Bipartite Perfect Matching is in quasi-NC

Stephen Fenner[1]

[1]Computer Science and Engineering Department
University of South Carolina
fenner@cse.sc.edu

Institute for Advanced Study, Princeton, February 8, 2016

Joint work with Rohit Gurjar and Thomas Thierauf (University of Aalen, Germany).
http://eccc.hpi-web.de/report/2015/177/.

Matching

$G = (V, E)$ is a graph with n nodes and m edges.

Definition

A matching in G is a set $M \subseteq E$ such that each $v \in V$ is incident to at most one $e \in M$.

For a perfect matching (p.m.): substitute "exactly" for "at most" above.
The perfect matching decision problem, PM, asks whether a given graph has a p.m.
The search problem, SEARCH-PM, asks for a p.m. in a graph if it exists.
Matchings and perfect matchings have been widely studied in combinatorics and complexity theory.

Matching

$G = (V, E)$ is a graph with n nodes and m edges.

Definition

A matching in G is a set $M \subseteq E$ such that each $v \in V$ is incident to at most one $e \in M$.

For a perfect matching (p.m.): substitute "exactly" for "at most" above.

The perfect matching decision problem, PM, asks whether a given graph has a p.m.

The search problem, SEARCH-PM, asks for a p.m. in a graph if it exists.

Matchings and perfect matchings have been widely studied in combinatorics and complexity theory.

Matching

$G = (V, E)$ is a graph with n nodes and m edges.

Definition

A matching in G is a set $M \subseteq E$ such that each $v \in V$ is incident to at most one $e \in M$.

For a perfect matching (p.m.): substitute "exactly" for "at most" above.
The perfect matching decision problem, PM, asks whether a given graph has a p.m.
The search problem, SEARCH-PM, asks for a p.m. in a graph if it exists.
Matchings and perfect matchings have been widely studied in combinatorics and complexity theory.

Matching

$G = (V, E)$ is a graph with n nodes and m edges.

Definition

A matching in G is a set $M \subseteq E$ such that each $v \in V$ is incident to at most one $e \in M$.

For a perfect matching (p.m.): substitute "exactly" for "at most" above.
The perfect matching decision problem, PM, asks whether a given graph has a p.m.
The search problem, SEARCH-PM, asks for a p.m. in a graph if it exists.
Matchings and perfect matchings have been widely studied in combinatorics and complexity theory.

Matching

$G = (V, E)$ is a graph with n nodes and m edges.

Definition

A matching in G is a set $M \subseteq E$ such that each $v \in V$ is incident to at most one $e \in M$.

For a perfect matching (p.m.): substitute "exactly" for "at most" above.
The perfect matching decision problem, PM, asks whether a given graph has a p.m.
The search problem, SEARCH-PM, asks for a p.m. in a graph if it exists.
Matchings and perfect matchings have been widely studied in combinatorics and complexity theory.

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.
For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

*Is there a fast parallel nonrandomized (*NC*) algorithm for* PM*?*

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.
For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

*Is there a fast parallel nonrandomized (*NC*) algorithm for* PM*?*

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.
For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

Is there a fast parallel nonrandomized (NC) algorithm for PM?

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.
For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

*Is there a fast parallel nonrandomized (*NC*) algorithm for* PM*?*

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.

For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

Is there a fast parallel nonrandomized (NC) algorithm for PM?

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.
For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

Is there a fast parallel nonrandomized (NC) algorithm for PM?

Previous algorithms for PM and SEARCH-PM

- A polynomial-time algorithm for PM due to Edmonds [Edm65].
- A fast randomized parallel (RNC) algorithm for PM due to Lovász [Lov79] (also Chari, Rohatgi, & Srinivasan [CRS95]).
- An RNC algorithm for SEARCH-PM due to Karp, Upfal, & Wigderson [KUW86].
- Another RNC algorithm due to Mulmuley, Vazirani, & Vazirani [MVV87] using the Isolation Lemma.

NC is the class of problems with uniform polynomial size circuits with polylogarithmic depth.
For polylog-depth circuits solving PM, nothing better than exponential size was known.

Open

*Is there a fast parallel nonrandomized (*NC*) algorithm for* PM*?*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC.

That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$.

We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC.

That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$.

We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC.

That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$.

We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC.
That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$.
We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC.
That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$.
We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC.
That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$.
We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC*.*

That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$*.*

We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC*.*
That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$*.*
We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Deterministic parallel algorithms

There are NC algorithms for certain types of graphs:

- $K_{3,3}$-free graphs(Vazirani [Vaz89]),
- graphs having polynomially many p.m.'s (Grigoriev & Karpinski [GK87], also Agrawal, Hoang, & Thierauf [AHT07])
- bipartite d-regular graphs (Lev, Pippenger, & Valiant [LPV81], also Sharan & Wigderson [SW96])
- strongly chordal graphs (Dahlhaus & Karpinski [DK98]).
- planar biparite graphs (Datta, Kulkarni, & Roy [DKR10] and Tewari & Vinodchandran [TV12])

Our Work

Bipartite PM *and* SEARCH-PM *are in* quasi-NC*.*
That is, PM *and* SEARCH-PM *on bipartite graphs have uniform circuits of depth* $O(\log^2 n)$ *and size* $2^{O(\log^2 n)}$*.*
We also give an RNC^2 *algorithm for bipartite* PM *using* $O(\log^2 n)$ *random bits.*

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)}.$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)}.$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)}\,.$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)}\ .$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)} .$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)} .$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Bipartite perfect matching in RNC

