When Combinatorics and Flow Networks Intersect

Gerald Huang

UNSW Computer Science and Engineering Society,
UNSW Competitive Programming and Mathematics Society

March 28, 2023

G'day!

- **About me**: Sixth year Computer Science / Mathematics student, specialising in algorithm design, computational complexity theory, combinatorics, number theory, and graph theory.

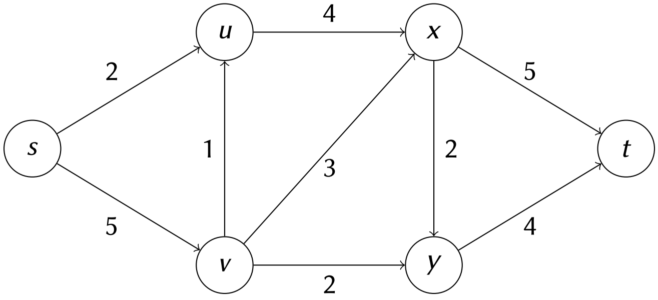
G'day!

- **About me**: Sixth year Computer Science / Mathematics student, specialising in algorithm design, computational complexity theory, combinatorics, number theory, and graph theory.
- I like:
  - Ruining my sleep schedule from time to time.
  - Teaching and learning about new things.
- Nom nom.

Introduction to Maximum Flow

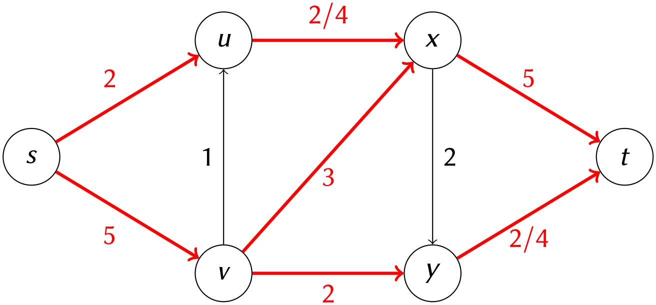
Introduction to Maximum Flow

A **flow network** is a directed and weighted graph $G = (V, E)$, where each edge $(u, v) \in E$ has a weight $w_{u,v}$. This is called the *capacity*.



The Maximum Flow Problem

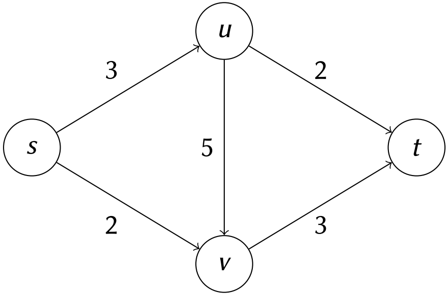
- Given a flow network, how much flow can we send from s to t assuming we have an infinite supply in s?

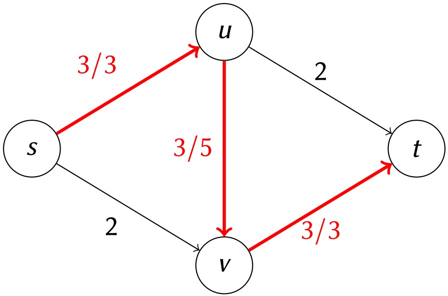


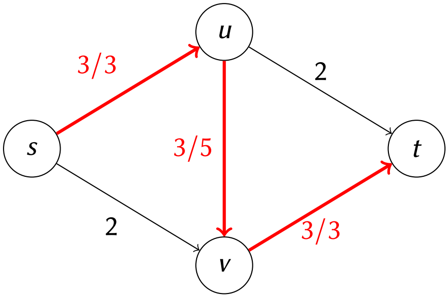
Maximum flow: 7.

Ford-Fulkerson

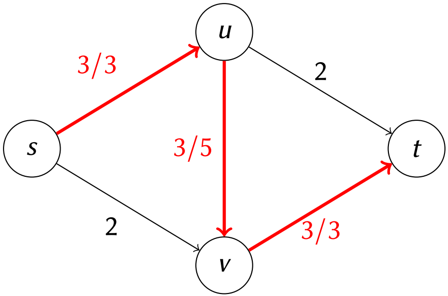
- Try as many paths as possible!
- Find $s - t$ paths and send flow down the path.
- When updating flows and capacities, send flow back an edge.


u
3
2
s
5
t
2
3
v

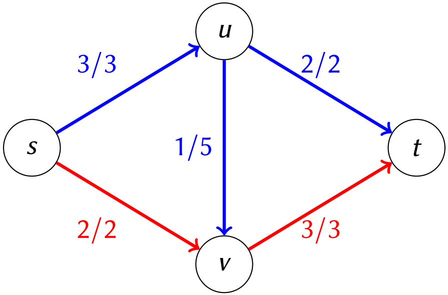

u
3/3
2
s
3/5
t
2
3/3
v

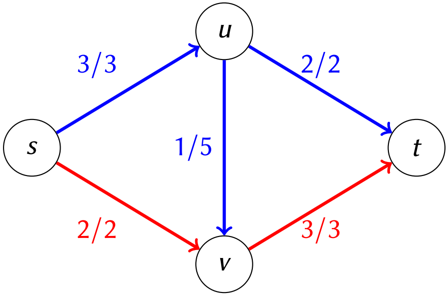


Flow: 3.

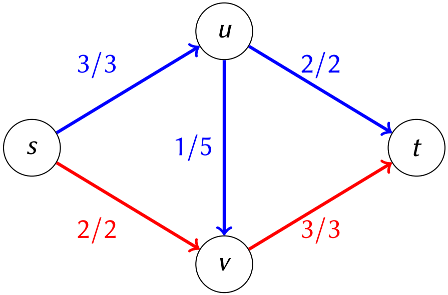


Flow: 3. Hmmm... can we do better?


u
3/3
2/2
s
1/5
t
2/2
3/3
v



Flow: 5.



