4. Inclusion-Exclusion

COMP6741: Parameterized and Exact Computation

Serge Gaspers\textsuperscript{1,2}

\textsuperscript{1}School of Computer Science and Engineering, UNSW Australia
\textsuperscript{2}Data61, Decision Sciences Group, CSIRO

Semester 2, 2016
Outline

1. The Principle of Inclusion-Exclusion
2. Counting Hamiltonian Cycles
3. Coloring
4. Counting Set Covers
5. Counting Set Partitions
6. Further Reading
Outline

1. The Principle of Inclusion-Exclusion
2. Counting Hamiltonian Cycles
3. Coloring
4. Counting Set Covers
5. Counting Set Partitions
6. Further Reading
... for 3 sets

$$|A \cup B \cup C| =$$
... for 3 sets

$$|A \cup B \cup C| = |A| + |B| + |C|$$
... for 3 sets

\[ |A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| \]
... for 3 sets

\[ |A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C| \]
... for 3 sets

\[ |A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C| \]

\[ |A \cup B \cup C| = \sum_{X \subseteq \{A,B,C\}} (-1)^{|X|+1} \cdot \left| \bigcap X \right| \]
\[ |A \cap B \cap C| = \]

\[ \sum_{X \subseteq \{A,B,C\}} (-1)^{|X|} |\bigcap X| \]
\[ |A \cap B \cap C| = |U| \]
\[ |A \cap B \cap C| = |U| - |\overline{A}| - |\overline{B}| - |\overline{C}| \]
... intersection version

\[ |A \cap B \cap C| = |U| - |\overline{A}| - |\overline{B}| - |\overline{C}| + |\overline{A} \cap \overline{B}| + |\overline{A} \cap \overline{C}| + |\overline{B} \cap \overline{C}| \]
... intersection version

\[ |A \cap B \cap C| = |U| - |\overline{A}| - |\overline{B}| - |\overline{C}| + |A \cap \overline{B}| + |A \cap C| + |B \cap C| - |A \cap B \cap C| \]
\[ |A \cap B \cap C| = |U| - |\overline{A}| - |\overline{B}| - |\overline{C}| + |\overline{A} \cap \overline{B}| + |\overline{A} \cap \overline{C}| + |\overline{B} \cap \overline{C}| - |\overline{A} \cap \overline{B} \cap \overline{C}| \]

\[ |A \cap B \cap C| = \sum_{X \subseteq \{A, B, C\}} (-1)^{|X|} \cdot \left| \bigcap X \right| \]
Inclusion-Exclusion Principle – intersection version

Theorem 1 (IE-theorem – intersection version)

Let \( U = A_0 \) be a finite set, and let \( A_1, \ldots, A_k \subseteq U \).

\[
\left| \bigcap_{i \in \{1, \ldots, k\}} A_i \right| = \sum_{J \subseteq \{1, \ldots, k\}} (-1)^{|J|} \left| \bigcap_{i \in J} \overline{A_i} \right|
\]

where \( \overline{A_i} = U \setminus A_i \) and \( \bigcap_{i \in \emptyset} = U \).
Inclusion-Exclusion Principle – intersection version

**Theorem 1 (IE-theorem – intersection version)**

Let $U = A_0$ be a finite set, and let $A_1, \ldots, A_k \subseteq U$.

$$\left| \bigcap_{i \in \{1, \ldots, k\}} A_i \right| = \sum_{J \subseteq \{1, \ldots, k\}} (-1)^{|J|} \left| \bigcap_{i \in J} \overline{A_i} \right|,$$

where $\overline{A_i} = U \setminus A_i$ and $\bigcap_{i \in \emptyset} = U$.

