Algebraic approaches to exact algorithms, part I: Inclusion-Exclusion

Łukasz Kowalik

University of Warsaw

ADFOCS, Saarbrücken, August 2013

Inclusion-Exclusion Principle

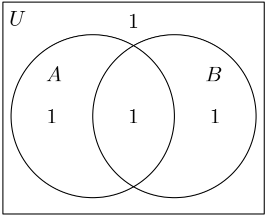
Theorem (Inclusion-Exclusion Principle, intersection version)

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

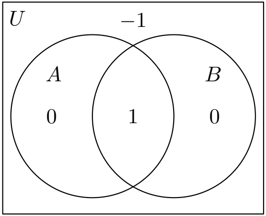
$$\left|\bigcap_{i\in\{1,\ldots,n\}} A_i\right| = \sum_{X\subseteq\{1,\ldots,n\}} (-1)^{|X|}\left|\bigcap_{i\in X}\overline{A_i}\right|$$

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

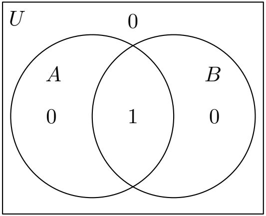
Example. $|A\cap B| = |U| - |\overline{A}| - |\overline{B}| + |\overline{A}\cap\overline{B}|$



$|U|$



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



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

Inclusion-Exclusion Principle, intersection version

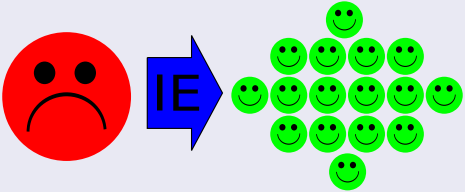
Theorem (Inclusion-Exclusion Principle, intersection version)

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:

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

A common algorithmic application

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



Iverson's Notation

$$[\alpha] = \begin{cases} 1 & \alpha \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Example:

$$\sum_{i=1}^{100} [i \text{ is even}] = 50$$

The number of Hamiltonian cycles (Karp 1982)

Hamiltonian cycle: a cycle that contains all the vertices.

Problem

Given an n-vertex undirected graph $G = (V, E)$ compute the number of Hamiltonian cycles.

The number of Hamiltonian cycles (Karp 1982)

Hamiltonian cycle: a cycle that contains all the vertices.

Problem

Given an n-vertex undirected graph $G = (V, E)$ compute 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 (Karp 1982)

Hamiltonian cycle: a cycle that contains all the vertices.

Problem

Given an n-vertex undirected graph $G = (V, E)$ compute 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 (Karp 1982)

Hamiltonian cycle: a cycle that contains all the vertices.

Problem

Given an n-vertex undirected graph $G = (V, E)$ compute 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 (Karp 1982)

Hamiltonian cycle: a cycle that contains all the vertices.

Problem

Given an n-vertex undirected graph $G = (V, E)$ compute 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 (Karp 1982)

Hamiltonian cycle: a cycle that contains all the vertices.

Problem

Given an n-vertex undirected graph $G = (V, E)$ compute 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.

Corollary

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

Notation: $f(n)n^{O(1)} = O^*(f(n))$.

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.

Note: If we can do it in time $T(n)$ then we can also **find** the coloring in $O^*(T(n))$ time when it exists, due to self-reducibility.

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.

Note: If we can do it in time $T(n)$ then we can also **find** the coloring in $O^*(T(n))$ time when it exists, due to self-reducibility.

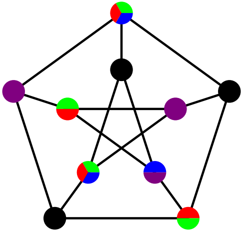
History

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

Coloring via inclusion-exclusion in $O^*(2^n)$ time

Observation

We can color a vertex with many colors at the same time – existence of such a coloring is equivalent to the existence of the classic coloring.



Coloring in 2^n, cont'd

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

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- 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 (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- 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 (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- 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 (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- 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 (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- 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**) $O^*(2^n)$, since the number of covers takes $O(n \log k)$ bits.