G bipartite with bipartition $L = \{u_1, \ldots, u_{n/2}\}$ and $R = \{v_1, \ldots, v_{n/2}\}$.
Given a weight function $w : E \to \mathbb{Z}^+$, we extend w to sets of edges: for $S \subseteq E$, define $w(S) := \sum_{e \in S} w(e)$.
Define the $n/2 \times n/2$ matrix $A_w = [a_{i,j}]$ as

$$a_{ij} = \begin{cases} 2^{w(e)} & \text{if } e = (u_i, v_j) \in E, \\ 0 & \text{if } (u_i, v_j) \notin E. \end{cases}$$

Then

$$\det(A_w) = \sum_{M \text{ a p.m. of } G} \operatorname{sgn}(M)\, 2^{w(M)} .$$

If G has no p.m., then $\det(A_w) = 0$ for any w.
If G does have a p.m., then $\det(A_w)$ may still be 0 because of cancellations.

Definition

A weight function w is isolating if G has a unique minimum weight p.m. with respect to w.

If w is isolating, then $\det(A_w) \neq 0$, because the minimum weight term in $\det(A_w)$ does not cancel with other terms, which are strictly higher powers of 2.

Lemma (Isolation Lemma [MVV87])

Let $w(e)$ chosen uniformly at random from $\{1, \ldots, 2m\}$ for each edge e independently. Then w is isolating with probability $\geq 1/2$.

If w is isolating, then computing $\det(A_w)$ gives the correct answer. This can be done in NC^2 (Berkowitz [Ber84]).

Definition

A weight function w is isolating if G has a unique minimum weight p.m. with respect to w.

If w is isolating, then $\det(A_w) \neq 0$, because the minimum weight term in $\det(A_w)$ does not cancel with other terms, which are strictly higher powers of 2.

Lemma (Isolation Lemma [MVV87])

Let $w(e)$ chosen uniformly at random from $\{1, \ldots, 2m\}$ for each edge e independently. Then w is isolating with probability $\geq 1/2$.

If w is isolating, then computing $\det(A_w)$ gives the correct answer. This can be done in NC^2 (Berkowitz [Ber84]).

Definition

A weight function w is isolating if G has a unique minimum weight p.m. with respect to w.

If w is isolating, then $\det(A_w) \neq 0$, because the minimum weight term in $\det(A_w)$ does not cancel with other terms, which are strictly higher powers of 2.

Lemma (Isolation Lemma [MVV87])

Let $w(e)$ chosen uniformly at random from $\{1, \ldots, 2m\}$ for each edge e independently. Then w is isolating with probability $\geq 1/2$.

If w is isolating, then computing $\det(A_w)$ gives the correct answer. This can be done in NC^2 (Berkowitz [Ber84]).

Definition

A weight function w is isolating if G has a unique minimum weight p.m. with respect to w.

If w is isolating, then $\det(A_w) \neq 0$, because the minimum weight term in $\det(A_w)$ does not cancel with other terms, which are strictly higher powers of 2.

Lemma (Isolation Lemma [MVV87])

Let $w(e)$ chosen uniformly at random from $\{1, \ldots, 2m\}$ for each edge e independently. Then w is isolating with probability $\geq 1/2$.

If w is isolating, then computing $\det(A_w)$ gives the correct answer. This can be done in NC^2 (Berkowitz [Ber84]).

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.
w is clearly isolating, ...
but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.
Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.
For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.

w is clearly isolating, ...

but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.

Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.

For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.
w is clearly isolating, ...
but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.
Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.
For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.
w is clearly isolating, ...
but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.
Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.
For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.
w is clearly isolating, ...
but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.
Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.
For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.
w is clearly isolating, ...
but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.
Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.
For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

We want to derandomize this lemma!

Let $E = \{e_0, \ldots, e_{m-1}\}$, and define $w(e_i) = 2^i$ for all $i < m$.
w is clearly isolating, ...
but we cannot compute $\det(A_w)$ efficiently, because the matrix entries are too big.
Instead, we reduce the weights modulo small numbers j:

Definition

Fix an integer $j > 1$. Define the weight function $w_{\bmod j}$ as

$$w_{\bmod j}(e) := w(e) \bmod j$$

for all $e \in E$.
For some t we choose later, define the set of weight functions

$$W_t := \{w_{\bmod j} \mid 2 \leq j \leq t\} .$$

Circulation

Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order.
(p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:

Circulation

Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order.
(p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:

Circulation

Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order. (p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:

Circulation

Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order. (p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:

Circulation

Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order. (p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.

Then M_1 and M_2 differ on disjoint cycles with zero circulation:

Circulation

Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order.
(p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:

