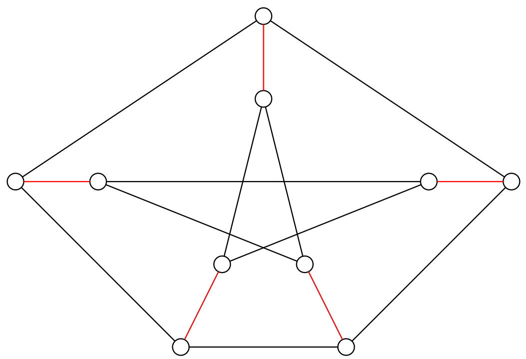
Matching

Input Given a (undirected) graph $G = (V, E)$

Goal Find a matching of maximum cardinality

- A matching is $M \subseteq E$ such that at most one edge in M is incident on any vertex



Bipartite Matching

Input Given a bipartite graph $G = (L \cup R, E)$

Goal Find a matching of maximum cardinality

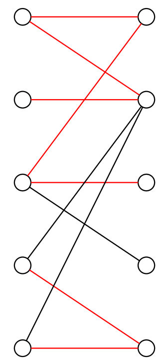
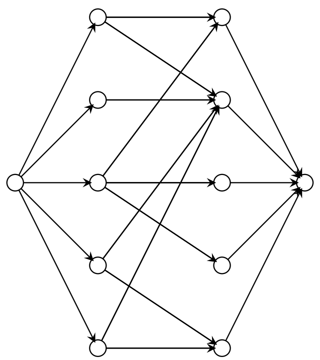


Figure : Maximum matching has 4 edges

Reduction to Max-Flow

Max-Flow Construction

Given graph $G = (L \cup R, E)$ create flow-network $G' = (V', E')$ as follows:



- $V' = L \cup R \cup \{s, t\}$ where s and t are the new source and sink
- Direct all edges in E from L to R, and add edges from s to all vertices in L and from each vertex in R to t
- Capacity of every edge is 1

Correctness: Matching to Flow

Proposition

If G has a matching of size k then G' has a flow of value k.

Proof.

Let M be matching of size k. Construct flow that send on unit along each edge in M, and in edges to and from vertices of $L \cup R$ that has some edge in M. This flow has value k. □

Correctness: Flow to Matching

Proposition

If G' has a flow of value k then G has a matching of size k.

Proof.

Consider flow f of value k.

- Observe that f is an integral flow. Thus each edge has flow 1 or 0
- Consider the set M of edges from L to R that have flow 1
  - M has k edges because value of flow is equal to the number of non-zero flow edges crossing cut $(L \cup \{s\}, R \cup \{t\})$
  - Each vertex has at most one edge in M incident upon it

□

Correctness of Reduction

Theorem

The maximum flow value in G' = maximum cardinality of matching in G

Consequence

Thus, to find maximum cardinality matching in G, we construct G' and find the maximum flow in G'

Running Time

For graph G with n vertices and m edges G' has $O(n+m)$ edges, and $O(n)$ vertices.

- Generic Ford-Fulkerson: Running time is $O(mC) = O(nm)$ since $C = n$
- Capacity scaling: Running time is $O(m^2 \log C) = O(m^2 \log n)$

Better known running times: $O(m\sqrt{n})$ and $O(n^{2.344})$

Perfect Matchings

Definition

A matching M is said to be perfect if every vertex has one edge in M incident upon it.

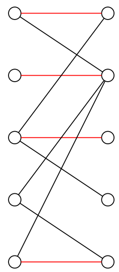


Figure : This graph does not have a perfect matching

Characterizing Perfect Matchings

Problem

When does a bipartite graph have a perfect matching?

- Clearly $|L| = |R|$
- Are there any necessary and sufficient conditions?

A Necessary Condition

Lemma

If $G = (L \cup R, E)$ has a perfect matching then for any $X \subseteq L$, $|N(X)| \geq |X|$, where $N(X)$ is the set of neighbors of vertices in X

Proof.

Since G has a perfect matching, every vertex of X is matched to a different neighbor, and so $|N(X)| \geq |X|$ □

Hall's Theorem

Theorem (Frobenius-Hall)

Let $G = (L \cup R, E)$ be a bipartite graph with $|L| = |R|$. G has a perfect matching if and only if for every $X \subseteq L$, $|N(X)| \geq |X|$

One direction is the necessary condition.
For the other direction we will show the following:

- create flow network G' from G
- if $|N(X)| \geq |X|$ for all X, show that minimum $s - t$ cut in G' is of capacity $n = |L| = |R|$
- implies that G has a perfect matching

Proof of Sufficiency

Assume $|N(X)| \geq |X|$ for each $X \in L$. Then show that min $s - t$ cut in G' is of capacity n.