Coloring in 2^n, cont'd

- U is the set of tuples $(I_1, \ldots, I_k)$, where I_j are independent sets (not necessarily disjoint nor even different!)
- $A_v = \{(I_1, \ldots, I_k) \in U \ : \ v \in \bigcup_{j=1}^{k} I_j\}$
- 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)** $O^*(2^n)$, since the number of covers takes $O(n \log k)$ bits.
- Next, we compute $|\bigcap_{v \in X} \overline{A_v}|$ easily in $O^*(1)$ time, so we get $|\bigcap_{v \in V} A_v|$ in $O^*(2^n)$ time.

Coloring in 2^n, cont'd

Theorem

In $O^*(2^n)$ 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 $O^*(2^n)$ time and space we can

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

Theorem

In $O^*(2.25^n)$ 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 $|E(T)| \leq c$?

Weighted version

Additionally: weights on edges $w : E \to \mathbb{N}$. Is there a tree $T \subseteq G$ such that $K \subseteq V(T)$ and $w(E(T)) \leq c$?

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 $|E(T)| \leq c$?

Weighted version

Additionally: weights on edges $w : E \to \mathbb{N}$. Is there a tree $T \subseteq G$ such that $K \subseteq V(T)$ and $w(E(T)) \leq c$?

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

The classical algorithm [Dreyfus, Wagner 1972]

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

Branching walks

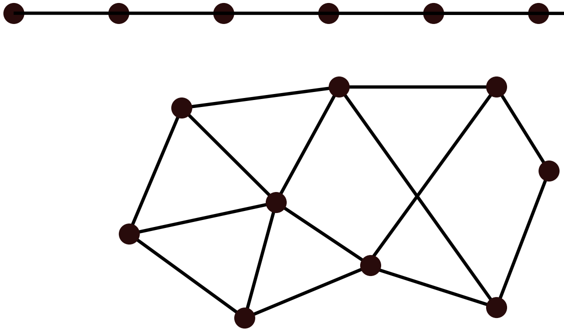
Definition

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 $h(x)h(y) \in E(G)$.

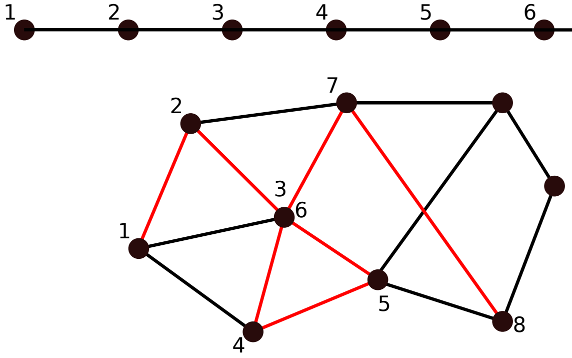
We say that B is from s, when $h(r) = s$, where r is the root of T.
The length of B is defined as $|E(T)|$.