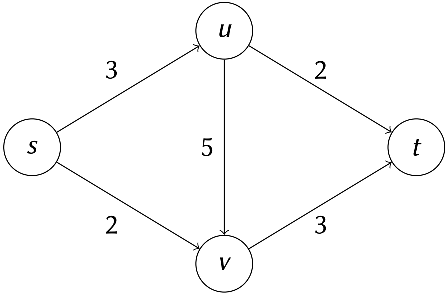
Flow: 5. So... what went wrong?

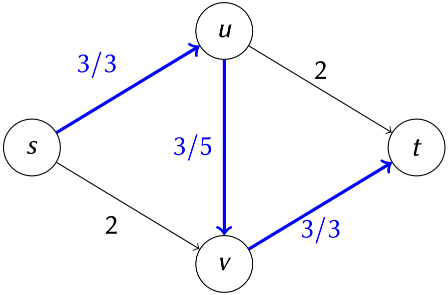
Ford-Fulkerson

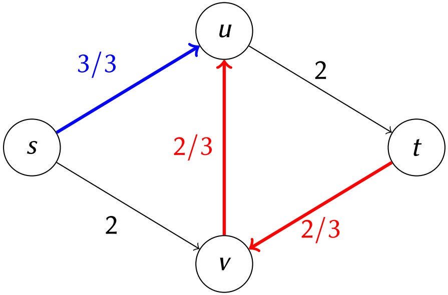
- We need a way to “undo” flow.

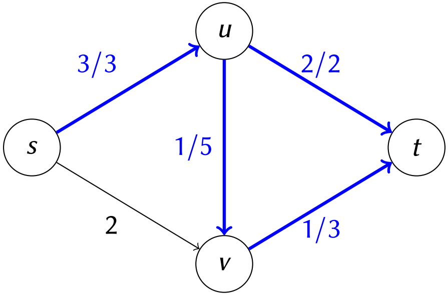
Ford-Fulkerson

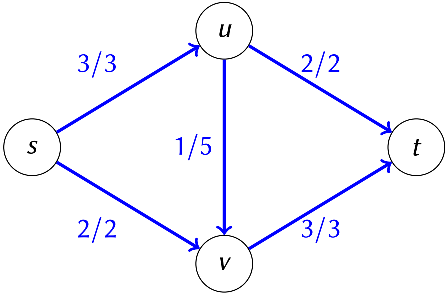
- We need a way to “undo” flow.
  - We can denote the amount of flow we can send back with an arrow in the *reverse* direction.
  - Keep finding $s - t$ paths this way until no more paths are available.

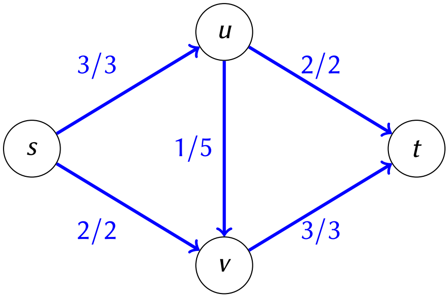

u
3
2
s
5
t
2
3
v


u
s
t
v
3/3
2
3/5
2
3/3


u
3/3
2
s
2/3
t
2
2/3
v


u
3/3
2/2
s
1/5
t
2
1/3
v


u
s
t
v
3/3
2/2
1/5
2/2
3/3



Terminate with maximum flow of 5.

Ford-Fulkerson

- Note that there are finite many paths from s to t; therefore, the algorithm must terminate.
- Every time we "reuse" an edge, we send flow back to try for a better $s - t$ path.
- The final output of the Ford-Fulkerson algorithm is a set of "saturated" edges which correspond to the edges that are used in the maximum flow of the flow network.
- **Running time**: $O(|E| \cdot |f|)$, where $|f|$ is the flow of the graph.

Other algorithms

Other algorithms exist that solve the Maximum Flow problem with various running times.

- Edmonds-Karp – special modification of Ford-Fulkerson: $O(|E| \cdot \min\{|V| \cdot |E|, |f|\})$.
- Dinic's algorithm – $O(|V|^2 \cdot |E|)$.
- Preflow push algorithm – $O(|V|^2 \cdot |E|)$.

Maximum Flow-Minimum Cut

Cuts in a Flow Network

A *cut* in a flow network is a partition of vertices into two sets S and T such that:

- $S \cup T = V$.
- $S \cap T = \emptyset$.
- $s \in S, t \in T$.

Maximum Flow-Minimum Cut

Cuts in a Flow Network

A *cut* in a flow network is a partition of vertices into two sets S and T such that:

- $S \cup T = V$.
- $S \cap T = \emptyset$.
- $s \in S$, $t \in T$.

The cut splits the graph into two parts such that s and t are completely separated!

Maximum Flow-Minimum Cut

Cuts in a Flow Network

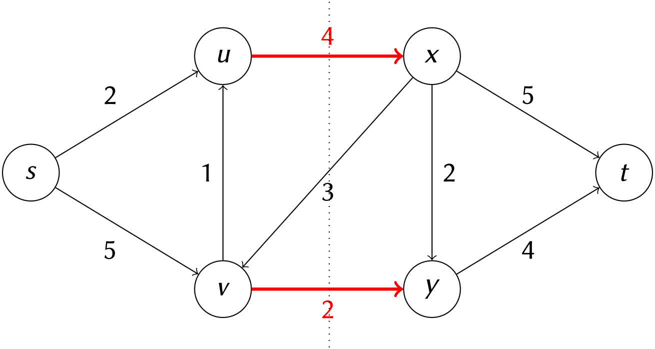
A *cut* in a flow network is a partition of vertices into two sets S and T such that:

- $S \cup T = V$.
- $S \cap T = \emptyset$.
- $s \in S, t \in T$.

