Michael Forbes
mforbes@illinois.edu
2019-10-22.4 →
2019-10-24.2 →
2019 - 10 - 24.1
cs473

cs473 Algorithms - Lecture 17

logistics - pset 7 due W10

last time - reductions to max flow
- multi source / sink [in lots of detail]
- bipartite matching

today: more flow applications [in less detail]

[edge flows = path flow]

Prop: $f : E \to \mathbb{R}_{\geq 0}$ flow [edge flow] $\mathcal{P} = \{$ simple $s \rightsquigarrow t$ paths in $G\}$ [no repeated vertices]

exists $g : \mathcal{P} \cup \mathcal{C} \to \mathbb{R}_{\geq 0}$ st $\mathcal{C} = \{$ simple cycles in $G\}$

- $\forall e \in E \quad f(e) = \sum_{p \ni e} g(p) + \sum_{C \ni e} g(C)$ [g has conservation for free]
- $|\{p : g(p) > 0\} \cup \{C : g(C) > 0\}| \leq m$ [# edges]
- $|f| = |g| = \sum_{p \in \mathcal{P}} g(p)$
- f integral $\Rightarrow$ g integral
- g computable in $O(mn)$ time

Sketch: [run Ford Fulkerson backwards]

algo: find simple path/cycle and subtract from f, until $f = 0$

correctness: ↳ removes all flow from ≥ 1 edge
$\Rightarrow \leq m$ paths/cycles

rmk: attention needed to turn $t \rightsquigarrow s$ flow into cycles

runtime: $O(m^2)$ easy
$O(mn)$, carefully ← adjacency lists over edges w/ positive flow

integrality - by induction ∎

def: network $G = (V, E)$ capacities c

A circulation is a flow w/ value 0.

ex: s ⟳ t

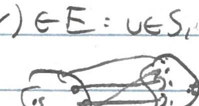
Cor: A circulation is a positive sum of simple cycles

Sketch: flow decomposition ← no paths as they add value. ∎

Q: $G = (V, E)$ directed $s \neq t \in V$. What is the maximum number of [directed] edge disjoint $s \rightsquigarrow t$ paths? [fault tolerant routing]

def: $E' \subseteq E$ is an (s,t)-edge-cut if $G - E'$ has no $s \rightsquigarrow t$ path.

ex: $V = S \sqcup T$ s-t cut $e(S,T) = \{(u,v) \in E : u \in S, v \in T\}$
$S \ni s$, $T \ni t$
is edge cut



Michael Forbes
miforbes@illinois.edu
2019-10-24.2 $\leftarrow$ 2019-10-24.1
$\rightarrow$ 2019-10-24.3
CS473

[hence only vertex partitions are minimal]

lem: $E' \subseteq E$ (s,t) edge cut, $\exists\ V = \underset{s}{\hat{S}} \sqcup \underset{t}{\hat{T}}$ w/ $E' \supseteq e(S,T)$

Pf: $S = \{ v : s \rightsquigarrow v \text{ path in } G - E' \}$

$\underset{S}{(v}, \underset{T}{w)} \in e(S,T) \Rightarrow (v,w) \in E'$ as $s \overset{G-E'}{\rightsquigarrow} v$ but $s \overset{G-E'}{\not\rightsquigarrow} w$ ∎

thm (menger): $G = (V,E)$ directed. $s \neq t \in V$.

$$\underbrace{\min_{\substack{E' \subseteq E \\ E'\ (s,t) \text{ edge cut}}} |E'|} = \max \#\ \text{edge-disjoint } s \rightsquigarrow t \text{ paths}$$

Sketch: view G as network, unit capacities [capacity 1]

$$= \min_{\substack{V = S \sqcup T \\ s \in S,\ t \in T}} e(S,T) = \min c(S,T)$$

$$= \max |f|$$

k edge disjoint $s \rightsquigarrow t$ paths in $G \Rightarrow$ flow value k in G [clear]

integral flow value k in $G \Rightarrow k$ edge disjoint paths [discard cycles]

apply flow decomposition to get paths [integrality is key]

unit capacities $\Rightarrow$ paths are edge disjoint

rmk: - Menger's thm predated max flow = min cut

- also works for undirected graphs

$$(u) - (v) \mapsto (u) \underset{1}{\overset{1}{\rightleftarrows}} (v) \mapsto (u) \underset{0}{\overset{0}{\rightleftarrows}} (v)$$

Q: who will make the world series? [irrelevant this year...]

fact: ≤ 1969: two baseball leagues (National, American) advanced their team to world series based on $\underbrace{\text{best overall season record.}}_{\text{"pennant race"}}$

≥ 1969: [chaos]

Q: can Baltimore Orioles still win the pennant (without tie)?

	wins	unplayed
New York	91	4
Boston	91	4
Baltimore	88	2

A: no, cannot exceed $90 < 91$

Q: still win?

	wins	unplayed	NYC	BOS	BWI
NYC	91	4	—	3	1
BOS	91	4	3	—	1
BWI	90	2	1	1	—

Michael Forbes
miforbes@illinois.edu
2019-10-24.2 → 2019-10-24.3
2019-10-24.4 ←
CS 473

A: suppose BWI wins both games — BWI 92
need BOS, NYC to lose all remaining games — BOS 91 + ?/3, NYC 91 + ?/3

↳ not possible as someone must win ≥ 2 of those 3 games

↳ $\geq 91+2 = 93 > 92$ wins.

