Algebraic techniques in parameterized algorithms, Part II: Polynomials over finite fields of characteristic two

Łukasz Kowalik

University of Warsaw

FPT School, Bedlewo, August 2014

Field

Field is a triple $(F, +, \cdot)$, where

- F is a set, $+$ and $\cdot$ are binary operations
- associativity: $(a+b)+c = a+(b+c)$, $\quad (a \cdot b) \cdot c = a \cdot (b \cdot c)$
- commutativity: $a+b = b+a$, $\quad a \cdot b = b \cdot a$
- distributivity: $a \cdot (b+c) = a \cdot b + a \cdot c$.
- additive identity: $\exists\ 0 \in F$ s.t. $0 + a = a$.
- multiplicative identity: $\exists\ 1 \in F$ s.t. $\forall a \in F \setminus \{0\}: \ 1 \cdot a = a$.
- additive inverses: $\forall\ a \in F\ \exists b \in F$ s.t. $a + b = 0$;
- multiplicative inverses: $\forall\ a \in F \setminus \{0\}\ \exists\ b \in F$ s.t. $a \cdot b = 1$;

Some familiar (infinite) fields: $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$.

Finite fields of characteristic 2

In what follows, we use finite fields of size $|F| = 2^\ell$.
We need to know just three things about such fields:

- They exist (for every $\ell \in \mathbb{N}$),
- We can perform arithmetic operations fast, in $O(\log |F|(\log \log |F|)^2)$ time,
- They are of characteristic two, i.e. $1 + 1 = 0$.
  In particular, for any element a, we have

$$a + a = a \cdot (1 + 1) = a \cdot 0 = 0$$

Multivariate polynomials

Fix a field F.

Monomial

Monomial is an expression of the form $m = a x_1^{c_1} x_2^{c_2} \cdots x_n^{c_n}$, where

- $a \in F$
- $x_1, \ldots, x_n$ are variables
- $c_1, \ldots, c_n \in \mathbb{N} \cup \{0\}$.

Degree of m is $\sum_{i=1}^{n} c_i$.

Examples:

$$5x_2^3 x_3^7 \qquad \text{(degree 10)},$$

$$x_1 x_2 x_3 \cdots x_{2014} \qquad \text{(degree 2014)},$$

Multivariate polynomials

Polynomial

A **polynomial** is a finite sum of monomials.

Multivariate polynomials

Polynomial

A **polynomial** is a finite sum of monomials.

Polynomial (more precisely)

A **polynomial** an expression of the form

$$p = \sum_{(c_1,\ldots,c_n)\in(\mathbb{N}\cup\{0\})^n} a_{c_1,\ldots,c_n} x_1^{c_1} x_2^{c_2} \cdots x_n^{c_n}, \tag{1}$$

where the coefficients $a_{c_1,\ldots,c_n}$ are non-zero only for a finite number of tuples $(c_1, \ldots, c_n)$.

Degree of p is the maximum degree of its monomials.

Multivariate polynomials

Examples:

$$x_1 + x_2 \qquad \text{(degree 1)},$$

$$x_1^3 x_2 + 5x_2^3 x_3^7 - x_1 x_2^5 x_3 \qquad \text{(degree 10)},$$

$$x_1 x_2 x_3 \cdots x_{2014} \qquad \text{(degree 2014)},$$

$$0 \qquad \text{(the zero polynomial)}.$$

The Schwartz-Zippel Lemma

Lemma [DeMillo and Lipton 1978, Zippel 1979, Schwartz 1980]

Let $p(x_1, x_2, \ldots, x_n)$ be a non-zero polynomial of degree at most d over a field F and let S be a finite subset of F. Sample values $a_1, a_2, \ldots, a_n$ from S uniformly at random. Then,

$$Pr[p(a_1, a_2, \ldots, a_n)] = 0] \leq d/|S|.$$

The Schwartz-Zippel Lemma

Lemma [DeMillo and Lipton 1978, Zippel 1979, Schwartz 1980]

Let $p(x_1, x_2, \ldots, x_n)$ be a non-zero polynomial of degree at most d over a field F and let S be a finite subset of F. Sample values $a_1, a_2, \ldots, a_n$ from S uniformly at random. Then,

$$Pr[p(a_1, a_2, \ldots, a_n)] = 0] \leq d/|S|.$$

A typical application

- We can efficiently **evaluate** a polynomial p of degree d.
- We want to test whether p is a non-zero polynomial.
- Then, we pick S so that $|S| \geq 2d$ and we evaluate p on a random vector $\mathbf{x} \in S^n$. We answer YES iff we got $p(\mathbf{x}) \neq 0$.
- If p is the zero polynomial we always get NO, otherwise we get YES with probability at least $\frac{1}{2}$.
- This is called a Monte-Carlo algorithm with one-sided error.

Schwartz-Zippel Lemma: Take-home message

Message

We can test whether a polynomial P is non-zero by a **single evaluation** of P in a random vector.

The Schwartz-Zippel Lemma: Example

Polynomial equality testing

INPUT: Two multivariate polynomials P, Q given as an arithmetic circuit.
QUESTION: Does $P \equiv Q$?

Note: A polynomial described by an arithmetic circuit of size s can have $2^{\Omega(s)}$ different monomials: $(x_1 + x_2)(x_1 - x_3)(x_2 + x_4) \cdots$.

Solution

The Schwartz-Zippel Lemma: Example

Polynomial equality testing

INPUT: Two multivariate polynomials P, Q given as an arithmetic circuit.
QUESTION: Does $P \equiv Q$?

Note: A polynomial described by an arithmetic circuit of size s can have $2^{\Omega(s)}$ different monomials: $(x_1 + x_2)(x_1 - x_3)(x_2 + x_4) \cdots$.

Solution

Test whether the polynomial $P - Q$ is non-zero using the Schwartz-Zippel Lemma.

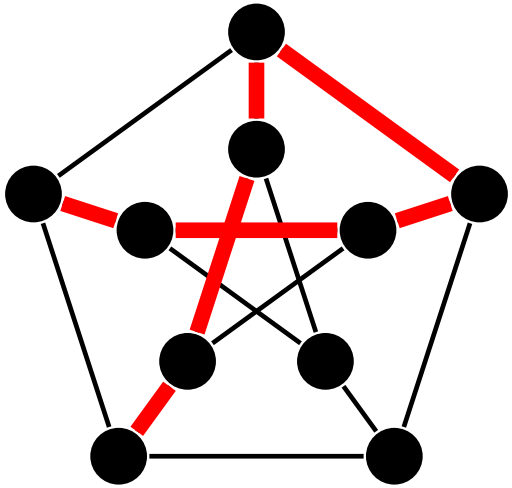
Theorem

Polynomial equality testing for two polynomials represented by circuits of size at most s can be solved in $O(s)$ time with a Monte Carlo algorithm with one-sided error probability bounded by $1/2$.