The cut splits the graph into two parts such that s and t are completely separated!

Capacity of a cut

The *capacity of a cut* is the sum of the capacity of the edges that "pass" through the cut in the forward direction (i.e. a directed edge from $u \in S$ *to* $v \in T$).



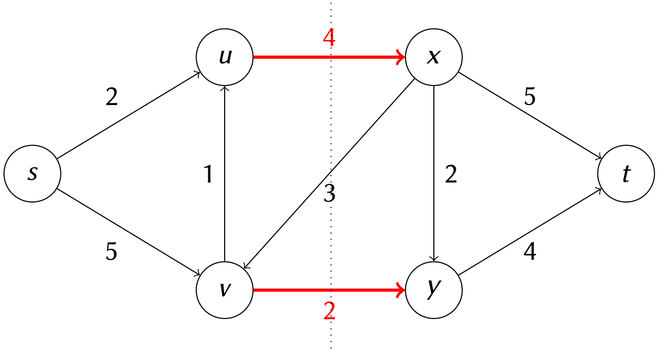
Capacity of cut: 6.

Maximum Flow-Minimum Cut Theorem

Maximum Flow-Minimum Cut Theorem

The maximum flow of a flow network corresponds to the minimum capacity cut of the flow network.

Maximum Flow-Minimum Cut Theorem



- All $s - t$ paths must pass through the red edges.
  - Minimum cut – limits the amount of flow that can be sent to these edges.
  - Maximum flow – must send flow along the edges along the minimum cut.

The Combinatorial Results

General structure of the theorems

Given a structure, the maximum of A corresponds to the minimum of B.

Given a flow network F, the maximum flow of F corresponds to the minimum cut of F.

It turns out there are many other theorems that have this same shape!

Hall's Marriage Theorem

Let $\mathcal{F}$ be a family (or *collection*) of sets and let X be the union of elements in all sets of $\mathcal{F}$.

Transversal of a set

We say that a subset $S \subseteq X$ is a *transversal* for $\mathcal{F}$ if S is comprised of one element from each set in $\mathcal{F}$.

In other words, for each set F in $\mathcal{F}$, pick one element from F to represent the set.

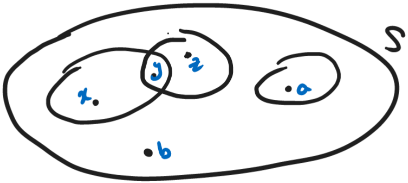
Hall's Marriage Theorem

When does a transversal exist? Let's consider a subcollection $\mathcal{G}$ of sets in $\mathcal{F}$.

Hall's Marriage Theorem

When does a transversal exist? Let's consider a subcollection $\mathcal{G}$ of sets in $\mathcal{F}$.

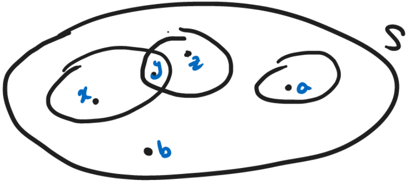
- Assign an element from S to represent a set in $\mathcal{G}$.



Hall's Marriage Theorem

When does a transversal exist? Let's consider a subcollection $\mathcal{G}$ of sets in $\mathcal{F}$.

- Assign an element from S to represent a set in $\mathcal{G}$.



Hmm... assigning an element directly from S might not give us the right assignment because we could accidentally choose an element that doesn't appear in any set in $\mathcal{G}$. Oops...
Let's fix this!

Hall's Marriage Theorem

Let's try again!

When does a transversal exist? Let's consider a subcollection $\mathcal{G}$ of sets in $\mathcal{F}$. We denote Y to be the set of elements that belong to at least one set in $\mathcal{G}$.

Hall's Marriage Theorem

Let's try again!

When does a transversal exist? Let's consider a subcollection $\mathcal{G}$ of sets in $\mathcal{F}$. We denote Y to be the set of elements that belong to at least one set in $\mathcal{G}$.

- Assign an element from Y to represent a set in $\mathcal{G}$.

Hall's Marriage Theorem

Let's try again!
When does a transversal exist? Let's consider a subcollection $\mathcal{G}$ of sets in $\mathcal{F}$. We denote Y to be the set of elements that belong to at least one set in $\mathcal{G}$.

- Assign an element from Y to represent a set in $\mathcal{G}$.

We now have limited our choice of elements to all elements that belong in some set in $\mathcal{G}$. However, what if we don't have enough elements?

Let's enforce that! If a transversal exists, then we need $|\mathcal{G}| \leq |Y|$.

Let's enforce that! If a transversal exists, then we need $|\mathcal{G}| \leq |Y|$. It turns out this is both sufficient and necessary! Therefore, a transversal exists if and only if

$$|\mathcal{G}| \leq |Y| = \left| \bigcup_{S \in \mathcal{G}} S \right|,$$

for every subcollection $\mathcal{G} \subseteq \mathcal{F}$.

Let's enforce that! If a transversal exists, then we need $|\mathcal{G}| \leq |Y|$. It turns out this is both sufficient and necessary! Therefore, a transversal exists if and only if

$$|\mathcal{G}| \leq |Y| = \left| \bigcup_{S \in \mathcal{G}} S \right|,$$

for every subcollection $\mathcal{G} \subseteq \mathcal{F}$.

Our theorem!

