

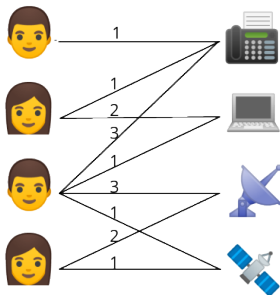
Maximum Cardinality Matching Problem

Narek Bojikian, Piotr Witkowski, Martin Vogel



June 11, 2019

Assignment Problem

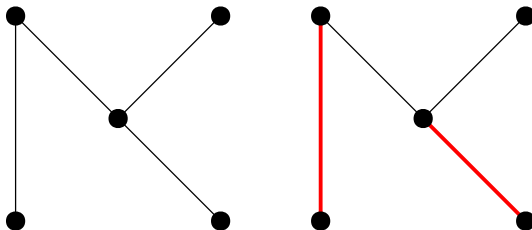


Definitions

Matching

Definition

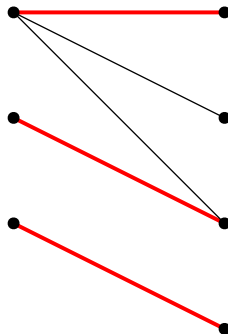
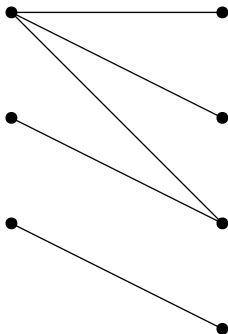
Set of non overlapping edges.



Bipartite Matching

Definition

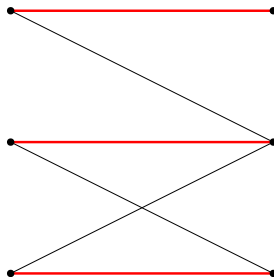
Matching on a bipartite graph.



Perfect Matching

Definition

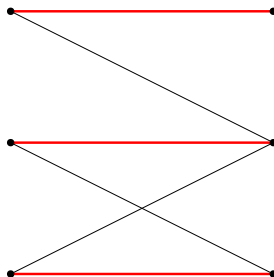
Matching of size $\frac{|V(G)|}{2}$.



Maximum Matching

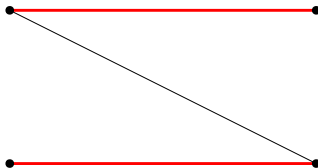
Definition

Matching of the maximum size.

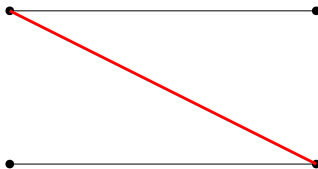


Maximum Matching vs Maximal Matching

Maximum: matching of the maximum size:



Maximal: no more edges can be added:



Bipartite graphs

Hall's Theorem

Definition

A bipartite graph $G(U, W, E)$, has a matching satisfying U if and only if $|N(X)| \geq |X|$ for every nonempty subset $X \subseteq U$.

