

## Maximum Flow and Minimum Cut

Max flow and min cut.

- Two very rich algorithmic problems.
- Cornerstone problems in combinatorial optimization.
- Beautiful mathematical duality.

Nontrivial applications / reductions.

- Data mining.
- Open-pit mining

Project selection

- Airline scheduling.
- Bipartite matching.

Baseball elimination.

- Image segmentation.
- Network connectivity.
- Network reliability.
- Distributed computing.
- Egalitarian stable matching.

Security of statistical data.

- Network intrusion detection.

Multi-camera scene reconstruction

- Many many more . . .



## Cuts

Def. An s-t cut is a partition $(A, B)$ of $V$ with $s \in A$ and $t \in$ B.
$\operatorname{cap}(A, B)=\sum_{e \text { out of } A} c(e)$
Def. The capacity of a cut (A, B) is:





## Towards a Max Flow Algorithm

Greedy algorithm.

- Start with $f(e)=0$ for all edge $e \in E$.
- Find an s-t path $P$ where each edge has $f(e)<c(e)$.
- Augment flow along path P .
- Repeat until you get stuck.



## Towards a Max Flow Algorithm

Greedy algorithm.

- Start with $f(e)=0$ for all edge $e \in E$.
- Find an s-t path $P$ where each edge has $f(e)<c(e)$.
- Augment flow along path $P$.
- Repeat until you get stuck. \ locally optimality $\Rightarrow$ global optimality

opt $=30$


## Residual Graph

Original edge: $\mathrm{e}=(\mathrm{u}, \mathrm{v}) \in \mathrm{E}$.

- Flow f(e), capacity c(e).

Residual edge.

- "Undo" flow sent.
- $e=(u, v)$ and $e^{R}=(v, u)$.
- Residual capacity:

$$
c_{f}(e)= \begin{cases}l(e)-f(e) & \text { if } e \in E \\ f(e) & \text { if } e^{R} \in E\end{cases}
$$



Residual graph: $\mathrm{G}_{\mathrm{f}}=\left(\mathrm{V}, \mathrm{E}_{\mathrm{f}}\right)$.

- Residual edges with positive residual capacity.
- $\mathrm{E}_{\mathrm{f}}=\{\mathrm{e}: \mathrm{f}(\mathrm{e})<\mathrm{c}(\mathrm{e})\} \cup\left\{\mathrm{e}^{\mathrm{R}}: \mathrm{c}(\mathrm{e})>0\right\}$.



## Augmenting Path Algorithm

```
Augment(f, C, P) {
    b}\leftarrow\mathrm{ bottleneck (P)
    foreach e\in \inP{
        if (e\inE) f(e)}\leftarrowf(e)+
        lif(e leE) f(e)\leftarrowf(e)+b
    }
    return f
}
```

Ford-Fulkerson ( $\mathbf{G}, \mathbf{s}, \mathrm{t}, \mathrm{c}$ ) $\{$
foreach $e \in E \quad f(e) \leftarrow 0$
$\mathrm{G}_{\mathrm{f}} \leftarrow$ residual graph
while (there exists augmenting path P) \{ $\mathrm{f} \leftarrow \operatorname{Augment}(\mathrm{f}, \mathrm{c}, \mathrm{P})$
update $\mathrm{G}_{\mathrm{f}}$
\}
return $f$
\}

## Flows and Cuts

Flow value lemma. Let f be any flow, and let (A, B) be any s-t cut. Then, the net flow sent across the cut is equal to the amount leaving s.

$$
\sum_{e \text { out of } A} f(e)-\sum_{e \text { into } A} f(e)=v(f)
$$



## Flows and Cuts

Flow value lemma. Let f be any flow, and let (A, B) be any s-t cut. Then, the net flow sent across the cut is equal to the amount leaving s .

$$
\sum_{e \text { out of } A} f(e)-\sum_{e \text { in to } A} f(e)=v(f)
$$



## Flows and Cuts

Flow value lemma. Let f be any flow, and let (A, B) be any s-t cut. Then, the net flow sent across the cut is equal to the amount leaving s.

$$
\sum_{e \text { out of } A} f(e)-\sum_{e \text { into } A} f(e)=v(f)
$$



## Flows and Cuts

Weak duality. Let f be any flow, and let (A, B) be any s-t cut. Then the value of the flow is at most the capacity of the cut.