Hall's Marriage Theorem

Let $\mathcal{F}$ be a family (collection) of finite sets. Then $\mathcal{F}$ has a transversal if and only if, for every subcollection $\mathcal{G} \subseteq \mathcal{F}$,

$$|\mathcal{G}| \leq \left| \bigcup_{S \in \mathcal{G}} S \right|.$$

Reformulating Hall's Marriage Theorem

In the original formulation of *Hall's Marriage Theorem*, we started off with a family of sets.

- How could we represent this information as a graph?

Reformulating Hall's Marriage Theorem

In the original formulation of *Hall's Marriage Theorem*, we started off with a family of sets.

- How could we represent this information as a graph?
  - Each set in $\mathcal{F}$ represents a *woman* with a list of *men* they wouldn't mind marrying.

Reformulating Hall’s Marriage Theorem

In the original formulation of *Hall’s Marriage Theorem*, we started off with a family of sets.

- How could we represent this information as a graph?
  - Each set in $\mathcal{F}$ represents a *woman* with a list of *men* they wouldn’t mind marrying.
  - Therefore, an edge represents the possibility of a married couple.

Reformulating Hall's Marriage Theorem

In the original formulation of *Hall's Marriage Theorem*, we started off with a family of sets.

- How could we represent this information as a graph?
  - Each set in $\mathcal{F}$ represents a *woman* with a list of *men* they wouldn't mind marrying.
  - Therefore, an edge represents the possibility of a married couple.
  - For *any* collection of women, we need to have enough men to match to each woman.

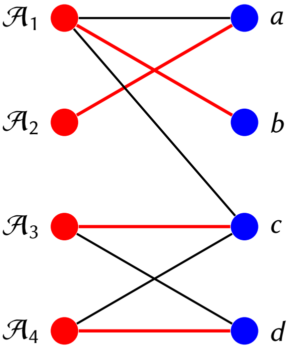
Reformulating Hall's Marriage Theorem

In the original formulation of *Hall's Marriage Theorem*, we started off with a family of sets.

- How could we represent this information as a graph?
    - Each set in $\mathcal{F}$ represents a *woman* with a list of *men* they wouldn't mind marrying.
    - Therefore, an edge represents the possibility of a married couple.
    - For *any* collection of women, we need to have enough men to match to each woman.
- This forms a bipartite graph, where one partition of vertices represents possible women and the other partition of vertices represents possible men. Every woman can be matched with a man if $|W| \leq |N(W)|$, where W is a set of women and $N(W)$ represents the men that is connected to at least one woman in W.

$$
\begin{aligned}
\mathcal{F} &= \{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4\},\\
\mathcal{A}_1 &= \{a, b, c\},\\
\mathcal{A}_2 &= \{a\},\\
\mathcal{A}_3 &= \{c, d\},\\
\mathcal{A}_4 &= \{c, d\}.
\end{aligned}
$$

$$\mathcal{F} = \{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4\},$$
$$\mathcal{A}_1 = \{a, {\color{red}b}, c\},$$
$$\mathcal{A}_2 = \{{\color{red}a}\},$$
$$\mathcal{A}_3 = \{{\color{red}c}, d\},$$
$$\mathcal{A}_4 = \{c, {\color{red}d}\}.$$



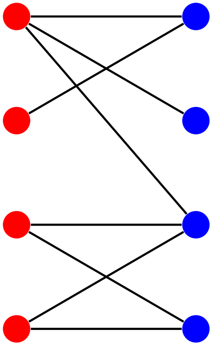
Graph-theoretic formulation of Hall's Marriage Theorem

Hall's Marriage Theorem

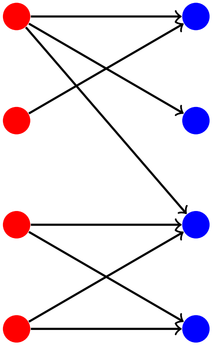
More formally, let $G = (V, E)$ be a bipartite graph with partition V_1 and V_2 such that $V_1 \cup V_2 = V$. Also, suppose that $|V_1| = |V_2|$. Then G has a *perfect* matching if and only if, for every $S \subseteq V_1$,