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?



Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$
- Alon, Yuster, Zwick 1994: $O((2e)^k n^{O(1)})$ (color coding $\rightarrow$ Tuesday)

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$
- Alon, Yuster, Zwick 1994: $O((2e)^k n^{O(1)})$ (color coding $\rightarrow$ Tuesday)
- Kneis et al. 2006, Chen et al. 2007: $O(4^k n^{O(1)})$
  (divide-and-color $\rightarrow$ book)

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$
- Alon, Yuster, Zwick 1994: $O((2e)^k n^{O(1)})$ (color coding $\rightarrow$ Tuesday)
- Kneis et al. 2006, Chen et al. 2007: $O(4^k n^{O(1)})$ (divide-and-color $\rightarrow$ book)
- Koutis 2008: $O(2^{3/2k} n^{O(1)})$ (group algebras $\rightarrow$ Friday)

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$
- Alon, Yuster, Zwick 1994: $O((2e)^k n^{O(1)})$ (color coding $\rightarrow$ Tuesday)
- Kneis et al. 2006, Chen et al. 2007: $O(4^k n^{O(1)})$ (divide-and-color $\rightarrow$ book)
- Koutis 2008: $O(2^{3/2k} n^{O(1)})$ (group algebras $\rightarrow$ Friday)
- Williams 2009: $O(2^k n^{O(1)})$ (group algebras $\rightarrow$ Friday)

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$
- Alon, Yuster, Zwick 1994: $O((2e)^k n^{O(1)})$ (color coding $\rightarrow$ Tuesday)
- Kneis et al. 2006, Chen et al. 2007: $O(4^k n^{O(1)})$
  (divide-and-color $\rightarrow$ book)
- Koutis 2008: $O(2^{3/2k} n^{O(1)})$ (group algebras $\rightarrow$ Friday)
- Williams 2009: $O(2^k n^{O(1)})$ (group algebras $\rightarrow$ Friday)
- Björklund 2010: $O(1.66^n n^{O(1)})$, undirected Hamiltonian cycle ($k = n$)
  (polynomials over finite fields of characteristic two)

Longest Path problem

Problem

Input: directed/undirected graph G, integer k.
Question: Does G contain a k-vertex path (shortly: k-path)?

Progress

- Monien 1985: $O(k!n^{O(1)})$
- Alon, Yuster, Zwick 1994: $O((2e)^k n^{O(1)})$ (color coding $\rightarrow$ Tuesday)
- Kneis et al. 2006, Chen et al. 2007: $O(4^k n^{O(1)})$
  (divide-and-color $\rightarrow$ book)
- Koutis 2008: $O(2^{3/2k} n^{O(1)})$ (group algebras $\rightarrow$ Friday)
- Williams 2009: $O(2^k n^{O(1)})$ (group algebras $\rightarrow$ Friday)
- Björklund 2010: $O(1.66^n n^{O(1)})$, undirected Hamiltonian cycle ($k = n$)
  (polynomials over finite fields of characteristic two)
- Björklund, Husfeldt, Kaski, Koivisto 2010: $O(1.66^k n^{O(1)})$, undirected

Longest Path in time $O(2^k k|E|)$

Notation

$$[k] = \{1, \ldots, k\}$$

$2^k(kn)^{O(1)}$-time algorithm for Longest Path

Rough idea

- Want to construct a polynomial P, $P \not\equiv 0$ iff G has a k-path.

$2^k(kn)^{O(1)}$-time algorithm for Longest Path

Rough idea

- Want to construct a polynomial P, $P \not\equiv 0$ iff G has a k-path.
- First try: $P(\cdots) = \sum_{k\text{-path } R \text{ in } G} \text{monomial}(R)$.

  Seems good, but how to evaluate it?

$2^k(kn)^{O(1)}$-time algorithm for Longest Path

Rough idea

- Want to construct a polynomial P, $P \not\equiv 0$ iff G has a k-path.
- First try: $P(\cdots) = \sum_{k\text{-path } R \text{ in } G} \text{monomial}(R)$.

  Seems good, but how to evaluate it?
- Second try: $P(\cdots) = \sum_{k\text{-walk } W \text{ in } G} \text{monomial}(W)$.

  Now we **can** evaluate it but we may get false positives.

$2^k(kn)^{O(1)}$-time algorithm for Longest Path

Rough idea

- Want to construct a polynomial P, $P \not\equiv 0$ iff G has a k-path.
- First try: $P(\cdots) = \sum_{k\text{-path } R \text{ in } G} \text{monomial}(R)$.

  Seems good, but how to evaluate it?
- Second try: $P(\cdots) = \sum_{k\text{-walk } W \text{ in } G} \text{monomial}(W)$.

  Now we **can** evaluate it but we may get false positives.
- Final try: $P(\cdots) = \sum_{k\text{-walk } W \text{ in } G} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{monomial}(w, \ell)$.
  - We still can evaluate it,
  - It turns out that every monomial corresponding to a walk which is not a path appears an even number of times so it cancels-out!

Our Hero

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}}_{\text{mon}_{W,\ell}}$$

Variables:

- a variable x_e for every $e \in E$,
- a variable $y_{v,\ell}$ for every $v \in V$ and $\ell \in [k]$.

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).
- We define $\ell' : [k] \to [k]$ as follows:

$$\ell'(x) = \begin{cases} \ell(b) & \text{if } x = a, \\ \ell(a) & \text{if } x = b, \\ \ell(x) & \text{otherwise.} \end{cases}$$

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).
- We define $\ell' : [k] \to [k]$ as follows:

$$\ell'(x) = \begin{cases} \ell(b) & \text{if } x = a, \\ \ell(a) & \text{if } x = b, \\ \ell(x) & \text{otherwise.} \end{cases}$$

- $(W, \ell) \neq (W, \ell')$ since ℓ is injective.

