6.006 Lecture 15: Shortest Paths (Intro)

- Motivation
- Weighted graphs
- General Structure of S.P. algorithms
- Optimal substructure

Motivation

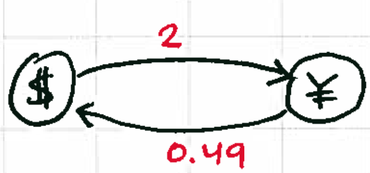
• PS5: find shotest path from Caltech To MIT.

-Map is a graph of road segments

-last segment is 0.7 mile on Mass Ave, but some interstate segments are many miles long; we clearly have to represent length of road segment in the graph.

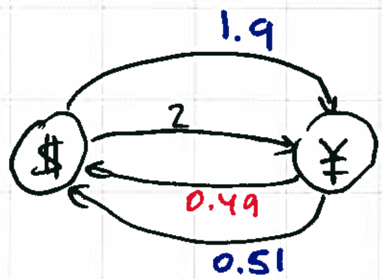
• Arbitrage

I have $1 and can convert it to Yens at the local bank; I get ¥2. For each ¥, I can get $0.49 (the bank needs to make money).



If I traverse the cycle, I end back with $'s, but only $0.98: $1 \times 2 \times 0.49 = 0.98$.

Not good. But suppose that a bank in Japan has slightly different exchange rates:



Now there is a path from $ to $ (a cycle) that makes me $0.02 on the dollar. People exploit that and this is what causes exchange rates to converge.

Clearly, weighted graphs are useful...

Definitions

A directed graph $G(V,E)$

Edges have weight $w(u,v)$ $(u,v \in V,$
$(u,v) \in E)$

Weight is a function $w: E \rightarrow \mathbb{R}$

Path $p = \langle v_0, v_1, v_2, \cdots, v_k \rangle$ where

$(v_i, v_{i+1}) \in E$ for $0 \le i \le k$; $v_0 \overset{p}{\rightsquigarrow} v_k$

Weight of a path $w(p) = \sum_{i=0}^{k-1} w(v_i, v_{i+1})$

Shortest paths:

$$\delta(u,v) = \begin{cases} \min_p \{ w(p) : u \overset{p}{\rightsquigarrow} v \} & \\ \infty & \text{if } v \text{ is unreachable from } u \\ & \text{(set above is empty).} \end{cases}$$

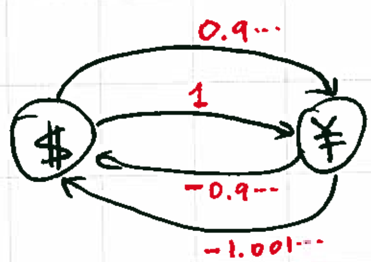
Back to Modelling

This clearly works for geographic routes:

we want to minimize the sum of the segment lengths on the path.

But not for arbitrage: there we multiplied the rates.

Solution: use $w(u,v) = \log(\text{exchage rate})$



Now an additive cycle with negative weight means we can make money.

Single-Source Shortest Paths

Input: $G=(V,E)$, w, a source vertex s

Goal: Find $\delta(s,v)$ for every $v \in V$

and the best path from s to v

Data structures:

$d[v]$ = length of best path to v

<u>so far</u>

initialization: $d[v] = \begin{cases} 0 & v = s \\ \infty & \text{otherwise} \end{cases}$