Circulation

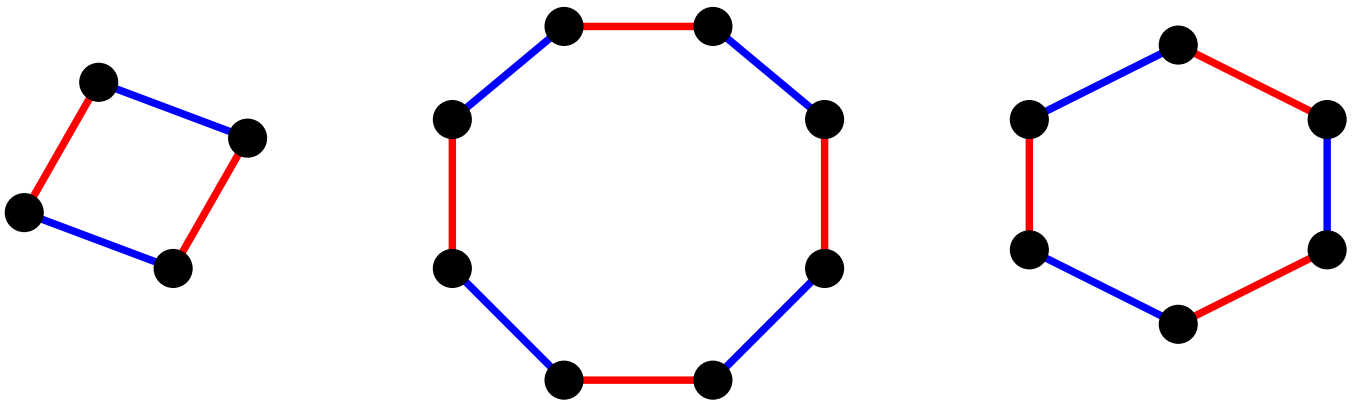
Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order. (p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:



Circulation

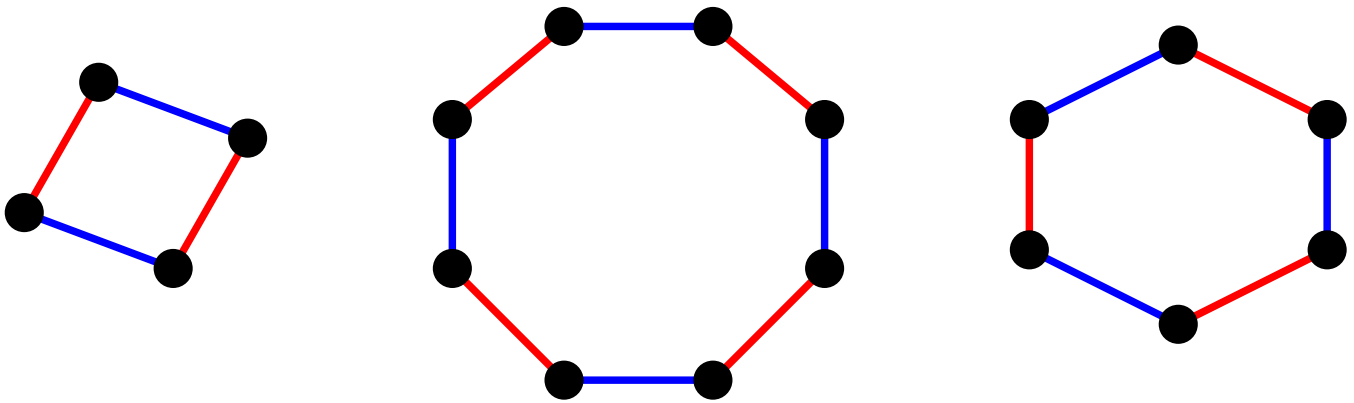
Let $C = \langle e_1, \ldots, e_p \rangle$ be a cycle of G with edges given in cyclic order.
(p is even because G is bipartite.)

Definition

Given a weight function w, the circulation of C with respect to w is

$$c_w(C) := \left| \sum_{i=1}^{p} (-1)^i w(e_i) \right| .$$

Given w, suppose $M_1 \neq M_2$ are min weight p.m.'s of G.
Then M_1 and M_2 differ on disjoint cycles with zero circulation:



Forcing nonzero circulation

We would like to choose a weight function from W_t that gives nonzero circulation to as many cycles as possible.
We cannot do this for all cycles at once, so we work in stages, starting with short cycles.

Lemma ([CRS95])

Let s be a positive integer, and let $t = n^2 s$. Then for any set of s many cycles $\{C_1, \ldots, C_s\}$ there exists a weight function $w \in W_t$ that gives nonzero circulation to all of $C_1, \ldots, C_s$.

We will apply this lemma with $s := n^4$.
Each weight of w is taken modulo some $j \leq t = n^2 s = n^6$, so needs only $6 \log n$ bits.

Forcing nonzero circulation

We would like to choose a weight function from W_t that gives nonzero circulation to as many cycles as possible.

We cannot do this for all cycles at once, so we work in stages, starting with short cycles.

Lemma ([CRS95])

Let s be a positive integer, and let $t = n^2 s$. Then for any set of s many cycles $\{C_1, \ldots, C_s\}$ there exists a weight function $w \in W_t$ that gives nonzero circulation to all of $C_1, \ldots, C_s$.

We will apply this lemma with $s := n^4$.

Each weight of w is taken modulo some $j \leq t = n^2 s = n^6$, so needs only $6 \log n$ bits.

Forcing nonzero circulation

We would like to choose a weight function from W_t that gives nonzero circulation to as many cycles as possible.
We cannot do this for all cycles at once, so we work in stages, starting with short cycles.

Lemma ([CRS95])

Let s be a positive integer, and let $t = n^2 s$. Then for any set of s many cycles $\{C_1, \ldots, C_s\}$ there exists a weight function $w \in W_t$ that gives nonzero circulation to all of $C_1, \ldots, C_s$.

We will apply this lemma with $s := n^4$.
Each weight of w is taken modulo some $j \leq t = n^2 s = n^6$, so needs only $6 \log n$ bits.

Forcing nonzero circulation

We would like to choose a weight function from W_t that gives nonzero circulation to as many cycles as possible.
We cannot do this for all cycles at once, so we work in stages, starting with short cycles.