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).
- We define $\ell' : [k] \to [k]$ as follows:

$$\ell'(x) = \begin{cases} \ell(b) & \text{if } x = a, \\ \ell(a) & \text{if } x = b, \\ \ell(x) & \text{otherwise.} \end{cases}$$

- $(W, \ell) \neq (W, \ell')$ since ℓ is injective.
- $\mathrm{mon}_{W,\ell} = \prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)} =$

$$\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i \in [k] \setminus \{a,b\}} y_{v_i,\ell(i)} \underbrace{y_{v_a,\ell(a)}}_{y_{v_b\ell'(b)}} \underbrace{y_{v_b,\ell(b)}}_{y_{v_a\ell'(a)}} = \mathrm{mon}_{W,\ell'}$$

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).
- We define $\ell' : [k] \to [k]$ as follows:

$$\ell'(x) = \begin{cases} \ell(b) & \text{if } x = a, \\ \ell(a) & \text{if } x = b, \\ \ell(x) & \text{otherwise.} \end{cases}$$

- $(W, \ell) \neq (W, \ell')$ since ℓ is injective.
- $\mathrm{mon}_{W,\ell} = \mathrm{mon}_{W,\ell'}$

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).
- We define $\ell' : [k] \to [k]$ as follows:

$$\ell'(x) = \begin{cases} \ell(b) & \text{if } x = a, \\ \ell(a) & \text{if } x = b, \\ \ell(x) & \text{otherwise.} \end{cases}$$

- $(W, \ell) \neq (W, \ell')$ since ℓ is injective.
- $\mathrm{mon}_{W,\ell} = \mathrm{mon}_{W,\ell'}$
- If we start from (W, ℓ') and follow the same way of assignment we get (W, ℓ) back. (This is called a fixed-point free involution)

Monomials corresponding to non-path walks cancel-out

- Let $W = v_1, \ldots, v_k$ be a walk, and a bijection $\ell : [k] \to [k]$.
- Assume $v_a = v_b$ for some $a < b$, if many such pairs take the lexicographically first pair (a, b).
- We define $\ell' : [k] \to [k]$ as follows:

$$\ell'(x) = \begin{cases} \ell(b) & \text{if } x = a, \\ \ell(a) & \text{if } x = b, \\ \ell(x) & \text{otherwise.} \end{cases}$$

- $(W, \ell) \neq (W, \ell')$ since ℓ is injective.
- $\mathrm{mon}_{W,\ell} = \mathrm{mon}_{W,\ell'}$
- If we start from (W, ℓ') and follow the same way of assignment we get (W, ℓ) back. (This is called a fixed-point free involution)
- Since the field is of characteristic 2, $\mathrm{mon}_{W,\ell}$ and $\mathrm{mon}_{W,\ell'}$ cancel out!

Half the way...

Corollary

If $P \not\equiv 0$ then there is a k-path.

The second half

Recall:

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}}_{\text{mon}_{W,\ell}}$$

Question

Why do we need exactly $\text{mon}_{W,\ell} = \prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}$?

What if, say, $\text{mon}_{W,\ell} = \prod_{i=1}^{k} y_{v_i,\ell(i)}$?

The second half

Recall:

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}}_{\text{mon}_{W,\ell}}$$

Question

Why do we need exactly $\text{mon}_{W,\ell} = \prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}$?
What if, say, $\text{mon}_{W,\ell} = \prod_{i=1}^{k} y_{v_i,\ell(i)}$?

Answer

Now, every labelled walk which is a path gets a **unique** monomial.

The second half

Recall:

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}}_{\text{mon}_{W,\ell}}$$

Question

Why do we need exactly $\text{mon}_{W,\ell} = \prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}$?
What if, say, $\text{mon}_{W,\ell} = \prod_{i=1}^{k} y_{v_i,\ell(i)}$?

Answer

Now, every labelled walk which is a path gets a **unique** monomial.

Corollary

If there is a k-path in G then $P \not\equiv 0$.

Where are we?

Corollary

There is a k-path in G iff $P \not\equiv 0$.

The missing element

How to evaluate P efficiently?
($2^k(kn)^{O(1)}$ is efficiently enough.)

Weighted inclusion-exclusion

Let $A_1, \ldots, A_n \subseteq U$, where U is a finite set.
Let $w : U \to F$ be a weight function.
For any $X \subseteq U$ denote $w(X) = \sum_{x \in X} w(x)$.
Let us also denote $\bigcap_{i \in \emptyset}(U - A_i) = U$.

Then,

$$w\left(\bigcap_{i \in [n]} A_i\right) = \sum_{X \subseteq [n]} (-1)^{|X|} w\left(\bigcap_{i \in X}(U - A_i)\right).$$

Weighted inclusion-exclusion

Let $A_1, \ldots, A_n \subseteq U$, where U is a finite set.
Let $w : U \to F$ be a weight function.
For any $X \subseteq U$ denote $w(X) = \sum_{x \in X} w(x)$.
Let us also denote $\bigcap_{i \in \emptyset}(U - A_i) = U$.

Then,

$$w\left(\bigcap_{i \in [n]} A_i\right) = \sum_{X \subseteq [n]} (-1)^{|X|} w\left(\bigcap_{i \in X}(U - A_i)\right).$$

Counting over a field of characteristic 2 we know that $-1 = 1$ so we can remove the $(-1)^{|X|}$:

$$w\left(\bigcap_{i \in [n]} A_i\right) = \sum_{X \subseteq [n]} w\left(\bigcap_{i \in X}(U - A_i)\right).$$

Evaluating $P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$

Fix a walk W.

- $U = \{\ell : [k] \to [k]\}$ (all functions)
- for $\ell \in U$, define the weight $w(\ell) = \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$.
- for $i = 1, \ldots, k$ let $A_i = \{\ell \in U \ : \ \ell^{-1}(i) \neq \emptyset\}$.
- Then,

$$\sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is surjective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{\ell \in \bigcap_{i=1}^{k} A_i} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = w(\bigcap_{i=1}^{k} A_i).$$

- By weighted I-E,

$$\sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is surjective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{X \subseteq [k]} w\left(\bigcap_{i \in X} (U - A_i)\right) =$$

$$\sum_{X \subseteq [k]} \sum_{\ell:[k]\to[k]\setminus X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$$

Evaluating $P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$

Fix a walk W.

