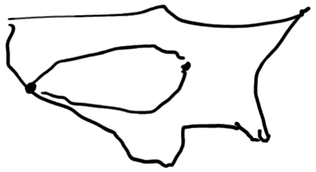
cs473 Algorithms : Lecture 6 (2022-02-03)

logistics: - pset 1 due F17, groups ≤ 3
- pset 2 out F17
- in person next week

last lecture: dynamic programming
- edit distance
  - $O(nm)$ time, $O(nm)$ space - value, - alignment
  - ——— , $O(n+m)$ space - value, - alignment

today: dynamic programming

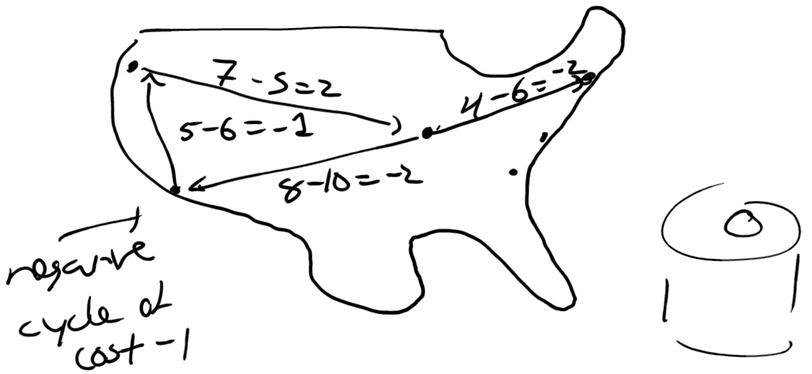
Q: resolve supply chain costs?



Q: shortest path?

physical costs are often (non-negative)

Q: make money to ——— ?



⇒ shortest paths are [not] definable

⇒ negative cycles are definable, repeatable arbitrage

Q: compute shortest paths in graphs w/ negative edge weights?

def: $G=(V,E)$ directed (simple) graph

$|V|=n$

$|E|=m$

↳ cost function $c: E \to \mathbb{Z}$

A **path** in G is a sequence of vertices

$v_1 \to v_2 \to \cdots \to v_k$ s.t. $(v_i, v_{i+1}) \in E$ all i

↳ is **cycle** if $v_1 = v_k$

the **cost** of path p is $|p| = \sum_{e \in p} c_e$

the distance from $s \in V$ to $t \in V$ is

$$dist(s,t) = \min_{\substack{\text{paths} \\ p: s \to t}} |p|$$

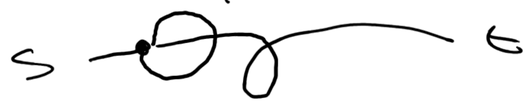
the argmin is the **shortest path**

rmk: no $s \to t$ path $\Rightarrow dist(s,t) = \infty$

lem: if exists $s \to t$ path sharing a vertex v a cycle C w/ $|C| < 0$, then $dist(s,t) = -\infty$

pf:

$|C| < 0$



$p + C$

$p + 2C$

so $p + k \cdot C$ is an $s \to t$ path

w/ cost $|p| + k|C| \xrightarrow{k \to \infty} -\infty$

($|C| < 0$)

Lem: no $s \to t$ path touching negative cycle

$\Rightarrow$ shortest path exists has $\le n-1$ edges

pf: $\le n-1$:

Clm: any finite p $s \to t$ exists p' $s \to t$ $\rightsquigarrow$ p' has $\le n-1$ edges. $(|p'| \le |p|)$

pf: Clm: path q $\rightsquigarrow$ $\ge n$ edges $\Rightarrow$ q has repeated vertex

pf: $q = v_1 \to v_2 \to v_3 \to \cdots \to v_{k-1} \to v_k$

$v_1 \to v_2$: 1, $v_2 \to v_3$: 2, v_3: 3, $v_{k-1} \to v_k$: $(k-1)$ st edge $\ge n$

$\Rightarrow k \ge n+1 > n$ pigeonhole $\Rightarrow \ge 1$ repeat ☐

Clm: q $s \to t$ path $\rightsquigarrow$ $k \ge n$ edges, exists $s \to t$ path q' $\rightsquigarrow$ $< k$ edges, $|q'| \le |q|$

pf: q has cycle C, $|k| \ge 0$

let $q' = q - C$ $\Rightarrow$ $|q'| = |q| + |C| \le |q|$

(# edges in q') = (# edges in q) $= k$ $-$ (# edges in C) ≥ 1 $< k$ ☐

shortest paths exist: finite # of ($\le n-1$) edge paths, min cost $s \to t$ path appear ☐

goal: given s, t either
- capture $s \to t$ shortest path
- capture $s \to t$ path touching negative cycle

Q: greedily?