Lemma ([CRS95])

Let s be a positive integer, and let $t = n^2 s$. Then for any set of s many cycles $\{C_1, \ldots, C_s\}$ there exists a weight function $w \in W_t$ that gives nonzero circulation to all of $C_1, \ldots, C_s$.

We will apply this lemma with $s := n^4$.
Each weight of w is taken modulo some $j \leq t = n^2 s = n^6$, so needs only $6 \log n$ bits.

Forcing nonzero circulation

We would like to choose a weight function from W_t that gives nonzero circulation to as many cycles as possible.
We cannot do this for all cycles at once, so we work in stages, starting with short cycles.

Lemma ([CRS95])

Let s be a positive integer, and let $t = n^2 s$. Then for any set of s many cycles $\{C_1, \ldots, C_s\}$ there exists a weight function $w \in W_t$ that gives nonzero circulation to all of $C_1, \ldots, C_s$.

We will apply this lemma with $s := n^4$.
Each weight of w is taken modulo some $j \leq t = n^2 s = n^6$, so needs only $6 \log n$ bits.

Forcing nonzero circulation

We would like to choose a weight function from W_t that gives nonzero circulation to as many cycles as possible.
We cannot do this for all cycles at once, so we work in stages, starting with short cycles.

Lemma ([CRS95])

Let s be a positive integer, and let $t = n^2 s$. Then for any set of s many cycles $\{C_1, \ldots, C_s\}$ there exists a weight function $w \in W_t$ that gives nonzero circulation to all of $C_1, \ldots, C_s$.

We will apply this lemma with $s := n^4$.
Each weight of w is taken modulo some $j \leq t = n^2 s = n^6$, so needs only $6 \log n$ bits.

The derived graph

Suppose G has a p.m., and w is a weight function on G.

Definition

The derived graph of G with respect to w is the subgraph $G^{(w)} := (V, E')$, where E' is the union of all w-min weight p.m.'s of G.

Key Lemma

All cycles in $G^{(w)}$ have zero circulation with respect to w.

We proved this lemma using linear algebra.
Later, an alternate combinatorial proof was found by Rao, Shpilka, & Wigderson (reported in Goldwasser & Grossman [GG15]).

Corollary

All p.m.'s in $G^{(w)}$ are min weight p.m.'s of G.

The derived graph

Suppose G has a p.m., and w is a weight function on G.

Definition

The derived graph of G with respect to w is the subgraph $G^{(w)} := (V, E')$, where E' is the union of all w-min weight p.m.'s of G.

Key Lemma

All cycles in $G^{(w)}$ have zero circulation with respect to w.

We proved this lemma using linear algebra.
Later, an alternate combinatorial proof was found by Rao, Shpilka, & Wigderson (reported in Goldwasser & Grossman [GG15]).

Corollary

All p.m.'s in $G^{(w)}$ are min weight p.m.'s of G.

The derived graph

Suppose G has a p.m., and w is a weight function on G.

Definition

The derived graph of G with respect to w is the subgraph $G^{(w)} := (V, E')$, where E' is the union of all w-min weight p.m.'s of G.

Key Lemma

All cycles in $G^{(w)}$ have zero circulation with respect to w.

We proved this lemma using linear algebra.
Later, an alternate combinatorial proof was found by Rao, Shpilka, & Wigderson (reported in Goldwasser & Grossman [GG15]).

Corollary

All p.m.'s in $G^{(w)}$ are min weight p.m.'s of G.

The derived graph

Suppose G has a p.m., and w is a weight function on G.

Definition

The derived graph of G with respect to w is the subgraph $G^{(w)} := (V, E')$, where E' is the union of all w-min weight p.m.'s of G.

Key Lemma

All cycles in $G^{(w)}$ have zero circulation with respect to w.

We proved this lemma using linear algebra.

Later, an alternate combinatorial proof was found by Rao, Shpilka, & Wigderson (reported in Goldwasser & Grossman [GG15]).

Corollary

All p.m.'s in $G^{(w)}$ are min weight p.m.'s of G.

The derived graph

Suppose G has a p.m., and w is a weight function on G.

Definition

The derived graph of G with respect to w is the subgraph $G^{(w)} := (V, E')$, where E' is the union of all w-min weight p.m.'s of G.

Key Lemma

All cycles in $G^{(w)}$ have zero circulation with respect to w.

We proved this lemma using linear algebra.
Later, an alternate combinatorial proof was found by Rao, Shpilka, & Wigderson (reported in Goldwasser & Grossman [GG15]).

Corollary

All p.m.'s in $G^{(w)}$ are min weight p.m.'s of G.

The derived graph

Suppose G has a p.m., and w is a weight function on G.

Definition

The derived graph of G with respect to w is the subgraph $G^{(w)} := (V, E')$, where E' is the union of all w-min weight p.m.'s of G.

Key Lemma

All cycles in $G^{(w)}$ have zero circulation with respect to w.

We proved this lemma using linear algebra.
Later, an alternate combinatorial proof was found by Rao, Shpilka, & Wigderson (reported in Goldwasser & Grossman [GG15]).

Corollary

All p.m.'s in $G^{(w)}$ are min weight p.m.'s of G.

Removing short cycles from $G^{(w)}$

In steps, we remove cycles from derived graphs whose lengths are increasing powers of 2.

The next lemma says there are not too many of these.

Lemma

Let G be a graph with no cycles of length $\leq r$ for some even r. Then G has $\leq n^4$ cycles of length $\leq 2r$.

We can give all these cycles nonzero circulation by some weight function $w \in W_t$, where $t = n^6$.

