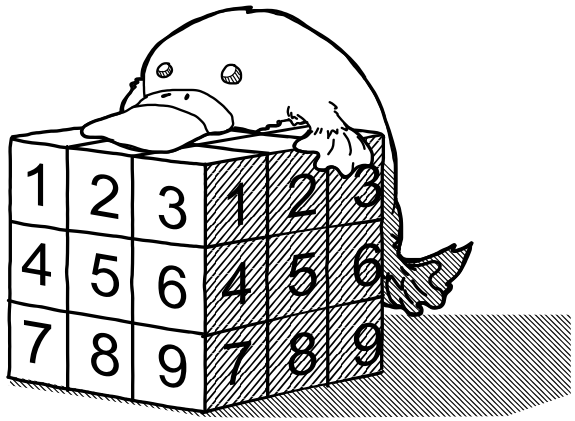
Algebraic techniques in parameterized algorithms

Łukasz Kowalik

University of Warsaw

FPT School, Będlewo, August 2014

Overview



- Inclusion-exclusion principle (Tuesday)
- Polynomials over finite fields of characteristic two (Thursday)
- Group algebras (Friday)

Algebraic techniques in parameterized algorithms, Part I: Inclusion-Exclusion

Łukasz Kowalik

University of Warsaw

FPT School, Będlewo, August 2014

Inclusion-Exclusion Principle

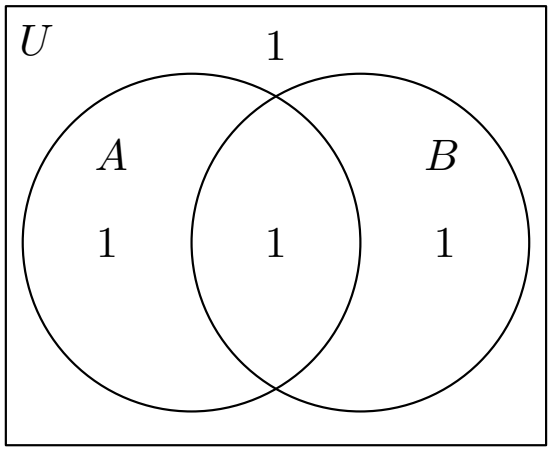
Theorem (Inclusion-Exclusion Principle, intersection version)

Let $A_1, \ldots, A_n \subseteq U$, where U is a finite set. Then:

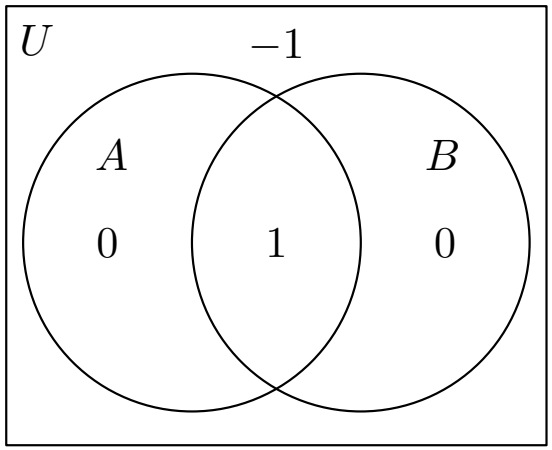
$$\left|\bigcap_{i\in\{1,\ldots,n\}} A_i\right| = \sum_{X\subseteq\{1,\ldots,n\}} (-1)^{|X|}\left|\bigcap_{i\in X} \overline{A_i}\right|$$

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

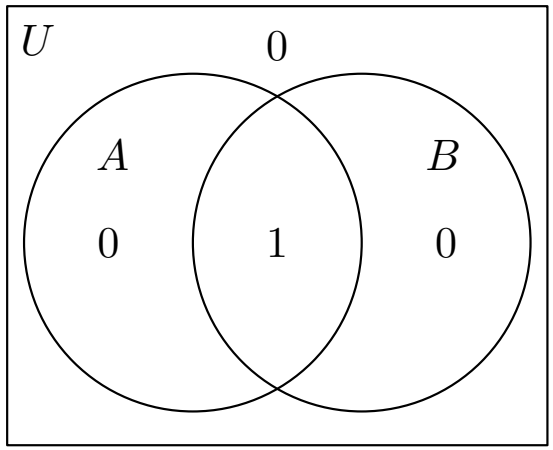
Example. $|A \cap B| = |U| - |\overline{A}| - |\overline{B}| + |\overline{A} \cap \overline{B}|$



$|U|$



$|U| - |\overline{A}| - |\overline{B}|$



$|U| - |\overline{A}| - |\overline{B}| + |\overline{A} \cap \overline{B}|$

Inclusion-Exclusion Principle, intersection version

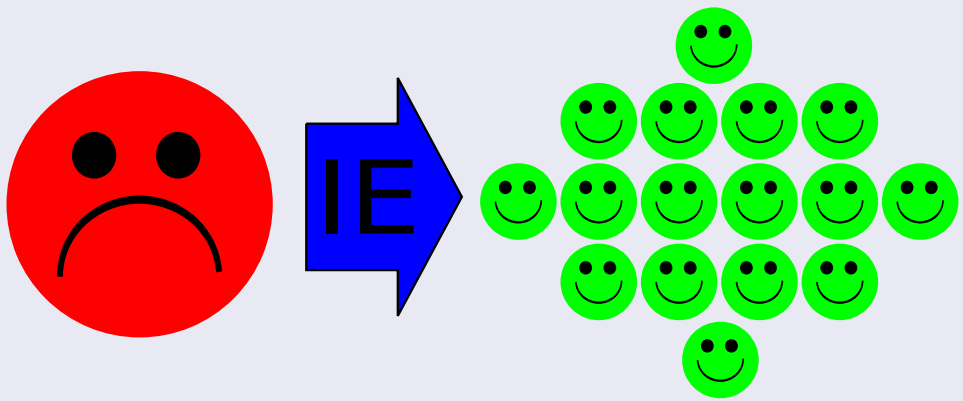
Theorem (Inclusion-Exclusion Principle, intersection version)

Let $A_1, \ldots, A_n \subseteq U$, where U is a finite set. *($\{A_i\}_{i=1}^n$ = "requirements".)*
Denote $\overline{A_i} = U - A_i$ and $\bigcap_{i \in \emptyset} \overline{A_i} = U$.
Then:

$$\left| \bigcap_{i \in \{1,\ldots,n\}} A_i \right| = \sum_{X \subseteq \{1,\ldots,n\}} (-1)^{|X|} \underbrace{\left| \bigcap_{i \in X} \overline{A_i} \right|}_{\text{"simplified problem"}}$$

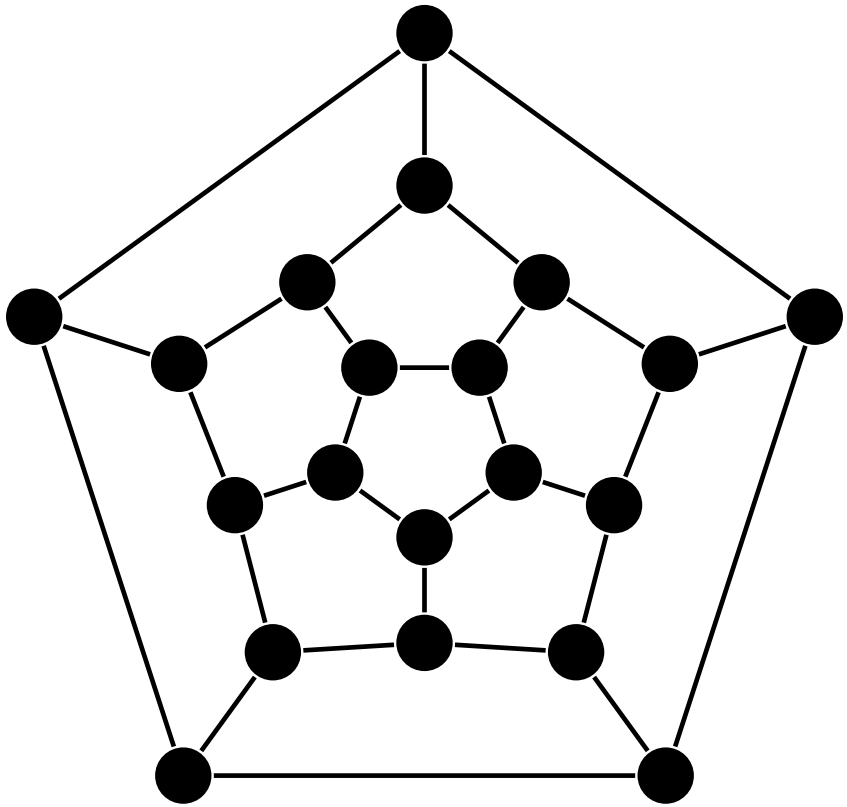
A common algorithmic application

Reduce a hard task to 2^n "simplified problems" (solvable in poly-time).



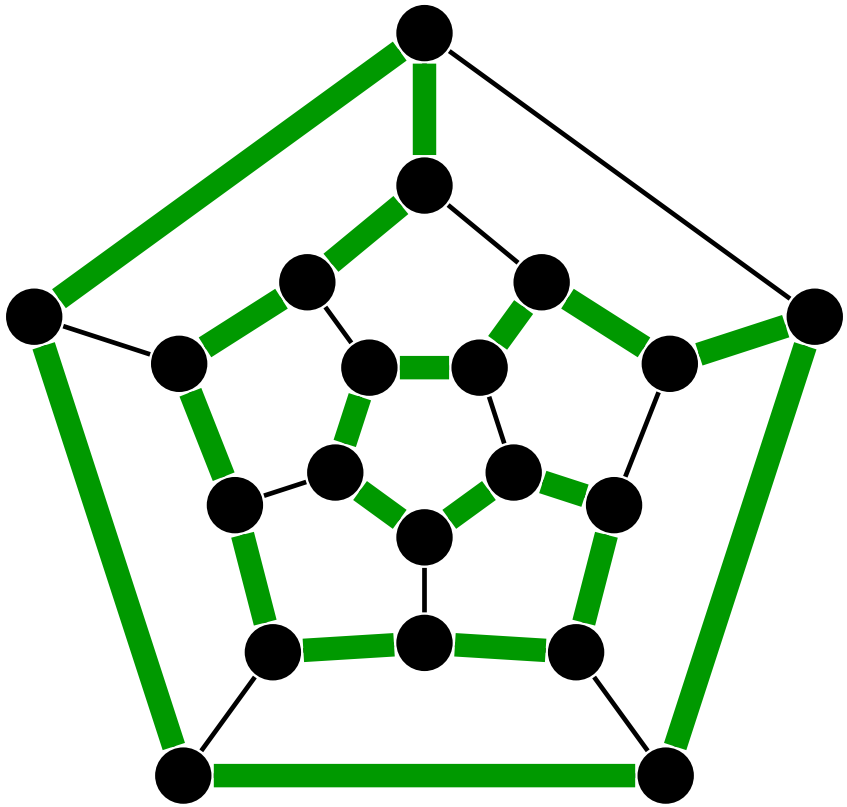
The number of Hamiltonian cycles

Hamiltonian cycle: a cycle that contains all the vertices.



