CPSC 420 Lecture 9 : Today's announcements:

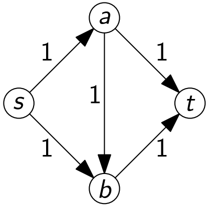
- HW2 available tonight on Gradescope, due Feb 9, 23:59
- Examlet 2 on Feb 17 in class. Closed book & no notes
- Reading: Maximum Flows & Minimum Cuts [Algorithms by Erickson Ch. 10]

Today's Plan

- Network Flow
  - Ford-Fulkerson algorithm
  - Edmonds-Karp algorithm
  - Maximum matching in bipartite graphs

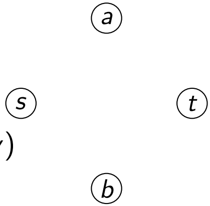
Max Flow via Path Augmentation [Ford & Fulkerson 1962]

1. Start with zero flow (a feasible solution)
2. Repeat until impossible
   - Choose an **augmenting path** from s to t
   - Increase flow on this path as much as possible



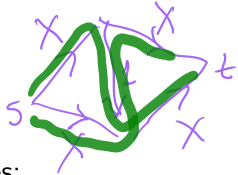
The **residual network** of flow network $G = (V, E)$ with flow f is $G^f = (V, E^f)$ where
$E^f = \{(u, v) | f(u, v) < c(u, v) \text{ or } f(v, u) > 0\}$
The **residual capacity** of an edge $(u, v) \in E^f$ is

$$c^f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } f(u, v) < c(u, v) \\ f(v, u) & \text{if } f(v, u) > 0 \end{cases}$$



An **augmenting path** in G is an $s \rightsquigarrow t$ path in G^f

Running time of Ford & Fulkerson



For flow networks (V, E) with integral capacities:

Running time $O(|E|\text{size}(f^*))$

For flow networks (V, E) with irrational capacities:

Running time ∞

Edmonds-Karp

Shortest augmenting path: Running time $O(|V||E|^2)$

regardless of capacities

Beyond Ford-Fulkerson

[Orlin 2012] Use this as NF runtime $\rightarrow$ $O(|V||E|)$

[Chen-Kyng-Liu-Peng-Gutenberg-Sachdeva 2022] $O(|E|^{1+o(1)})$

Shortest augmenting path [Edmonds & Karp]

by contradiction

Lemma: The length $d_1(s, v)$ (# edges) of the shortest path in residual network G_1 from s to v cannot decrease in residual network G_2 after a shortest aug. path augmentation. in G_2

Proof: Suppose $d_1(s, v)$ does decrease to $d_2(s, v) < d_1(s, v)$. Pick such a v with smallest $d_2(s, v)$. Let u be the vertex before v in this shortest aug. path in G_2 ($v \neq s$ so u exists).

so $d_2(s,u) \geq d_1(s,u)$

$$
\begin{aligned}
d_2(s, v) &= d_2(s, u) + 1 \\
&\geq d_1(s, u) + 1 && \rightarrow u \text{ is closer to } s \\
&\geq d_1(s, v) && \text{see (1) and (2)} \Rightarrow\Leftarrow \square
\end{aligned}
$$

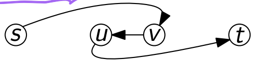
(1) or (2)

shortest

(1) If $(u, v) \in G_1$ then $d_1(s, v) \leq d_1(s, u) + 1$ $\quad s \longrightarrow u \longrightarrow v$

(2) If $(u, v) \notin G_1$ then augmentation created (u, v) in G_2

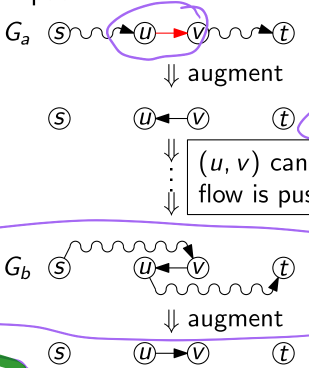
G_1



This is a shortest aug. path so $d_1(s, u) = d_1(s, v) + 1$ and $d_1(s, u) > d_1(s, v)$

Shortest augmenting path [Edmonds & Karp]