**Proof sketch.**

- An element $e \in \bigcap_{i \in \{1, \ldots, k\}} A_i$ is counted on the right only for $J = \emptyset$.
- An element $e \notin \bigcap_{i \in \{1, \ldots, k\}} A_i$ is counted on the right for all $J \subseteq I$, where $I$ is the set of indices $i$ such that $e \notin A_i$.
  - counted negatively for each odd-sized $J \subseteq I$, and positively for each even-sized $J \subseteq I$
  - a non-empty set has as many even-sized subsets as odd-sized subsets
Outline

1. The Principle of Inclusion-Exclusion
2. Counting Hamiltonian Cycles
3. Coloring
4. Counting Set Covers
5. Counting Set Partitions
6. Further Reading
A walk of length $k$ in a graph $G = (V, E)$ (short, a $k$-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i \in \{0, \ldots, k - 1\}$.
Walks and cycles

- A walk of length $k$ in a graph $G = (V, E)$ (short, a $k$-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i \in \{0, \ldots, k - 1\}$.
- A walk $(v_0, v_1, \ldots, v_k)$ is closed if $v_0 = v_k$. 

Diagram:

- Vertices: $a, b, c, d, e$
- Edges: $a, d, c, b, d, e, c, a$
Walks and cycles

- A **walk** of length $k$ in a graph $G = (V, E)$ (short, a $k$-**walk**) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i \in \{0, \ldots, k - 1\}$.
- A walk $(v_0, v_1, \ldots, v_k)$ is **closed** if $v_0 = v_k$.
- A **cycle** is a 2-regular subgraph of $G$.

![Diagram of a cycle](image-url)
Walks and cycles

- A **walk** of length $k$ in a graph $G = (V, E)$ (short, a $k$-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i \in \{0, \ldots, k-1\}$.
- A walk $(v_0, v_1, \ldots, v_k)$ is **closed** if $v_0 = v_k$.
- A **cycle** is a 2-regular subgraph of $G$.
- A **Hamiltonian cycle** of $G$ is a cycle of length $n = |V|$.

(a, d, e, c, b, a)
Input: A graph $G = (V, E)$
Output: The number of Hamiltonian cycles of $G$

This graph has 2 Hamiltonian cycles.
IE for \( \# \text{Hamiltonian-Cycles} \)

- **\( U \):** the set of closed \( n \)-walks starting at vertex 1
- **\( A_v \subseteq U \):** walks in \( U \) that visit vertex \( v \in V \)
- \( \Rightarrow \) number of Hamiltonian cycles is \( |\cap_{v \in V} A_v| \)

To use the IE-theorem, we need to compute \( |\cap_{v \in S} A_v| \), the number of walks from \( U \) in the graph \( G - S \).
A simpler problem

\#CLOSED $n$-WALKS

Input: An integer $n$, and a graph $G = (V, E)$ on $\leq n$ vertices
Output: The number of closed $n$-walks in $G$ starting at vertex 1
A simpler problem

### Closed $n$-Walks

**Input:** An integer $n$, and a graph $G = (V, E)$ on $\leq n$ vertices  
**Output:** The number of closed $n$-walks in $G$ starting at vertex $1$

#### Dynamic programming

- $T[d, v]$: number of $d$-walks starting at vertex $1$ and ending at vertex $v$
- Base cases: $T[0, 1] = 1$ and $T[0, v] = 0$ for all $v \in V \setminus \{1\}$
- DP recurrence: $T[d, v] = \sum_{uv \in E} T[d - 1, u]$
- Table $T$ is filled by increasing $d$
- Return $T[n, 1]$ in $O(n^3)$ time
Wrapping up

- Recall:
  - $U$: set of closed $n$-walks starting at vertex 1
  - $A_v$: set of closed $n$-walks that start at vertex 1 and visit vertex $v$

- By the IE-theorem, the number of Hamiltonian cycles is

$$\left| \bigcap_{v \in V} A_v \right| = \sum_{S \subseteq V} (-1)^{|S|} \left| \bigcap_{v \in S} \overline{A_v} \right|$$

Theorem 2

Hamiltonian-Cycles can be solved in $O(2^n n^3)$ time and polynomial space, where $n = |V|$. 