Hall's Theorem - Examples

- Examples on the board –

König's Theorem

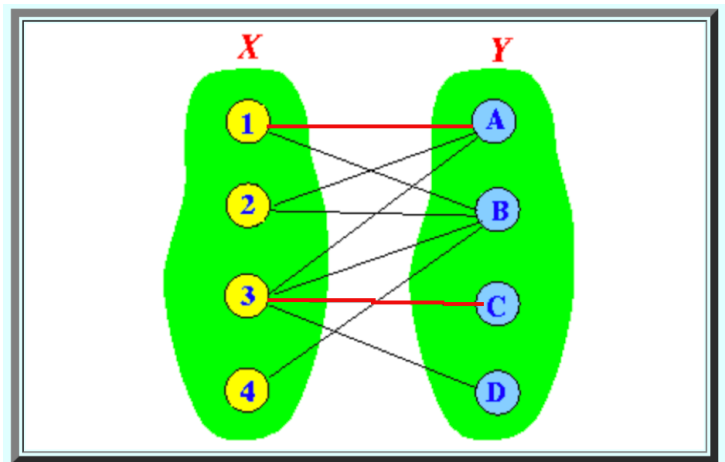
Definition

The maximum matching for a bipartite graph equals its minimum vertex cover

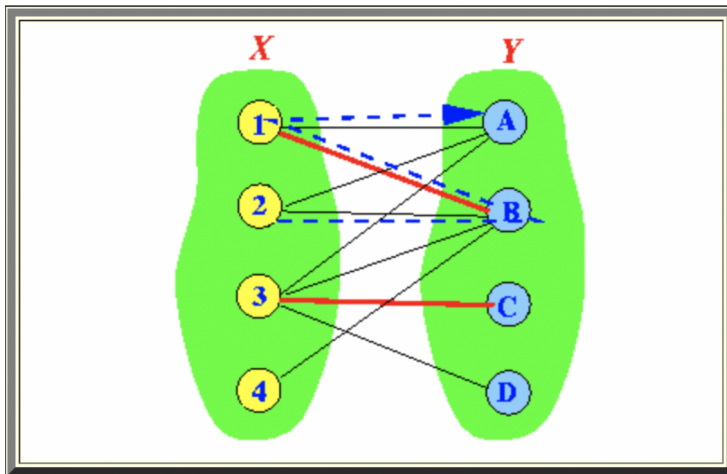
König's Theorem - Example

– Example on the Board –

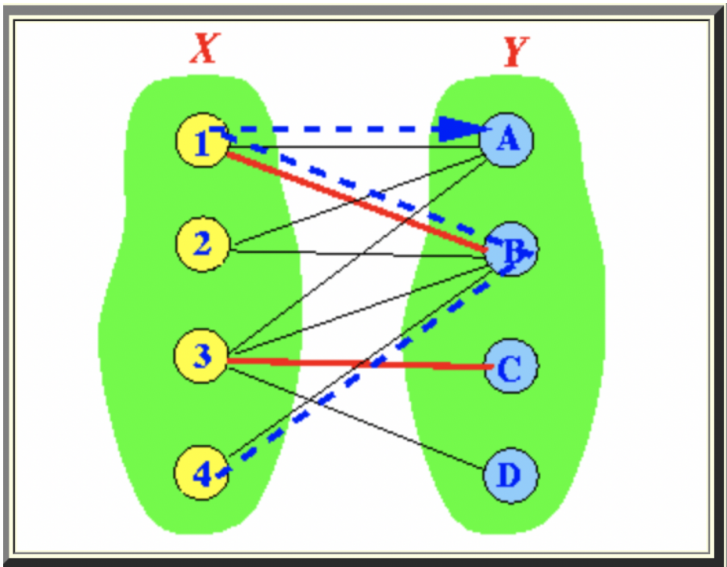
Alternating/Augmenting Path



Alternating/Augmenting Path



Alternating/Augmenting Path



Augmenting Path - Summary

- starts in a vertex element of X and ends in vertex element of Y
 - must have an odd-number of edges
 - will visit nodes in X and Y alternately
 - And it starts and ends in free/unmatched vertices
- To go forward, use an edge that is not part of the matching
- To go backward, use an edge that is part of the matching

Augmenting Path - example

– Example on the Board –

Breadth First Search - repetition

– Example on the Board –

Definition

A matching is a maximum matching if it contains no augmenting path.

Hungarian Method

Search augmenting paths in the graph until no augmenting path can be found

→ **Time complexity:** $O(|V|^3)$

Precise information can be found here:

<https://brilliant.org/wiki/hungarian-matching/>

Note: It is not an algorithm, so it does not specify a implementation

Maximum Flow Reduction

– Example on the Board –

Hopcraft-Carp Algorithm

Input: **A bipartite Graph** Initialize Matching

1. Repeat

- Build alternating level graph rooted at unmatched vertices using bfs
- Augment M via maximal set of vertex disjoint shortest-length paths
- until no augmenting paths exists

2. Return M

Time complexity: $O(|E| * \sqrt{|V|})$

Hopcraft-Carp Algorithm - bipartite Graph

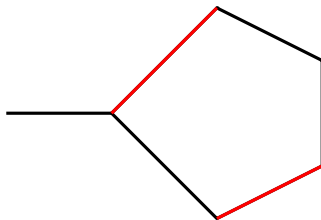
– Example on the Board –

General graphs

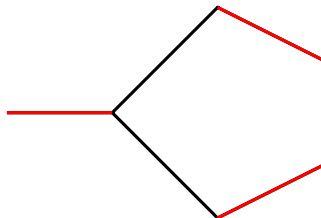
General graphs

Problem: odd-length cycles

Maximal: matching of size 2

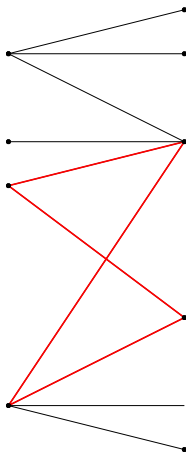


Optimum: matching of size 3



General graphs

Bipartite graphs can have cycles, but always only of even length:



Edmonds' Blossom algorithm (1961)

Blossom algorithm uses the idea of Berge's Theorem, that

Edmonds' Blossom algorithm (1961)

Blossom algorithm uses the idea of Berge's Theorem, that matching is a **maximum matching** iff there is **no augmenting path**.