$$|S| \leq |N(S)|.$$

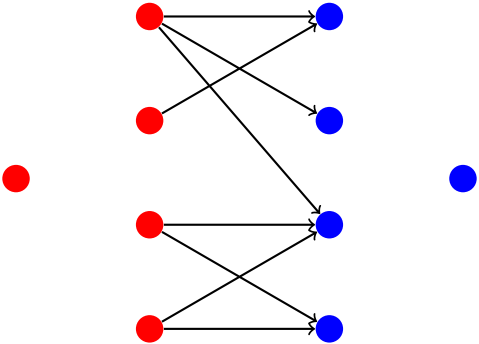
Maximum Flow-Minimum Cut $\implies$ HMT



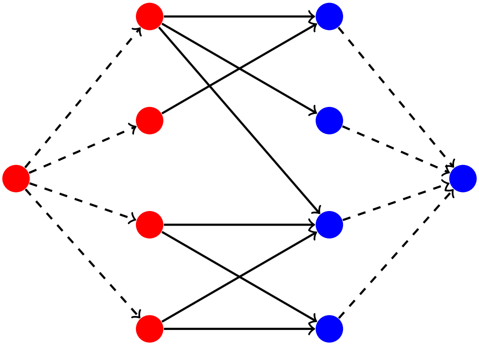
Maximum Flow-Minimum Cut $\implies$ HMT



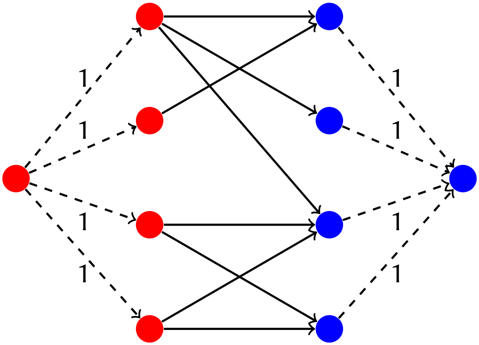
Maximum Flow-Minimum Cut $\implies$ HMT



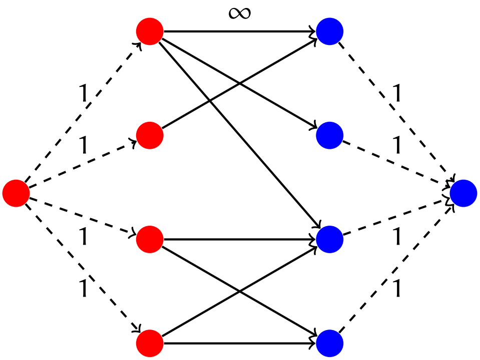
Maximum Flow-Minimum Cut $\implies$ HMT



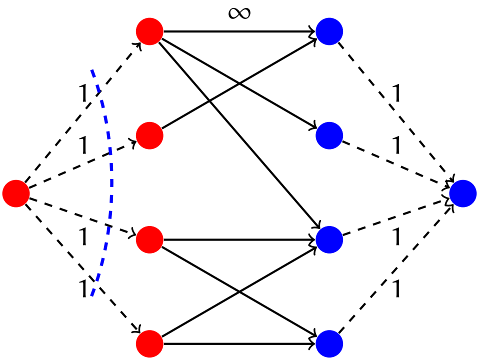
Maximum Flow-Minimum Cut $\implies$ HMT



Maximum Flow-Minimum Cut $\implies$ HMT



Maximum Flow-Minimum Cut $\implies$ HMT

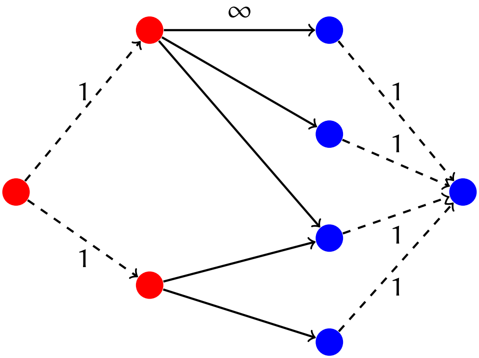


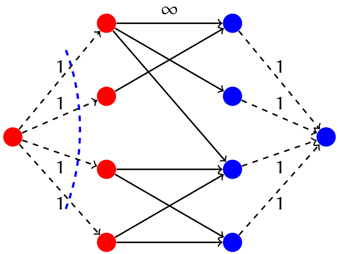
- Edges that cross the minimum cut can only belong to either the red or blue side but not both!

- Edges that cross the minimum cut can only belong to either the red or blue side but not both!
- Take some subset $S \subseteq V_1$. Then $N(S)$ must only belong to a subset of the blue vertices that is neighbours to at least one vertex in S.

- Edges that cross the minimum cut can only belong to either the red or blue side but not both!
- Take some subset $S \subseteq V_1$. Then $N(S)$ must only belong to a subset of the blue vertices that is neighbours to at least one vertex in S.
    - By only considering these vertices, then the maximum flow sends one unit of flow to each of these vertices.

- Edges that cross the minimum cut can only belong to either the red or blue side but not both!
- Take some subset $S \subseteq V_1$. Then $N(S)$ must only belong to a subset of the blue vertices that is neighbours to at least one vertex in S.
    - By only considering these vertices, then the maximum flow sends one unit of flow to each of these vertices.





An example of a bipartite graph that satisfies Hall's condition and an example of a bipartite graph that does not satisfy Hall's condition.

- Taking the last two vertices in the red vertex set does not satisfy Hall's condition. Note that the maximum flow of the second flow network is 3.

Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies:

Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies:

- **Reflexivity**: $R(x, x)$ for all $x \in P$.

Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies:

- **Reflexivity**: $R(x, x)$ for all $x \in P$.
- **Antisymmetry**: $R(x, y), R(y, x) \implies x = y$.

Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies:

- **Reflexivity**: $R(x, x)$ for all $x \in P$.
- **Antisymmetry**: $R(x, y), R(y, x) \implies x = y$.
- **Transitivity**: $R(x, y), R(y, z) \implies R(x, z)$.

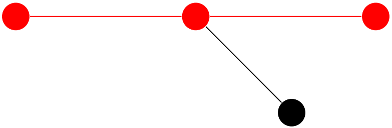
Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies:

- **Reflexivity**: $R(x, x)$ for all $x \in P$.
- **Antisymmetry**: $R(x, y), R(y, x) \implies x = y$.
- **Transitivity**: $R(x, y), R(y, z) \implies R(x, z)$.

Chains of P

Let P be a finite partially ordered set. A chain is a subset $C \subseteq P$ such that, for any two elements $x, y \in C$, either $R(x, y)$ or $R(y, x)$. We say that x and y are *comparable*.



Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies

- **Reflexivity**: $R(x, x)$ for all $x \in P$.
- **Antisymmetry**: $R(x, y), R(y, x) \implies x = y$.
- **Transitivity**: $R(x, y), R(y, z) \implies R(x, z)$.

Antichains of P

Let P be a finite partially ordered set. An antichain is a subset $\mathcal{A} \subseteq P$ such that, no two elements $x, y \in \mathcal{A}$ are comparable; that is, neither $R(x, y)$ nor $R(y, x)$.