S. Gaspers (UNSW)
Wrapping up

- Recall:
  - \( U \): set of closed \( n \)-walks starting at vertex 1
  - \( A_v \): set of closed \( n \)-walks that start at vertex 1 and visit vertex \( v \)
- By the IE-theorem, the number of Hamiltonian cycles is

\[
\left| \bigcap_{v \in V} A_v \right| = \sum_{S \subseteq V} (\text{size of } S) \times \left| \bigcap_{v \in S} A_v \right|
\]

- We have seen that \( \left| \bigcap_{v \in S} \overline{A_v} \right| \) can be computed in \( O(n^3) \) time.
- So, \( \sum_{S \subseteq V} (\text{size of } S) \times \left| \bigcap_{v \in S} \overline{A_v} \right| \) can be evaluated in \( O(2^n n^3) \) time
Wrapping up

- Recall:
  - $U$: set of closed $n$-walks starting at vertex 1
  - $A_v$: set of closed $n$-walks that start at vertex 1 and visit vertex $v$

- By the IE-theorem, the number of Hamiltonian cycles is

$$\left| \bigcap_{v \in V} A_v \right| = \sum_{S \subseteq V} (-1)^{|S|} \left| \bigcap_{v \in S} \overline{A_v} \right|$$

- We have seen that $\left| \bigcap_{v \in S} \overline{A_v} \right|$ can be computed in $O(n^3)$ time.
- So, $\sum_{S \subseteq V} (-1)^{|S|} \left| \bigcap_{v \in S} \overline{A_v} \right|$ can be evaluated in $O(2^n n^3)$ time

### Theorem 2

$\#\text{HAMILTONIAN-CYCLES}$ can be solved in $O(2^n n^3)$ time and polynomial space, where $n = |V|$.
Outline

1. The Principle of Inclusion-Exclusion
2. Counting Hamiltonian Cycles
3. Coloring
4. Counting Set Covers
5. Counting Set Partitions
6. Further Reading
A $k$-coloring of a graph $G = (V, E)$ is a function $f : V \rightarrow \{1, 2, ..., k\}$ assigning colors to $V$ such that no two adjacent vertices receive the same color.

**Coloring**

**Input:** Graph $G$, integer $k$

**Question:** Does $G$ have a $k$-coloring?
A \textit{k-coloring} of a graph \( G = (V, E) \) is a function \( f : V \rightarrow \{1, 2, \ldots, k\} \) assigning colors to \( V \) such that no two adjacent vertices receive the same color.

**COLORING**

Input: Graph \( G \), integer \( k \)

Question: Does \( G \) have a \( k \)-coloring?

**Exercise**

- Suppose \( A \) is an algorithm solving \textsc{Coloring} in \( O(f(n)) \) time, \( n = |V| \), where \( f \) is non-decreasing.

- Design a \( O^*(f(n)) \) time algorithm \( B \), which, for an input graph \( G \), finds a coloring of \( G \) with a minimum number of colors.
Observation: partitioning vs. covering

\[ G = (V, E) \text{ has a } k\text{-coloring} \]
\[ \iff \]
\[ G \text{ has independent sets } I_1, \ldots, I_k \text{ such that } \bigcup_{i=1}^{k} I_i = V. \]
Observation: partitioning vs. covering

\[ G = (V, E) \text{ has a } k\text{-coloring} \iff \]

\[ G \text{ has independent sets } I_1, \ldots, I_k \text{ such that } \bigcup_{i=1}^{k} I_i = V. \]

- \( U \): set of tuples \((I_1, \ldots, I_k)\), where each \( I_i, i \in \{1, \ldots, k\} \), is an independent set.
Observation: partitioning vs. covering

\( G = (V, E) \) has a \( k \)-coloring \( \iff \)

\( G \) has independent sets \( I_1, \ldots, I_k \) such that \( \bigcup_{i=1}^{k} I_i = V \).