Edmonds' Blossom algorithm (1961)

Blossom algorithm uses the idea of Berge's Theorem, that matching is a **maximum matching** iff there is **no augmenting path**.

Input: Graph \mathcal{G} , initial matching \mathcal{M} on \mathcal{G}

Output: maximum matching \mathcal{M}^* on \mathcal{G}

Edmonds' Blossom algorithm (1961)

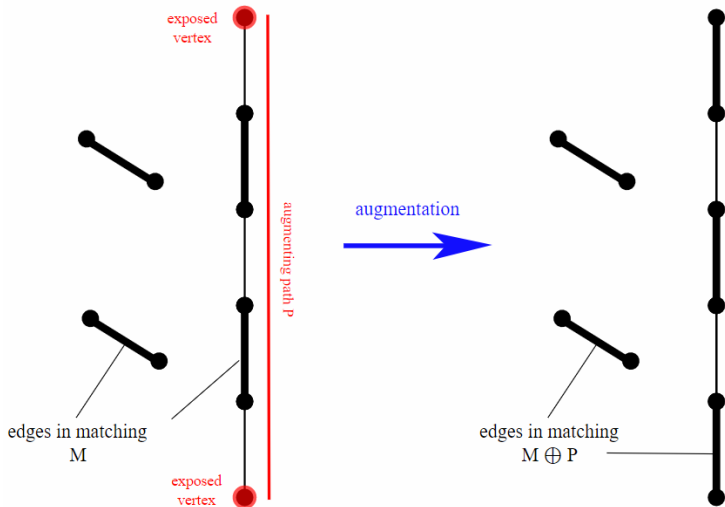
Blossom algorithm uses the idea of Berge's Theorem, that matching is a **maximum matching** iff there is **no augmenting path**.

Input: Graph \mathcal{G} , initial matching \mathcal{M} on \mathcal{G}

Output: maximum matching \mathcal{M}^* on \mathcal{G}

In other words, Blossom algorithm improves existing matching \mathcal{M} in \mathcal{G} as long as augmenting paths exist, then returns.

Edmonds' Blossom algorithm (1965)



Edmonds' Blossom algorithm (1965)

Problem: How to guarantee no augmenting paths in a graph?

Edmonds' Blossom algorithm (1965)

Problem: How to guarantee no augmenting paths in a graph?

Definition: Exposed vertex

Vertex v is exposed iff no edge of M is incident with v .

Edmonds' Blossom algorithm (1965)

Problem: How to guarantee no augmenting paths in a graph?

Definition: Exposed vertex

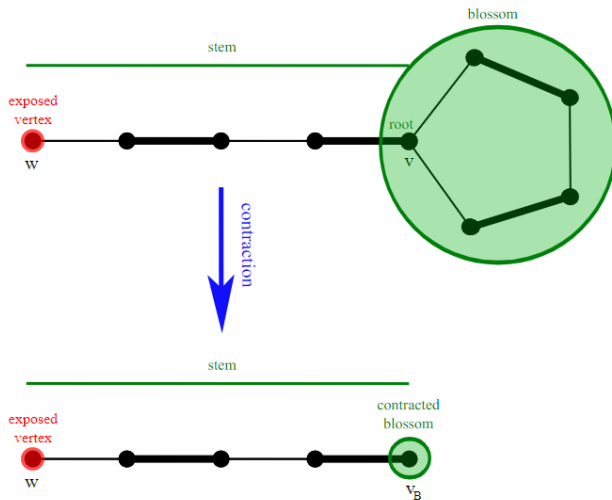
Vertex v is exposed iff no edge of M is incident with v .