- $U = \{\ell : [k] \to [k]\}$ (all functions)
- for $\ell \in U$, define the weight $w(\ell) = \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$.
- for $i = 1, \ldots, k$ let $A_i = \{\ell \in U \ : \ \ell^{-1}(i) \neq \emptyset\}$.
- Then,

$$\sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is surjective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{\ell \in \bigcap_{i=1}^{k} A_i} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = w(\bigcap_{i=1}^{k} A_i).$$

- By weighted I-E,

$$\sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is surjective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{X \subseteq [k]} w\left(\bigcap_{i \in X} (U - A_i)\right) =$$

$$\sum_{X \subseteq [k]} \sum_{\ell:[k]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$$

Evaluating $P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$

We got

$$\sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{X\subseteq[k]} \sum_{\ell:[k]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$$

Hence,

$$\begin{aligned} P(\mathbf{x},\mathbf{y}) &= \sum_{\text{walk } W} \sum_{X\subseteq[k]} \sum_{\ell:[k]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) \\ &= \sum_{X\subseteq[k]} \underbrace{\sum_{\text{walk } W} \sum_{\ell:[k]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})}_{P_X(\mathbf{x},\mathbf{y})} \end{aligned}$$

Evaluating $P_X(\mathbf{x}, \mathbf{y}) = \sum_{\substack{\text{walk } W \\ \text{of length } k}} \sum_{\ell:[k]\to X} \text{mon}_{W,\ell}(\mathbf{x}, \mathbf{y})$ in $n^{O(1)}$

We use dynamic programming. (How?)

Evaluating $P_X(\mathbf{x},\mathbf{y}) = \sum_{\substack{\text{walk } W \\ \text{of length } k}} \sum_{\ell:[k]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$ in $n^{O(1)}$

We use dynamic programming. (How?)
Fill the 2-dimensional table T,

$$T[v,d] = \sum_{\substack{\text{walk } W = v_1,\ldots,v_d \\ v_1 = v}} \sum_{\ell:[k]\to X} \prod_{i=1}^{d-1} x_{v_i,v_{i+1}} \prod_{i=1}^{d} y_{v_i,\ell(i)}$$

Then,

$$T[v,d] = \begin{cases} \sum_{l\in X} y_{vl} & \text{when } d = 1, \\ \sum_{l\in X} y_{vl} \sum_{(v,w)\in E} x_{vw}\cdot T[w,d-1] & \text{otherwise.} \end{cases}$$

Hence, $P_X(\mathbf{x},\mathbf{y}) = \sum_{s\in V} T[s,k]$ can be computed in $O(k|E|)$ time.

Cost of arithmetic

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k]\to[k] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k} y_{v_i,\ell(i)}}_{\text{mon}_{W,\ell}}$$

- Degree of P is $2k-1$.
- Pick a field F of size $2^{\lceil \log(4k) \rceil} > 4k$.
- Sample values of variables from set $S = F$.
- By Schwartz-Zippel Lemma, false-negatives with probability at most $(2k-1)/(4k) \leq 1/2$
- Arithmetic in $O(\log k (\log\log k)^2)$ time (cheap!).

Conclusion

Corollary

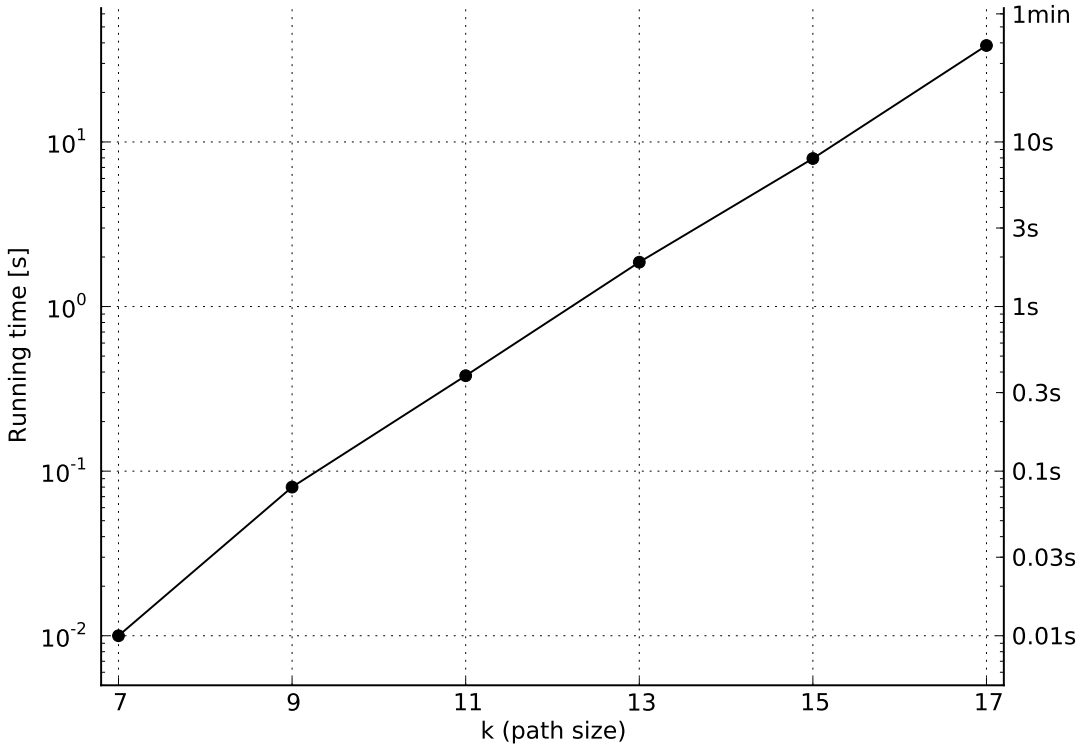
Longest Path can be solved by a $O(2^k k \log k (\log \log k)^2 |E|)$-time polynomial space one-sided error Monte-Carlo algorithm.

Conclusion

Corollary

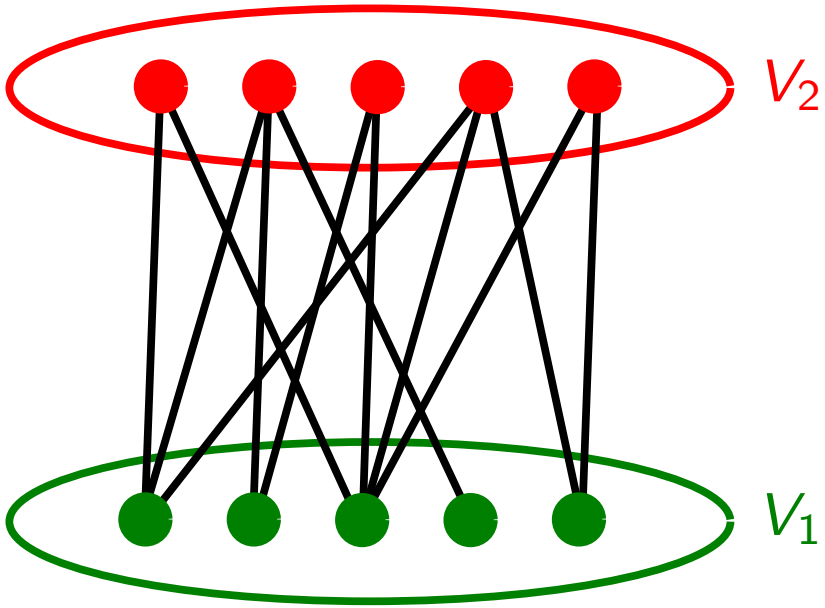
LONGEST PATH can be solved by a $O(2^k k \log k (\log \log k)^2 |E|)$-time polynomial space one-sided error Monte-Carlo algorithm.