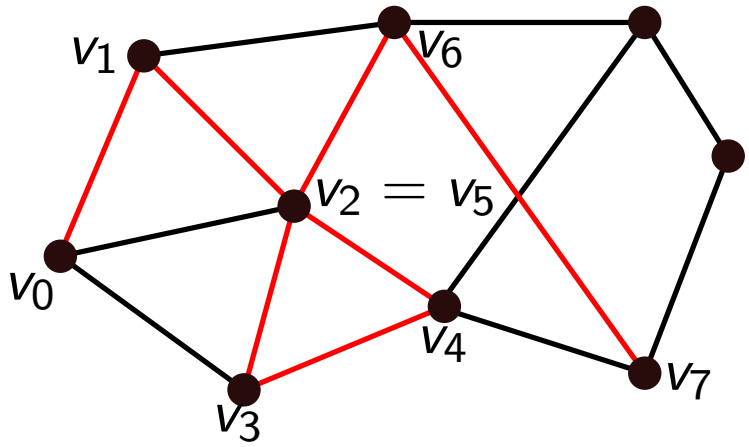
The number of Hamiltonian cycles

Hamiltonian cycle: a cycle that contains all the vertices.



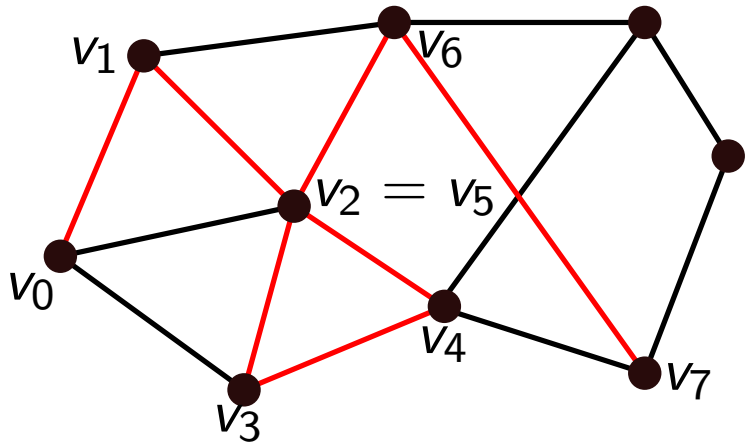
The number of Hamiltonian cycles

- A **walk** of length k in G (shortly, 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 $i = 0, \ldots, k-1$.



The number of Hamiltonian cycles

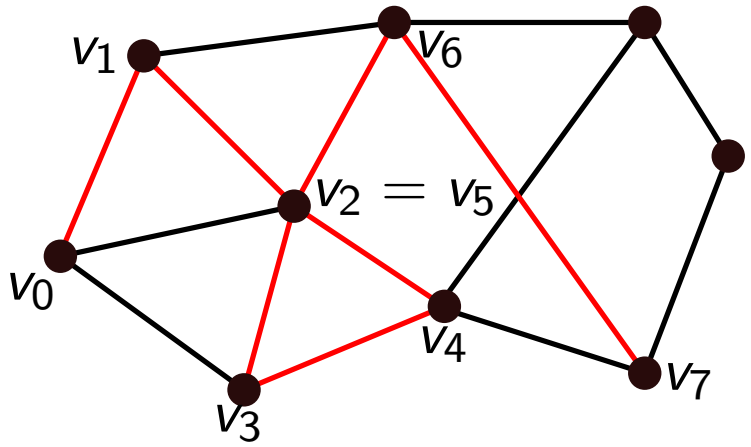
- A **walk** of length k in G (shortly, a k-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i = 0, \ldots, k-1$.



- A walk is **closed**, when $v_0 = v_k$.

The number of Hamiltonian cycles

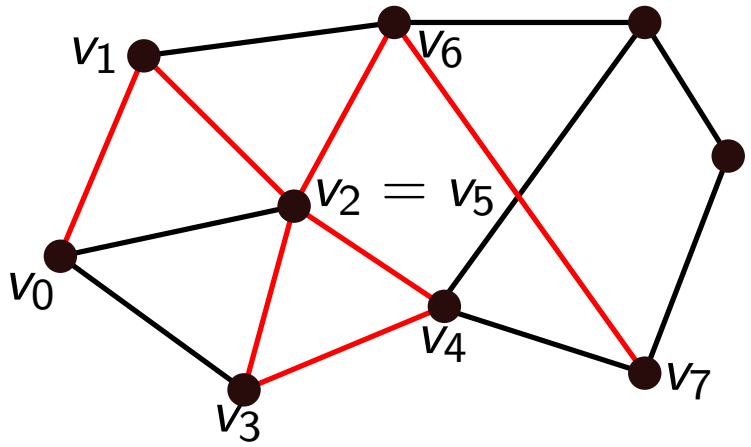
- A **walk** of length k in G (shortly, a k-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i = 0, \ldots, k-1$.



- A walk is **closed**, when $v_0 = v_k$.
- U is the set of closed n-walks from vertex 1.

The number of Hamiltonian cycles

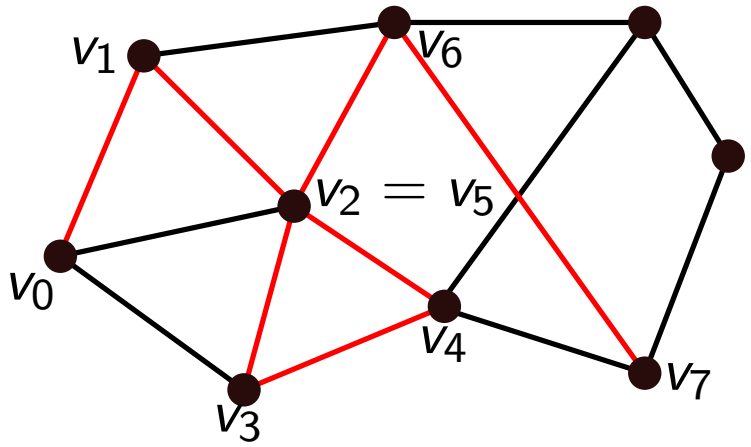
- A **walk** of length k in G (shortly, a k-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i = 0, \ldots, k-1$.



- A walk is **closed**, when $v_0 = v_k$.
- U is the set of closed n-walks from vertex 1.
- A_v = the walks from U that visit v, $v \in V$.

The number of Hamiltonian cycles

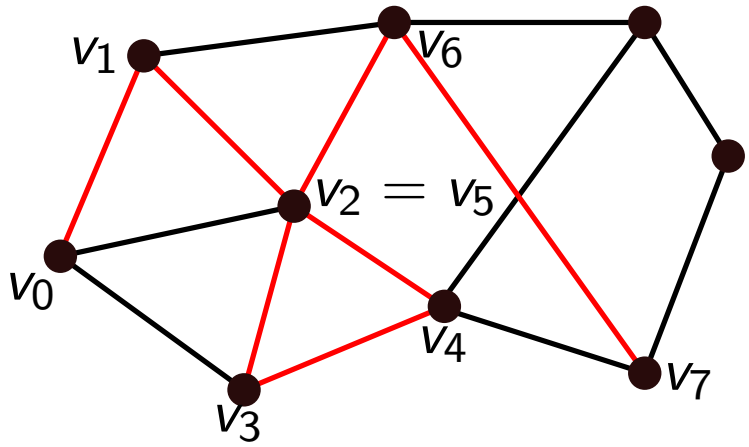
- A **walk** of length k in G (shortly, a k-walk) is a sequence of vertices $v_0, v_1, \ldots, v_k$ such that $v_i v_{i+1} \in E$ for each $i = 0, \ldots, k-1$.



- A walk is **closed**, when $v_0 = v_k$.
- U is the set of closed n-walks from vertex 1.
- A_v = the walks from U that visit v, $v \in V$.
- Then the solution is $|\bigcap_{v \in V} A_v|$.

The number of Hamiltonian cycles

- A **walk** of length k in G (shortly, 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 $i = 0, \ldots, k-1$.



- A walk is **closed**, when $v_0 = v_k$.
- U is the set of closed n-walks from vertex 1.
- A_v = the walks from U that visit v, $v \in V$.
- Then the solution is $|\bigcap_{v \in V} A_v|$.
- The simplified problem: $|\bigcap_{v \in X} \overline{A_v}|$ = the number of closed walks from U in $G' = G[V - X]$.

The number of Hamiltonian cycles, cont'd

The simplified problem

Compute the number of closed n-walks in G' that start at vertex 1.

Dynamic programming

- $T(d, x) =$ the number of length d walks from 1 to x.
- $T(d, x) = \sum_{yx \in E(G')} T(d-1, y)$.
- We return $T(n, 1)$, DP works in $O(n^3)$ time.

The number of Hamiltonian cycles, cont'd

The simplified problem

Compute the number of closed n-walks in G' that start at vertex 1.

Dynamic programming

- $T(d, x) =$ the number of length d walks from 1 to x.
- $T(d, x) = \sum_{yx \in E(G')} T(d-1, y)$.
- We return $T(n, 1)$, DP works in $O(n^3)$ time.

Theorem (Kohn, Gottlieb, Kohn 1969, Karp 1982, Bax 1993)

We can solve the Hamiltonian Cycle problem (and even find the number of such cycles) in $O(2^n n^3) = 2^n n^{O(1)}$ time and **polynomial space**.

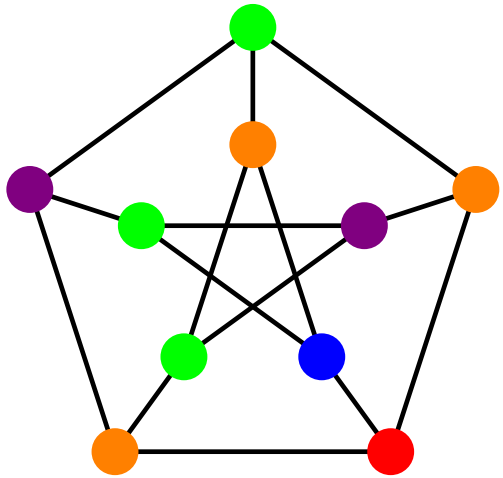
Coloring

k-coloring

k-coloring of a graph $G = (V, E)$ is a function $c : V \to \{1, \ldots, k\}$ such that for every edge $xy \in E$, $c(x) \neq c(y)$.

Problem

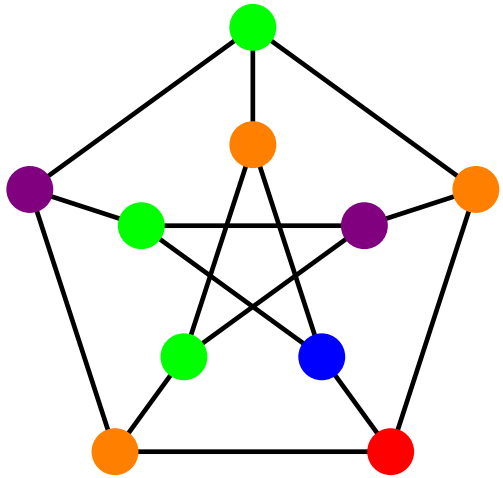
Given a graph $G = (V, E)$ and $k \in \mathbb{N}$ decide whether there is a k-coloring of G.



Coloring

k-coloring

k-coloring of a graph $G = (V, E)$ is a function $c : V \to \{1, \ldots, k\}$ such that for every edge $xy \in E$, $c(x) \neq c(y)$.



Problem

Given a graph $G = (V, E)$ and $k \in \mathbb{N}$ decide whether there is a k-coloring of G.

History

- (naive) $k^n n^{O(1)}$
- Lawler 1976: Dynamic programming $O(2.45^n)$
- Björklund, Husfeldt, Koivisto 2006: Inclusion-Exclusion $2^n n^{O(1)}$

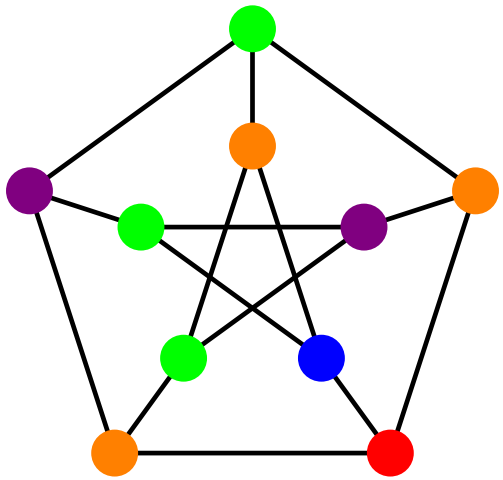
Coloring via inclusion-exclusion in $2^n n^{O(1)}$ time

Observation

There is a k-coloring of graph $G = (V, E)$

$$\Updownarrow$$

There is a partition of V into k independent sets



Coloring via inclusion-exclusion, first attempt

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)

Coloring via inclusion-exclusion, first attempt

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- U contains all colorings

Coloring via inclusion-exclusion, first attempt

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- U contains all colorings
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \$ there is exactly one j such that $v \in I_j\}$

Coloring via inclusion-exclusion, first attempt

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- U contains all colorings
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \$ there is exactly one j such that $v \in I_j\}$
- Then $|\bigcap_{v \in V} A_v|$ is the number of k-colorings.

Coloring via inclusion-exclusion, first attempt

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- U contains all colorings
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \$ there is exactly one j such that $v \in I_j\}$
- Then $|\bigcap_{v \in V} A_v|$ is the number of k-colorings.
- The simplified problem: $|\bigcap_{v \in X} \overline{A_v}| = ??$

Coloring via inclusion-exclusion in $2^n n^{O(1)}$ time

Observation

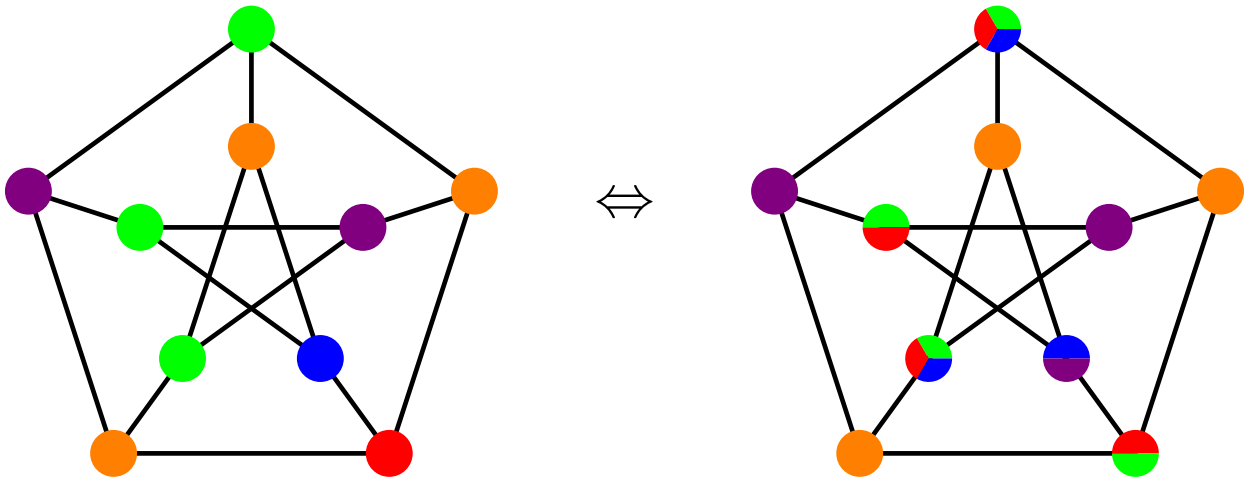
There is a k-coloring of graph $G = (V, E)$

$\Updownarrow$

There is a partition of V into k independent sets

$\Updownarrow$

there is a **cover** of V by k independent sets,
(i.e. k independent sets $I_1, \ldots, I_k$ such that $\bigcup_{j=1}^{k} I_j = V$.)



Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- Then $|\bigcap_{v \in V} A_v| \neq 0$ iff G is k-colorable.

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- Then $|\bigcap_{v \in V} A_v| \neq 0$ iff G is k-colorable.
- The simplified problem:

$$|\bigcap_{v \in X} \overline{A_v}| =$$

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- Then $|\bigcap_{v \in V} A_v| \neq 0$ iff G is k-colorable.
- The simplified problem:

$$|\bigcap_{v \in X} \overline{A_v}| = |\{(I_1, \ldots, I_k) \in U \ : \ I_1, \ldots, I_k \subseteq V - X\}|$$

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- Then $|\bigcap_{v \in V} A_v| \neq 0$ iff G is k-colorable.
- The simplified problem:

$$|\bigcap_{v \in X} \overline{A_v}| = |\{(I_1, \ldots, I_k) \in U \ : \ I_1, \ldots, I_k \subseteq V - X\}| = s(V - X)^k$$

where $s(Y) =$ the number of independent sets in $G[Y]$.

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- Then $|\bigcap_{v \in V} A_v| \neq 0$ iff G is k-colorable.
- The simplified problem:

$$|\bigcap_{v \in X} \overline{A_v}| = |\{(I_1, \ldots, I_k) \in U \ : \ I_1, \ldots, I_k \subseteq V - X\}| = s(V - X)^k$$

  where $s(Y) =$ the number of independent sets in $G[Y]$.
- $s(Y)$ can be computed at the beginning **for all subsets** $Y \subseteq V$: $s(Y) = s(Y - \{y\}) + s(Y - N[y])$. This takes time (**and space**) $2^n n^{O(1)}$. (Note: $s(Y)$ is stored using at most n bits).

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- Then $|\bigcap_{v \in V} A_v| \neq 0$ iff G is k-colorable.
- The simplified problem:

$$|\bigcap_{v \in X} \overline{A_v}| = |\{(I_1, \ldots, I_k) \in U \ : \ I_1, \ldots, I_k \subseteq V - X\}| = s(V - X)^k$$

  where $s(Y) =$ the number of independent sets in $G[Y]$.
- $s(Y)$ can be computed at the beginning **for all subsets** $Y \subseteq V$: $s(Y) = s(Y - \{y\}) + s(Y - N[y])$. This takes time (**and space**) $2^n n^{O(1)}$. (Note: $s(Y)$ is stored using at most n bits).
- Next, we compute $|\bigcap_{v \in X} \overline{A_v}|$ easily in $n^{O(1)}$ time, so we get $|\bigcap_{v \in V} A_v|$ in $2^n n^{O(1)}$ time.

Coloring in 2^n, cont'd

Theorem

In $2^n n^{O(1)}$ time and space we can

- find a k-coloring or conclude it does not exist,
- find the chromatic number.

Coloring in 2^n, cont'd

Theorem

In $2^n n^{O(1)}$ time and space we can

- find a k-coloring or conclude it does not exist,
- find the chromatic number.

Theorem

In $O(2.24^n)$ time and **polynomial space** we can find a k-coloring of a given graph G or conclude that it does not exist.

Proof

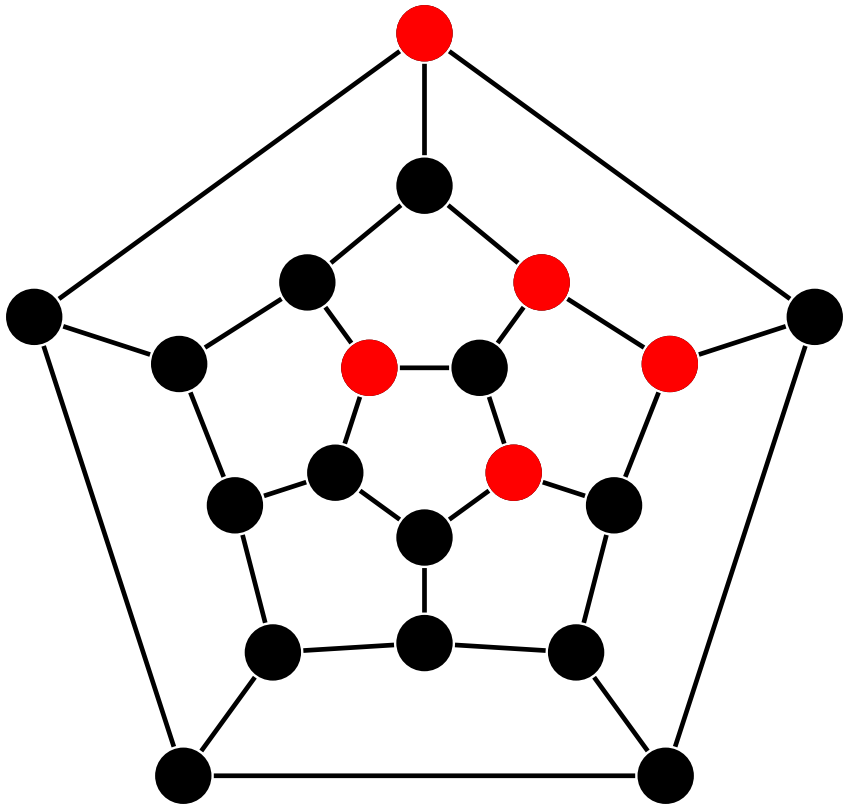
We compute $s(Y)$ in $O(1.2377^n)$ time and **polynomial space** by the algorithm of Wahlström (2008). Total time:

$$\sum_{X \subseteq V} 1.2377^{|X|} = \sum_{k=0}^{n} \binom{n}{k} 1.2377^k = (1 + 1.2377)^n = O(2.24^n).$$

Steiner Tree in 2^k, Nederlof 2009

Unweighted version

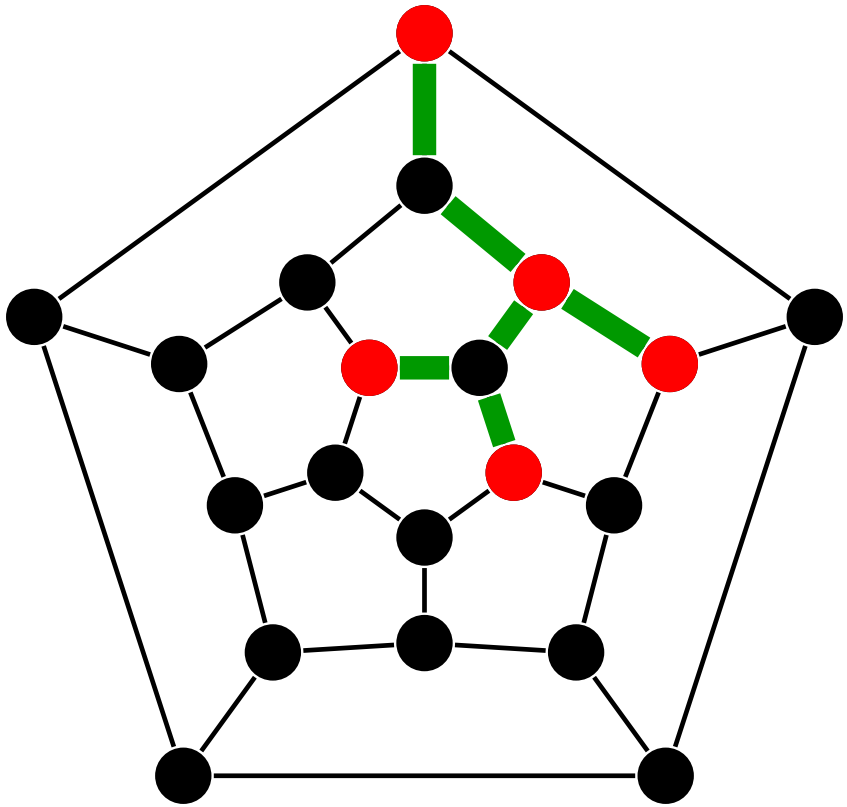
Given graph $G = (V, E)$, the set of terminals $K \subseteq V$ and a number $c \in \mathbb{N}$.
Is there a tree $T \subseteq G$ such that $K \subseteq V(T)$ and $|E(T)| \leq c$?



Steiner Tree in 2^k, Nederlof 2009

Unweighted version

Given graph $G = (V, E)$, the set of terminals $K \subseteq V$ and a number $c \in \mathbb{N}$.
Is there a tree $T \subseteq G$ such that $K \subseteq V(T)$ and $|E(T)| \leq c$?



Steiner Tree in 2^k, Nederlof 2009

Unweighted version

Given graph $G = (V, E)$, the set of terminals $K \subseteq V$ and a number $c \in \mathbb{N}$.
Is there a tree $T \subseteq G$ such that $K \subseteq V(T)$ and $|E(T)| \leq c$?

Denote $n = |V|$, $k = |K|$.

The classical algorithm [Dreyfus, Wagner 1972]

Dynamic programming, works in $3^k(nk)^{O(1)}$ time and $2^k(nk)^{O(1)}$ space, even in the weighted version.

Branching walks

Definition

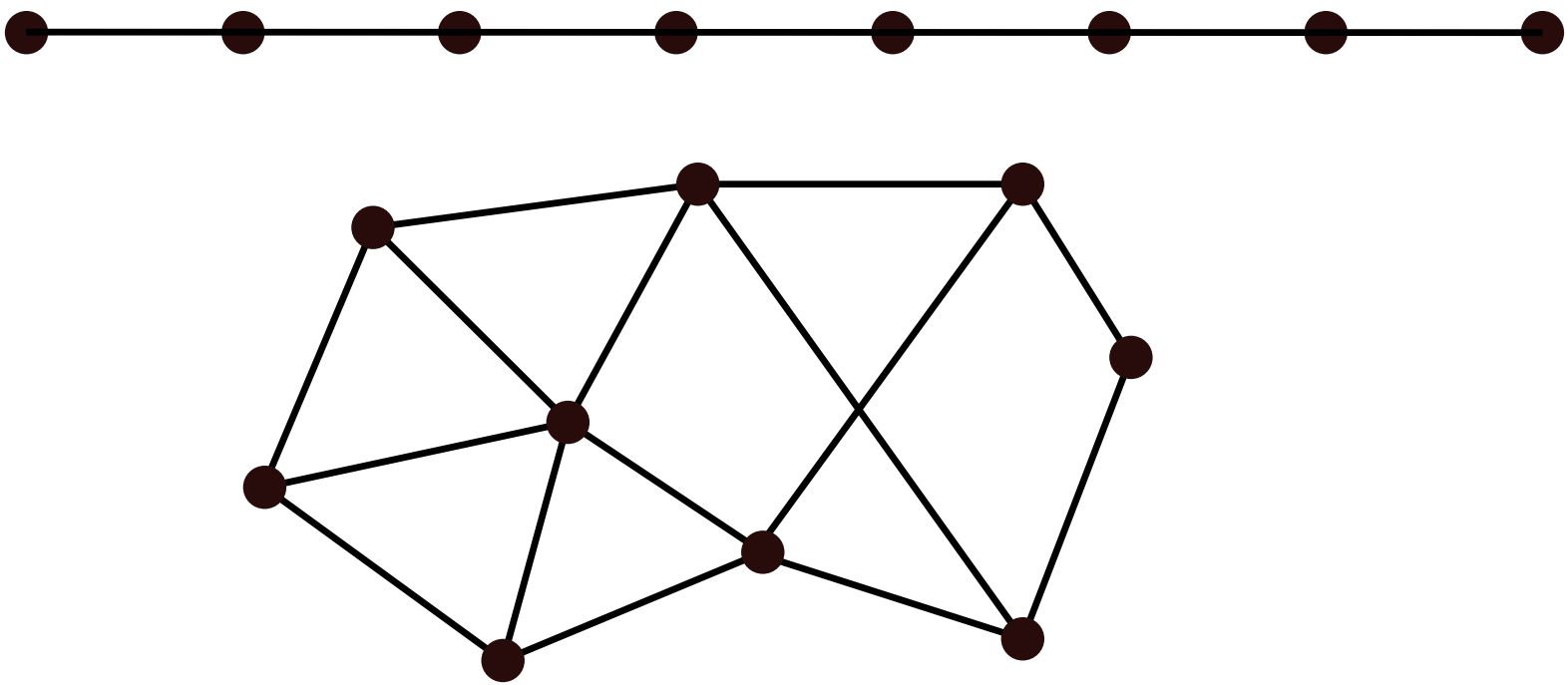
Let $G = (V, E)$ be an undirected graph and let $s \in V$.
A branching walk is a pair $B = (T, h)$, where

- T is an ordered rooted tree and
- $h : V(T) \to V$ is a homomorphism,
  i.e. if $(x, y) \in E(T)$ then $h(x)h(y) \in E(G)$.

We say that B is from s, when $h(r) = s$, where r is the root of T.
The length of B is defined as $|E(T)|$.