Blossom algorithm examines all exposed vertices v and

- if an augmenting path is found, improve matching
- if a "blossom" (odd cycle*) is found, temporarily remove cycle and execute algorithm on a modified graph
- (delete vertex v from exposed vertices)

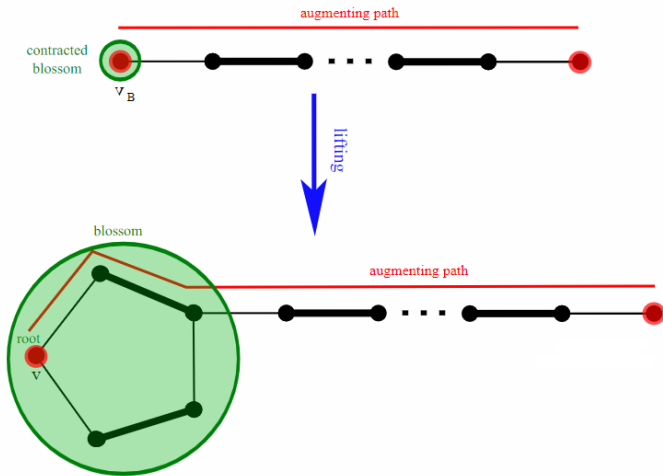
Edmonds' Blossom algorithm (1965)

Blossom contraction



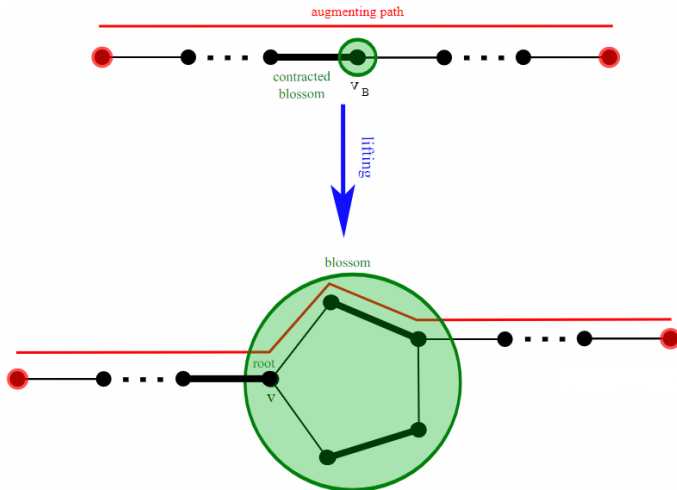
Edmonds' Blossom algorithm (1965)

Blossom lift



Edmonds' Blossom algorithm (1965)

Blossom lift (other variant)



Complexity: For general graphs a straightforward implementation of the maximum matching algorithm of Edmonds (1965) runs in $O(n^4)$ time (Papadimitriou and Steiglitz, 1982). More efficient general matching algorithms have been designed with the following running times:

- $O(n^3)$ - Gabow, 1976,
- $O(n * m)$ - Kameda and Munro, 1974,
- $O(n^{2.5})$ - Even and Kariv, 1975,
- $O(n^{1/2} * m)$ - Micali and Vazirani, 1980

Randomized Algorithms

Determinant and Permanent

Determinant and Permanent

A short tour in number theory,

Determinant and Permanent

A short tour in number theory,
Permutation

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

i	1	2	3	4	5
$\delta(i)$	3	2	4	1	5

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

\mathcal{S}_n - Set of all permutations of n elements.

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

\mathcal{S}_n - Set of all permutations of n elements.

For an $n \times n$ -matrix A , we define

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

\mathcal{S}_n - Set of all permutations of n elements.

For an $n \times n$ -matrix A , we define

Determinant of a matrix

$$\det(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

\mathcal{S}_n - Set of all permutations of n elements.

For an $n \times n$ -matrix A , we define

Determinant of a matrix

$$\det(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

Permanent of a matrix

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

S_n - Set of all permutations of n elements.

For an $n \times n$ -matrix A , we define

Determinant of a matrix

$$\det(A) = \sum_{\pi \in S} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

Permanent of a matrix

$$\text{perm}(A) = \sum_{\pi \in S} \prod_{i \in [n]} A_{i, \pi(i)}$$

Laplace expansion (on board)

Efficient Gaussian Elimination

$O(n^\omega), \omega = 2.373$

Matrix-Multiplication Exponent

Determinant and Permanent

A short tour in number theory,

Permutation - Bijection over n elements to themselves.

S_n - Set of all permutations of n elements.