Example 1 Every walk is a branching walk



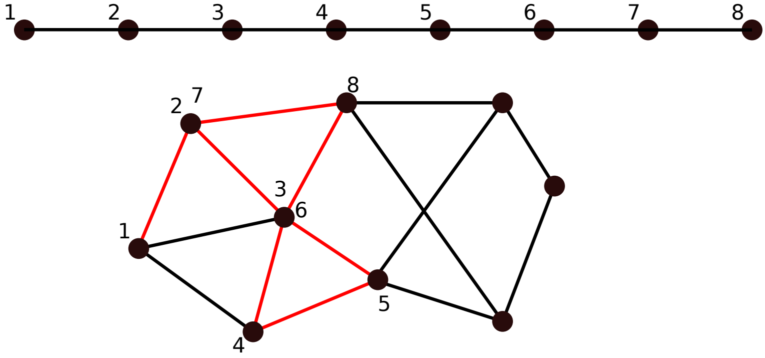
Branching walks

Example 1 Every walk is a branching walk

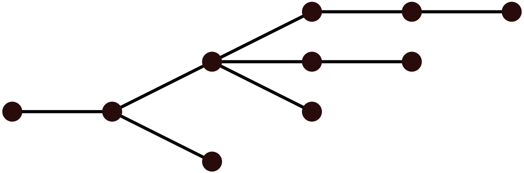


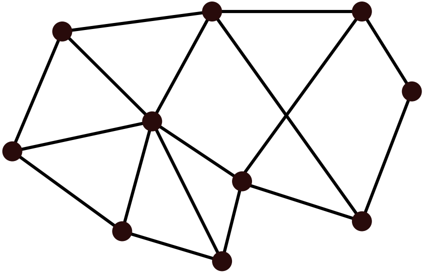
Branching walks

Example 2 Even this one.

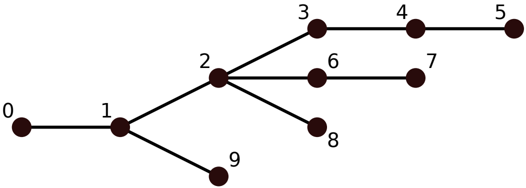


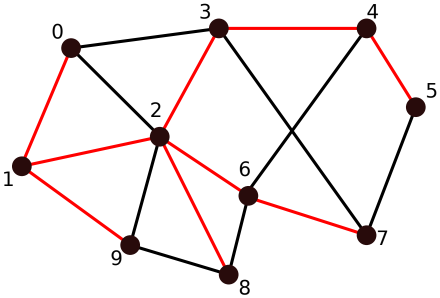
Branching walks



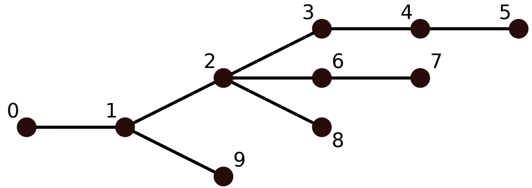


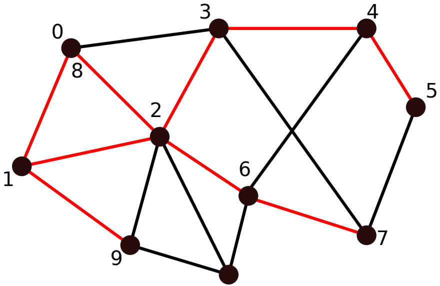
Example 3 An injective homomorphism.





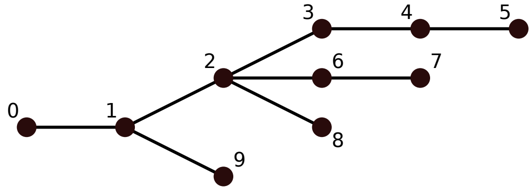
Example 4 A non-injective homomorphism.

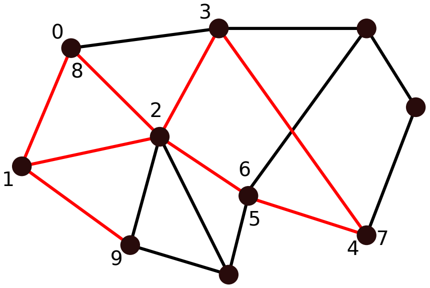




Branching walks

Example 5 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 $G[V \setminus X]$.

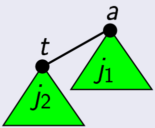
The simplified problem

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, finish

Corollary [Nederlof 2009]

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

Steiner Tree, finish

Corollary [Nederlof 2009]

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

Theorem [Nederlof 2009]

The **weighted** Steiner Tree problem can be solved in $O^*(C \cdot 2^k)$ time and $O^*(C)$ space. (We skip the proof here)

The zeta ζ transform and the Möbius μ transform

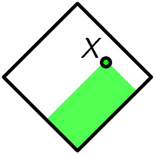
We consider functions from subsets of a finite set V to some ring – for simplicity let us fix the ring $(\mathbb{Z}, +, \cdot)$.

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

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 $f(X) = \mu\zeta f(X)$.