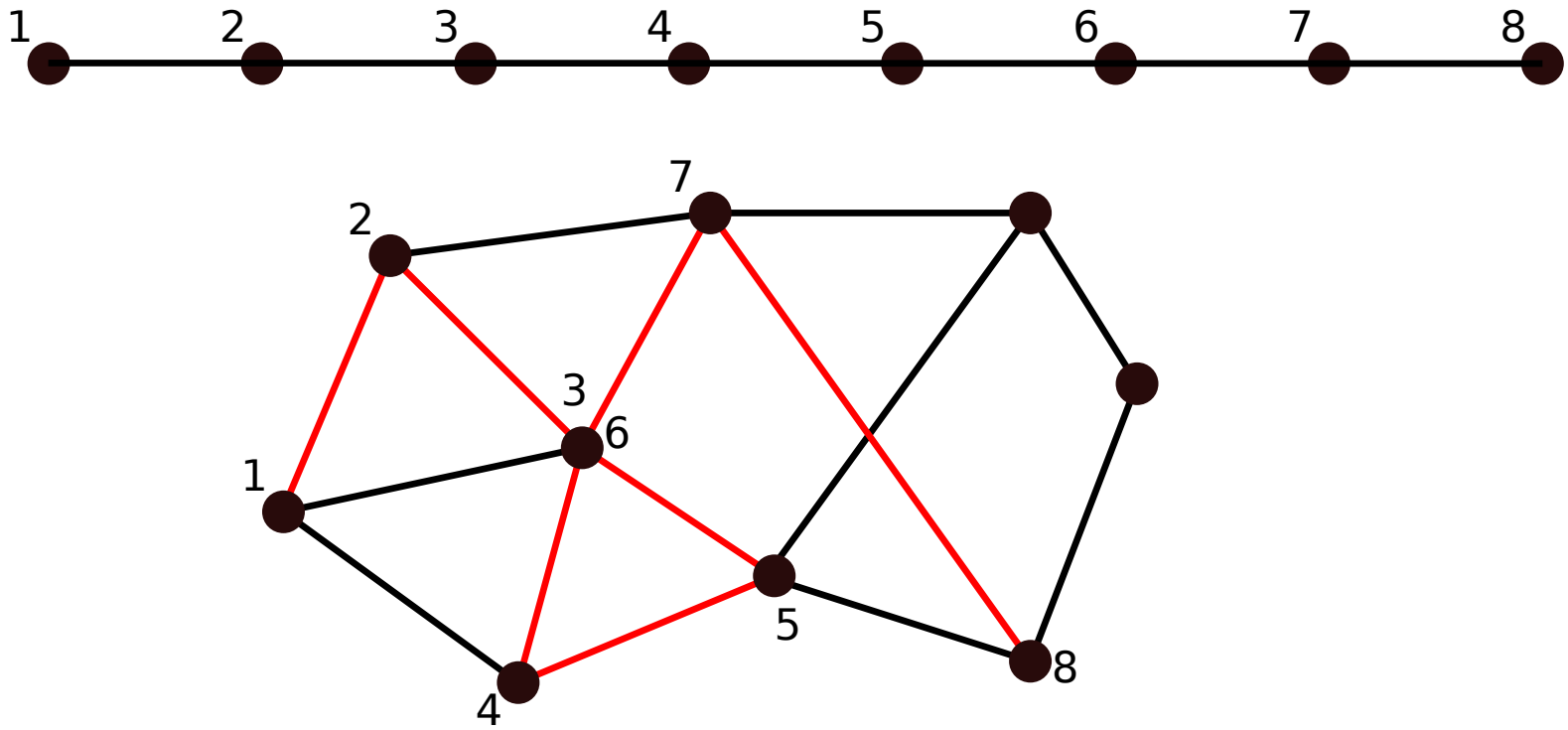
Branching walks

Example 1 Every walk is a branching walk

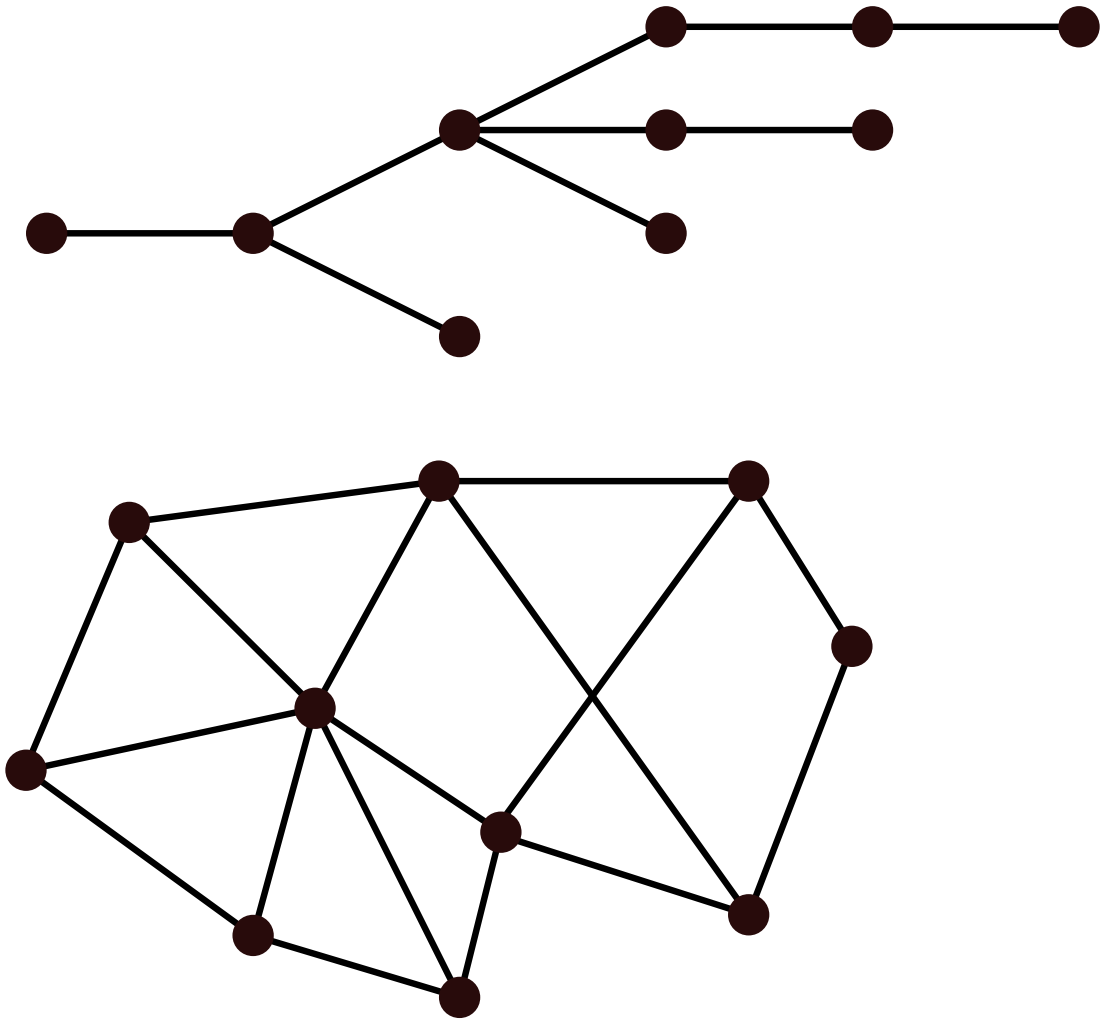


Branching walks

Example 1 Every walk is a branching walk

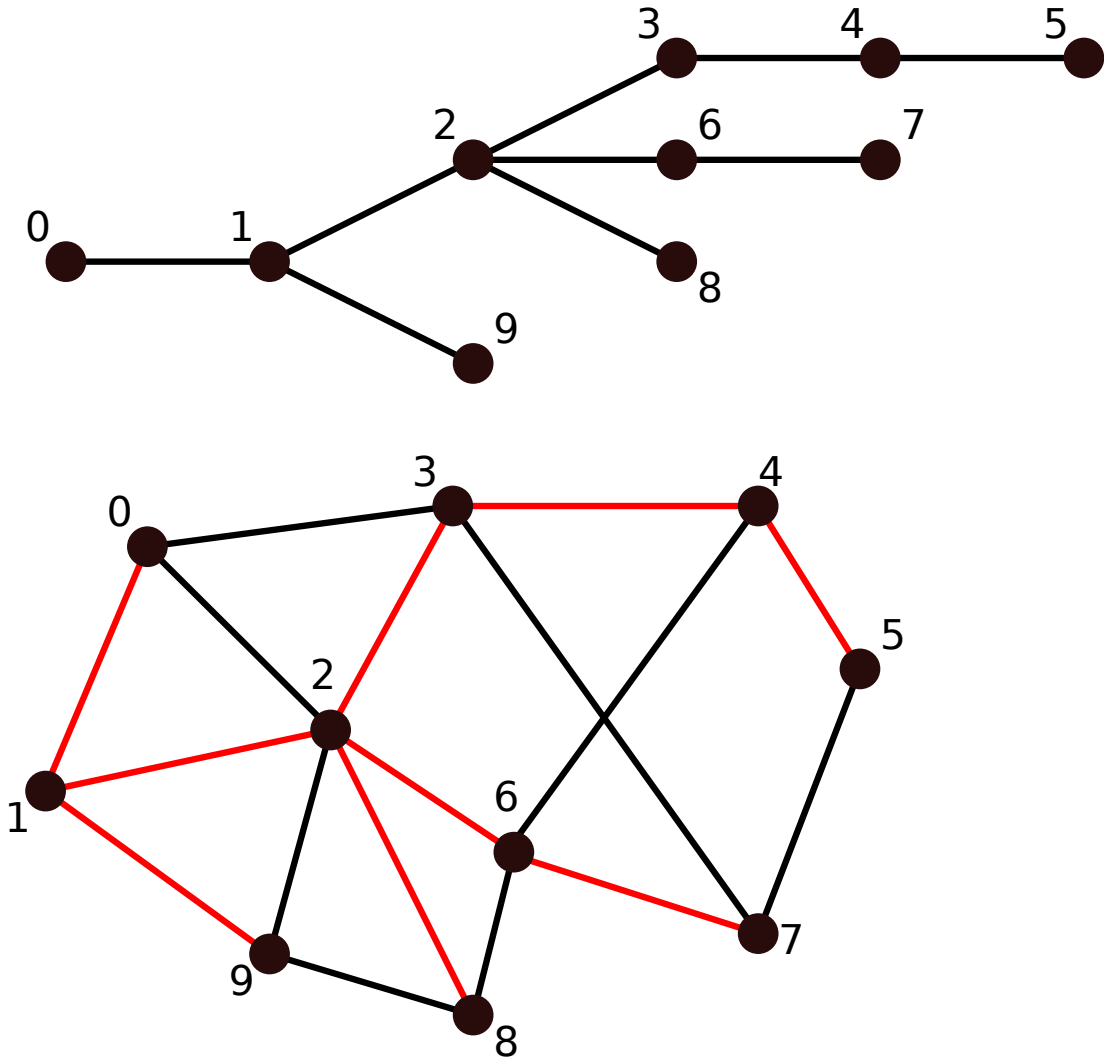


Branching walks



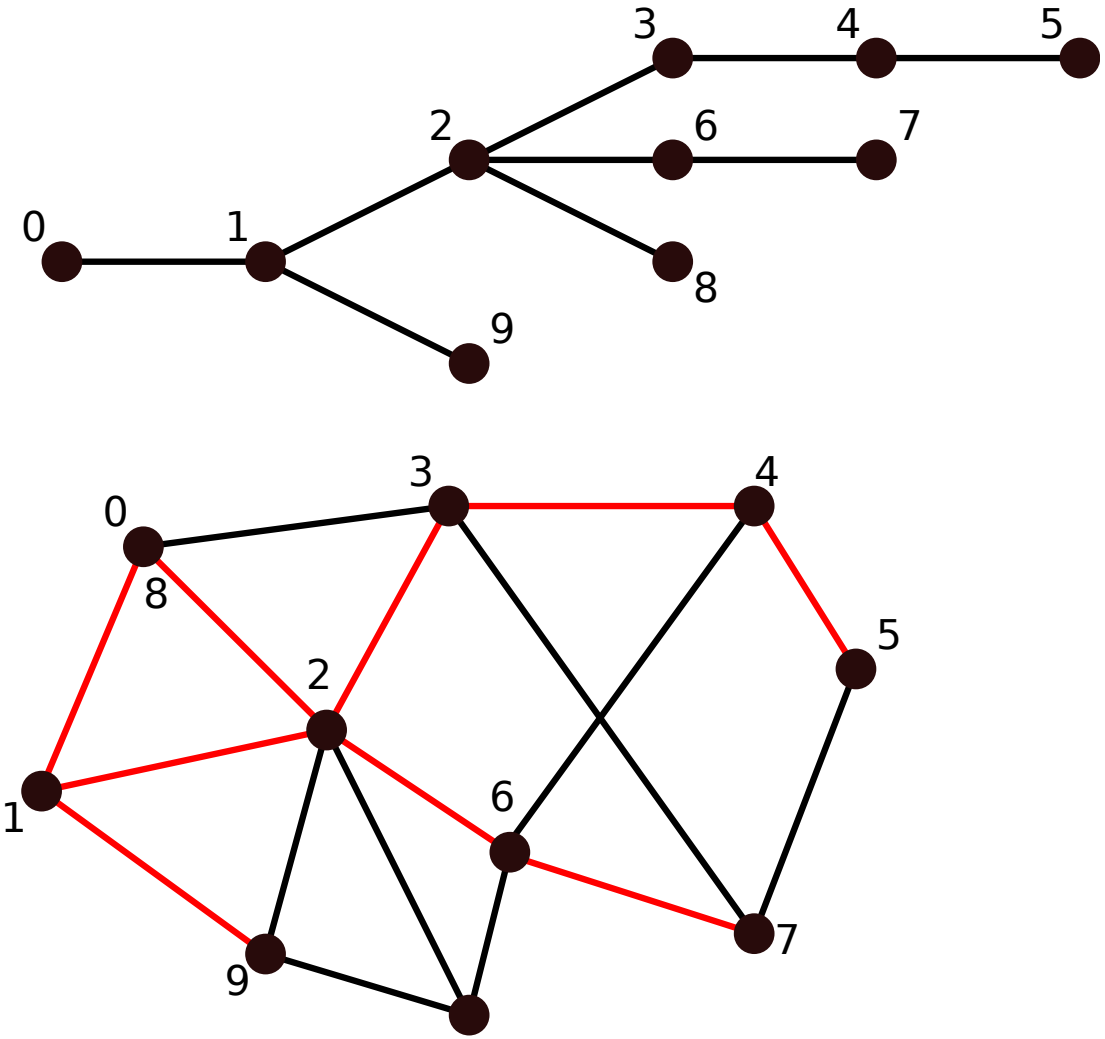
Branching walks

Example 2 An injective homomorphism.



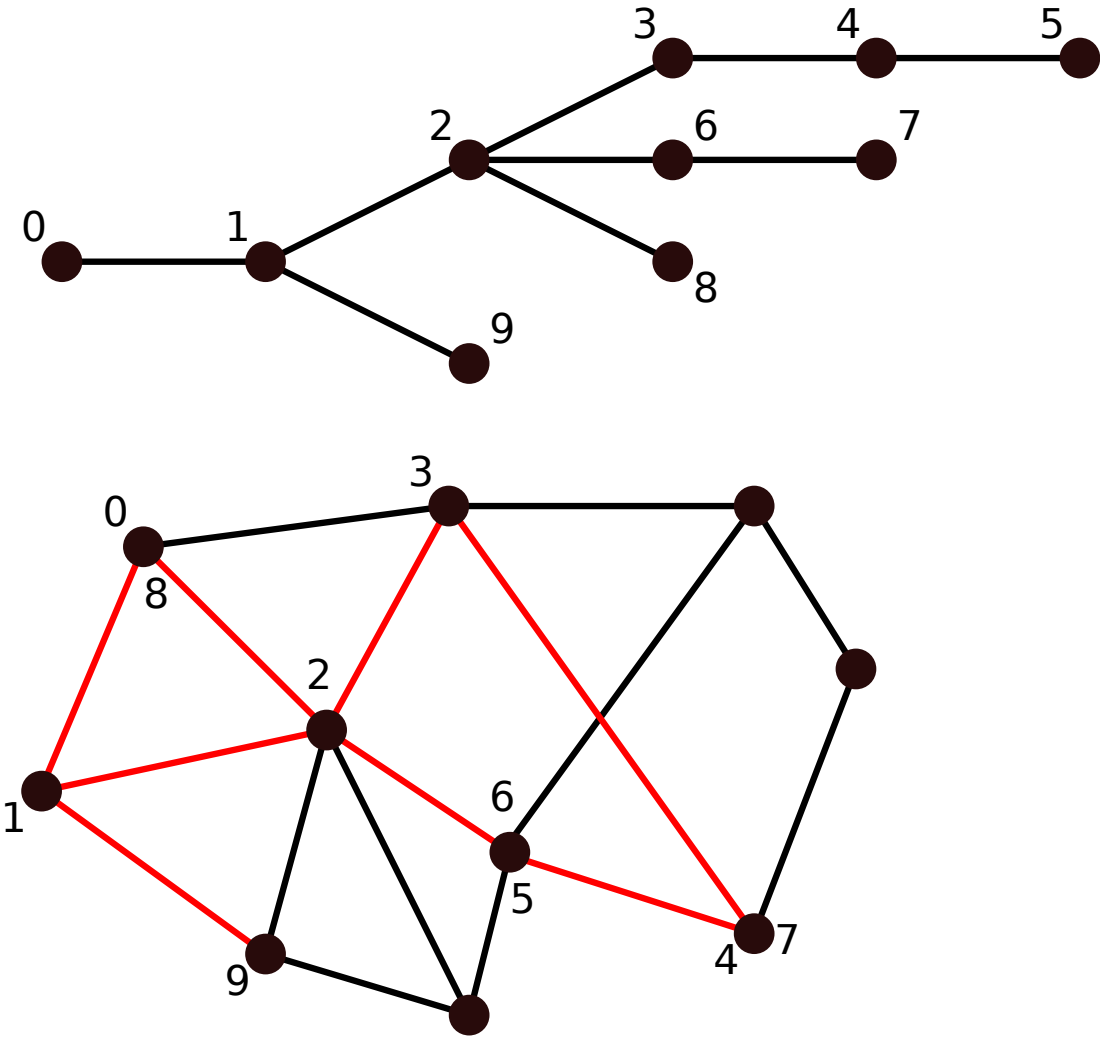
Branching walks

Example 3 A non-injective homomorphism.



Branching walks

Example 4 An even more non-injective homomorphism.



Steiner Tree, unweighted

For a branching walk $B = (T_B, h)$ denote $V(B) = h(V(T_B))$.
Let $s \in K$ be any terminal.

Observation

G contains a tree T such that $K \subseteq V(T)$ and $|E(T)| \leq c$ iff
G contains a branching walk $B = (T_B, h)$ from s in G such that
$K \subseteq V(B)$ and $|E(T_B)| \leq c$.

Steiner Tree, unweighted

For a branching walk $B = (T_B, h)$ denote $V(B) = h(V(T_B))$.
Let $s \in K$ be any terminal.

Observation

G contains a tree T such that $K \subseteq V(T)$ and $|E(T)| \leq c$ iff
G contains a branching walk $B = (T_B, h)$ from s in G such that
$K \subseteq V(B)$ and $|E(T_B)| \leq c$.

- U is the set of all length c branching walks from s.

Steiner Tree, unweighted

For a branching walk $B = (T_B, h)$ denote $V(B) = h(V(T_B))$.
Let $s \in K$ be any terminal.

Observation

G contains a tree T such that $K \subseteq V(T)$ and $|E(T)| \leq c$ iff
G contains a branching walk $B = (T_B, h)$ from s in G such that
$K \subseteq V(B)$ and $|E(T_B)| \leq c$.

- U is the set of all length c branching walks from s.
- $A_v = \{B \in U \ : \ v \in V(B)\}$ for $v \in K$.

Steiner Tree, unweighted

For a branching walk $B = (T_B, h)$ denote $V(B) = h(V(T_B))$.
Let $s \in K$ be any terminal.

Observation

G contains a tree T such that $K \subseteq V(T)$ and $|E(T)| \leq c$ iff
G contains a branching walk $B = (T_B, h)$ from s in G such that
$K \subseteq V(B)$ and $|E(T_B)| \leq c$.