$\implies$ BWI cannot (outright) win pennant or even tie

def: The baseball elimination problem

- teams $0, 1, \ldots, n$
- current wins $w_i \geq 0$, $i \geq 0$
- $0 \leq i < j \leq n$ remaining games $g_{ij} \geq 0$

want to know if team 0 can possibly achieve largest # wins (or tie) [otherwise eliminated]

prop: team 0 can win/tie pennant iff

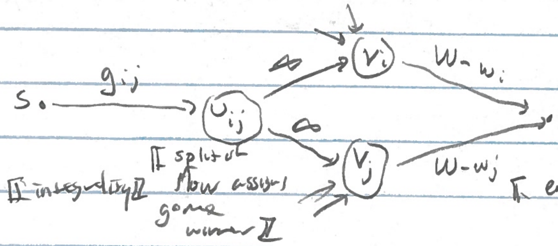
team 0 wins all remaining games, totals $W = w_0 + \sum_{k>0} g_{0k}$ wins

team $k > 0$ wins $\leq W$ games total $\equiv$ wins $\leq W - w_k$ remaining games [how to assign?]

Prop: baseball elimination reduces to max flow

Sketch: create $G = (V, E)$

$V =$
- s — source
- U_{ij}, $1 \leq i < j \leq n$ — games i vs j remaining
- V_k, $k \geq 1$ — team k wins in remaining games
- t — sink



[ensures no one wins too much]

Clm: $\underbrace{\text{max flow value}}_{= \text{min cut}} = \underbrace{\sum_{1 \leq i < j \leq n} g_{ij}}$ iff team 0 can win pennant (or tie)

value of $V = \{s\} \sqcup (V \setminus \{s\})$ cut

says all games can be assigned <u>and</u> no one wins too much

∎

rmk: small min cut gives succinct proof that team 0 cannot win [otherwise challenging]

---

def: A <u>network with prices</u>: $G = (V, E)$ directed

$s \neq t \in V$

$c : E \to \mathbb{R}_{\geq 0}$ capacities

$p : E \to \mathbb{R}$ prices [possibly negative]

The <u>cost</u> of a flow $f : E \to \mathbb{R}_{\geq 0}$ is $\sum_e p(e) f(e)$

Michael Forbes
mforbes@illinois.edu
2019-10-24.4 ⟵ 2019-10-24.3
⟶ 2019-10-29.1
CS473

Q: find min cost max flow?

rmk: - generalizes min weight perfect bipartite matching
- generalizes shortest $s \rightsquigarrow t$ path [use 1 unit of flow, cheaply]
- not known to be reducible to unweighted max flow
- special case of linear programming [next topic] [need more ideas]

thm: min cost max flow has poly(n, m, C, P) time algo when prices, capacities are integral

$C = \sum_e c(e)$, $P = \sum_e |p(e)|$ ⟵ prices can be negative

Sketch: [like Ford Fulkerson]

create residual graph $\; (u) \xrightarrow{f/c\,:\,p} (v) \;\longmapsto\; (u) \xrightarrow{c-f\,:\,p} (v),\; (v) \xrightarrow{f\,:\,-p} (u)$

key lem: f min cost max flow iff G_f has no negative cost cycle

Sketch: $\neg \Leftarrow \neg$: negative cost cycle D
$\Rightarrow$ $f+D$ [augment] - is max flow [capacity, conservation, value]
- has less cost [negative cycle]

$\neg \Rightarrow \neg$: f^* min cost max flow $p(f^*) < p(f)$
$f^* - f$ is flow in G_f [pset 6]
- value 0 $\Rightarrow$ circulation $\Rightarrow$ sum of simple cycles
and - negative total cost $\Rightarrow$ some negative cost cycle ☒

algo: find maxflow f in G
initialize G_f
while negative cost cycle D in G_f [each iteration is efficient]
$f \leftarrow f+D$
$G_f \leftarrow G_{f+D}$
return f

$\#$ iterations $\le \underbrace{(\text{max-cost maxflow})}_{\le C\cdot P} - \underbrace{(\text{min cost maxflow})}_{\ge -C\cdot P}$
$\le 2 \cdot C \cdot P$ ∎

rmks: - finding negative cost cycle uses Bellman Ford
- finding most-negative cost (simple) cycle is NP-hard ⟵ Hamiltonian cycle problem
- efficient algo via augmenting along cycle of $\underbrace{\text{minimum mean cost}}_{\text{cost per edge}}$ [can't do max capacity analysis]

logistics: pset 7 due W10
today: - flow decomposition
- edge disjoint paths
- baseball elimination
- min cost max flow
next time: - linear programming [generalizes maxflow]