- \( U \): set of tuples \((I_1, \ldots, I_k)\), where each \( I_i, i \in \{1, \ldots, k\} \), is an independent set
- \( A_v = \{(I_1, \ldots, I_k) \in U : v \in \bigcup_{i \in \{1, \ldots, k\}} I_i\} \)
Observation: partitioning vs. covering

$G = (V, E)$ has a $k$-coloring

$\iff$

$G$ has independent sets $I_1, \ldots, I_k$ such that $\bigcup_{i=1}^{k} I_i = V$.

- $U$: set of tuples $(I_1, \ldots, I_k)$, where each $I_i, i \in \{1, \ldots, k\}$, is an independent set
- $A_v = \{(I_1, \ldots, I_k) \in U : v \in \bigcup_{i \in \{1, \ldots, k\}} I_i\}$
- Note: $|\bigcap_{v \in V} A_v| \neq 0 \iff G$ has a $k$-coloring
IE formulation

Observation: partitioning vs. covering

\[ G = (V, E) \text{ has a } k\text{-coloring} \iff \]
\[ G \text{ has independent sets } I_1, \ldots, I_k \text{ such that } \bigcup_{i=1}^k I_i = V. \]

- **U**: set of tuples \((I_1, \ldots, I_k)\), where each \(I_i, i \in \{1, \ldots, k\}\), is an independent set
- \(A_v = \{(I_1, \ldots, I_k) \in U : v \in \bigcup_{i \in \{1, \ldots, k\}} I_i\}\)
- **Note**: \(\bigcap_{v \in V} A_v \neq \emptyset \iff G\) has a \(k\)-coloring
- To use the IE-theorem, we need to compute

\[
\left| \bigcap_{v \in S} \overline{A_v} \right| = |\{(I_1, \ldots, I_k) \in U : I_1, \ldots, I_k \subseteq V \setminus S\}| 
\]
Observation: partitioning vs. covering

\[ G = (V, E) \text{ has a } k\text{-coloring} \iff G \text{ has independent sets } I_1, \ldots, I_k \text{ such that } \bigcup_{i=1}^{k} I_i = V. \]

- **U**: set of tuples \((I_1, \ldots, I_k)\), where each \(I_i, i \in \{1, \ldots, k\}\), is an independent set
- \(A_v = \{(I_1, \ldots, I_k) \in U : v \in \bigcup_{i \in \{1, \ldots, k\}} I_i\}\)
- Note: \(\left| \bigcap_{v \in V} A_v \right| \neq 0 \iff G \text{ has a } k\text{-coloring} \)
- To use the IE-theorem, we need to compute

\[
\left| \bigcap_{v \in S} \overline{A_v} \right| = \left| \{(I_1, \ldots, I_k) \in U : I_1, \ldots, I_k \subseteq V \setminus S\} \right|
\]

\[
= s(V \setminus S)^k,
\]

where \(s(X)\) is the number of independent sets in \(G[X]\)
A simpler problem

#IS of Induced Subgraphs

Input: A graph $G = (V, E)$
Output: $s(X)$, the number of independent sets of $G[X]$, for each $X \subseteq V$
A simpler problem

\#IS of Induced Subgraphs

Input: A graph $G = (V, E)$

Output: $s(X)$, the number of independent sets of $G[X]$, for each $X \subseteq V$

Dynamic Programming

- $s(X)$: the number of independent sets of $G[X]$
- Base case: $s(\emptyset) = 1$
- DP recurrence: $s(X) = s(X \setminus N_G[v]) + s(X \setminus \{v\})$, where $v \in X$
- Table $s$ filled by increasing cardinalities of $X$
- Output $s(X)$ for each $X \subseteq V$ in time $O^*(2^n)$
Wrapping up

Now, evaluate

$$\left| \bigcap_{v \in V} A_v \right| = \sum_{S \subseteq V} (-1)^{|S|} \left| \bigcap_{v \in S} A_v \right| = \sum_{S \subseteq V} (-1)^{|S|} s(V \setminus S)^k,$$

in $O^*(2^n)$ time.