during the algorithm, $d[v] \geq \delta(s,v)$

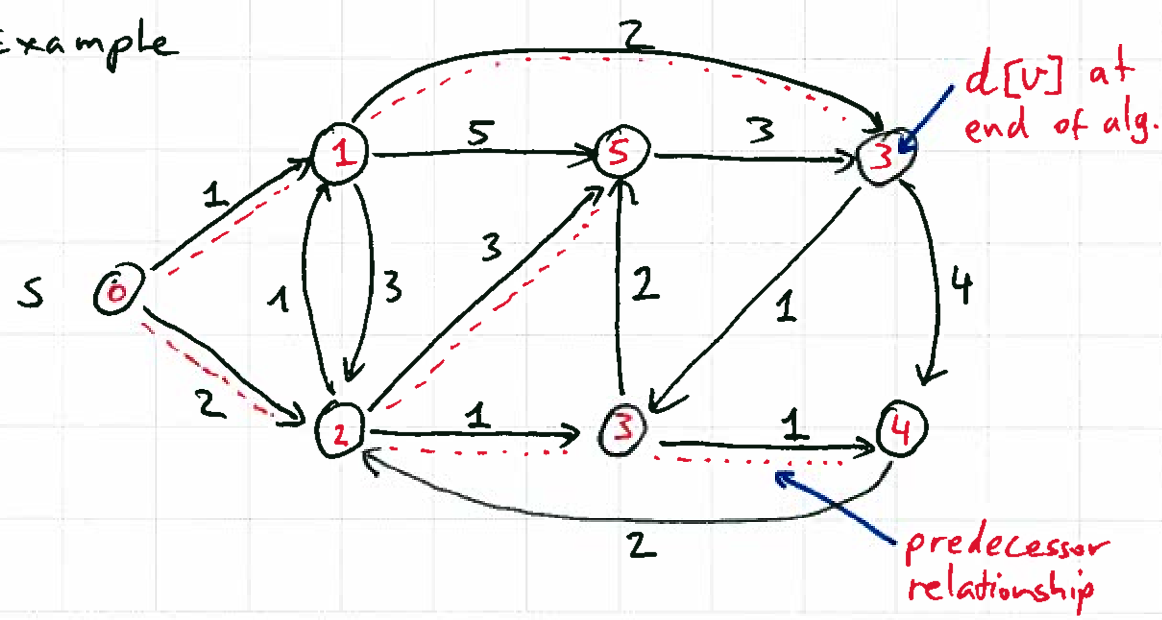
when the algorithm terminates,

$d[v]$ will be exactly $\delta(s,v)$

$\pi[v]$ = predecessor of v on best path

so far, initialize $\pi[v]$ = None.

Example



General structure of SSSP algorithms

Based on the observation that each $(u,v) \in E$ is a constraint

$$\delta(s,v) \leq \delta(s,u) + w(u,v)$$

We start with some $d[v]$ that satisfies constraints (and $\delta(s,s)=0$) but is not minimal, and relax the constraints until we are (hopefully) done

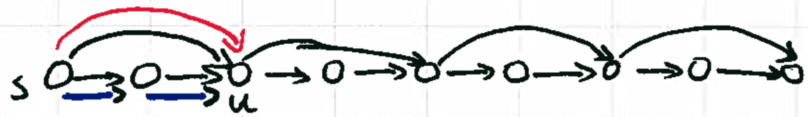
nondeterministic choice, not a real algorithm yet!!!

```
for v in V:
    d[v] = ∞
    π[v] = None
d[s] = 0
while d[v] > d[u] + w(u,v) for some v:
    d[v] = d[u] + w(u,v)
    π[v] = u
```

Does not terminate if ∃ negative cycle (need negative cycle detection)

Complexity may be exponential!

Complexity (bad case):



use $\rightarrow$ to set $d[u]$ and recurse from u, then

use $\rightarrow\rightarrow$ to lower ~~d[u]~~ $d[u]$ and recurse again.

$T(n) = 3 + 2\,T(n-2) = \Theta(2^{n/2})$

We'll see two algorithms that do much better

Bellman-Ford $O(VE)$, detects negative cycles

Dijkstra $O(V \lg V + E)$ but assumes that there are no negative cycles.

Two Structural Properties

Theorem: subpaths of shortest paths are also shortest paths.

Proof: Let $p = \langle v_0, v_1, \cdots v_i, v_{i+1}, \cdots v_j, \cdots v_k \rangle$

$$p_{ij} = \langle v_i, v_{i+1}, \cdots, v_j \rangle$$

$$\begin{tikzcd} p = v_0 \arrow[r, squiggly, "p_{0i}"] & v_i \arrow[r, squiggly, "p_{ij}"] \arrow[r, bend right, "p'_{ij}"'] & v_j \arrow[r, squiggly, "p_{jk}"] & v_k \end{tikzcd}$$

If $w(p'_{ij}) < w(p_{ij})$ then p is not shortest.

we can replace p_{ij} with p'_{ij} and get

p $v_0 \overset{p'}{\rightsquigarrow} v_k$ with $w(p') < w(p)$.

Theorem: triangle inequality, for all $u, v, x \in V$

we have $\delta(u,v) \le \delta(u,x) + \delta(x,v)$

Proof: