CS 163 & CS 265: Graph Algorithms

Week 9: Matching

Lecture 9a: Maximum matching, independent sets, and rectangle partition

David Eppstein
University of California, Irvine

Winter Quarter, 2024

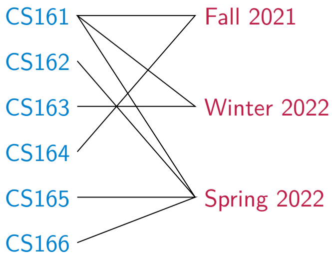
Overview

Bipartite graphs

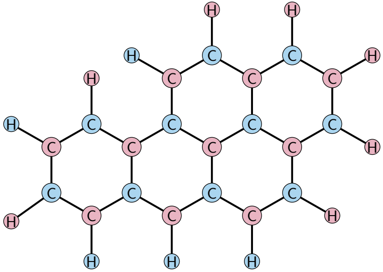
Remember cut property of minimum spanning trees?

"Cut": split vertices into two sets, look at edges from one to other

"Bipartite": all edges are part of a single cut
= can color vertices red and blue so each edge has both colors



Relations between two different kinds of things

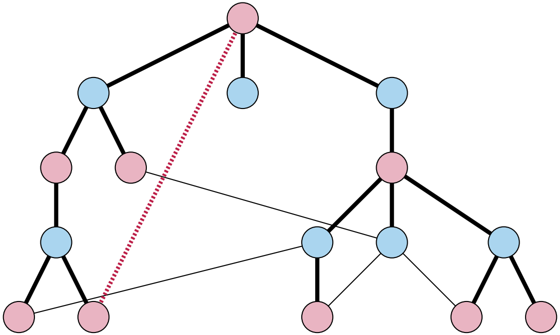


Structures where all cycles have even length

Easy linear-time algorithm for testing bipartiteness

Find any spanning tree (BFS tree, DFS tree, etc)

Color vertices at even distance from root red, odd distance blue



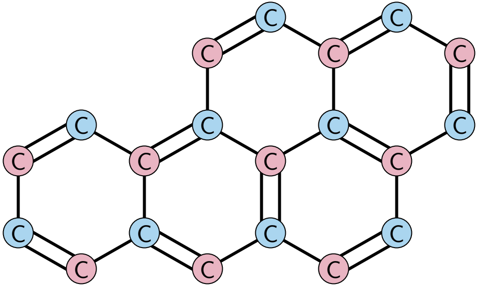
If all edges have both colors, graph is bipartite

If not, bad edge + tree path = odd cycle, graph is not bipartite

Matchings

A matching is a set of edges that don't touch each other
(no two matched edges share a vertex)

Like the double bonds in this benzopyrene:



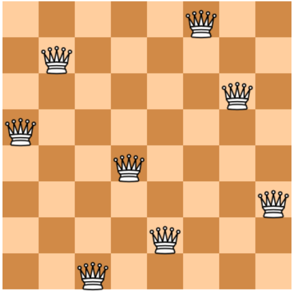
Maximum matching (a matching with as many edges as possible) can be found in polynomial time, in any graph

We will only prove the (easier) case of bipartite graphs

Independent sets

An independent set is a set of vertices that don't touch each other
(no two are endpoints of the same edge)

In bipartite graphs, both sets of vertices are independent, but there might be other larger independent sets



Eight queens puzzle:
place 8 queens on chessboard,
no two attacking each other

Independent set in graph of possible queen moves

Maximum independent sets are NP-hard to find in arbitrary graphs

But they can be found using matching in bipartite graphs!

Kőnig's theorem [Dénes Kőnig, 1931]

Let $M = \#$ edges in maximum matching,
$I = \#$ vertices in max independent set
Then in bipartite graphs, $M + I = n$

Easy direction: Independent set can only include one vertex from each matched edge, so $I \leq n - M$ (true for all graphs)

Hard direction: We can find an independent set with all unmatched vertices + one vertex from each matched edge, so $I \geq n - M$

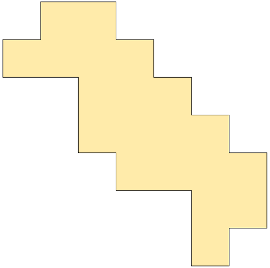
Proof: Later in lecture,
based on algorithm for finding matchings

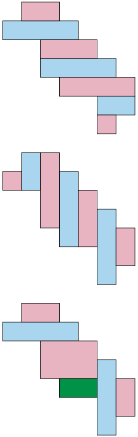
Dénes Kőnig
(1884–1944)

Father Gyula Kőnig was also a famous mathematician

Rectangle partitions

Geometric application of bipartite independent sets





Input: polygon with axis-parallel sides

Goal: slice into as few rectangles as possible

If we only slice horizontally or only vertically
$\Rightarrow$ too many pieces

We need a more careful algorithm

Why slice polygons into rectangles?

Decomposition of lithography masks into simple shapes for VLSI

DNA microarray design

Convolution operations in image processing

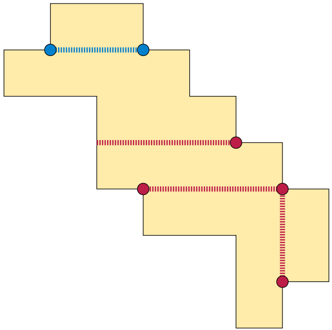
Compression of bitmap images

Radiation therapy planning

Robot self-assembly planning

(as listed in Eppstein [2009])

What's really going on here?



Polygon corners are convex (90° interior) or concave (270°)

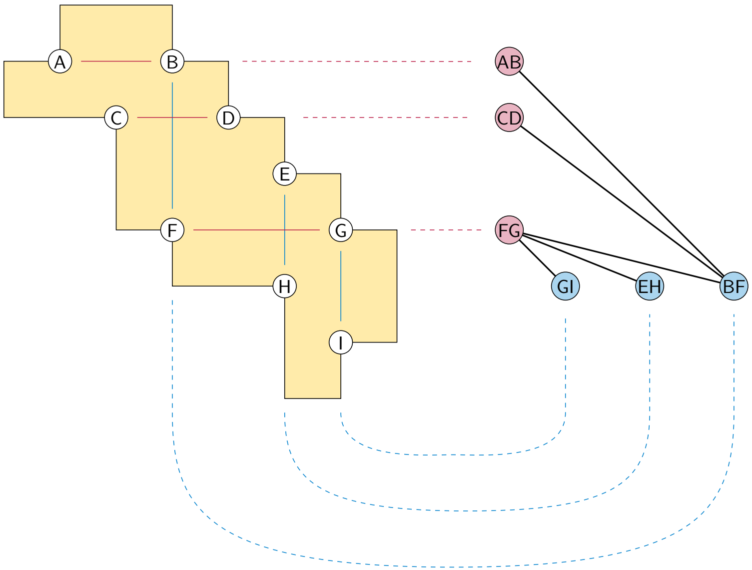
We need to slice horizontally or vertically at concave corners to make rectangular pieces

We can use fewer slices whenever we can slice two corners with a single cut

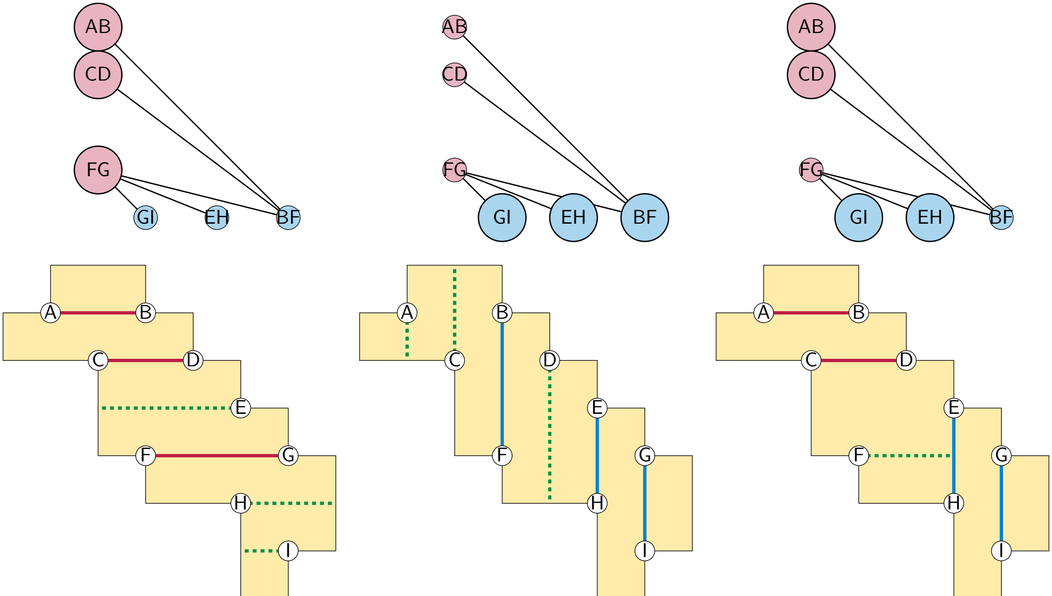
But we lose the improvement when two slices touch or cross each other or when a slice reaches only one concave corner

New goal: Find many non-touching two-concave-corner slices

Bipartite graph of 2-corner slices that touch or cross



Independent sets and the rectangles they produce

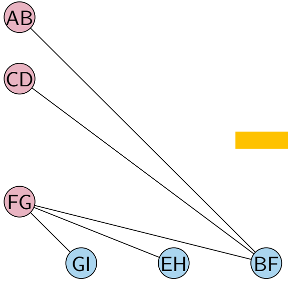


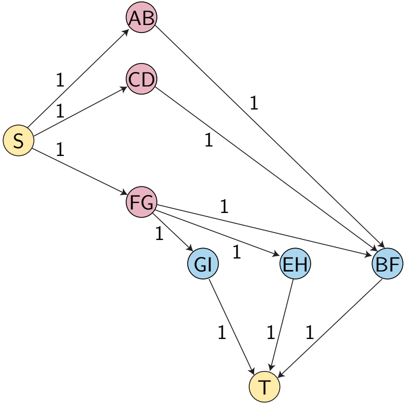
Finding bipartite maximum matchings

Maximum bipartite matching as flow

Given a bipartite graph G:

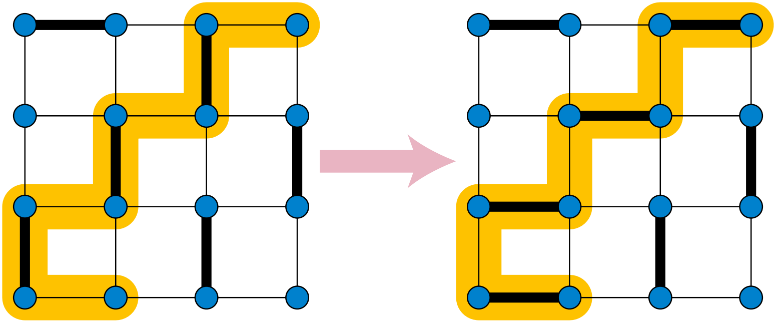
- Add new vertices s connected to all red vertices, t connected to all blue vertices
- Direct all edges $s \to$ red $\to$ blue $\to t$, with capacity $= 1$
- Find an integer maximum flow; matching $=$ edges with flow 1





Alternating paths

Suppose you have a matching that is not maximum, and you find a path starting and ending at unmatched vertices, alternating between matched and unmatched edges



Can improve matching by changing unmatched $\leftrightarrow$ matched along the path

This is just an augmenting path in the flow problem!