$G$ has a $k$-coloring iff $\left| \bigcap_{v \in V} A_v \right| > 0$.

**Theorem 3** ([Bjørklund & Husfeldt ’06], [Koivisto ’06])

**Coloring** can be solved in $O^*(2^n)$ time (and space).

**Corollary 4**

For a given graph $G$, a coloring with a minimum number of colors can be found in $O^*(2^n)$ time (and space).
Using an algorithm by [Gaspers, Lee, 2016], counting all independent sets in a graph on \( n \) vertices in \( O(1.2355^n) \) time, we obtain a polynomial-space algorithm for Coloring with running time

\[
\sum_{S \subseteq V} O(1.2355^{n-|S|}) = \sum_{s=0}^{n} \binom{n}{s} O(1.2377^{n-s}) = O(2.2355^n).
\]

Here, we used the Binomial Theorem:

\[
(x + y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k.
\]

**Theorem 5**

Coloring can be solved in \( O(2.2355^n) \) time and polynomial space.
Outline

1. The Principle of Inclusion-Exclusion
2. Counting Hamiltonian Cycles
3. Coloring
4. Counting Set Covers
5. Counting Set Partitions
6. Further Reading
#Set Covers

**Input:** A finite ground set $V$ of elements, a collection $H$ of subsets of $V$, and an integer $k$

**Output:** The number of ways to choose a $k$-tuple of sets $(S_1, \ldots, S_k)$ with $S_i \in H$, $i \in \{1, \ldots, k\}$, such that $\bigcup_{i=1}^{k} S_i = V$.

This instance has $1 \cdot 3! = 6$ covers with 3 sets and $3 \cdot 4! = 72$ covers with 4 sets.

We consider, more generally, that $H$ is given only implicitly, but can be enumerated in $O^*(2^n)$ time and space.
Algorithm for Counting Set Covers

- $U$: set of $k$-tuples $(S_1, \ldots, S_k)$, where $S_i \in H$, $i \in \{1, \ldots, k\}$,
- $A_v = \{(S_1, \ldots, S_k) \in U : v \in \bigcup_{i \in \{1, \ldots, k\}} S_i\}$,
- the number of covers with $k$ sets is

$$\left|\bigcap_{v \in V} A_v\right| = \sum_{S \subseteq V} (-1)^{|S|} \left|\bigcap_{v \in S} \overline{A_v}\right|$$

$$= \sum_{S \subseteq V} (-1)^{|S|} s(V \setminus S)^k,$$

where $s(X)$ is the number of sets in $H$ that are subsets of $X$. 
Compute $s(X)$

For each $X \subseteq V$, compute $s(X)$, the number of sets in $H$ that are subsets of $X$.

**Dynamic Programming**

- Arbitrarily order $V = \{v_1, v_2, \ldots, v_n\}$
- $g[X, i] = |\{S \in H : (X \cap \{v_i, \ldots, v_n\}) \subseteq S \subseteq X\}|$
- Note: $g[X, n + 1] = s(X)$
- Base case: $g[X, 1] = \begin{cases} 1 & \text{if } X \in H \\ 0 & \text{otherwise.} \end{cases}$
- DP recurrence: $g[X, i] = \begin{cases} g[X, i - 1] & \text{if } v_{i-1} \notin X \\ g[X \setminus \{v_{i-1}\}, i - 1] + g[X, i - 1] & \text{otherwise.} \end{cases}$
- Table filled by increasing $i$
Compute $s(X)$

For each $X \subseteq V$, compute $s(X)$, the number of sets in $H$ that are subsets of $X$.

