Theorem (Max-Flow Min-Cut Theorem)

The following statements are equivalent:

1. *f is a maximum flow on a flow network G;*
2. *There is an s-t cut (A, B) with $c(A, B) = |f|$;*
3. *There exists no augmenting path in the residual graph G_f.*

Proof.

$2 \Rightarrow 1$: cut capacities are upper bounds for flow values.
$1 \Rightarrow 3$: Augmenting along a path increases a flow's value.
$3 \Rightarrow 2$: The set of nodes reachable from the source in G_f gives the cut. □

Learning Goals

- Matching definition
- Reduction from bipartite matching to max flow
- Hall's theorem and its proof

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

- there are K viewers and L ads;

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

- there are K viewers and L ads;
- each viewer can be shown at most one ad, and each ad can be shown to at most one viewer (due to advertisers' budget constraints);

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

- there are K viewers and L ads;
- each viewer can be shown at most one ad, and each ad can be shown to at most one viewer (due to advertisers' budget constraints);
- each ad is interested in being shown to a certain group of customers.

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

- there are K viewers and L ads;
- each viewer can be shown at most one ad, and each ad can be shown to at most one viewer (due to advertisers' budget constraints);
- each ad is interested in being shown to a certain group of customers.
- We would like to show as many ads as possible.

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

- there are K viewers and L ads;
- each viewer can be shown at most one ad, and each ad can be shown to at most one viewer (due to advertisers' budget constraints);
- each ad is interested in being shown to a certain group of customers.
- We would like to show as many ads as possible.

Model: Bipartite graph $G = (U, V, E)$, U is the set of ads, V the set of viewers; $(u_i, v_j) \in E$ if ad i is interested in being shown to viewer j.

Bipartite Matching

Motivation problem: An ad exchange decides what ads to show to which viewers. At each minute,

- there are K viewers and L ads;
- each viewer can be shown at most one ad, and each ad can be shown to at most one viewer (due to advertisers' budget constraints);
- each ad is interested in being shown to a certain group of customers.
- We would like to show as many ads as possible.

Model: Bipartite graph $G = (U, V, E)$, U is the set of ads, V the set of viewers; $(u_i, v_j) \in E$ if ad i is interested in being shown to viewer j.

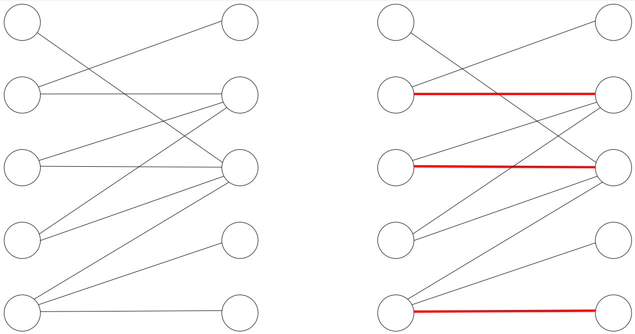
Definition

Given an undirected graph $G = (V, E)$, a set of edges $M \subseteq E$ is a *matching* if each node in V is incident to at most one edge in M.

Maximum Bipartite Matching Problem

Problem (The unweighted maximum bipartite matching problem)

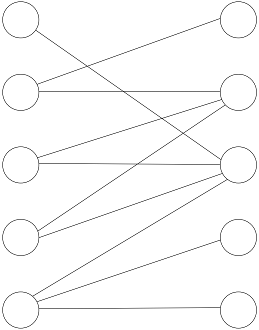
- *Input: a bipartite graph $G = (U, V, E)$. (Recall: this means all edges have one endpoint in U and the other in V.)*
- *Output: a matching M with the maximum cardinality.*

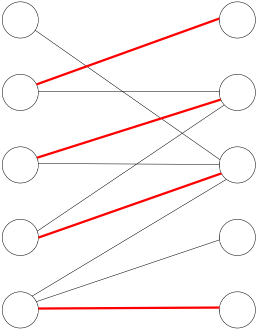


Maximum Bipartite Matching Problem

Problem (The unweighted maximum bipartite matching problem)

- *Input: a bipartite graph $G = (U, V, E)$. (Recall: this means all edges have one node in U and the other in V.)*
- *Output: a matching M with the maximum cardinality.*





Reducing Bipartite Matching to Max Flow

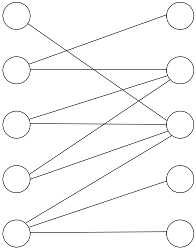
- Input: $G = (U, V, E)$.
- Construct a directed graph G' with node set $\{s\} \cup U \cup V \cup \{t\}$.