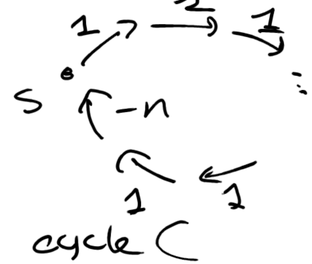
↳ local optimal choices ⇒ global optimal choices

A: yes, if non-negative costs

[Dijkstra, CS374]

A: no, if negative costs

eg:



cycle C

$|C| = (n-1)\cdot 1 + 1(-n)$

$= -1$

$\Rightarrow dist(s,s) = -\infty$

bad: local view would

claim "$dist(s,s)=0$"

idea: dynamic programming?

subproblems: distances $s \rightsquigarrow v$, for any v

↳ $\le k$ edges

def: $dist_{\le k}(s,v) =$

$$\min_{\text{paths } p \; s \rightsquigarrow t,\; p \text{ has} \le k \text{ edges}} |p|$$

base: $dist_{\le 0}(s,v) = \begin{cases} 0 & s = v \\ \infty & s \neq v \end{cases}$

prop: $\{$ length $\le k$ paths $s \rightsquigarrow v\}$

$= \bigcup_{(u,v)\in E} \{$ length $\le k-1$ paths $s \rightsquigarrow u\}$

$= \{ p \circ v : p$ length $\le k-1$ paths $s \rightsquigarrow u \}$

rmk: non disjoint decomposition

ex: $dist_{\le k}(s,v) = \min$ $dist_{\le k-1}(s,v)$

$\min_{(u,v)\in E} dist_{\le k-1}(s,u) + c_{uv}$

prop: if no path from s reaches negative cycle then $\forall t$ $dist(s,t) = dist_{\le n-1}(s,t)$

prop [Bellman Ford]: if no path from s reaches negative cycle, then $dist(s,t)$ can be computed in $O(n \cdot m)$ time

pf: also: $O(n)$ (1) for $v \in V$ $d[0][v] = \infty$

$O(1)$ (2) $d[0][s] = 0$

$O(n)$ (3) for $1 \le k \le n-1$

$O(n)$ (a) for $v \in V$

$O(n)$ (i) $d[k][v] = \min \begin{cases} d[k-1][v] \\ \min_u \left(d[k-1][u] + c_{uv} \right) \end{cases}$

(4) return $d[n-1][t]$

correctness: clear

complexity: clearly $O(n^3)$

better: $c_{uv} = \infty$ if $(u,v) \notin E$

so restrict u s.t. $(u,v) \in E$ by using adjacency list

is $O(m)$ as we change) to edge $(u,v) \in E$ ☒

prop: if no path from s to negative cycle then $\{ dist_{\le k}(s,v) \}_{k,v}$ we can compute for any t a shortest $s \to t$ path in time $O(n)$

Q: negative cycles?

prop: if no path $s \rightsquigarrow$ -- touches negative cycle, then $\forall t$

$$dist(s,t) = dist_{\le n-1}(s,t) = dist_{\le n}(s,t)$$

prop: If for all t $dist_{\le n-1}(s,t) = dist_{\le n}(s,t)$ then no path from s to negative cycle

pf - claim: all v, $k-1 \ge n$ $dist_{\le k}(s,v) = dist_{\le n-1}(s,v)$

pf: induction:

$k-1 = n$: by assumption

$k-1 > n$: $dist_{\le k}(s,v) = \min\{ \underbrace{dist_{\le k-1}(s,v)}_{= dist_{\le n-1}(s,v)}, \min_u (\underbrace{dist_{\le k-1}(s,u)}_{= dist_{\le n-1}(s,u)} + c_{uv}) \}$

$= dist_{\le n}(s,v)$

$= dist_{\le n-1}(s,v)$ □

$\Rightarrow$ all t all $k \ge n$ $dist_{\le k}(s,t)$

$= dist_{\le n-1}(s,t)$

$\Rightarrow$ $\lim_{k \to \infty} dist_{\le k}(s,t) = dist_{\le n-1}(s,t) > -\infty$

$\Rightarrow$ no negative cycles reachable by s

cor: s cannot reach negative cycle iff $\forall t$ $dist_{\le n-1}(s,t) = dist_{\le n}(s,t)$

cor: $s \in V$. in $O(nm)$ time either
- conclude s reaches negative cycle
- compute $\{dist(s,t)\}_{t \in V}$

Q: detect [any] negative cycle?

Prop: given G can decide if G has negative cycle in $O(n^2m)$ time

Sketch: $\forall s \in V$ test if s reaches negative cycle

$O(n)$ $\quad$ $O(nm)$ □

Q: can we do better?

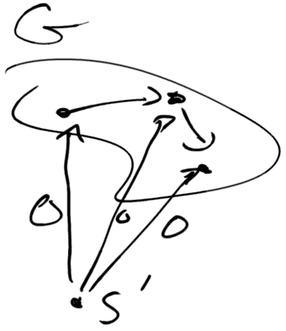
Prop: $O(nm)$ time.

idea: [reduce] to 1 instance of Bellman Ford

pf: consider G' w/ $V' = V \cup \{s'\}$

$E' = E \cup \{(s', v)\}_{v \in V}$

$c_{s',v} = 0$



Ch: G has neg cycle iff s' reaches neg cycle in G'

pf: $\Rightarrow$:

$\Leftarrow$: s' reaches neg cycle C' in G'

$\Rightarrow C' \not\ni s'$

$\Rightarrow C'$ in G □

also: (1) construct G'

(2) detect if G' has neg cycle reachable from s'

correctness: clear

complexity: (1) $O(m)$ time

(2) $O(n'm') = O(nm)$ □

today: dynamic programming
- Bellman Ford
  - shortest paths w/o negative cycle
  - negative cycle detection

next lecture: flows

logistics:
- pset 2 due Fri, groups ≤ 3
- pset 2 out Fri
- lecture in-person