## Flows and Cuts

Weak duality. Let f be any flow. Then, for any s-t cut (A, B) we have
$\mathrm{v}(\mathrm{f}) \leq \operatorname{cap}(\mathrm{A}, \mathrm{B})$.
Pf.
$v(f)=\sum^{f} f(e)-\sum_{i} f(e)$
$\leq \quad \sum_{e} f(e)$
$\leq \quad \sum c(e)$
$e$ out of $A$
$=\operatorname{cap}(A, B)$


## Certificate of Optimality

Corollary. Let f be any flow, and let (A, B) be any cut. If $v(f)=\operatorname{cap}(A, B)$, then $f$ is a max flow and $(A, B)$ is a min cut.

```
Value of flow = 28
Cut capacity =28 F Flow value \leq28
```



## Max-Flow Min-Cut Theorem

Augmenting path theorem. Flow f is a max flow iff there are no augmenting paths.

Max-flow min-cut theorem. [Ford-Fulkerson 1956] The value of the max flow is equal to the value of the min cut.

Proof strategy. We prove both simultaneously by showing the TFAE:
(i) There exists a cut $(A, B)$ such that $v(f)=\operatorname{cap}(A, B)$.
(ii) Flow f is a max flow.
(iii) There is no augmenting path relative to $f$.
(i) $\Rightarrow$ (ii) This was the corollary to weak duality lemma.
(ii) $\Rightarrow$ (iii) We show contrapositive.

- Let f be a flow. If there exists an augmenting path, then we can improve $f$ by sending flow along path.


## Proof of Max-Flow Min-Cut Theorem

$$
(\text { iii) } \Rightarrow \text { (i) }
$$

- Let f be a flow with no augmenting paths.
- Let A be set of vertices reachable from s in residual graph.
- By definition of $A, s \in A$.
- By definition of $f, t \notin A$.
$v(f)=\sum_{e \text { out of } A} f(e)-\sum_{e \text { in to } A} f(e)$
$=\quad \sum c(e)$
$=\quad \operatorname{cap}(A, B) \quad$.

original network


## Running Time

Assumption. All capacities are integers between 1 and C.
Invariant. Every flow value f(e) and every residual capacities $c_{f}(e)$ remains an integer throughout the algorithm.

Theorem. The algorithm terminates in at most $\mathrm{v}\left(\mathrm{f}^{*}\right) \leq \mathrm{nC}$ iterations.
Pf. Each augmentation increase value by at least 1. •
Corollary. If $\mathrm{C}=1$, Ford-Fulkerson runs in $\mathrm{O}(\mathrm{m})$ time.
Integrality theorem. If all capacities are integers, then there exists a max flow $f$ for which every flow value $f(e)$ is an integer.
Pf. Since algorithm terminates, theorem follows from invariant. •



## Matching

Matching.

- Input: undirected graph $\mathrm{G}=(\mathrm{V}, \mathrm{E})$.
- $\mathrm{M} \subseteq \mathrm{E}$ is a matching if each node appears in at most edge in M.
- Max matching: find a maxccardinality matching.



## Bipartite Matching

Bipartite matching.

- Input: undirected, bipartite graph $G=(L \cup R, E)$.
- $\mathrm{M} \subseteq \mathrm{E}$ is a matching if each node appears in at most edge in M.
- Max matching: find a max cardinality matching.



## Bipartite Matching

Bipartite matching.

- Input: undirected, bipartite graph $G=(L \cup R, E)$.
- $\mathrm{M} \subseteq \mathrm{E}$ is a matching if each node appears in at most edge in M.
- Max matching: find a max cardinality matching.



## Bipartite Matching

Max flow formulation.

- Create digraph $\mathrm{G}^{\prime}=\left(\mathrm{L} \cup \mathrm{R} \cup\{\mathrm{s}, \mathrm{t}\}, \mathrm{E}^{\prime}\right)$.
- Direct all edges from $L$ to $R$, and assign infinite (or unit) capacity.
- Add source s , and unit capacity edges from s to each node in L.
- Add $\operatorname{sink} \mathrm{t}$, and unit capacity edges from each node in R to t .



## Bipartite Matching: Proof of Correctness

Theorem. Max cardinality matching in $\mathrm{G}=$ value of max flow in $\mathrm{G}^{\prime}$.
Pf. $\leq$

- Given max matching M of cardinality k .
- Consider flow f that sends 1 unit along each of k paths.
- f is a flow, and has cardinality k. •

G



| 7.6 Disjoint Paths |
| :--- |
|  |
|  |
|  |

## Edge Disjoint Paths

Disjoint path problem. Given a digraph $\mathrm{G}=(\mathrm{V}, \mathrm{E})$ and two nodes $s$ and $t$, find the max number of edge-disjoint s-t paths.