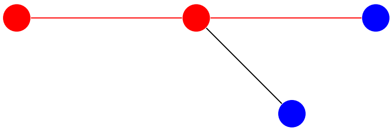
Dilworth's Theorem

Let P be a finite partially ordered set: a set structure with a binary operation R that satisfies

- **Reflexivity**: $R(x, x)$ for all $x \in P$.
- **Antisymmetry**: $R(x, y), R(y, x) \implies x = y$.
- **Transitivity**: $R(x, y), R(y, z) \implies R(x, z)$.

Antichains of P

Let P be a finite partially ordered set. An antichain is a subset $\mathcal{A} \subseteq P$ such that, no two elements $x, y \in \mathcal{A}$ are comparable; that is, neither $R(x, y)$ nor $R(y, x)$.

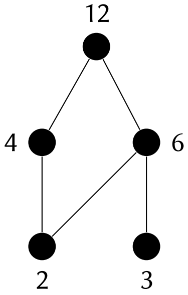


Dilworth's Theorem

It turns out there is a nice connection between the size of an antichain and the number of chains required to cover an entire set.

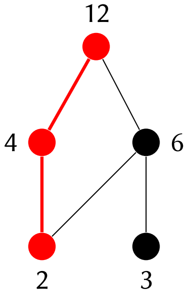
Dilworth's Theorem

It turns out there is a nice connection between the size of an antichain and the number of chains required to cover an entire set.



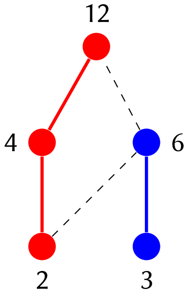
Dilworth's Theorem

It turns out there is a nice connection between the size of an antichain and the number of chains required to cover an entire set.



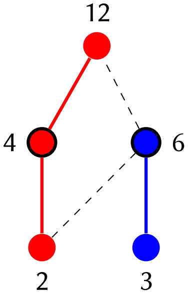
Dilworth's Theorem

It turns out there is a nice connection between the size of an antichain and the number of chains required to cover an entire set.



Dilworth's Theorem

It turns out there is a nice connection between the size of an antichain and the number of chains required to cover an entire set.



Dilworth's Theorem

It turns out that the largest sized antichain corresponds to the smallest number of chains required to cover P! This is our theorem that we want to explore.

Dilworth's Theorem

It turns out that the largest sized antichain corresponds to the smallest number of chains required to cover P! This is our theorem that we want to explore.

Dilworth's Theorem

Let P be a finite partially ordered set and suppose that C is the smallest collection of disjoint chains that partition P. Let $\mathcal{A}$ be a largest antichain of P. Then $|\mathcal{A}| = |C|$.

Reformulating Dilworth's Theorem

Reformulating Dilworth's Theorem

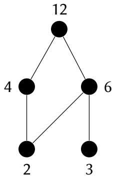
- Every point p in P corresponds to two vertices: p^- and p^+.

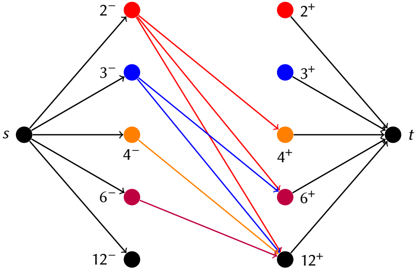
Reformulating Dilworth's Theorem

- Every point p in P corresponds to two vertices: p^- and p^+.
- In P, if $R(x, y)$ where $x \neq y$, then draw an edge with capacity 1 from x^- to y^+. There are additional source and sink vertices.

Reformulating Dilworth's Theorem

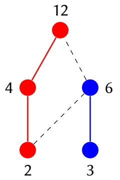
- Every point p in P corresponds to two vertices: p^- and p^+.
- In P, if $R(x, y)$ where $x \neq y$, then draw an edge with capacity 1 from x^- to y^+. There are additional source and sink vertices.

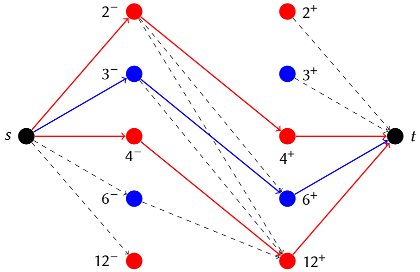


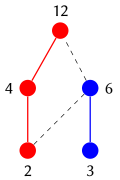


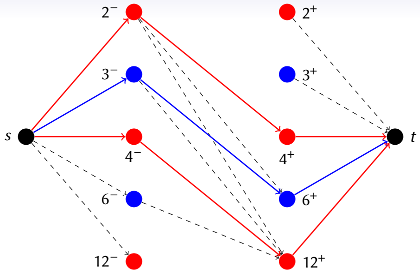
Reformulating Dilworth's Theorem

- Every point p in P corresponds to two vertices: p^- and p^+.
- In P, if $R(x, y)$ where $x \neq y$, then draw an edge with capacity 1 from x^- to y^+. There are additional source and sink vertices.

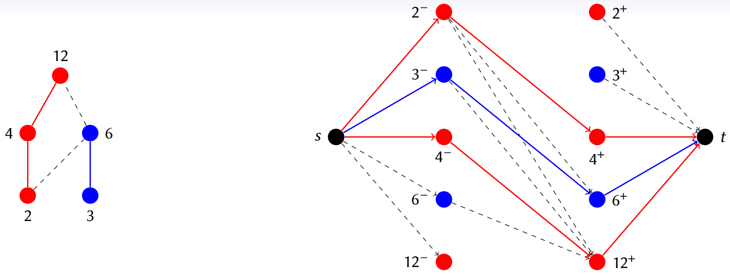




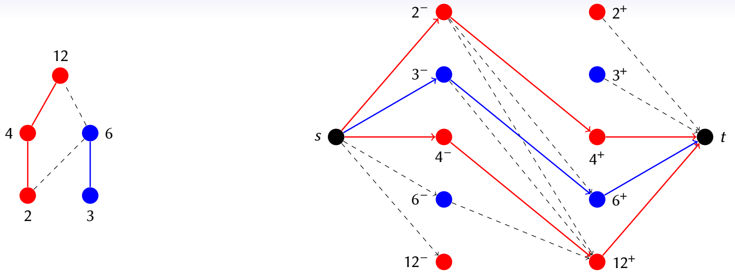




- Let $|f|$ denote the maximum flow of the flow network constructed by the Hasse diagram.