Dynamic Programming

- Arbitrarily order $V = \{v_1, v_2, \ldots, v_n\}$
- $g[X, i] = |\{S \in H : (X \cap \{v_i, \ldots, v_n\}) \subseteq S \subseteq X\}|$
- Note: $g[X, n + 1] = s(X)$
- Base case: $g[X, 1] = \begin{cases} 1 & \text{if } X \in H \\ 0 & \text{otherwise.} \end{cases}$
- DP recurrence: $g[X, i] = \begin{cases} g[X, i - 1] & \text{if } v_{i-1} \notin X \\ g[X \setminus \{v_{i-1}\}, i - 1] + g[X, i - 1] & \text{otherwise.} \end{cases}$
- Table filled by increasing $i$

Theorem 6

#Set Covers can be solved in $O^*(2^n)$ time and space, where $n = |V|$. 
1. The Principle of Inclusion-Exclusion
2. Counting Hamiltonian Cycles
3. Coloring
4. Counting Set Covers
5. Counting Set Partitions
6. Further Reading
# Ordered Set Partitions

Input: A finite ground set $V$ of elements, a collection $H$ of subsets of $V$, and an integer $k$.

Output: The number of ways to choose a $k$-tuple of pairwise disjoint sets $(S_1, \ldots, S_k)$ with $S_i \in H$, $i \in \{1, \ldots, k\}$, such that $\bigcup_{i=1}^{k} S_i = V$. (Now, $S_i \cap S_j = \emptyset$, if $i \neq j$.)

This instance has $1 \cdot 3! = 6$ ordered partitions with 3 sets.
Lemma 7

The number of ordered $k$-partitions of a set system $(V, H)$ is

$$\sum_{S \subseteq V} (-1)^{|S|} a_k(V \setminus S),$$

where $a_k(X)$ denotes the number of $k$-tuples of sets $S_1, \ldots, S_k \subseteq X$ with $\sum_{i=1}^{k} |S_i| = |V|$. 
Lemma 7

The number of ordered \( k \)-partitions of a set system \((V, H)\) is

\[
\sum_{S \subseteq V} (-1)^{|S|} a_k(V \setminus S),
\]

where \( a_k(X) \) denotes the number of \( k \)-tuples of sets \( S_1, \ldots, S_k \subseteq X \) with \( \sum_{i=1}^k |S_i| = |V| \).

Proof (Sketch).

- \( U \): set of tuples \((S_1, \ldots, S_k)\), where \( S_i \in H, i \in \{1, \ldots, k\} \), and \( \sum_{i=1}^k |S_i| = |V| \).
- \( A_v = \{(S_1, \ldots, S_k) \in U : v \in \bigcup_{i \in \{1, \ldots, k\}} S_i\} \).
- the number of ordered partitions with \( k \) sets is

\[
\left| \bigcap_{v \in V} A_v \right| = \sum_{S \subseteq V} (-1)^{|S|} \left| \bigcap_{v \in S} A_v \right| = \sum_{S \subseteq V} (-1)^{|S|} a_k(V \setminus S).
\]
For each $X \subseteq V$, we need to compute $a_k(X)$, the number of $k$-tuples of sets $S_1, \ldots, S_k \subseteq X$ with $\sum_{i=1}^{k} |S_i| = |V|$.
For each $X \subseteq V$, we need to compute $a_k(X)$, the number of $k$-tuples of sets $S_1, \ldots, S_k \subseteq X$ with $\sum_{i=1}^{k} |S_i| = |V|$.

**Dynamic Programming**

(1) Compute $s[X, i] = |\{Y \in H : Y \subseteq X \text{ and } |Y| = i\}|$ for each $X \subseteq V$ and each $i \in \{0, \ldots, n\}$:

- The entries $s[\cdot, i]$ are computed the same ways as $s[\cdot]$ in the previous section, but keep only the sets in $H$ of size $i$. 

For each $X \subseteq V$, we need to compute $a_k(X)$, the number of $k$-tuples of sets $S_1, \ldots, S_k \subseteq X$ with $\sum_{i=1}^{k} |S_i| = |V|$.