So these cycles cannot exist in the derived graph with respect to w.

Removing short cycles from $G^{(w)}$

In steps, we remove cycles from derived graphs whose lengths are increasing powers of 2.
The next lemma says there are not too many of these.

Lemma

Let G be a graph with no cycles of length $\leq r$ for some even r. Then G has $\leq n^4$ cycles of length $\leq 2r$.

We can give all these cycles nonzero circulation by some weight function $w \in W_t$, where $t = n^6$.
So these cycles cannot exist in the derived graph with respect to w.

Removing short cycles from $G^{(w)}$

In steps, we remove cycles from derived graphs whose lengths are increasing powers of 2.
The next lemma says there are not too many of these.

Lemma

Let G be a graph with no cycles of length $\leq r$ for some even r. Then G has $\leq n^4$ cycles of length $\leq 2r$.

We can give all these cycles nonzero circulation by some weight function $w \in W_t$, where $t = n^6$.
So these cycles cannot exist in the derived graph with respect to w.

Removing short cycles from $G^{(w)}$

In steps, we remove cycles from derived graphs whose lengths are increasing powers of 2.
The next lemma says there are not too many of these.

Lemma

Let G be a graph with no cycles of length $\leq r$ for some even r. Then G has $\leq n^4$ cycles of length $\leq 2r$.

We can give all these cycles nonzero circulation by some weight function $w \in W_t$, where $t = n^6$.
So these cycles cannot exist in the derived graph with respect to w.

Removing short cycles from $G^{(w)}$

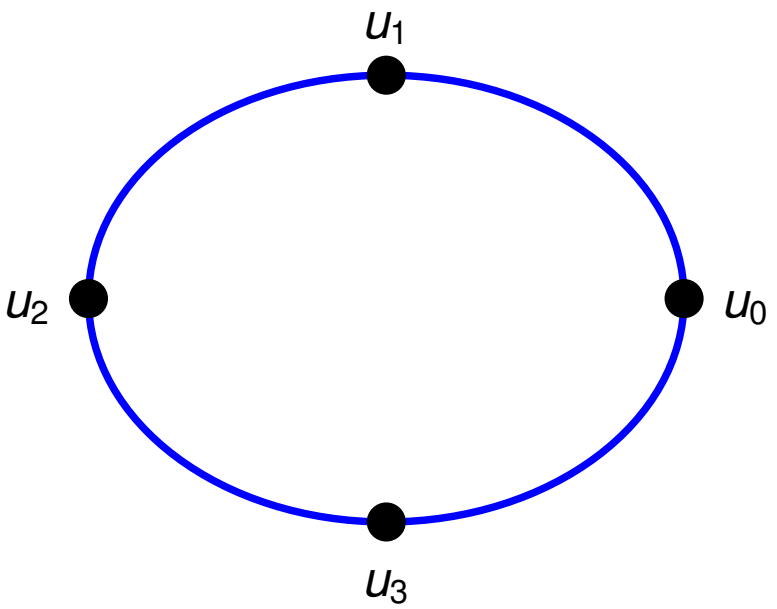
In steps, we remove cycles from derived graphs whose lengths are increasing powers of 2.
The next lemma says there are not too many of these.

Lemma

Let G be a graph with no cycles of length $\leq r$ for some even r. Then G has $\leq n^4$ cycles of length $\leq 2r$.

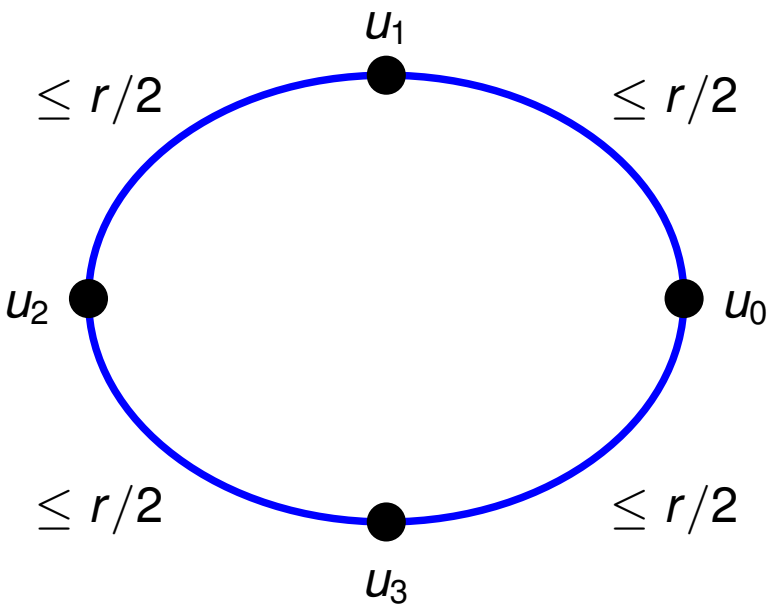
We can give all these cycles nonzero circulation by some weight function $w \in W_t$, where $t = n^6$.
So these cycles cannot exist in the derived graph with respect to w.