Intuition why it is useful

- Assume we want to compute $f(X)$ efficiently, but we do not know how to do it.
- Say that we can compute $(\zeta f)(Y)$ for all $Y \subseteq X$ efficiently. So we compute, and we get the function $g = \zeta f$...
- ... and we compute $\mu g(X)$ in $O^*(2^{|V|})$ time (say it is efficient).

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 $f(X) = \mu\zeta f(X)$.

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 $f(V) = \mu\zeta f(V)$ in $O^*(2^n)$ time and polynomial space.

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

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

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 $O^*(2^n)$ 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:

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

Consider fixing the last $n - j$ bits:

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

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

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

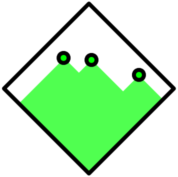
Fast zeta transform trimmed from above

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

DP in subset notation

$$\zeta_j(X) = \begin{cases} \zeta_{j-1}(X) & \text{when } j \notin X, \\ \zeta_{j-1}(X) + \zeta_{j-1}(X - \{j\}) & \text{when } j \in X. \end{cases}$$

If we need to find $\zeta(X)$ only for $X \in \mathcal{G}$ for some $\mathcal{G} \subset 2^V$, it suffices to compute $\zeta_j(X)$ only for $X \in \downarrow\mathcal{G}$;



Lower closure

$\downarrow\mathcal{G} = \{Y \subseteq V \ : \ \text{for some } X \in \mathcal{G},\ Y \subseteq X\}$.

Corollary (Björklund, Husfeldt, Kaski, Koivisto)

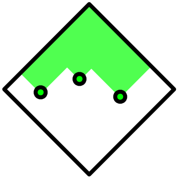
If we store the values of $\zeta_j(X)$ for $X \subseteq \downarrow\mathcal{G}$ in a dictionary, all values of $(\zeta f)(X)$ for $X \in \mathcal{G}$ can be computed in $O^*(|\downarrow\mathcal{G}|)$ time. Similarly for μf.

Fast zeta transform trimmed from below

Support, upper closure

For $f : 2^V \to \mathbb{Z}$ and $\mathcal{F} \subseteq 2^V$ define

- $\mathrm{supp}(f) = \{X \subseteq V \ : \ f(X) \neq 0\}$,
- $\uparrow\mathcal{F} = \{Y \subseteq V \ : \text{ for some } X \in \mathcal{F},\ X \subseteq Y\}$.



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

Observation

- $\mathrm{supp}(\zeta f) \subseteq \uparrow\mathrm{supp}(f)$.
- $\mathrm{supp}(\zeta_j f) \subseteq \ \mathrm{supp}(\zeta f) \subseteq \uparrow\mathrm{supp}(f)$.

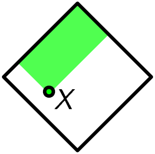
Corollary (Björklund, Husfeldt, Kaski, Koivisto)

If we store only the nonzero values of $\zeta_j(X)$ in a dictionary, **all the values** of ζf can be computed in $O^*(|\uparrow\mathrm{supp}(f)|)$ time. Similarly for μf.

Up-zeta transform $\zeta^{\uparrow}$

Definition

$(\zeta^{\uparrow} f)(X) = \sum_{Y \supseteq X} f(Y)$.



Trimmed up-zeta transform (Björklund, Husfeldt, Kaski, Koivisto)

- **(Trimming from above)** For any set family $\mathcal{G} \subseteq 2^V$ we can compute **all values** of $\zeta^{\uparrow} f|_{\mathcal{G}}$ in $O^*(|\uparrow\mathcal{G}|)$ time.
- **(Trimming from below)** We can compute **all the values** of $\zeta^{\uparrow} f$ in $O^*(|\downarrow \mathrm{supp}(f)|)$ time.