For an $n \times n$ -matrix A , we define

Determinant of a matrix

$$\det(A) = \sum_{\pi \in S} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

Laplace expansion (on board)

Efficient Gaussian Elimination

$O(n^\omega)$, $\omega = 2.373$

Matrix-Multiplication Exponent

Permanent of a matrix

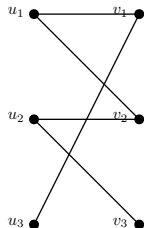
$$\text{perm}(A) = \sum_{\pi \in S} \prod_{i \in [n]} A_{i, \pi(i)}$$

NP-Hard problem.

Presumably no polynomial time algorithm to compute.

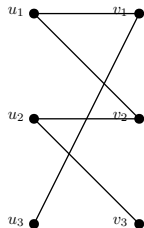
Randomized Perfect Matching

How to check if a bipartite graph admits a perfect matching?



Randomized Perfect Matching

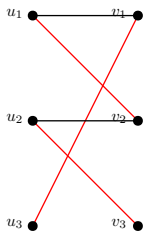
How to check if a bipartite graph admits a perfect matching?



$$A^G = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Randomized Perfect Matching

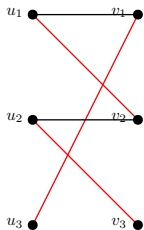
How to check if a bipartite graph admits a perfect matching?



$$A^G = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Randomized Perfect Matching

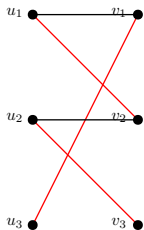
How to check if a bipartite graph admits a perfect matching?



$$A^G = \begin{bmatrix} 1 & \textcolor{red}{1} & 0 \\ 0 & 1 & \textcolor{red}{1} \\ \textcolor{red}{1} & 0 & 0 \end{bmatrix}$$

Randomized Perfect Matching

How to check if a bipartite graph admits a perfect matching?

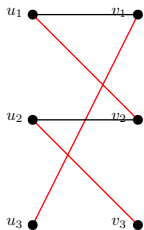


$$A^G = \begin{bmatrix} 1 & \textcolor{red}{1} & 0 \\ 0 & 1 & \textcolor{red}{1} \\ \textcolor{red}{1} & 0 & 0 \end{bmatrix}$$

u_i	1	2	3
$\pi(u_i)$	2	3	1

Randomized Perfect Matching

How to check if a bipartite graph admits a perfect matching?



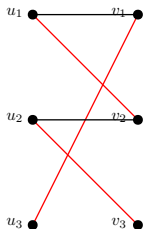
$$A^G = \begin{bmatrix} 1 & \textcolor{red}{1} & 0 \\ 0 & 1 & \textcolor{red}{1} \\ \textcolor{red}{1} & 0 & 0 \end{bmatrix}$$

u_i	1	2	3
$\pi(u_i)$	2	3	1

The graph admits a perfect matching \iff

Randomized Perfect Matching

How to check if a bipartite graph admits a perfect matching?



$$A^G = \begin{bmatrix} 1 & \textcolor{red}{1} & 0 \\ 0 & 1 & \textcolor{red}{1} \\ \textcolor{red}{1} & 0 & 0 \end{bmatrix}$$

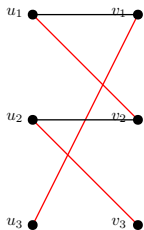
u_i	1	2	3
$\pi(u_i)$	2	3	1

The graph admits a perfect matching \iff

There is a permutation π , s.t. $\prod_{i \in [n]} A^G_{i, \pi(i)} = 1 \iff$

Randomized Perfect Matching

How to check if a bipartite graph admits a perfect matching?



$$A^G = \begin{bmatrix} 1 & \textcolor{red}{1} & 0 \\ 0 & 1 & \textcolor{red}{1} \\ \textcolor{red}{1} & 0 & 0 \end{bmatrix}$$

u_i	1	2	3
$\pi(u_i)$	2	3	1

The graph admits a perfect matching \iff

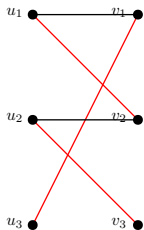
There is a permutation π , s.t. $\prod_{i \in [n]} A^G_{i, \pi(i)} = 1 \iff$

For \mathcal{S} the set of all permutations on n elements

$$\sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A^G_{i, \pi(i)} > 0$$

Randomized Perfect Matching

How to check if a bipartite graph admits a perfect matching?