Finding k-paths in a 1000-vertex graph on a 2.53-GHz Intel Xeon CPU:



Longest Path in undirected bipartite graphs in $2^{k/2}(kn)^{O(1)}$ time

Longest Path in undirected bipartite graphs in $2^{k/2}(kn)^{O(1)}$ time



A new hero

Idea

Label vertices of V_1 only.

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k/2]\to[k/2]\\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1},\ell(i)}}_{\text{mon}_{W,\ell}}$$

Variables:

- a variable x_e for every $e \in E$ ($x_{uv} = x_{vu}$),
- a variable $y_{v,\ell}$ for every $v \in V_1$ and $\ell \in [k/2]$.

Checking the hero

$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k/2]\to[k/2]\\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1},\ell(i)}}_{\text{mon}_{W,\ell}}$$

Paths do not cancel-out

If there is a k-path with an endpoint in V_1 then $P \not\equiv 0$.
(Proof: We can recover (W,ℓ) from $\text{mon}_{W,\ell}$ as before.)

Checking the hero

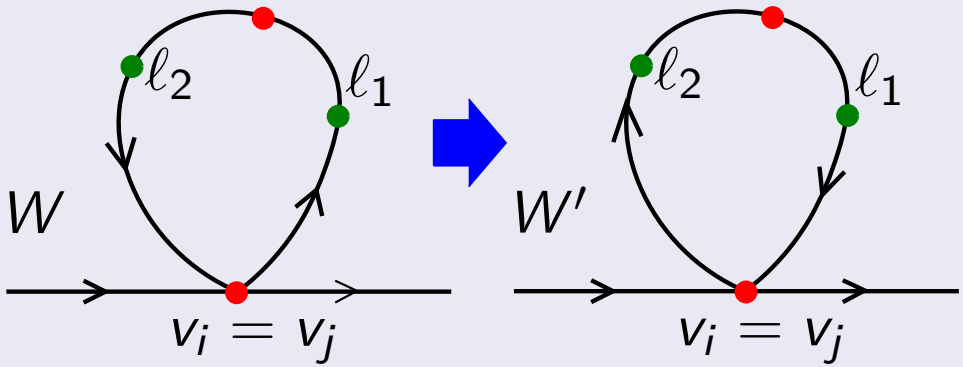
$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k/2]\to[k/2]\\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1},\ell(i)}}_{\text{mon}_{W,\ell}}$$

Do non-path walks cancel-out?

Consider a non-path labelled walk (W,ℓ), $W = v_1,\ldots,v_k$.

Case 1 If exist i,j, $i<j$ s.t. $v_i = v_j$, $v_i \in V_1$:
pick lexicographically first such pair;
both v_i and v_j have labels so we **swap labels** as before.

Case 2 As in Case 1, but $v_i \in V_2$ and Case 1 does not occur:
reverse the cycle:



- $\text{mon}_{W,\ell} = \text{mon}_{W',\ell'}$,
- from (W',ℓ') we get (W,ℓ),
- Does $(W,\ell) \neq (W',\ell')$?

Checking the hero

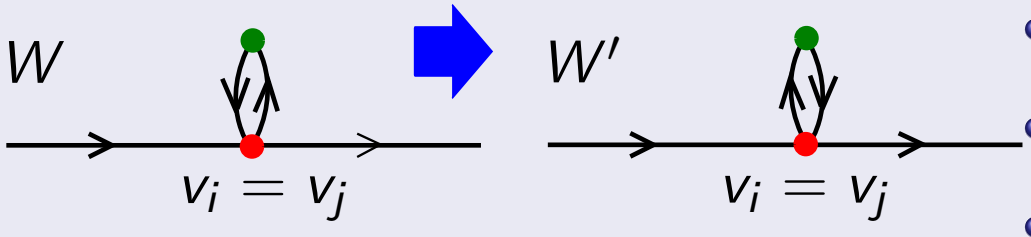
$$P(\mathbf{x},\mathbf{y}) = \sum_{\text{walk } W=v_1,\ldots,v_k} \sum_{\substack{\ell:[k/2]\to[k/2]\\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1},\ell(i)}}_{\text{mon}_{W,\ell}}$$

Do non-path walks cancel-out?

Consider a non-path labelled walk (W,ℓ), $W = v_1,\ldots,v_k$.

Case 1 If exist i,j, $i<j$ s.t. $v_i = v_j$, $v_i \in V_1$:
pick lexicographically first such pair;
both v_i and v_j have labels so we **swap labels** as before.

Case 2 As in Case 1, but $v_i \in V_2$ and Case 1 does not occur:
reverse the cycle:

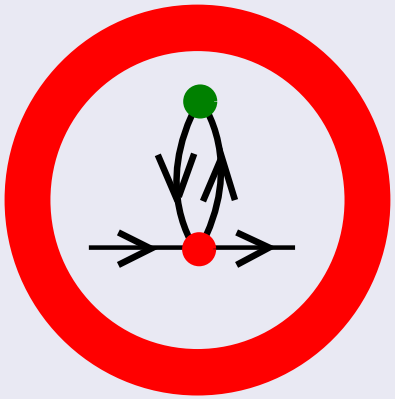


- $\text{mon}_{W,\ell} = \text{mon}_{W',\ell'}$,
- from (W',ℓ') we get (W,ℓ),
- Does $(W,\ell) \neq (W',\ell')$? NO!