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 $\bigcup_{j=1}^{k} I_j \subseteq X$.
- As before, all 2^n values of ζf can be found in $O^*(2^n)$ 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 $\bigcup_{j=1}^{k} I_j \subseteq X$.
- As before, all 2^n values of ζf can be found in $O^*(2^n)$ 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 $\bigcup_{j=1}^{k} I_j \subseteq X$.
- As before, all 2^n values of ζf can be found in $O^*(2^n)$ 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 $O^*(2^n)$ 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 $Y \subseteq V$,

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

Why do we define it? Because it is natural. Besides, e.g.:

Let $\mathcal{F}$ be the family of all independent sets in a given graph G. Let $\mathbf{1}_{\mathcal{F}} : 2^V \to \{0,1\}$ be the characteristic function of $\mathcal{F}$, i.e. $\mathbf{1}_{\mathcal{F}}(X) = [X \in \mathcal{F}]$.

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 $Y \subseteq V$,

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

Why do we define it? Because it is natural. Besides, e.g.:

Let $\mathcal{F}$ be the family of all independent sets in a given graph G. Let $\mathbf{1}_{\mathcal{F}} : 2^V \to \{0,1\}$ be the characteristic function of $\mathcal{F}$, i.e. $\mathbf{1}_{\mathcal{F}}(X) = [X \in \mathcal{F}]$. Then

$$\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V) \neq 0 \text{ iff } G \text{ is } k\text{-colorable.}$$

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 $(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$. We use the Yates' algorithm 3x and we get $O^*(2^n)$ time (and space).

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 $(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$. We use the Yates' algorithm 3x and we get $O^*(2^n)$ time (and space).

Corollary

In order to compute $\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V)$ it suffices to perform $O(\log k)$ such operations. Hence we obtain yet another algorithm which finds all k-colorable induced subgraphs in $O^*(2^n)$ 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...

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

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...

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

Why do we define it? Because it is natural. Besides, e.g.:

if $k = \chi(G)$ then $\underbrace{\mathbf{1}_{\mathcal{F}} * \mathbf{1}_{\mathcal{F}} * \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V)$ is the number of k-colorings of G.

Computing the subset convolution

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

$$f_k(S) = f(S) \cdot [|S| = k].$$

Computing the subset convolution

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

$$f_k(S) = f(S) \cdot [|S| = k].$$

Then

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

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

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

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

Corollary

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

Coloring below the 2^n barrier: the bounded degree case

Observation

Let $\mathcal{F}$ be the family of (inclusion-wise) **maximal** independent sets.

$$\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V) \neq 0 \text{ iff } G \text{ is } k\text{-colorable.}$$

Coloring below the 2^n barrier: the bounded degree case

Observation

Let $\mathcal{F}$ be the family of (inclusion-wise) **maximal** independent sets.

$$\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V) \neq 0 \text{ iff } G \text{ is } k\text{-colorable.}$$

Denote $\mathbf{1}_{\mathcal{F}}^r = \underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{r \text{ times}}.$

Recall

$(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$, so
$(\mathbf{1}_{\mathcal{F}}^r *_c \mathbf{1}_{\mathcal{F}}^s)(Y) = \mu((\zeta \mathbf{1}_{\mathcal{F}}^r(Y))(\zeta \mathbf{1}_{\mathcal{F}}^s(Y)))$.

Coloring below the 2^n barrier: the bounded degree case

Observation

Let $\mathcal{F}$ be the family of (inclusion-wise) **maximal** independent sets.

$$\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V) \neq 0 \text{ iff } G \text{ is } k\text{-colorable.}$$

Denote $\mathbf{1}^r_{\mathcal{F}} = \underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{r \text{ times}}$.

Recall

$(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$, so
$(\mathbf{1}^r_{\mathcal{F}} *_c \mathbf{1}^s_{\mathcal{F}})(Y) = \mu((\zeta \mathbf{1}^r_{\mathcal{F}}(Y))(\zeta \mathbf{1}^s_{\mathcal{F}}(Y)))$.

- $\text{supp}\mathbf{1}_{\mathcal{F}} = \mathcal{F}$,

Observation

Let $\mathcal{F}$ be the family of (inclusion-wise) **maximal** independent sets.

$$\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V) \neq 0 \text{ iff } G \text{ is } k\text{-colorable.}$$

Denote $\mathbf{1}_{\mathcal{F}}^r = \underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{r \text{ times}}$.