Proof



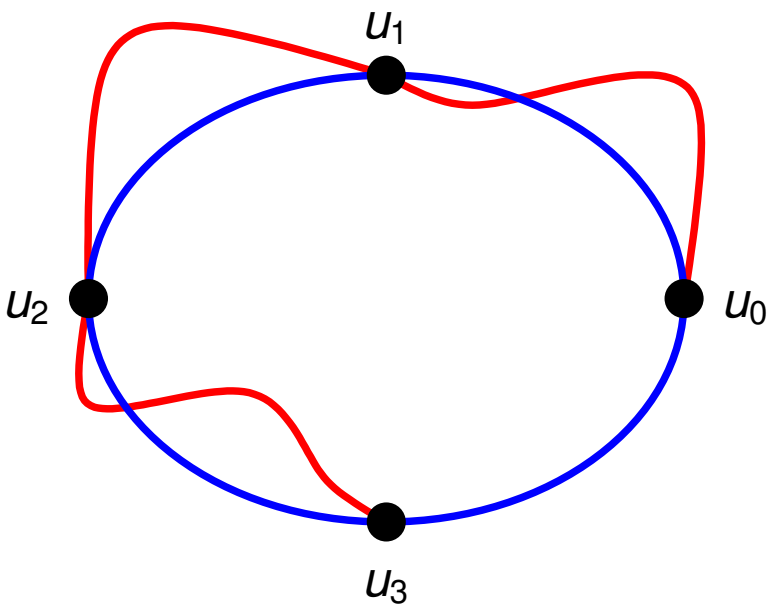
- Given a cycle C of length $\leq 2r$, choose four vertices u_0, u_1, u_2, u_3 on the cycle
- such that the distance between adjacent vertices is $\leq r/2$.
- This is the only such cycle given $(u_0, \ldots, u_3)$. If there is another such cycle C', then
- C' forms a cycle with C of length $\leq r$. Contradiction.

Proof



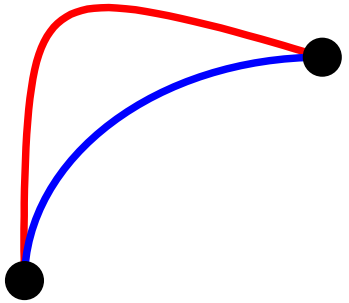
- Given a cycle C of length $\leq 2r$, choose four vertices u_0, u_1, u_2, u_3 on the cycle
- such that the distance between adjacent vertices is $\leq r/2$.
- This is the only such cycle given $(u_0, \ldots, u_3)$. If there is another such cycle C', then
- C' forms a cycle with C of length $\leq r$. Contradiction.

Proof



- Given a cycle C of length $\leq 2r$, choose four vertices u_0, u_1, u_2, u_3 on the cycle
- such that the distance between adjacent vertices is $\leq r/2$.
- This is the only such cycle given $(u_0, \ldots, u_3)$. If there is another such cycle C', then
- C' forms a cycle with C of length $\leq r$. Contradiction.

Proof



- Given a cycle C of length $\leq 2r$, choose four vertices u_0, u_1, u_2, u_3 on the cycle
- such that the distance between adjacent vertices is $\leq r/2$.
- This is the only such cycle given $(u_0, \ldots, u_3)$. If there is another such cycle C', then
- C' forms a cycle with C of length $\leq r$. Contradiction.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- ...
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- ...

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- $\ldots$
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- $\ldots$

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- $\ldots$
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- $\ldots$

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

The sequence of derived graphs

Start with $G_0 := G$, a bipartite graph with a p.m.

- Choose $w_1 \in W_t$ such that all cycles in G_0 of length ≤ 4 have nonzero circulation.
- Let $G_1 := G^{(w_1)}$. G_1 has a p.m. and no cycles of length ≤ 4.
- Choose $w_2 \in W_t$ such that all cycles in G_1 of length ≤ 8 have nonzero circulation.
- Let $G_2 := G_1^{(w_2)}$. G_2 has a p.m. and no cycles of length ≤ 8.
- $\ldots$
- Choose $w_i \in W_t$ such that all cycles in G_{i-1} of length $\leq 2^{i+1}$ have nonzero circulation.
- Let $G_i := G_{i-1}^{(w_i)}$. G_i has a p.m. and no cycles of length $\leq 2^{i+1}$.
- $\ldots$

Proceed for $k := \lceil \log n \rceil - 1$ rounds to obtain G_k, which is a p.m.

An isolating weight function for G

We must glue the weight functions $w_1, \ldots, w_k$ together into a single weight function.

Let B be a strict bound on any edge weight from $w_1, \ldots, w_k$ (we may take $B := n^6$).

For every $e \in E$, define

$$w(e) = B^{k-1} w_1(e) + B^{k-2} w_2(e) + \cdots + B^0 w_k(e) .$$

Lemma

If G has a p.m., then w is isolating for G.

An isolating weight function for G

We must glue the weight functions $w_1, \ldots, w_k$ together into a single weight function.

Let B be a strict bound on any edge weight from $w_1, \ldots, w_k$ (we may take $B := n^6$).

For every $e \in E$, define

$$w(e) = B^{k-1} w_1(e) + B^{k-2} w_2(e) + \cdots + B^0 w_k(e) .$$

Lemma

If G has a p.m., then w is isolating for G.

An isolating weight function for G

We must glue the weight functions $w_1, \ldots, w_k$ together into a single weight function.
Let B be a strict bound on any edge weight from $w_1, \ldots, w_k$ (we may take $B := n^6$).
For every $e \in E$, define

$$w(e) = B^{k-1} w_1(e) + B^{k-2} w_2(e) + \cdots + B^0 w_k(e) .$$

Lemma

If G has a p.m., then w is isolating for G.