Fixing the hero

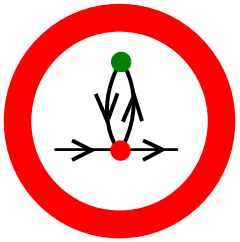
Admissible walks

Walk $v_1, \ldots, v_k$ is admissible if:
For every $i = 1, \ldots, k-2$, if $v_i \in V_2$ and $v_{i+1} \in V_1$ then $v_{i+2} \neq v_i$.



$$P(\mathbf{x}, \mathbf{y}) = \sum_{\substack{\text{walk } W = v_1, \ldots, v_k \\ W \text{ is admissible}}} \sum_{\substack{\ell:[k/2]\to[k/2] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i, v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1}, \ell(i)}}_{\text{mon}_{W,\ell}}$$

Checking the fixed hero



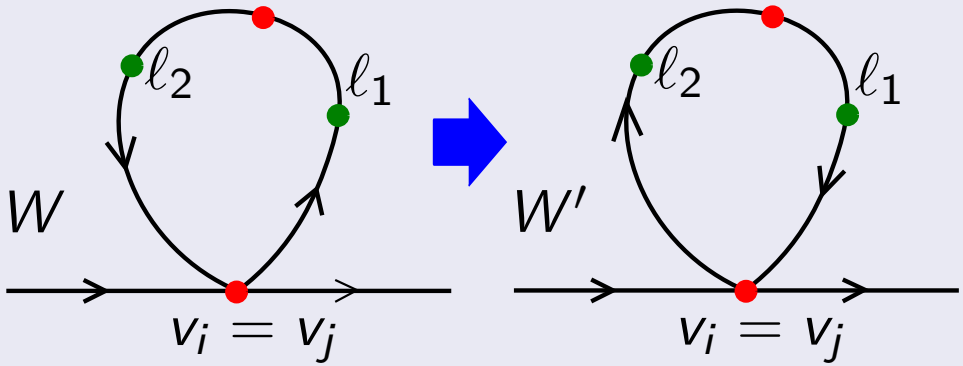
$$P(\mathbf{x},\mathbf{y}) = \sum_{\substack{\text{walk } W=v_1,\ldots,v_k \\ W \text{ is admissible}}} \sum_{\substack{\ell:[k/2]\to[k/2] \\ \ell \text{ is bijective}}} \underbrace{\prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1},\ell(i)}}_{\text{mon}_{W,\ell}}$$

Do non-path walks cancel-out?

Consider a non-path labelled walk (W,ℓ), $W = v_1,\ldots,v_k$.

Case 1 If exist i,j, $i<j$ s.t. $v_i = v_j$, $v_i \in V_1$:
pick lexicographically first such pair;
both v_i and v_j have labels so we **swap labels** as before.

Case 2 As in Case 1, but $v_i \in V_2$ and Case 1 does not occur:
reverse the cycle:



- $\text{mon}_{W,\ell} = \text{mon}_{W',\ell'}$,
- from (W',ℓ') we get (W,ℓ),
- $(W,\ell) \neq (W',\ell')$ because W admissible,
- W' is admissible.

Evaluating $P(\mathbf{x},\mathbf{y}) = \sum_{\text{admissible walk } W} \sum_{\substack{\ell:[k/2]\to[k/2] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$

As before, from inclusion-exclusion principle we can get

$$\sum_{\substack{\ell:[k/2]\to[k/2] \\ \ell \text{ is bijective}}} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) = \sum_{X\subseteq[k/2]} \sum_{\ell:[k/2]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})$$

Hence, as before:

$$\begin{aligned} P(\mathbf{x},\mathbf{y}) &= \sum_{\text{admissible walk } W} \sum_{X\subseteq[k/2]} \sum_{\ell:[k/2]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y}) \\ &= \sum_{X\subseteq[k/2]} \underbrace{\sum_{\text{admissible walk } W} \sum_{\ell:[k/2]\to X} \text{mon}_{W,\ell}(\mathbf{x},\mathbf{y})}_{P_X(\mathbf{x},\mathbf{y})} \end{aligned}$$

Note: Only $2^{k/2}$ polynomials P_X to evaluate.

Evaluating $P_X(\mathbf{x},\mathbf{y}) = \sum_{\substack{\text{admissible walk } W \\ \text{of length } k}} \sum_{\ell:[k/2]\to X} \text{mon}_{W,\ell}$ in poly-time

Dynamic programming:

$$T[v,w,d] = \sum_{\substack{\text{admissible walk } W=v_1,\ldots,v_d \\ v_1=v \\ v_2=w}} \sum_{\ell:[k/2]\to X} \prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{k/2} y_{v_{2i-1},\ell(i)}$$

Then,

$$T[v,w,d] = \begin{cases} x_{vw}\sum_{l\in X} y_{vl} & \text{when } d=2 \text{ and } v\in V_1, \\ x_{vw}\sum_{l\in X} y_{wl} & \text{when } d=2 \text{ and } v\in V_2, \\ \sum_{l\in X} y_{vl} \sum_{(w,u)\in E} x_{vw}\cdot T[w,u,d-1] & \text{when } d>2 \text{ and } v\in V_1, \\ \sum_{\substack{(w,u)\in E \\ u\neq v}} x_{vw}\cdot T[w,u,d-1] & \text{when } d>2 \text{ and } v\in V_2. \end{cases}$$

Conclusion

Theorem (Björklund, Husfeldt, Kaski, Koivisto 2010)

Longest Path in undirected bipartite graphs can be solved in $2^{k/2}(kn)^{O(1)} = 1.42^k(kn)^{O(1)}$ time and polynomial space.

Conclusion

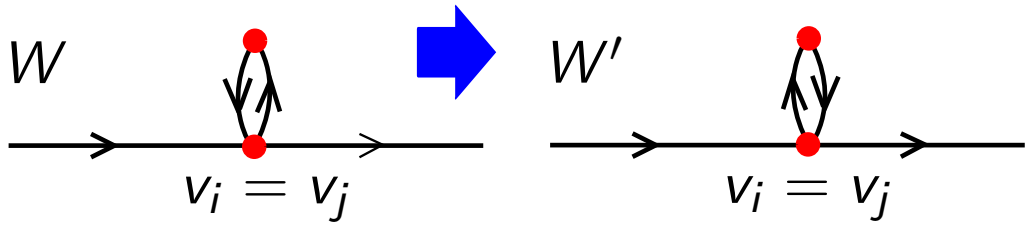
Theorem (Björklund, Husfeldt, Kaski, Koivisto 2010)

Longest Path in undirected bipartite graphs can be solved in $2^{k/2}(kn)^{O(1)} = 1.42^k(kn)^{O(1)}$ time and polynomial space.