Def. Two paths are edge-disjoint if they have no edge in common.

Ex: communication networks.


## Edge Disjoint Paths

Disjoint path problem. Given a digraph $G=(V, E)$ and two nodes $s$ and $t$, find the max number of edge-disjoint $s-t$ paths.

Def. Two paths are edge-disjoint if they have no edge in common.

Ex: communication networks.


## Edge Disjoint Paths

Max flow formulation: assign unit capacity to every edge.


Theorem. Max number edge-disjoint s-t paths equals max flow value.
Pf. $\leq$

- Suppose there are k edge-disjoint paths $\mathrm{P}_{1}, \ldots, \mathrm{P}_{\mathrm{k}}$.
- Set $f(e)=1$ if e participates in some path $P_{i}$; else set $f(e)=$ 0 .
- Since paths are edge-disjoint, f is a flow of value k .


## Edge Disjoint Paths

Max flow formulation: assign unit capacity to every edge.


Theorem. Max number edge-disjoint s-t paths equals max flow value.
Pf. $\geq$

- Suppose max flow value is k .
- Integrality theorem $\Rightarrow$ there exists $0-1$ flow f of value k .
- Consider edge $(\mathrm{s}, \mathrm{u})$ with $\mathrm{f}(\mathrm{s}, \mathrm{u})=1$.
-by conservation, there exists an edge ( $u, v$ ) with $f(u, v)=$ 1




## Edge Disjoint Paths and Network Connectivity

Theorem. [Menger 1927] The max number of edge-disjoint s-t paths is equal to the min number of edges whose removal disconnects t from s .

Pf. $\leq$

- Suppose the removal of $\mathrm{F} \subseteq \mathrm{E}$ disconnects t from s , and $|\mathrm{F}|=$ k.
- All s-t paths use at least one edge of F. Hence, the number of edge-disjoint paths is at most k. .



| 7.10 Image Segmentation |
| :---: |
|  |
|  |

## Image Segmentation

Image segmentation.

- Central problem in image processing.
- Divide image into coherent regions.

Ex: Three people standing in front of complex background scene. Identify each person as a coherent object.

## Image Segmentation

Foreground / background segmentation.

- Label each pixel in picture as belonging to foreground or background.
- $\mathrm{V}=$ set of pixels, $\mathrm{E}=$ pairs of neighboring pixels.
- $a_{i} \geq 0$ is likelihood pixel $i$ in foreground.
- $b_{i} \geq 0$ is likelihood pixel $i$ in background.
- $\mathrm{p}_{\mathrm{ij}} \geq 0$ is separation penalty for labeling one of i
and j as foreground, and the other as background.
Goals.
- Accuracy: if $a_{i}>b_{i}$ in isolation, prefer to label $i$ in foreground.





## Image Segmentation

Formulate as min cut problem.

- Maximization.
- No source or sink.
- Undirected graph.

Turn into minimization problem.

- Maximizing $\sum_{i \in A} a_{i}+\sum_{j \in B} b_{j}-\sum_{(i, j) \in E} p_{i}$
- Maximizing $\underbrace{}_{i \in A} \underset{j \in B}{ } \xrightarrow{j} \begin{aligned} & (i, j) \in E \\ & \mid A \cap\{i, j\}=1\end{aligned}$
is equivalent to minimizing $\underset{\text { aconstant }}{\left(\sum_{i \in V} a_{i}+\sum_{j \in V} b_{j}\right)}-\sum_{i \in A} a_{i}-\sum_{j \in B} b_{j}+\underbrace{\sum_{i j} p_{i j}}_{\substack{(i, j) \in E \\|A \cap\{i, j\}|=1}}$
- or alternatively

$$
\sum_{j \in B} a_{j}+\sum_{i \in A} b_{i}+\sum_{\substack{(i, j) \in E \\|A \cap\{i, j\}|=1}} p_{i j}
$$

## Image Segmentation

Formulate as min cut problem.

- $\mathrm{G}^{\prime}=\left(\mathrm{V}^{\prime}, \mathrm{E}^{\prime}\right)$.
- Add source to correspond to foreground;
 add sink to correspond to background
- Use two anti-parallel edges instead of undirected edge.
$\mathbf{G}^{\prime}$