Proof of Kőnig's theorem

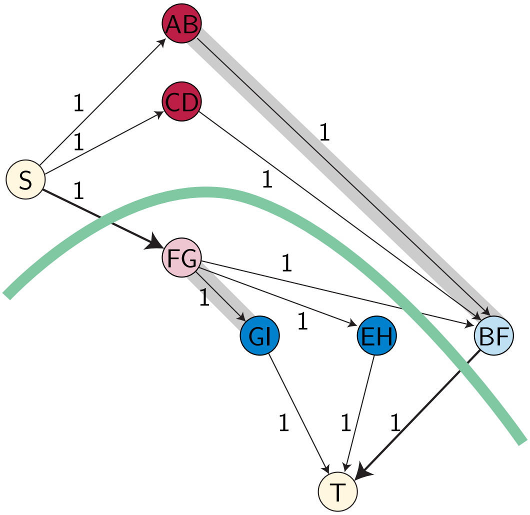
Max flow min cut theorem $\Rightarrow$ cut S–T with no residual capacity across cut

Independent set: red in S + blue in T

Includes all unmatched vertices (with residual edge from s or to t)

Cannot avoid both endpoints of a matched edge (would give a residual edge across the cut)

So independent set size $\geq$ unmatched vertices + matched edges



Time bounds for bipartite matching

Repeated augmenting paths:

- Each path increases size of matching
- $\leq n/2$ paths, BFS per path $\Rightarrow O(mn)$

Hopcroft–Karp–Karzanov [Hopcroft and Karp 1973; Karzanov 1973]:

- Use BFS to find many short augmenting paths in linear time
- After using these paths, shortest augmenting path gets longer
- Once length becomes $\geq \sqrt{n}$, only $O(\sqrt{n})$ paths left to find
- $\Rightarrow$ time $O(m\sqrt{n})$

New breakthrough [Chen et al. 2022]

- Convert bipartite matching to flow
- Fast new flow algorithm
- $O(m^{1+\varepsilon})$ for any $\varepsilon > 0$

Non-bipartite matching

Algorithms

Non-bipartite version of Hopcroft–Karp–Karzanov by Silvio Micali and Vijay Vazirani (now a UCI professor) from 1980, time $O(m\sqrt{n})$

An application

Kidney transplant donors and recipients enter the system together (usually the donor is a friend or relative of the patient) but often don't have compatible immune systems

Form a graph whose vertices are donor-recipients pairs and whose edges represent double compatibility (donor of one pair is compatible with recipients of the other and vice versa)

Set up a double transplant operation with four patients (two donors and two recipients) for each matched edge

Morals of the story

Bipartiteness is easy to test, natural in many matching problems, and makes those problems simpler

In bipartite graphs, matching and independent sets are equivalent, but in other graphs independent sets are hard

Application to optimal rectangle partition

Solve by converting to max flow

References

Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *Proc. 63rd IEEE Symposium on Foundations of Computer Science*, 2022.

David Eppstein. Graph-theoretic solutions to computational geometry problems. In *Proc. 35th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2009), Montpellier, France, 2009*, volume 5911 of *Lecture Notes in Computer Science*, pages 1–16. Springer-Verlag, 2009. doi:10.1007/978-3-642-11409-0_1.

John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, 2:225–231, 1973. doi:10.1137/0202019.

Alexander V. Karzanov. An exact estimate of an algorithm for finding a maximum flow, applied to the problem on representatives. *Problems in Cybernetics*, 5:66–70, 1973.

Dénes Kőnig. Gráfok és mátrixok. *Matematikai és Fizikai Lapok*, 38:116–119, 1931.

Silvio Micali and Vijay V. Vazirani. An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs. In *Proc. 21st IEEE Symp. Foundations of Computer Science (FOCS 1980)*, pages 17–27, 1980. doi:10.1109/SFCS.1980.12.