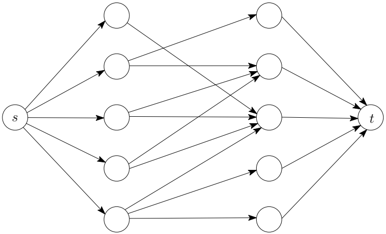
Reducing Bipartite Matching to Max Flow

- Input: $G = (U, V, E)$.
- Construct a directed graph G' with node set $\{s\} \cup U \cup V \cup \{t\}$.
- Add an edge from s to every node in U, and an edge to t from every node in V, each with capacity 1.

Reducing Bipartite Matching to Max Flow

- Input: $G = (U, V, E)$.
- Construct a directed graph G' with node set $\{s\} \cup U \cup V \cup \{t\}$.
- Add an edge from s to every node in U, and an edge to t from every node in V, each with capacity 1.
- For every $(u, v) \in E, u \in U, v \in V$, add directed edge (u, v) to G', with capacity 1.





How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.

How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.
- Moreover, this flow is integral on all edges. Hence on each edge it is either 0 or 1.

How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.
- Moreover, this flow is integral on all edges. Hence on each edge it is either 0 or 1.
- Let $M^* \subseteq E$ be the set of edges in E whose copies in G' carry a flow of 1 in f^*.

How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.
- Moreover, this flow is integral on all edges. Hence on each edge it is either 0 or 1.
- Let $M^* \subseteq E$ be the set of edges in E whose copies in G' carry a flow of 1 in f^*.

Proposition

M^* is a maximum matching in G.

How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.
- Moreover, this flow is integral on all edges. Hence on each edge it is either 0 or 1.
- Let $M^* \subseteq E$ be the set of edges in E whose copies in G' carry a flow of 1 in f^*.

Proposition

M^* is a maximum matching in G.

Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.

How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.
- Moreover, this flow is integral on all edges. Hence on each edge it is either 0 or 1.
- Let $M^* \subseteq E$ be the set of edges in E whose copies in G' carry a flow of 1 in f^*.

Proposition

M^* is a maximum matching in G.

Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.
2. For every integer valued flow f in G' there is a matching M in G, with $|M| = |f|$.

How the reduction works

- Ford-Fulkerson finds a max flow f^* in G' in time $O(|E| \cdot (|U| + |V|) + (|U| + |V|)^2) = O(mn + n^2)$.
- Moreover, this flow is integral on all edges. Hence on each edge it is either 0 or 1.
- Let $M^* \subseteq E$ be the set of edges in E whose copies in G' carry a flow of 1 in f^*.

Proposition

M^* is a maximum matching in G.

Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.
2. For every integer valued flow f in G' there is a matching M in G, with $|M| = |f|$.
3. In particular, $|M^*| = |f^*|$.

□

Proposition

M^* is a maximum matching in G.

Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.
2. For every integer valued flow f in G' there is a matching M in G, with $|M| = |f|$.
3. In particular, $|M^*| = |f^*|$.

Proposition

M^* is a maximum matching in G.

Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.
2. For every integer valued flow f in G' there is a matching M in G, with $|M| = |f|$.
3. In particular, $|M^*| = |f^*|$.

1. Given M, for each $(u, v) \in M$, let $f(s, u) = f(u, v) = f(v, t) = 1$. All other edges carry flow 0. Check f is a flow and $|f| = |M|$.

Proposition

M^* is a maximum matching in G.

Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.
2. For every integer valued flow f in G' there is a matching M in G, with $|M| = |f|$.
3. In particular, $|M^*| = |f^*|$.