- U is the set of all length c branching walks from s.
- $A_v = \{B \in U \ : \ v \in V(B)\}$ for $v \in K$.
- Then $|\bigcap_{v \in K} A_v| \neq 0$ iff there is the desired Steiner Tree.

Steiner Tree, unweighted

For a branching walk $B = (T_B, h)$ denote $V(B) = h(V(T_B))$.
Let $s \in K$ be any terminal.

Observation

G contains a tree T such that $K \subseteq V(T)$ and $|E(T)| \leq c$ iff
G contains a branching walk $B = (T_B, h)$ from s in G such that
$K \subseteq V(B)$ and $|E(T_B)| \leq c$.

- U is the set of all length c branching walks from s.
- $A_v = \{B \in U \ : \ v \in V(B)\}$ for $v \in K$.
- Then $|\bigcap_{v \in K} A_v| \neq 0$ iff there is the desired Steiner Tree.
- The simplified problem: for every $X \subseteq K$ compute

$$|\bigcap_{v \in X} \overline{A_v}| = b_c^{V \setminus X}(s),$$

where $b_j^{V \setminus X}(a)$ = the number of length j branching walks from a in $G[V \setminus X]$.

Steiner Tree, the simplified problem

$b_j^{V \setminus X}(a)$ = the number of length j branching walks from a in $G[V \setminus X]$.

The simplified problem

For any $X \subseteq K$ compute $b_c^{V \setminus X}(s)$.

Steiner Tree, the simplified problem

$b_j^{V\setminus X}(a)$ = the number of length j branching walks from a in $G[V \setminus X]$.

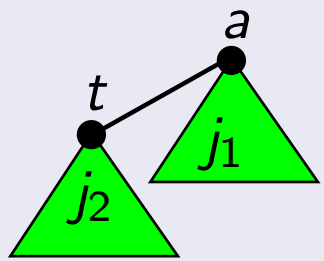
The simplified problem

For any $X \subseteq K$ compute $b_c^{V\setminus X}(s)$.

Dynamic Programming: computing $b_c^{V\setminus X}(s)$ in polynomial time

Compute $b_j^{V\setminus X}(a)$ for all $j = 0, \ldots, c$ and $a \in V \setminus X$ using DP:

$$b_j^{V\setminus X}(a) = \begin{cases} 1 & \text{if } j = 0, \\ \sum_{t \in N(a)\setminus X} \sum_{j_1+j_2=j-1} b_{j_1}^{V\setminus X}(a) b_{j_2}^{V\setminus X}(t) & \text{otherwise.} \end{cases}$$



Steiner Tree, conclusion

Corollary [Nederlof 2009]

The unweighted Steiner Tree problem can be solved in $2^k(nk)^{O(1)}$ time and polynomial space.

The zeta ζ transform and the Möbius μ transform

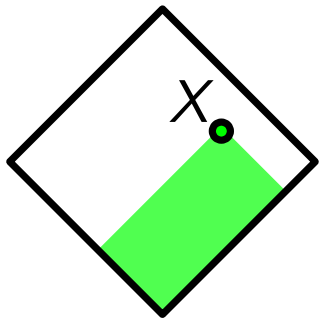
We consider functions from subsets of a finite set V to some ring – for simplicity let us fix the ring $(\mathbb{Z}, +, \cdot)$.

$$f \ : \ 2^V \to \mathbb{Z}$$

The transforms below transform f into another function $g \ : \ 2^V \to \mathbb{Z}$.

The Zeta transform

$(\zeta f)(X) = \sum_{Y \subseteq X} f(Y)$.



The Möbius transform

$(\mu f)(X) = \sum_{Y \subseteq X} (-1)^{|X \setminus Y|} f(Y)$.

Why ζ and μ are cool?

The Zeta and Möbius transforms

$(\zeta f)(X) = \sum_{Y \subseteq X} f(Y)$ $\qquad$ $(\mu f)(X) = \sum_{Y \subseteq X} (-1)^{|X \setminus Y|} f(Y)$.

Inversion formula

For every $X \subseteq V$, we have $f(X) = (\mu\zeta f)(X) = (\zeta\mu f)(X)$.

Proof.

$$\begin{aligned}
\mu\zeta f(X) &= \sum_{Y \subseteq X} (-1)^{|X \setminus Y|} (\zeta f)(Y) = \sum_{Y \subseteq X} (-1)^{|X \setminus Y|} \sum_{Z \subseteq Y} f(Z) \\
&= \sum_{Z \subseteq X} f(Z) \cdot \sum_{Z \subseteq Y \subseteq X} (-1)^{|X \setminus Y|} \\
&= f(X) + \sum_{Z \subsetneq X} f(Z) \cdot \sum_{Z \subseteq Y \subseteq X} (-1)^{|X \setminus Y|} \\
&= f(X) + \sum_{Z \subsetneq X} f(Z) \cdot \underbrace{\sum_{X \setminus Y \subseteq X \setminus Z} (-1)^{|X \setminus Y|}}_{0}
\end{aligned}$$

Hamiltonian cycle revisited

Counting HCs in a directed graph $G = (V, E)$, $V = \{1, \ldots, n\}$

For $X \subseteq V$, let $f(X)$ be the number of closed n-walks W from vertex 1 such that $V(W) = X$.
Then:

Hamiltonian cycle revisited

Counting HCs in a directed graph $G = (V, E)$, $V = \{1, \ldots, n\}$

For $X \subseteq V$, let $f(X)$ be the number of closed n-walks W from vertex 1 such that $V(W) = X$.
Then:

- $f(V)$ is the number of Hamiltonian cycles in G.

Hamiltonian cycle revisited

Counting HCs in a directed graph $G = (V, E)$, $V = \{1, \ldots, n\}$

For $X \subseteq V$, let $f(X)$ be the number of closed n-walks W from vertex 1 such that $V(W) = X$.
Then:

- $f(V)$ is the number of Hamiltonian cycles in G.
- $(\zeta f)(X) = \sum_{S \subseteq X} f(X)$ is the number of closed n-walks W from vertex 1 such that $V(W) \subseteq X$.

Hamiltonian cycle revisited

Counting HCs in a directed graph $G = (V, E)$, $V = \{1, \ldots, n\}$

For $X \subseteq V$, let $f(X)$ be the number of closed n-walks W from vertex 1 such that $V(W) = X$.
Then:

- $f(V)$ is the number of Hamiltonian cycles in G.
- $(\zeta f)(X) = \sum_{S \subseteq X} f(X)$ is the number of closed n-walks W from vertex 1 such that $V(W) \subseteq X$.
- Hence for every X, the value of $(\zeta f)(X)$ can be computed in $O(n^3)$ time (DP).

Hamiltonian cycle revisited

Counting HCs in a directed graph $G = (V, E)$, $V = \{1, \ldots, n\}$

For $X \subseteq V$, let $f(X)$ be the number of closed n-walks W from vertex 1 such that $V(W) = X$.
Then:

- $f(V)$ is the number of Hamiltonian cycles in G.
- $(\zeta f)(X) = \sum_{S \subseteq X} f(X)$ is the number of closed n-walks W from vertex 1 such that $V(W) \subseteq X$.
- Hence for every X, the value of $(\zeta f)(X)$ can be computed in $O(n^3)$ time (DP).
- So we compute $f(V) = (\mu\zeta f)(V) = \sum_{X \subseteq V} (-1)^{|V \setminus X|} (\zeta f)(X)$ in $2^n n^{O(1)}$ time and polynomial space.

Computing ζ and μ for all subsets $X \subseteq V$

$$(\zeta f)(X) = \sum_{Y \subseteq X} f(Y) \qquad (\mu f)(X) = \sum_{Y \subseteq X} (-1)^{|X \setminus Y|} f(Y).$$

Naive algorithm

- evaluating at single X: $O(2^{|X|})$.
- evaluating at **all** $X \subseteq V$: $O(\sum_{X \subseteq V} 2^{|X|}) = O(3^{|V|})$.

Yates' algorithm (1937), described in Knuth's TAOCP

Given a function $f : 2^V \to \mathbb{Z}$, we can compute **all** the 2^n values of ζf in $2^n n^{O(1)}$ time. Similarly μf.

Fast Zeta Transform: all values of ζf in $O(2^n \cdot n)$ time

Let $V = \{1, \ldots, n\}$. Represent subsets as characteristic vectors:

$$(\zeta f)(x_1, \ldots, x_n) = \sum_{\substack{y_1, \ldots, y_n \in \{0,1\} \\ y_1 \leq x_1, \ldots, y_n \leq x_n}} f(y_1, \ldots, y_n).$$

Consider fixing the last $n - j$ bits:

$$\zeta_j(x_1, \ldots, x_n) = \sum_{\substack{y_1, \ldots, y_j \in \{0,1\} \\ y_1 \leq x_1, \ldots, y_j \leq x_j}} f(y_1, \ldots, y_j, \underbrace{x_{j+1}, \ldots, x_n}_{\text{fixed}}).$$

Consistently, $\zeta_0(x_1, \ldots, x_n) := f(x_1, \ldots, x_n)$. Note that $\zeta_n(X) = \zeta f(X)$.
Dynamic programming:

$$\zeta_j(x_1, \ldots, x_n) = \begin{cases} \zeta_{j-1}(x_1, \ldots, x_n) & \text{when } x_j = 0, \\ \zeta_{j-1}(x_1, \ldots, x_{j-1}, 1, x_{j+1}, \ldots, x_n) + & \\ \zeta_{j-1}(x_1, \ldots, x_{j-1}, 0, x_{j+1}, \ldots, x_n) & \text{when } x_j = 1. \end{cases}$$

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

- $f(X) \neq 0$ iff $G[X]$ is k-colorable.

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

- $f(X) \neq 0$ iff $G[X]$ is k-colorable.
- $\zeta f(X) = \sum_{S \subseteq X} f(X)$ is the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j \subseteq X$.

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

- $f(X) \neq 0$ iff $G[X]$ is k-colorable.
- $\zeta f(X) = \sum_{S \subseteq X} f(X)$ is the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j \subseteq X$.
- so, $\zeta f(X) = s(X)^k$.