Longest Path in undirected graphs in $2^{\frac{3}{4}k}(kn)^{O(1)}$ time

Longest Path in undirected graphs in $2^{\frac{3}{4}k}(kn)^{O(1)}$ time

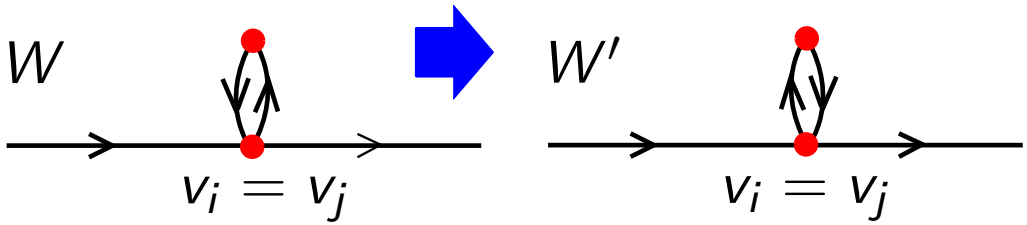
- Choose a **random bipartition** $V = V_1 \cup V_2$, $||V_1| - |V_2|| \leq 1$. (V_1 and V_2 need not be independent now.)
- Where does the bipartite case algorithm fail?



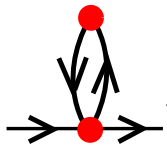
Then $(W, \ell) = (W', \ell')$.

Longest Path in undirected graphs in $2^{\frac{3}{4}k}(kn)^{O(1)}$ time

- Choose a **random bipartition** $V = V_1 \cup V_2$, $||V_1| - |V_2|| \leq 1$. (V_1 and V_2 need not be independent now.)
- Where does the bipartite case algorithm fail?

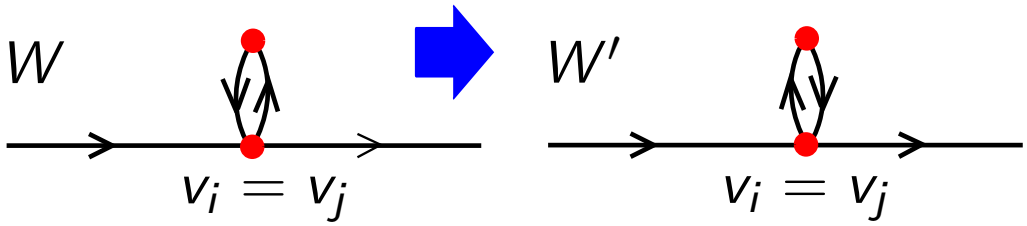


Then $(W, \ell) = (W', \ell')$.

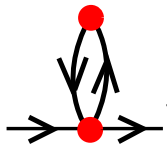
- What if we forbid also  ?

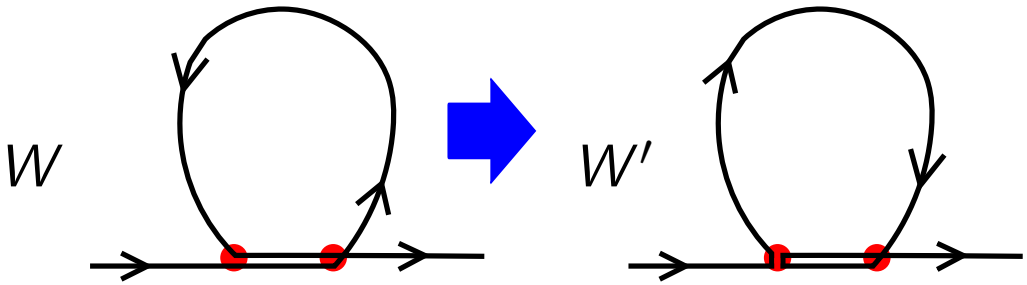
Longest Path in undirected graphs in $2^{\frac{3}{4}k}(kn)^{O(1)}$ time

- Choose a **random bipartition** $V = V_1 \cup V_2$, $||V_1| - |V_2|| \leq 1$. (V_1 and V_2 need not be independent now.)
- Where does the bipartite case algorithm fail?



Then $(W, \ell) = (W', \ell')$.

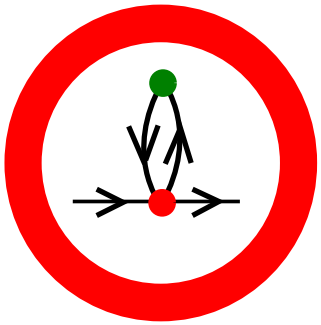
- What if we forbid also  ?
- Then we run into another trouble:



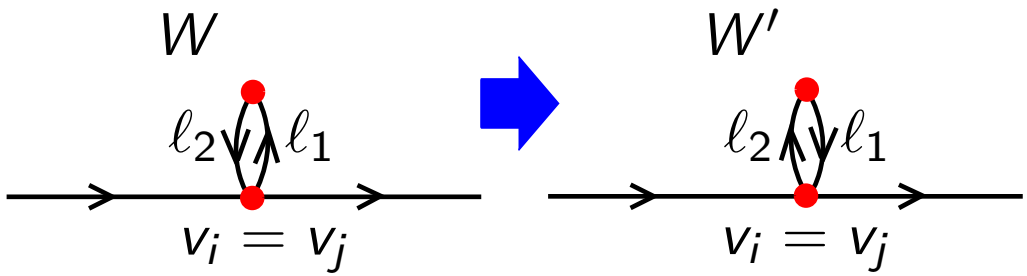
W' contains the forbidden configuration.

The solution

- Forbidden configuration as before:



- Add more labels:
  label each V_2V_2-edge:



Now $\ell' \neq \ell$.

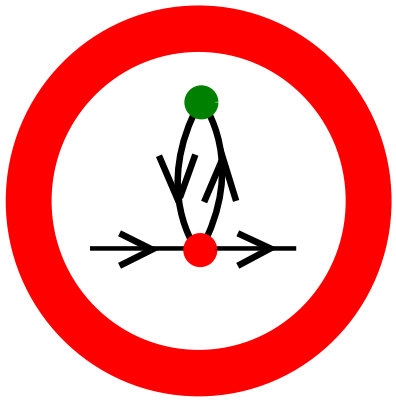
How many labels do we need now?

- a different label for each $i = 1, \ldots, k$ s.t. $v_i \in V_1$
- a different label for each $i = 1, \ldots, k$ s.t. $v_i, v_{i+1} \in V_2$

L-admissible walks

Walk $W = v_1, \ldots, v_k$ is L-admissible when

- For every $i = 1, \ldots, k-2$, if $v_i \in V_2$ and $v_{i+1} \in V_1$ then $v_{i+2} \neq v_i$.