Let (A, B) be an *arbitrary* $s - t$ cut in G'

- let $X = A \cap L$ and $Y = A \cap R$
- cut capacity is equal to $(|L| - |X|) + |Y| + |N(X) - Y|$
- $|N(X) - Y| \geq |N(X)| - |Y|$ and by assumption $|N(X)| \geq |X|$ and hence $|N(X) - Y| \geq |X| - |Y|$
- cut capacity is therefore at least $|L| - |X| + |Y| + |X| - |Y| \geq |L| = n$.

Application: assigning jobs to people

- n jobs or tasks
- m people
- for each job a set of people who can do that job
- for each person a limit on number of jobs
- Goal: find an assignment of jobs to people so that all jobs are assigned and no person is overloaded

Reduce to max-flow similar to matching.

Arises in many settings. Using *minimum-cost flows* can also handle the case when assigning a job i to person j costs c_{ij} and goal is assign all jobs but minimize cost of assignment.

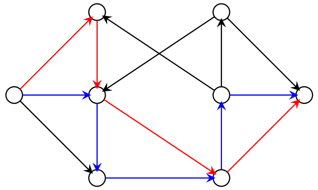
Matchings in General Graphs

Matchings in general graphs more complicated.

There is a polynomial time algorithm to compute a maximum matching in a general graph. Running time is $O(m\sqrt{n})$.

Edge-Disjoint Paths in Directed Graphs

Definition



A set of paths is edge disjoint if no two paths share an edge.

Problem

Given a directed graph with two special vertices s and t, find the *maximum* number of edge disjoint paths from s to t

Applications: Fault tolerance in routing — edges/nodes in networks can fail. Disjoint paths allow for planning backup routes in case of failures.

Reduction to Max-Flow

Problem

Given a directed graph G with two special vertices s and t, find the maximum number of edge disjoint paths from s to t

Reduction

Consider G as a flow network with edge capacities 1, and find max-flow.

Correctness of Reduction

Lemma
If G has k edge disjoint paths then there is a flow of vlaue k

Proof.
Set $f(e) = 1$ if e belongs to the set of edge disjoint paths; other-wise set $f(e) = 0$. This defines a flow of value k. □

Lemma
If G has a flow of value k then there are k edge disjoint paths.

Proof.
Left as exercise. □

Running Time

Theorem
The number of edge disjoint paths in G can be found in $O(mn)$ time

Run Ford-Fulkerson algorithm. Maximum possible flow is n and hence run-time is $O(nm)$.

Menger's Theorem

Theorem (Menger)
Let G be a directed graph. Size of the minimum-cut between s and t is equal to the number of edge-disjoint paths in G between s and t.

Proof.
Maxflow-mincut theorem and integrality of flow. □

Menger proved his theorem before Maxflow-Mincut theorem! Maxflow-Mincut theorem is a generalization of Menger's theorem to capacitated graphs.

Edge Disjoint Paths in Undirected Graphs

Problem
Given an undirected graph G, find the maximum number of edge disjoint paths in G

Reduction:

- create directed graph H by adding directed edges (u, v) and (v, u) for each edge uv in G.
- compute maximum $s - t$ flow in H

Problem: Both edges (u, v) and (v, u) may have non-zero flow!

Fixing the Solution: Acyclicit of Flows

Proposition

In any flow network, there is a maximum flow f that is acyclic. *Further if all the capacities are integral, then there is such a flow f that is also integral.*

Proof.

- Let f be a maximum flow. $E' = \{e \in E \mid f(e) > 0\}$
- Suppose there is a directed cycle C in E'
- Let e' be the edge in C with least amount of flow
- For each $e \in C$, reduce flow $f(e')$. Remains a flow
- flow on e' is reduced to 0
- Claim: flow value from s to t does not change (why?)
- iterate till no cycles

□

Multiple Sources and Sinks

- Directed graph G with edge capacities $c(e)$
- source nodes $S = \{s_1, s_2, \ldots, s_k\}$
- sink nodes $t_1, t_2, \ldots, t_\ell$
- sources and sinks are *disjoint*

Maximum Flow: send as much flow as possible from the sources to the sinks. Sinks don't care which source they get flow from.

Minimum Cut: find a minimum capacity set of edge E' such that removing E' disconnects every source from every sink.

Reduction to Single-Source Single-Sink

- Add a *source* node s and a *sink* node t
- Add edges $(s, s_1), (s, s_2), \ldots, (s, s_k)$
- Add edges $(t_1, t), (t_2, t), \ldots, (t_\ell, t)$
- Set the capacity of the new edges to be ∞

Supplies and Demands

A further generalization:

- source s_i has a supply of $S_i \geq 0$
- sink t_j has a demand of $D_j \geq 0$ units

Question: is there a flow from source to sinks such that supplies are not exceeded and demands are met?