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

- $f(X) \neq 0$ iff $G[X]$ is k-colorable.
- $\zeta f(X) = \sum_{S \subseteq X} f(X)$ is the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j \subseteq X$.
- so, $\zeta f(X) = s(X)^k$.
- As before, all 2^n values of ζf can be found in $2^n n^{O(1)}$ time and space.

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

- $f(X) \neq 0$ iff $G[X]$ is k-colorable.
- $\zeta f(X) = \sum_{S \subseteq X} f(X)$ is the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j \subseteq X$.
- so, $\zeta f(X) = s(X)^k$.
- As before, all 2^n values of ζf can be found in $2^n n^{O(1)}$ time and space.
- Using the Yates' algorithm we find $f = \mu \zeta f$.

k-coloring, revisited

For $X \subseteq V$, let $f(X)$ be the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j = X$.
Then:

- $f(X) \neq 0$ iff $G[X]$ is k-colorable.
- $\zeta f(X) = \sum_{S \subseteq X} f(X)$ is the number of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets in G and $\bigcup_{j=1}^{k} I_j \subseteq X$.
- so, $\zeta f(X) = s(X)^k$.
- As before, all 2^n values of ζf can be found in $2^n n^{O(1)}$ time and space.
- Using the Yates' algorithm we find $f = \mu \zeta f$.
- Thus we found **all** the induced k-colorable subgraphs of G in $2^n n^{O(1)}$ time and space.

The cover product

The cover product

The cover product of two functions $f, g : 2^V \to \mathbb{Z}$ is a function $(f *_c g) : 2^V \to \mathbb{Z}$ such that for every $Y \subseteq V$,

$$(f *_c g)(Y) = \sum_{A \cup B = Y} f(A)g(B).$$

The cover product

The cover product

The cover product of two functions $f, g : 2^V \to \mathbb{Z}$ is a function $(f *_c g) : 2^V \to \mathbb{Z}$ such that for every $Y \subseteq V$,

$$(f *_c g)(Y) = \sum_{A \cup B = Y} f(A)g(B).$$

Computing the cover product

As usual: We cannot compute $(f *_c g)$? Then we compute $\zeta(f *_c g)$.

$$\zeta(f *_c g)(Y) = \sum_{X \subseteq Y} \sum_{A \cup B = X} f(A)g(B) = \sum_{A \cup B \subseteq Y} f(A)g(B) =$$

$$= \left(\sum_{A \subseteq Y} f(A) \right) \left(\sum_{B \subseteq Y} g(B) \right) = (\zeta f(Y))(\zeta g(Y)).$$

Computing the cover product

As usual: We cannot compute $(f *_c g)$? Then we compute $\zeta(f *_c g)$.

$$\zeta(f *_c g)(Y) = \sum_{X \subseteq Y} \sum_{A \cup B = X} f(A)g(B) = \sum_{A \cup B \subseteq Y} f(A)g(B) =$$

$$= \left(\sum_{A \subseteq Y} f(A) \right) \left(\sum_{B \subseteq Y} g(B) \right) = (\zeta f(Y))(\zeta g(Y)).$$

Hence $(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$.

Fast cover product: an FFT-style algorithm

1. Compute ζf using Yates' algorithm in time $2^n n^{O(1)}$,
2. Compute ζg using Yates' algorithm in time $2^n n^{O(1)}$,
3. Compute $h = \zeta(f *_c g)$ using pointwise multiplication in time $2^n n^{O(1)}$,
4. Compute $f *_c g = \mu h$ using Yates' algorithm in time $2^n n^{O(1)}$.

Cover product: an application in graph coloring

Define $s : 2^V \to \{0, 1\}$ where $s(X) = \begin{cases} 1 & \text{if } X \text{ is independent} \\ 0 & \text{otherwise.} \end{cases}$.

Cover product: an application in graph coloring

Define $s : 2^V \to \{0, 1\}$ where $s(X) = \begin{cases} 1 & \text{if } X \text{ is independent} \\ 0 & \text{otherwise.} \end{cases}$.

Then

$$(s *_c s)(Y) = \sum_{A \cup B = Y} s(A)s(B) = \sum_{\substack{A \cup B = Y \\ A, B \text{ independent}}} 1.$$

$$(\underbrace{s *_c s *_c \cdots s}_{k \text{ times}})(Y) = \sum_{\substack{I_1 \cup \cdots \cup I_k = Y \\ I_1, \ldots, I_k \text{ independent}}} 1.$$

Cover product: an application in graph coloring

Define $s : 2^V \to \{0,1\}$ where $s(X) = \begin{cases} 1 & \text{if } X \text{ is independent} \\ 0 & \text{otherwise.} \end{cases}$.

Then

$$(s *_c s)(Y) = \sum_{A \cup B = Y} s(A)s(B) = \sum_{\substack{A \cup B = Y \\ A,B \text{ independent}}} 1.$$

$$(\underbrace{s *_c s *_c \cdots s}_{k \text{ times}})(Y) = \sum_{\substack{I_1 \cup \cdots \cup I_k = Y \\ I_1, \ldots, I_k \text{ independent}}} 1.$$

$\underbrace{s *_c s *_c \cdots s}_{k \text{ times}}(Y) \neq 0$ iff $G[Y]$ is k-colorable.

Cover product: an application in graph coloring

Define $s : 2^V \to \{0,1\}$ where $s(X) = \begin{cases} 1 & \text{if } X \text{ is independent} \\ 0 & \text{otherwise.} \end{cases}$.

Then

$$(s *_c s)(Y) = \sum_{A \cup B = Y} s(A)s(B) = \sum_{\substack{A \cup B = Y \\ A,B \text{ independent}}} 1.$$

$$(\underbrace{s *_c s *_c \cdots s}_{k \text{ times}})(Y) = \sum_{\substack{I_1 \cup \cdots \cup I_k = Y \\ I_1, \ldots, I_k \text{ independent}}} 1.$$

$$\underbrace{s *_c s *_c \cdots s}_{k \text{ times}}(Y) \neq 0 \text{ iff } G[Y] \text{ is } k\text{-colorable.}$$

By performing cover product $k-1$ times (even $O(\log k)$ times suffices) we obtain yet another algorithm which finds all k-colorable induced subgraphs in $2^n n^{O(1)}$ time.

Subset convolution

Subset convolution

The subset convolution of two functions $f, g : 2^V \to \mathbb{Z}$ is a function $(f * g) : 2^V \to \mathbb{Z}$ such that for every $Y \subseteq V$,

$$(f * g)(Y) = \sum_{X \subseteq Y} f(X)g(Y - X).$$

Equivalently...

$$(f * g)(Y) = \sum_{\substack{A \cup B = Y \\ A \cap B = \emptyset}} f(A)g(B).$$

Computing the subset convolution

For $f : 2^V \to \mathbb{Z}$ let f_k denote f trimmed to the cardinality k subsets, i.e.:

$$f_k(S) = \begin{cases} f(S) & \text{if } |S| = k, \\ 0 & \text{otherwise.} \end{cases}$$

Computing the subset convolution

For $f : 2^V \to \mathbb{Z}$ let f_k denote f trimmed to the cardinality k subsets, i.e.:

$$f_k(S) = \begin{cases} f(S) & \text{if } |S| = k, \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$(f * g)(Y) = \sum_{\substack{A \cup B = Y \\ A \cap B = \emptyset}} f(A)g(B) =$$

$$= \sum_{i=0}^{|Y|} \sum_{\substack{A \cup B = Y \\ A \cap B = \emptyset \\ |A| = i}} f(A)g(B) = \sum_{i=0}^{|Y|} \sum_{\substack{A \cup B = Y \\ |A| = i \\ |B| = |Y| - i}} f(A)g(B) =$$

$$= \sum_{i=0}^{|Y|} \sum_{A \cup B = Y} f_i(A) g_{|Y|-i}(B) = \sum_{i=0}^{|Y|} (f_i *_c g_{|Y|-i})(Y).$$

Computing the subset convolution

We got:

$$(*) \qquad (f * g)(Y) = \sum_{i=0}^{|Y|} (f_i *_c g_{|Y|-i})(Y).$$

Algorithm:

1. Compute and store $f_i *_c g_j(Y)$ for all $i, j = 0, \ldots, n$ and $Y \subseteq 2^V$.
2. Compute $(f * g)(Y)$ for all $Y \subseteq 2^V$ using $(*)$.

Corollary

One can compute $f * g$ in $2^n n^{O(1)}$ time.

Applications of fast subset convolution

Recall: $$(f * g)(Y) = \sum_{\substack{A \cup B = Y \\ A \cap B = \emptyset}} f(A)g(B).$$

An application in graph coloring

$\underbrace{s * s * \cdots s}_{k \text{ times}}(V)$ is the number of k-colorings of G.

Corollary: There is an algorithm which, for every induced subgraph H of G, finds the number of k-colorings of H in total $2^n n^{O(1)}$ time and space.

Speeding-up dynamic programming over tree decomposition

We can speed-up processing the join nodes (see the book and Marcin's talk on Thursday):

- $2^{\mathrm{tw}}\mathrm{tw}^{O(1)}n$-time algorithm for counting perfect matchings,
- $3^{\mathrm{tw}}\mathrm{tw}^{O(1)}n$-time algorithm for dominating set.

Exercises

- (Ex 10.6, ?) Extend the $2^n n^{O(1)}$ algorithm for Hamiltonian cycle to weighted version (aka TSP), i.e., find a minimum weight Hamiltonian cycle in a graph $G = (V, E)$ with weight function $w : E \to \{0, \ldots, W\}$ in time $2^n W n^{O(1)}$.
- (Ex 10.7,) Extend the $2^n n^{O(1)}$ algorithm for coloring to list coloring.
- (Ex 10.8) Show an algorithm which computes the number of perfect matchings in a given n-vertex bipartite graph in $2^{n/2} n^{O(1)}$ time. (The solution is called Ryser's Formula.)
- (Ex 10.12,) Show a $2^n n^{O(1)}$ algorithm for computing packing product defined as

$$(f *_p g)(Y) = \sum_{\substack{A,B \subseteq Y \\ A \cap B = \emptyset}} f(A) g(B).$$