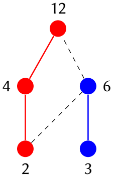


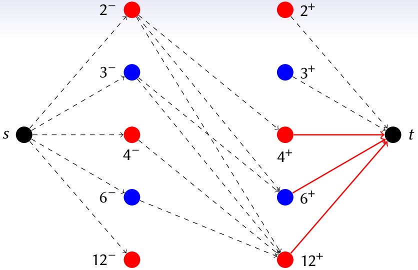
- Let $|f|$ denote the maximum flow of the flow network constructed by the Hasse diagram.
- Then P is partitioned into $|B| = |P| - |f|$ chains.



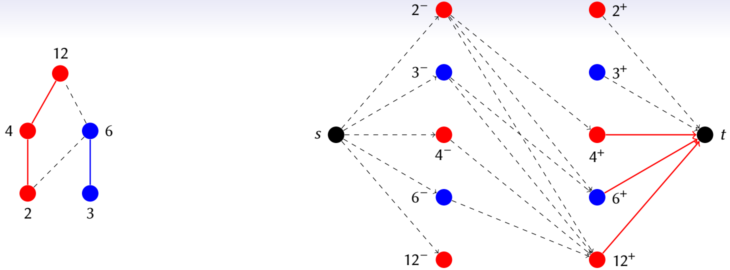
- Let $|f|$ denote the maximum flow of the flow network constructed by the Hasse diagram.
- Then P is partitioned into $|B| = |P| - |f|$ chains.
    - We obtain the two chains in the flow network by following along the paths:

$$\{s \longrightarrow 2^- \longrightarrow 4^+ \longrightarrow 4^- \longrightarrow 12^+ \longrightarrow t\}, \qquad (2 \longrightarrow 4 \longrightarrow 12)$$
$$\{s \longrightarrow 3^- \longrightarrow 6^+ \longrightarrow t\}. \qquad (3 \longrightarrow 6)$$

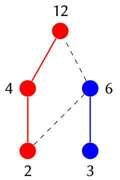


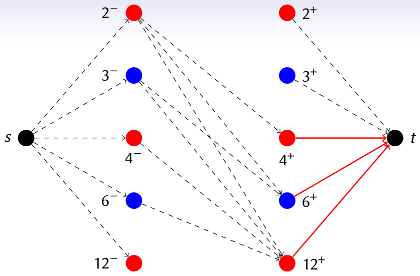


- We now compute the size of the largest antichain.

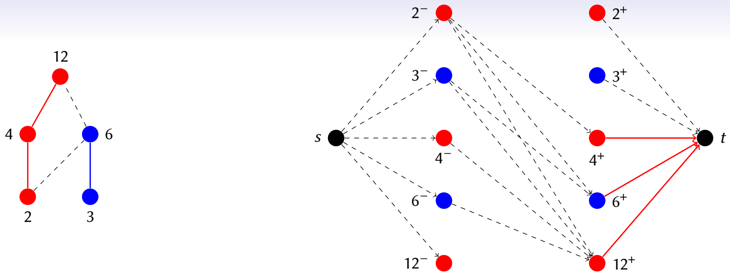


- We now compute the size of the largest antichain.
  - Consider a cut (S, T) in the flow network. Consider all vertices $p \in P$ such that $p^- \in S$ and $p^+ \in T$. Call it A.



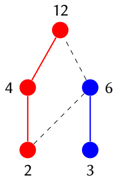


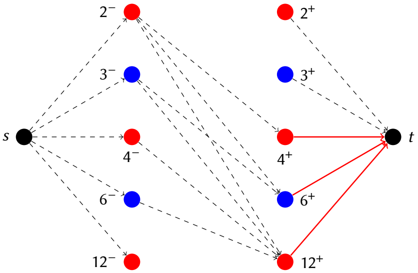
- We now compute the size of the largest antichain.
  - Consider a cut (S, T) in the flow network. Consider all vertices $p \in P$ such that $p^- \in S$ and $p^+ \in T$. Call it A.
  - If $a, b \in A$, then $a^- \in S$ and $b^+ \in T$. If (a^-, b^+) was an edge, then s and t would have to be connected. Therefore, a^- and b^+ has no edge. In other words, a, b are incomparable.