1. Given M, for each $(u, v) \in M$, let $f(s, u) = f(u, v) = f(v, t) = 1$. All other edges carry flow 0. Check f is a flow and $|f| = |M|$.
2. Given f, let M be the set of edges between U and V that carry one unit of flow in f. Check: M is a matching and $|M| = |f|$.

Proposition

M^* is a maximum matching in G.

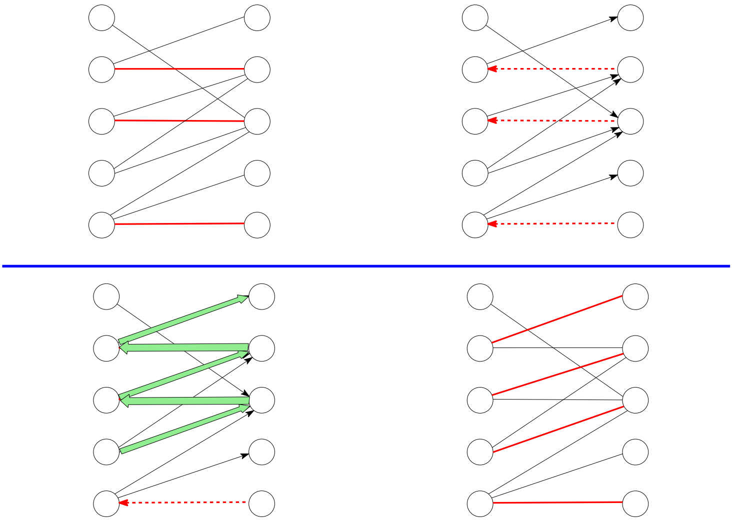
Proof.

1. For every matching M in G, there is a flow f in G' with $|f| = |M|$.
2. For every integer valued flow f in G' there is a matching M in G, with $|M| = |f|$.
3. In particular, $|M^*| = |f^*|$.

1. Given M, for each $(u, v) \in M$, let $f(s, u) = f(u, v) = f(v, t) = 1$. All other edges carry flow 0. Check f is a flow and $|f| = |M|$.
2. Given f, let M be the set of edges between U and V that carry one unit of flow in f. Check: M is a matching and $|M| = |f|$.
3. Special case of 2.

□

Illustration of a step from the algorithm



Hall's Theorem

Definition

In a biparitite graph $G = (U, V, E)$, a matching M is said to be *complete* on U if $|M| = |U|$. When $|U| = |V|$, such a matching is called *perfect*.

Hall's Theorem

Definition

In a biparitite graph $G = (U, V, E)$, a matching M is said to be *complete* on U if $|M| = |U|$. When $|U| = |V|$, such a matching is called *perfect*.

Notation: For a set of nodes $S \subseteq U$, denote by $\Gamma(S)$ the "neighbors" of S, i.e., $\Gamma(S) = \{v \in V \mid \exists u \in S \text{ s.t. } (u, v) \in E\}$.

Hall's Theorem

Definition

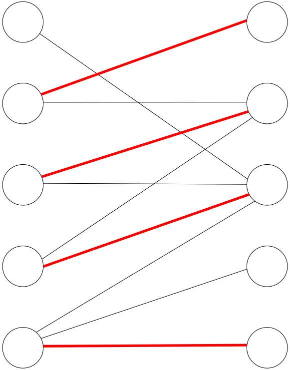
In a biparitite graph $G = (U, V, E)$, a matching M is said to be *complete* on U if $|M| = |U|$. When $|U| = |V|$, such a matching is called *perfect*.

Notation: For a set of nodes $S \subseteq U$, denote by $\Gamma(S)$ the "neighbors" of S, i.e., $\Gamma(S) = \{v \in V \mid \exists u \in S \text{ s.t. } (u, v) \in E\}$.