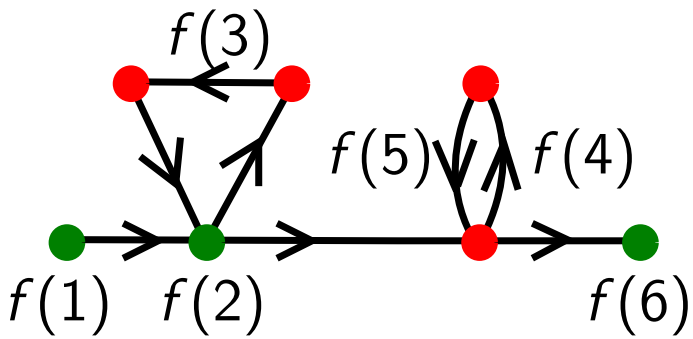


- $|\{i \ : \ v_i \in V_1\}| + |\{i \ : \ v_i v_{i+1} \in V_2\}| = L$

The ultimate hero

$$P_L(\mathbf{x},\mathbf{y}) = \sum_{\substack{\text{walk } W=v_1,\ldots,v_k \\ W \text{ is } L\text{-admissible}}} \sum_{\substack{\ell:[L]\to[L] \\ \ell \text{ is bijective}}} \prod_{i=1}^{k-1} x_{v_i,v_{i+1}} \prod_{i=1}^{L} y_{f(i),\ell(i)},$$

where $f(i) = i$-th labeled object (V_1-vertex or V_2V_2-edge) in walk W.



$$P = \sum_{L=k/2}^{\lceil \frac{3}{4}k \rceil} P_L$$

Correctness

- We have checked that:
  $P \not\equiv 0 \Rightarrow$ exists k-path
  (i.e. non-path walks cancel-out)

Correctness

- We have checked that:
  $P \not\equiv 0 \Rightarrow$ exists k-path
  (i.e. non-path walks cancel-out)
- The opposite implication not always true! (why?)

Correctness

- We have checked that:
  $P \not\equiv 0 \Rightarrow$ exists k-path
  (i.e. non-path walks cancel-out)
- The opposite implication not always true! (why?)
  it may happen that the only (say) solution P is not L-admissible for all $L \leq \lceil \frac{3}{4}k \rceil$.

Correctness

- We have checked that:
  $P \not\equiv 0 \Rightarrow$ exists k-path
  (i.e. non-path walks cancel-out)
- The opposite implication not always true! (why?)
  it may happen that the only (say) solution P is not L-admissible for all $L \leq \lceil \frac{3}{4}k \rceil$.
- But...
  - $\mathbb{E}[|\{i \ : \ v_i \in V_1\}| + |\{i \ : \ v_i v_{i+1} \in V_2\}|] = \frac{k}{2} + \frac{k-1}{4} = \frac{3k-1}{4}$

Correctness

- We have checked that:
  $P \not\equiv 0 \Rightarrow$ exists k-path
  (i.e. non-path walks cancel-out)
- The opposite implication not always true! (why?)
  it may happen that the only (say) solution P is not L-admissible for all $L \leq \lceil \frac{3}{4}k \rceil$.
- But...
  - $\mathbb{E}[|\{i \ : \ v_i \in V_1\}| + |\{i \ : \ v_i v_{i+1} \in V_2\}|] = \frac{k}{2} + \frac{k-1}{4} = \frac{3k-1}{4}$
- So, by Markov inequality

$$Pr[P \text{ is not } L\text{-admissible for all } L \leq \lceil \tfrac{3}{4}k \rceil] \leq \frac{(3k-1)/4}{\lceil \frac{3}{4}k \rceil + 1} = 1 - 1/O(k)$$

Correctness

- We have checked that:
  $P \not\equiv 0 \Rightarrow$ exists k-path
  (i.e. non-path walks cancel-out)
- The opposite implication not always true! (why?)
  it may happen that the only (say) solution P is not L-admissible for all $L \leq \lceil \frac{3}{4}k \rceil$.
- But...
  - $\mathbb{E}[|\{i \ : \ v_i \in V_1\}| + |\{i \ : \ v_i v_{i+1} \in V_2\}|] = \frac{k}{2} + \frac{k-1}{4} = \frac{3k-1}{4}$
- So, by Markov inequality

$$Pr[P \text{ is not } L\text{-admissible for all } L \leq \lceil \tfrac{3}{4}k \rceil] \leq \frac{(3k-1)/4}{\lceil \frac{3}{4}k \rceil + 1} = 1 - 1/O(k)$$

- If we repeat the algorithm $k \log n$ times this probability drops to

$$(1 - 1/O(k))^{k \log n} = (e^{-1/O(k)})^{k \log n} = e^{-O(\log n)} = 1/n^{\Omega(1)}$$

Conclusion

Theorem (Björklund, Husfeldt, Kaski, Koivisto 2010)

Longest Path in undirected graphs can be solved in $2^{3k/4} = 1.682^k(kn)^{O(1)}$ time and polynomial space.

Conclusion

Theorem (Björklund, Husfeldt, Kaski, Koivisto 2010)

Longest Path in undirected graphs can be solved in $2^{3k/4} = 1.682^k (kn)^{O(1)}$ time and polynomial space.

Note: using a simple trick one can tune the algorithm to get $1.66^k n^{O(1)}$.

Corollary (Björklund 2009)

The Hamiltonian Cycle problem in undirected graphs can be solved in $1.66^k n^{O(1)}$ time and polynomial space.

More applications of the same technique

- 3-dimensional matching,
- k-packing,
- edge coloring,
- Steiner cycle (aka K-cycle),
- rural postman,
- graph motif and related problems,
- ...

Open problems

- Improve over $O(1.66^n)$ for Hamiltonian cycle in undirected graphs
- Get $O(1.99^n)$ for Hamiltonian cycle in directed graphs.
- Faster deteministic algorithms for Longest Path (Best known: $2.86^k n^{O(1)}$, Fomin, Lokshtanov, Saurabh 2013)

Exercises

- (Ex 10.16, 🍃) Show an algorithm running in time $2^k(nk)^{O(1)}$ **and polynomial space** for finding a colorful k-path in a vertex colored graph.
- (Ex 10.17) Extend the $2^k(nk)^{O(1)}$ algorithm for k-path to **weighted** case for weight function $w : E \to [W]$. Your algorithm should run in time $2^k W(nk)^{O(1)}$.
- (Ex 10.18) Show a $2^{3k}(nk)^{O(1)}$ algorithm which determines if a given graph contains k vertex disjoint triangles.