Lecture #7

Last Time: nearest neighbor search and LSH

i.e. how to exploit collisions, to get sublinear search time

Today: Maximum Flow (seen before?)

Input: (1) directed graph $G=(V,E)$

(2) capacity fcn $u: A \rightarrow \mathbb{R}^+$

(3) source $s$, sink $t$ ($s,t \in V$

Goal: Send as much flow as possible from $s$ to $t$, without violating constraints

\[
\max |f| = \sum_{a} f(s,a) - \sum_{a} f(a,s)
\]

s.t.

(1) $\sum_{a} f(v,a) - \sum_{a} f(a,v) = 0 \forall v \neq s,t$

(2) $0 \leq f(a,b) \leq u(a,b) \forall (a,b) \in A$

(1) is called conservation, (2) is called capacity constraints
Examples? Packets in computer network, cars/trains in transportation network

Generalizations? multiple sources/sinks if we will study these

Many applications in combinatorial optimization e.g. matching, scheduling, partitioning, etc

Powerful tool for reasoning about flows.
* Assume no arc into \( s \) has flow, no arc out of \( t \) has flow.

Flow decomposition: Any \( s \rightarrow t \) flow can be decomposed into at most \( m = |A| \) \( s \rightarrow t \) paths and cycles

Proof: Induction on number of arcs with non-zero flow.

Let \( (a,b) \) be any arc with \( f(a,b) > 0 \). Then trace backwards from \( a \) and forwards from \( b \)

\[
\begin{align*}
\text{flow out of } a \quad + \quad \text{conservation of flow} \quad \Rightarrow \quad \text{flow into } a
\end{align*}
\]

\[
\begin{align*}
\text{flow into } b \quad + \quad \text{flow out of } b
\end{align*}
\]

Thus we can trace backwards from \( a \) and forwards from \( b \) until we (1) reach a cycle or (2) reach both \( s \) and \( t \)
Let $W$ be the arcs in the cycle, or in the s-t path

$$\Delta(W) = \min_{(a, b) \in W} f(a, b)$$

Decrease flow on cycle/path by $\Delta(W)$ on each arc; add cycle/path with $\Delta(W)$ flow to flow decomposition.

The # of edges with positive flow has strictly Decreased $\Delta$.

Question: If s-t flow $\Rightarrow$ s-t paths + cycles, what if we zero out cycles? Would we get on s-t flow? And of what value?

Now let's apply flow decomposition.

def: An s-t cut is a set $S \subseteq V$ where $s \in S$ and $t \in V \setminus S$. The capacity is

$$\text{cap}(S, V \setminus S) = \sum_{(a, b) \in A} \min\{u(a, b)\}$$

Lemma: max flow $\leq$ min cut

$$(\max |S|)$$

$$(\min \text{cap}(S, V \setminus S))$$

Proof: Take any max flow, and decompose it into paths and cycles.

(1) Each s-t path must cross the cut $(S, V \setminus S)$ at least once.
Hence a path that contributes \( p \) to total flow uses up at least \( p \) units of capacity in \((S,V)\).

(2) cycles do not contribute to \( |f| \)

(If flow > cut, we used up more capacity than is available)

Max-Flow Attempt #1

If \( W \) is a path, let \( \delta(W) = \min_{(a,b) \in W} u(a,b) - f(a,b) \)

while \( \exists \) s-t path \( W \) with \( \delta(W) > 0 \)

increase flow on each arc in \( W \) by \( \delta(W) \)

Does this algorithm succeed in finding the max flow? What can go wrong?

![Diagram](image.png)

augment along one path, but when we're done no more s-t paths left

This motivates the notion of residual graph
Residual Graph: Given instance of max flow problem

\[ G = (V, A) \] and s-t flow \( f \)

\[ \text{arc } u(a, b) \Rightarrow \text{arc } u(a, b) - f(a, b) \]

The meaning of \( \text{arc } (b, a) \) is that \( f(a, b) \) units of flow in \( f \) can be undone.

Max Flow Attempt #2:

Let \( \Delta_H(w) = \min_{(a, b) \in W} u_{H}(a, b) \)

While \( \exists \) s-t path \( W \) in \( H \) with \( \Delta_H(W) > 0 \)

- augment along \( W \) by \( \Delta_H(W) \)
- update residual

What if we reach a point where there is no path in the residual?

Let \( S \) be the set of nodes reachable (in residual) from s. Then

1. Every arc \( (a, b) \) with \( a \in S \) has \( f(a, b) = u(a, b) \) (else we could reach b in residual)
2. Every arc \( (b, a) \) with \( a \in S \) has \( f(b, a) = 0 \) (else there would be an arc \( (a, b) \) in residual)
Thus \( \text{cap}(S, V \setminus S) = \sum_{(a, b) \in S, \bar{b} \in V} f(a, b) \)

This proves \( \text{max flow} = \text{min cut} \)

The following conditions are equivalent:

1. \( f \) is a max flow.
2. There is no \( s-t \) path with \( \Delta_{\text{res}}(u) \leq 0 \) in residual.
3. \( 1f \) = \( \text{cap}(S, V \setminus S) \) for some \( s-t \) cut.

Clearly \( (3) \Rightarrow (1) \).

\( - (2) \Rightarrow - (1) \)

and \( (2) \Rightarrow (3) \) by claim.

There are many complications to how you choose augmenting paths.

**Table: Path and Flow**

<table>
<thead>
<tr>
<th>Path</th>
<th>Sent flow</th>
<th>Residual capacities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( e_1 )</td>
<td>( e_2 )</td>
</tr>
<tr>
<td>( s \rightarrow 2 \rightarrow 3 \rightarrow t )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( P_1 )</td>
<td>( r )</td>
<td>( r^2 )</td>
</tr>
<tr>
<td>( P_2 )</td>
<td>( r )</td>
<td>( r^2 )</td>
</tr>
<tr>
<td>( P_3 )</td>
<td>( r^2 )</td>
<td>0</td>
</tr>
</tbody>
</table>

\( P_1 = s \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow t \)

\( P_2 = s \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow t \)

\( P_3 = s \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow t \)

Flow converges to \( 1 + 2 \frac{r^2}{r} = 3 + 2r \leq 2m^2 - 1 \).
For irrational capacities, augmenting paths need not converge.

But for integer capacities, it always finds an integer-valued max flow, and this is important in many settings.