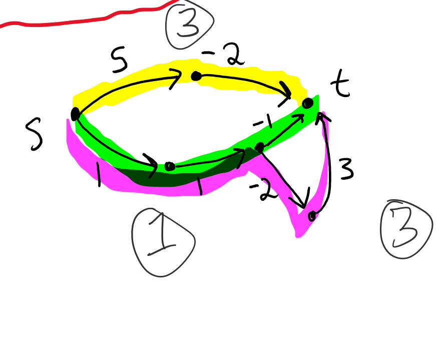
Subproblem: $P_{s,u}$ = shortest path from s to u

Difficulty: $P_{s,u}$ is not a smaller subproblem
⇓
never reach base case

Idea: Limit # of edges on our path

What is the length of the shortest path from s to t with at most 2 edges? 3 edges? 4 edges?

A) 3, 1, 1 (circled) B) 3, 1, 3 C) 3, 2, 1 D) 2, 3, 4



Define $P_{v,i}$ = shortest path (adding weights on edges) from s to v with at most i edges.

Note: a path with n or more edges must contain a cycle. Since there are no negative cyles → optimal path will never have more than n-1 edges.

Group work

1. $P_{v,i} = \begin{cases} \_\_\_\_\_\_, & \text{if shortest path with } i \text{ edges goes through } u \text{ immediately prior to } v \\ \_\_\_\_\_\_, & \text{" } w \text{ "} \\ \vdots & \vdots \\ \_\_\_\_\_\_, & \text{if shortest } s \to v \text{ path with at most } i \text{ edges uses fewer than } i \text{ edges.} \end{cases}$