A critical edge on an aug. path is edge with smallest residual capacity on path.



shortest aug.path

(u, v) critical

$(u, v) \notin$ res. network

$\boxed{\text{$(u, v)$ can't be in aug. path until flow is pushed back from v to u}}$

shortest aug.path

$$
\begin{aligned}
d_a(s, v) &= d_a(s, u) + 1 && \text{shortest path prefix is shortest}\\
d_b(s, u) &= d_b(s, v) + 1 &&\\
&\geq d_a(s, v) + 1 && \text{Lemma}\\
&= d_a(s, u) + 2 &&
\end{aligned}
$$

Shortest augmenting path [Edmonds & Karp]

A critical edge on an aug. path is edge with smallest residual capacity on path.

$$
\begin{aligned}
d_a(s,v) &= d_a(s,u) + 1 && \text{shortest path prefix is shortest} \\
d_b(s,u) &= d_b(s,v) + 1 && \\
&\geq d_a(s,v) + 1 && \text{Lemma} \\
&= d_a(s,u) + 2 &&
\end{aligned}
$$

From the time (u,v) was critical to the time when (u,v) could next be critical, the shortest path from s to u increases by at least 2
$\Rightarrow$ # times (u,v) can be critical $\leq \frac{|V|-1}{2}$
$\Rightarrow$ # augmentations $\leq \frac{|V|-1}{2} \cdot |E| \in O(|V| \cdot |E|)$

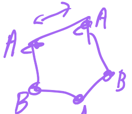
Since finding a shortest augmenting path (using BFS) takes time $O(|E|)$, total runtime is $O(|V||E|^2)$.

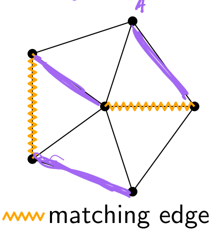
Maximum Matching in Bipartite Graphs

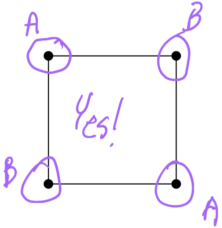
A **matching** in a graph G is a subset M of its edges with no vertex the endpoint of more than one edge in M.

A **maximum matching** is a matching with the maximum number of edges.

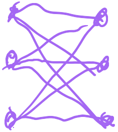
A **maximal matching** is a matching to which another edge cannot be added to form a new matching.







A **bipartite graph** is a graph $G = (V, E)$ where V can be partitioned into A and B such that $\forall (u, v) \in E$, either $u \in A$ and $v \in B$ or $u \in B$ and $v \in A$.



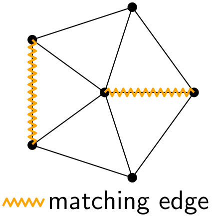
$K_{3,3}$

Maximum Matching in Bipartite Graphs

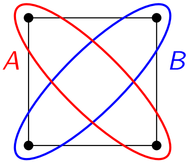
A **matching** in a graph G is a subset M of its edges with no vertex the endpoint of more than one edge in M.

A **maximum matching** is a matching with the maximum number of edges.

A **maximal matching** is a matching to which another edge cannot be added to form a new matching.



A **bipartite graph** is a graph $G = (V, E)$ where V can be partitioned into A and B such that $\forall(u, v) \in E$, either $u \in A$ and $v \in B$ or $u \in B$ and $v \in A$.



Given bipartite graph $G = (V, E)$ with partitions A and B find maximum matching in G.

Maximum Matching in Bipartite Graphs Algorithm

Given bipartite graph $G = (V, E)$ with partitions A and B:

1. Create a flow network $F = (V', E')$

$$
\begin{aligned}
V' =& V \cup \{s, t\} && \text{add source and sink} \\
E' =& \{(u, v) | u \in A, v \in B, (u, v) \in E\} && \text{edges from } A \text{ to } B \\
& \cup \{(s, u) | u \in A\} && \text{edges from } s \text{ to } A \\
& \cup \{(v, t) | v \in B\} && \text{edges from } B \text{ to } t
\end{aligned}
$$

   Set all capacities to 1.

2. Find maximum flow f in F
3. Output edges $(u, v) \in E$ such that $f(u, v) = 1$