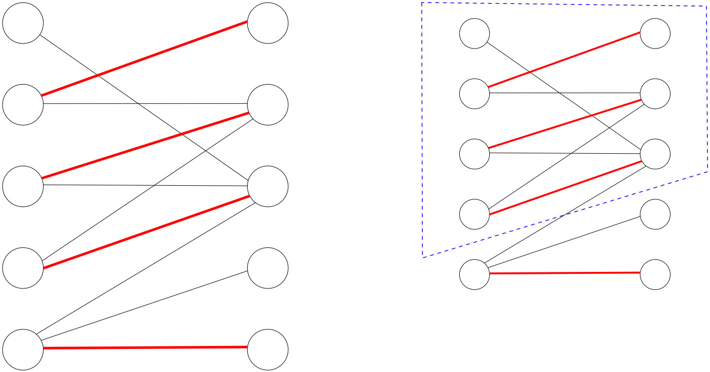
Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

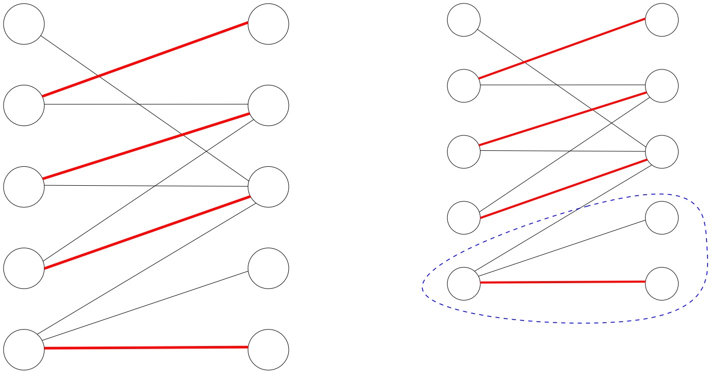
Hall's Theorem Illustration



Hall's Theorem Illustration



Hall's Theorem Illustration



Proof of Hall's Theorem

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

$\Rightarrow$: If G has a complete matching M, for any $u \in U$, let $\varphi(u) \in V$ be the vertex matched to u in M. Then $\varphi(u) \neq \varphi(u')$ for any $u \neq u'$.

Proof of Hall's Theorem

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

$\Rightarrow$: If G has a complete matching M, for any $u \in U$, let $\varphi(u) \in V$ be the vertex matched to u in M. Then $\varphi(u) \neq \varphi(u')$ for any $u \neq u'$.
For any $S \subseteq U$, $|\Gamma(S)| \geq |\{\varphi(u)\}_{u \in S}| = |S|$.

Proof of Hall's Theorem

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

$\Rightarrow$: If G has a complete matching M, for any $u \in U$, let $\varphi(u) \in V$ be the vertex matched to u in M. Then $\varphi(u) \neq \varphi(u')$ for any $u \neq u'$.
For any $S \subseteq U$, $|\Gamma(S)| \geq |\{\varphi(u)\}_{u \in S}| = |S|$.
$\Leftarrow$: Consider the flow network G' in the reduction. If the max matching in G has fewer than $|U|$ edges, the max flow in G' is smaller than $|U|$.

Proof of Hall's Theorem

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

$\Rightarrow$: If G has a complete matching M, for any $u \in U$, let $\varphi(u) \in V$ be the vertex matched to u in M. Then $\varphi(u) \neq \varphi(u')$ for any $u \neq u'$.
For any $S \subseteq U$, $|\Gamma(S)| \geq |\{\varphi(u)\}_{u \in S}| = |S|$.
$\Leftarrow$: Consider the flow network G' in the reduction. If the max matching in G has fewer than $|U|$ edges, the max flow in G' is smaller than $|U|$.
By Max Flow Min Cut Theorem, there is an s-t cut (A, B) in G' with $c(A, B) < |U|$.

Proof of Hall's Theorem cont.

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

Claim: There exists a min cut (A, B) such that $\Gamma(A \cap U) \subseteq A \cap V$.

Proof of Hall's Theorem cont.

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

Claim: There exists a min cut (A, B) such that $\Gamma(A \cap U) \subseteq A \cap V$.
Reason: If $(u, v) \in E$, $u \in A \cap U, v \in B \cap V$, then edges into v contribute ≥ 1 to $c(A, B)$, and edges leaving v contribute 0. Moving v to A, then edges into v contribute 0, and edges leaving v contribute 1.