$$A^G = \begin{bmatrix} 1 & \textcolor{red}{1} & 0 \\ 0 & 1 & \textcolor{red}{1} \\ \textcolor{red}{1} & 0 & 0 \end{bmatrix}$$

u_i	1	2	3
$\pi(u_i)$	2	3	1

The graph admits a perfect matching \iff

There is a permutation π , s.t. $\prod_{i \in [n]} A^G_{i, \pi(i)} = 1 \iff$

For \mathcal{S} the set of all permutations on n elements

$$\underbrace{\sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A^G_{i, \pi(i)}}_{\text{perm}(A^G)} > 0$$

Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

... is a known hard problem

Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

... is a known hard problem

$$\text{det}(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

... is a known hard problem

$$\det(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

We don't have to bound ourselves with ones in the matrix..

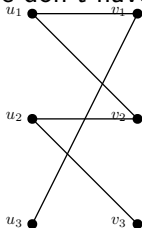
Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

... is a known hard problem

$$\det(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

We don't have to bound ourselves with ones in the matrix..



$$A'^G = \begin{bmatrix} x_{11} & x_{12} & 0 \\ 0 & x_{22} & x_{23} \\ x_{31} & 0 & 0 \end{bmatrix}$$

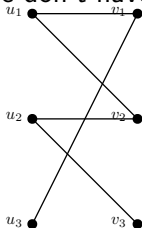
Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

... is a known hard problem

$$\det(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

We don't have to bound ourselves with ones in the matrix..



$$A'^G = \begin{bmatrix} x_{11} & x_{12} & 0 \\ 0 & x_{22} & x_{23} \\ x_{31} & 0 & 0 \end{bmatrix}$$

$\det(A'^G)$ is a polynomial of degree n .

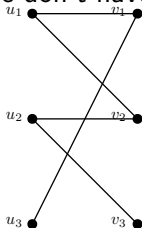
Permanent and Determinant

$$\text{perm}(A) = \sum_{\pi \in \mathcal{S}} \prod_{i \in [n]} A_{i, \pi(i)}$$

... is a known hard problem

$$\det(A) = \sum_{\pi \in \mathcal{S}} \text{sign}(\pi) \prod_{i \in [n]} A_{i, \pi(i)}$$

We don't have to bound ourselves with ones in the matrix..



$$A'^G = \begin{bmatrix} x_{11} & x_{12} & 0 \\ 0 & x_{22} & x_{23} \\ x_{31} & 0 & 0 \end{bmatrix}$$

$\det(A'^G)$ is a polynomial of degree n .

The graph admits a perfect matching \iff the polynomial $\det(A'^G)$ is not identical zero.

Schwartz-Zippel lemma

Schwartz-Zippel lemma

Let ρ be a non-zero polynomial of n variables and degree d over a field \mathbb{F} .
Let $\mathcal{S} \subseteq \mathbb{F}^n$, then for $x_0 \underset{\text{u.a.r}}{\in} \mathcal{S}$

$$\Pr[\rho(x_0) = 0] \leq \frac{d}{|\mathcal{S}|}$$

Schwartz-Zippel lemma

Schwartz-Zippel lemma

Let ρ be a non-zero polynomial of n variables and degree d over a field \mathbb{F} .
Let $\mathcal{S} \subseteq \mathbb{F}^n$, then for $x_0 \underset{\text{u.a.r}}{\in} \mathcal{S}$

$$\Pr[\rho(x_0) = 0] \leq \frac{d}{|\mathcal{S}|}$$

Choosing $|\mathcal{S}| = 2d$, we get a method that tells with probability at least $1/2$, if the polynomial is identical zero.

Schwartz-Zippel lemma

Schwartz-Zippel lemma

Let ρ be a non-zero polynomial of n variables and degree d over a field \mathbb{F} .
Let $\mathcal{S} \subseteq \mathbb{F}^n$, then for $x_0 \in \mathcal{S}$
u.a.r

$$\Pr[\rho(x_0) = 0] \leq \frac{d}{|\mathcal{S}|}$$

Choosing $|\mathcal{S}| = 2d$, we get a method that tells with probability at least $1/2$, if the polynomial is identical zero.

Since one non-zero answer is enough to know the polynomial is a non-zero, we can repeat the operation a couple of times magnifying the probability.