Recall

$(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$, so
$(\mathbf{1}_{\mathcal{F}}^r *_c \mathbf{1}_{\mathcal{F}}^s)(Y) = \mu((\zeta \mathbf{1}_{\mathcal{F}}^r(Y))(\zeta \mathbf{1}_{\mathcal{F}}^s(Y)))$.

- $\text{supp} \mathbf{1}_{\mathcal{F}} = \mathcal{F}$,
- $\text{supp} \mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}} \subseteq \uparrow \text{supp} \mathbf{1}_{\mathcal{F}} = \uparrow \mathcal{F}$,

Coloring below the 2^n barrier: the bounded degree case

Observation

Let $\mathcal{F}$ be the family of (inclusion-wise) **maximal** independent sets.

$$\underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}(V) \neq 0 \text{ iff } G \text{ is } k\text{-colorable.}$$

Denote $\mathbf{1}^r_{\mathcal{F}} = \underbrace{\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{r \text{ times}}$.

Recall

$(f *_c g)(Y) = \mu((\zeta f(Y))(\zeta g(Y)))$, so
$(\mathbf{1}^r_{\mathcal{F}} *_c \mathbf{1}^s_{\mathcal{F}})(Y) = \mu((\zeta \mathbf{1}^r_{\mathcal{F}}(Y))(\zeta \mathbf{1}^s_{\mathcal{F}}(Y)))$.

- $\text{supp}\mathbf{1}_{\mathcal{F}} = \mathcal{F}$,
- $\text{supp}\mathbf{1}_{\mathcal{F}} *_c \mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}} \subseteq \uparrow\text{supp}\mathbf{1}_{\mathcal{F}} = \uparrow\mathcal{F}$,
- **Corollary:** One can compute $\underbrace{\mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}$ in $O^*(|\uparrow\mathcal{F}|)$ time.

Coloring below the 2^n barrier: the bounded degree case

Corollary

One can compute $\underbrace{\mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}$ in $O^*(|\uparrow\mathcal{F}|)$ time and space.

Coloring below the 2^n barrier: the bounded degree case

Corollary

One can compute $\underbrace{\mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}$ in $O^*(|\uparrow\mathcal{F}|)$ time and space.

Theorem (Björklund, Husfeldt, Kaski, Koivisto 2008)

In any n-vertex graph of maximum degree Δ there are $\leq (2^{\Delta+1} - 1)^{n/(\Delta+1)}$ dominating sets.

Coloring below the 2^n barrier: the bounded degree case

Corollary

One can compute $\underbrace{\mathbf{1}_{\mathcal{F}} *_c \cdots \mathbf{1}_{\mathcal{F}}}_{k \text{ times}}$ in $O^*(|\uparrow\mathcal{F}|)$ time and space.

Theorem (Björklund, Husfeldt, Kaski, Koivisto 2008)

In any n-vertex graph of maximum degree Δ there are $\leq (2^{\Delta+1} - 1)^{n/(\Delta+1)}$ dominating sets.

Aaaaha!

But $\uparrow\mathcal{F}$ contains only dominating sets!

Corollary (Björklund, Husfeldt, Kaski, Koivisto 2008)

One can find a k-coloring of a graph of maximum degree Δ in $O^*((2^{\Delta+1} - 1)^{n/(\Delta+1)})$ time.

Coloring below the 2^n barrier: the bounded degree case

Theorem (Björklund, Husfeldt, Kaski, Koivisto 2008)

One can find a k-coloring of a graph of maximum degree Δ in $O^*((2^{\Delta+1} - \Delta - 1)^{n/(\Delta+1)})$ time.

Δ	$(2^{\Delta+1} - \Delta - 1)^{n/(\Delta+1)}$
3	1.86121
4	1.93318
5	1.96745
6	1.98400
7	1.99208
8	1.99606
9	1.99804
10	1.99902
11	1.99951
12	1.99976