An isolating weight function for G

We must glue the weight functions $w_1, \ldots, w_k$ together into a single weight function.
Let B be a strict bound on any edge weight from $w_1, \ldots, w_k$ (we may take $B := n^6$).
For every $e \in E$, define

$$w(e) = B^{k-1} w_1(e) + B^{k-2} w_2(e) + \cdots + B^0 w_k(e) \; .$$

Lemma

If G has a p.m., then w is isolating for G.

An isolating weight function for G

We must glue the weight functions $w_1, \ldots, w_k$ together into a single weight function.
Let B be a strict bound on any edge weight from $w_1, \ldots, w_k$ (we may take $B := n^6$).
For every $e \in E$, define

$$w(e) = B^{k-1} w_1(e) + B^{k-2} w_2(e) + \cdots + B^0 w_k(e) \; .$$

Lemma

If G has a p.m., then w is isolating for G.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

Proof

- Notice that the edge sets of the G_i form a descending chain, ending in a p.m. M of G (the edge set of G_k).
- Let $M' \neq M$ be some other p.m. of G.
- There must be some stage $i < k$ where M and M' are both in G_i but M' is not in G_{i+1}.
- Since M and M' are in $G_1, \ldots, G_i$, they both have the same minimum weight with respect to $w_1, \ldots, w_i$.
- But since M' is not in G_{i+1} (but M is), it must be that $w_{i+1}(M') > w_{i+1}(M)$.
- This implies $w(M') > w(M)$, and so w is isolating.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel.

For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel.
For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits.
Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel. For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel. For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel. For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel. For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel. For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

The algorithm

We do not know which $w_1, \ldots, w_k$ work, so we try them all in parallel. For all $w_1, \ldots, w_k \in W_{n^6}$ in parallel:

- Compute w as above. (One of these choices of w must be isolating.)
- Compute $\det(A_w)$ as in the RNC algorithm of [MVV87].
- If we ever find a nonzero determinant, answer "yes."
- Else, answer "no."

Each w_i takes $6 \log n$ bits to store, so w takes $O(\log^2 n)$ bits. Processing them all in parallel can be done with circuits of size $2^{O(\log^2 n)}$ and depth $O(\log^2 n)$.

Proving the Key Lemma: The perfect matching polytope

Let $\mathbb{R}^E$ be the m-dimensional real vector space with standard basis labeled by the edges of G.
Then any set $S \subseteq E$ of edges naturally corresponds to its characteristic vector $(s_e)_{e \in E}$, where, for each edge $e \in E$,

$$s_e = \begin{cases} 1 & \text{if } e \in S, \\ 0 & \text{if } e \notin S. \end{cases}$$

Definition

The perfect matching polytope $\mathrm{PM}(G)$ is the convex hull of all the perfect matchings of G.

Proving the Key Lemma: The perfect matching polytope

Let $\mathbb{R}^E$ be the m-dimensional real vector space with standard basis labeled by the edges of G.

Then any set $S \subseteq E$ of edges naturally corresponds to its characteristic vector $(s_e)_{e \in E}$, where, for each edge $e \in E$,

$$s_e = \begin{cases} 1 & \text{if } e \in S, \\ 0 & \text{if } e \notin S. \end{cases}$$

Definition

The perfect matching polytope $\mathrm{PM}(G)$ is the convex hull of all the perfect matchings of G.

Proving the Key Lemma: The perfect matching polytope

Let $\mathbb{R}^E$ be the m-dimensional real vector space with standard basis labeled by the edges of G.
Then any set $S \subseteq E$ of edges naturally corresponds to its characteristic vector $(s_e)_{e \in E}$, where, for each edge $e \in E$,

$$s_e = \begin{cases} 1 & \text{if } e \in S, \\ 0 & \text{if } e \notin S. \end{cases}$$

Definition

The perfect matching polytope PM(G) is the convex hull of all the perfect matchings of G.

Proving the Key Lemma: The perfect matching polytope

Let $\mathbb{R}^E$ be the m-dimensional real vector space with standard basis labeled by the edges of G.
Then any set $S \subseteq E$ of edges naturally corresponds to its characteristic vector $(s_e)_{e \in E}$, where, for each edge $e \in E$,

$$s_e = \begin{cases} 1 & \text{if } e \in S, \\ 0 & \text{if } e \notin S. \end{cases}$$

Definition

The perfect matching polytope PM(G) is the convex hull of all the perfect matchings of G.

Lemma ([LP86])

If G is bipartite, then a vector $\vec{x} = (x_e)_e$ is in $\mathrm{PM}(G)$ if and only if

$$\begin{aligned} x_e &\geq 0 \,, \\ \sum_{e' \in \delta(v)} x_{e'} &= 1 \,, \end{aligned}$$

for all $e \in E$ and $v \in V$, where $\delta(v)$ is the set of edges incident to v.

The $\Rightarrow$ direction is clear for any graph (not necessarily bipartite). The converse does not hold for general graphs.

We can extend any weight function w to $\mathbb{R}^m$ by linearity:

$$w(\vec{x}) = \sum_{e \in E} w(e) x_e \,.$$

Lemma ([LP86])

If G is bipartite, then a vector $\vec{x} = (x_e)_e$ is in $\mathrm{PM}(G)$ if and only if

$$\begin{aligned} x_e &\geq 0 \,, \\ \sum_{e' \in \delta(v)} x_{e'} &= 1 \,, \end{aligned}$$

for all $e \in E$ and $v \in V$, where $\delta(v)$ is the set of edges incident to v.

The $\Rightarrow$ direction is clear for any graph (not necessarily bipartite). The converse does not hold for general graphs.