Proof of Hall's Theorem cont.

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

Claim: There exists a min cut (A, B) such that $\Gamma(A \cap U) \subseteq A \cap V$.
Reason: If $(u, v) \in E$, $u \in A \cap U, v \in B \cap V$, then edges into v contribute ≥ 1 to $c(A, B)$, and edges leaving v contribute 0. Moving v to A, then edges into v contribute 0, and edges leaving v contribute 1.
So for any s-t cut (A, B), we can move all $v \in \Gamma(U \cap A)$ to A without increasing the cut's capacity.

Proof of Hall's Theorem cont.

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

Claim: There exists a min cut (A, B) such that $\Gamma(A \cap U) \subseteq A \cap V$.
Reason: If $(u, v) \in E$, $u \in A \cap U, v \in B \cap V$, then edges into v contribute ≥ 1 to $c(A, B)$, and edges leaving v contribute 0. Moving v to A, then edges into v contribute 0, and edges leaving v contribute 1.
So for any s-t cut (A, B), we can move all $v \in \Gamma(U \cap A)$ to A without increasing the cut's capacity.
So there is an s-t cut (A, B) with $c(A, B) < |U|$ and $\Gamma(A \cap U) \subseteq A$.

Proof of Hall's Theorem cont.

Theorem (Hall's Theorem)

A bipartite graph $G = (U, V, E)$ has a complete matching on U if and only if for any $S \subseteq U$, $|\Gamma(S)| \geq |S|$.

Proof.

Claim: There exists a min cut (A, B) such that $\Gamma(A \cap U) \subseteq A \cap V$.
Reason: If $(u, v) \in E$, $u \in A \cap U, v \in B \cap V$, then edges into v contribute ≥ 1 to $c(A, B)$, and edges leaving v contribute 0. Moving v to A, then edges into v contribute 0, and edges leaving v contribute 1.
So for any s-t cut (A, B), we can move all $v \in \Gamma(U \cap A)$ to A without increasing the cut's capacity.
So there is an s-t cut (A, B) with $c(A, B) < |U|$ and $\Gamma(A \cap U) \subseteq A$.

$$|U| > c(A, B) = |U \setminus A| + |A \cap V| \geq |U \setminus A| + |\Gamma(A \cap U)|.$$
$$\Rightarrow |U| - |U \setminus A| = |A \cap U| > |\Gamma(A \cap U)|.$$

A Glimpse at Better Algorithms

- Reducing to network flows and solving by Ford-Fulkerson is not the fastest algorithm to find maximum bipartite matchings.

A Glimpse at Better Algorithms

- Reducing to network flows and solving by Ford-Fulkerson is not the fastest algorithm to find maximum bipartite matchings.
- Fastest algorithm known: Hopcroft-Karp algorithm, which runs in time $O(m\sqrt{n})$.

A Glimpse at Better Algorithms

- Reducing to network flows and solving by Ford-Fulkerson is not the fastest algorithm to find maximum bipartite matchings.
- Fastest algorithm known: Hopcroft-Karp algorithm, which runs in time $O(m\sqrt{n})$.
- Basic idea: In each iteration, instead of augmenting along a path, look for a *maximal* set of vertex-disjoint *shortest* augmenting paths, and augment along all of them.

A Glimpse at Better Algorithms

- Reducing to network flows and solving by Ford-Fulkerson is not the fastest algorithm to find maximum bipartite matchings.
- Fastest algorithm known: Hopcroft-Karp algorithm, which runs in time $O(m\sqrt{n})$.
- Basic idea: In each iteration, instead of augmenting along a path, look for a *maximal* set of vertex-disjoint *shortest* augmenting paths, and augment along all of them.
- Similar ideas (of augmenting along a collection of shortest paths that "block" s from t) lead to faster algorithms for the max flow problem: Dinic's algorithm, running in time $O(mn^2)$.
- (The algorithm by Edmonds and Karp that run in time $O(m^2n)$ is an important predecessor.)