Dynamic Programming

1. Compute $s[X, i] = |\{Y \in H : Y \subseteq X \text{ and } |Y| = i\}|$ for each $X \subseteq V$ and each $i \in \{0, \ldots, n\}$:
   - The entries $s[\cdot, i]$ are computed the same ways as $s[\cdot]$ in the previous section, but keep only the sets in $H$ of size $i$.

2. $A[\ell, m, X]$: number of tuples $(S_1, \ldots, S_\ell)$ with $S_i \in H$, $S_i \subseteq X$, and $\sum_{i=1}^{\ell} |S_i| = m$.
   - Base case: $A[1, m, X] = s[X, m]$
   - DP recurrence: $A[\ell, m, X] = \sum_{i=1}^{m-1} s[X, i] \cdot A[\ell - 1, m - i, X]$
   - Table filled by increasing $\ell$
   - Note: $a_k(X) = A[k, |V|, X]$
Algorithm for Counting Set Partitions

Theorem 8

\#Ordered Set Partitions can be solved in $O^*(2^n)$ time and space.
Algorithm for Counting Set Partitions

**Theorem 8**

#Ordered Set Partitions can be solved in $O^*(2^n)$ time and space.

**Corollary 9**

There is an algorithm computing the number of $k$-colorings of an input graph on $n$ vertices in $O^*(2^n)$ time and space.
Theorem 10

The number of covers with $k$ sets and the number of ordered partitions with $k$ sets of a set system $(V, H)$ can be computed in polynomial space and
Theorem 10

The number of covers with \( k \) sets and the number of ordered partitions with \( k \) sets of a set system \((V, H)\) can be computed in polynomial space and \( O^*(2^n |H|) \) time, assuming that \( H \) can be enumerated in \( O^*(|H|) \) time and polynomial space.
Theorem 10

The number of covers with \( k \) sets and the number of ordered partitions with \( k \) sets of a set system \((V, H)\) can be computed in polynomial space and

1. \( O^*(2^n|H|) \) time, assuming that \( H \) can be enumerated in \( O^*(|H|) \) time and polynomial space

2. \( O^*(3^n) \) time, assuming membership in \( H \) can be decided in polynomial time, and
Theorem 10

The number of covers with \( k \) sets and the number of ordered partitions with \( k \) sets of a set system \((V, H)\) can be computed in polynomial space and

1. \( O^*(2^n|H|) \) time, assuming that \( H \) can be enumerated in \( O^*(|H|) \) time and polynomial space

2. \( O^*(3^n) \) time, assuming membership in \( H \) can be decided in polynomial time, and

3. \( \sum_{j=0}^{n} \binom{n}{j} T_H(j) \) time, assuming there is a \( T_H(j) \) time and polynomial space algorithm to count for any \( W \subseteq V \) with \( |W| = j \) the number of sets \( S \in H \) satisfying \( S \cap W = \emptyset \).
A graph $G = (V, E)$ is bipartite if $V$ can be partitioned into two independent sets. A matching in a graph $G = (V, E)$ is a set of edges $M \subseteq E$ such that no two edges of $M$ have an end-point in common. The matching $M$ in $G$ is perfect if every vertex of $G$ is contained in an edge of $M$.

<table>
<thead>
<tr>
<th># Bipartite Perfect Matchings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: Bipartite graph $G = (V, E)$</td>
</tr>
<tr>
<td>Output: The number of perfect matchings in $G$.</td>
</tr>
</tbody>
</table>

1. Design an algorithm with running time $O^* \left( \left( \frac{n}{2} \right)! \right)$, where $n = |V|$.
2. Design a polynomial-space $O^*(2^{n/2})$-time inclusion-exclusion algorithm.
Reading

- Chapter 4, *Inclusion-Exclusion* in


**Advanced Reading**

- Chapter 7, *Subset Convolution* in