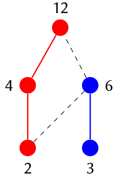
- We now compute the size of the largest antichain.
  - Consider a cut (S, T) in the flow network. Consider all vertices $p \in P$ such that $p^- \in S$ and $p^+ \in T$. Call it A.
  - If $a, b \in A$, then $a^- \in S$ and $b^+ \in T$. If (a^-, b^+) was an edge, then s and t would have to be connected. Therefore, a^- and b^+ has no edge. In other words, a, b are incomparable.
- The only edges that contribute towards the capacity cut are the edges (s, a^-) and (a^+, t). Therefore, this excludes all of the elements in A; that is,

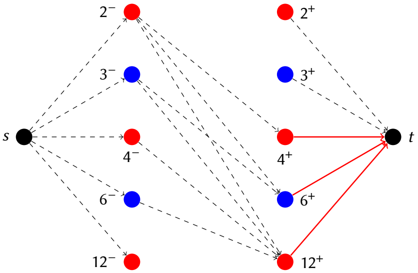
$$c(S, T) = |P| - |A| \implies |A| = |P| - c(S, T).$$



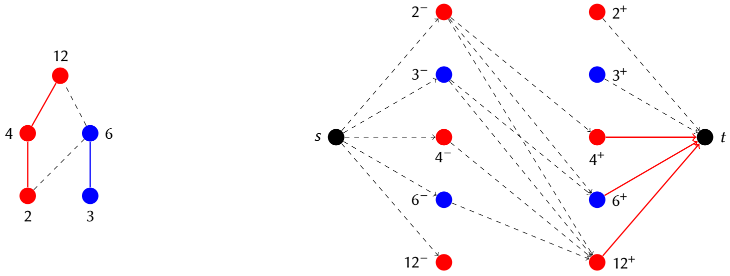


- **Maximum Flow**: $|P| - |B|$ number of partitions. So $|B|$ is minimised (i.e. minimum number of chains).





- **Maximum Flow**: $|P| - |B|$ number of partitions. So $|B|$ is minimised (i.e. minimum number of chains).
- **Minimum Cut**: $|P| - |\mathcal{A}|$; size of an antichain. So $|\mathcal{A}|$ is maximised (i.e. the largest antichain size).

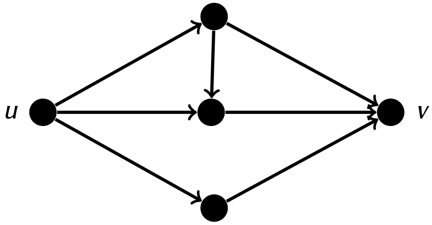


- **Maximum Flow**: $|P| - |B|$ number of partitions. So $|B|$ is minimised (i.e. minimum number of chains).
- **Minimum Cut**: $|P| - |\mathcal{A}|$; size of an antichain. So $|\mathcal{A}|$ is maximised (i.e. the largest antichain size).
- Therefore, the largest sized antichain corresponds to the smallest number of chains that partition P.

Menger's Theorem

In this problem, we are given a directed and unweighted graph $G = (V, E)$ where $u, v \in V$ are two non-adjacent vertices.

- **Question**: How many edge-disjoint paths are there from u to v?



Menger's Theorem

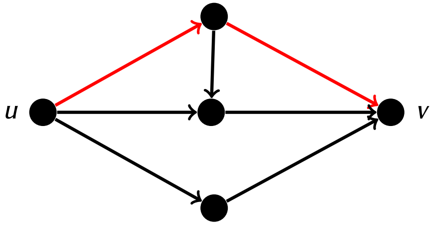
In this problem, we are given a directed and unweighted graph $G = (V, E)$ where $u, v \in V$ are two non-adjacent vertices.

- **Question**: How many edge-disjoint paths are there from u to v?

Menger's Theorem

In this problem, we are given a directed and unweighted graph $G = (V, E)$ where $u, v \in V$ are two non-adjacent vertices.

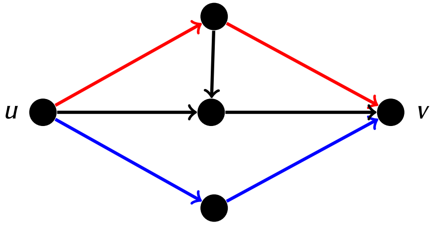
- **Question**: How many edge-disjoint paths are there from u to v?



Menger's Theorem

In this problem, we are given a directed and unweighted graph $G = (V, E)$ where $u, v \in V$ are two non-adjacent vertices.

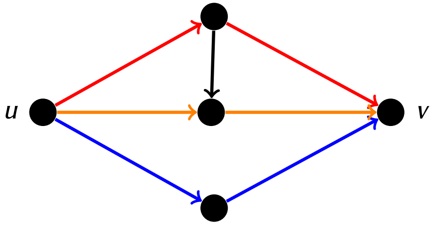
- **Question**: How many edge-disjoint paths are there from u to v?



Menger's Theorem

In this problem, we are given a directed and unweighted graph $G = (V, E)$ where $u, v \in V$ are two non-adjacent vertices.

- **Question**: How many edge-disjoint paths are there from u to v?



Menger's Theorem

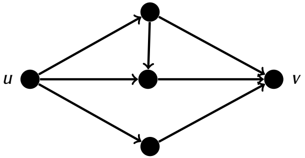
It turns out that the maximum number of edge-disjoint paths from u to v corresponds to the minimum number of edges required to separate u and v!

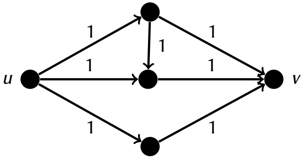
Menger's Theorem

If $u, v \in V$, then there is a (u, v)-separating set of edges S and a collection of edge-disjoint paths $\mathcal{P}$ from u to v such that $|S| = |\mathcal{P}|$.

Reformulating Menger's Theorem

- u is the source and v is the sink vertex.
- Each edge has capacity 1.





- Note that no two $u - v$ paths can share an edge.
  - Therefore, the maximum flow corresponds to the maximum number of edge-disjoint paths from u to v.
- Since each edge has capacity 1, a cut counts the number of edges that pass through the cut.
  - Therefore, the minimum cut corresponds to the minimum number of edges to remove from the graph.

Concluding Remarks

Other theorems that have relations to maximum flow.

- *König's Theorem* – maximal matching.
- *Mirsky's Theorem* – dual of Dilworth's Theorem.
- *Greene's Theorem* – Generalisation of Dilworth's Theorem.