We can extend any weight function w to $\mathbb{R}^m$ by linearity:

$$w(\vec{x}) = \sum_{e \in E} w(e) x_e \,.$$

Lemma ([LP86])

If G is bipartite, then a vector $\vec{x} = (x_e)_e$ is in $\mathrm{PM}(G)$ if and only if

$$\begin{aligned} x_e &\geq 0 \,, \\ \sum_{e' \in \delta(v)} x_{e'} &= 1 \,, \end{aligned}$$

for all $e \in E$ and $v \in V$, where $\delta(v)$ is the set of edges incident to v.

The $\Rightarrow$ direction is clear for any graph (not necessarily bipartite). The converse does not hold for general graphs.
We can extend any weight function w to $\mathbb{R}^m$ by linearity:

$$w(\vec{x}) = \sum_{e \in E} w(e) x_e \,.$$

Key Lemma (cont.)

Let $\vec{x}_1, \ldots, \vec{x}_t \in \mathrm{PM}(G)$ be vectors corresponding to all the p.m.'s of G with the same minimum weight q.

Set

$$\vec{x} = (x_e)_e = \frac{\vec{x}_1 + \cdots + \vec{x}_t}{t} .$$

Then $\vec{x} \in \mathrm{PM}(M)$, and $w(\vec{x}) = q$.

Also, every entry of $\vec{x}$ in the derived graph G' satisfies $x_e \geq \frac{1}{t}$.

Key Lemma (cont.)

Let $\vec{x}_1, \ldots, \vec{x}_t \in \mathrm{PM}(G)$ be vectors corresponding to all the p.m.'s of G with the same minimum weight q.
Set

$$\vec{x} = (x_e)_e = \frac{\vec{x}_1 + \cdots + \vec{x}_t}{t} .$$

Then $\vec{x} \in \mathrm{PM}(M)$, and $w(\vec{x}) = q$.
Also, every entry of $\vec{x}$ in the derived graph G' satisfies $x_e \geq \frac{1}{t}$.

Key Lemma (cont.)

Let $\vec{x}_1, \ldots, \vec{x}_t \in \mathrm{PM}(G)$ be vectors corresponding to all the p.m.'s of G with the same minimum weight q.
Set

$$\vec{x} = (x_e)_e = \frac{\vec{x}_1 + \cdots + \vec{x}_t}{t} .$$

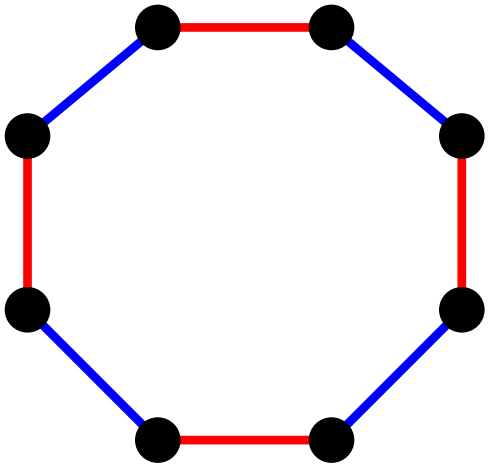
Then $\vec{x} \in \mathrm{PM}(M)$, and $w(\vec{x}) = q$.
Also, every entry of $\vec{x}$ in the derived graph G' satisfies $x_e \geq \frac{1}{t}$.

Key Lemma (cont.)

Let $\vec{x}_1, \ldots, \vec{x}_t \in \mathrm{PM}(G)$ be vectors corresponding to all the p.m.'s of G with the same minimum weight q.
Set

$$\vec{x} = (x_e)_e = \frac{\vec{x}_1 + \cdots + \vec{x}_t}{t} .$$

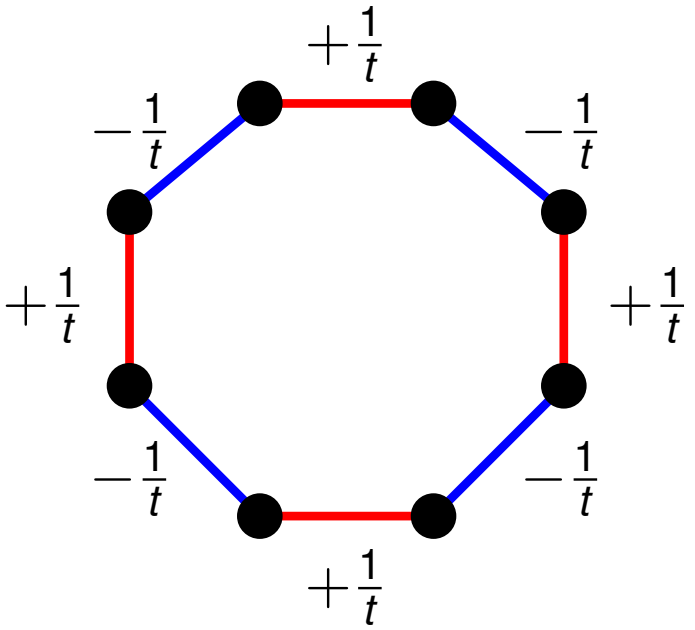
Then $\vec{x} \in \mathrm{PM}(M)$, and $w(\vec{x}) = q$.
Also, every entry of $\vec{x}$ in the derived graph G' satisfies $x_e \geq \frac{1}{t}$.



- Suppose some cycle C in the derived graph G' has nonzero circulation. W.l.o.g., the blue edges outweigh the red edges.
- Let $\vec{y} = (y_e)_e$ be the vector obtained from $\vec{x}$ by subtracting $\frac{1}{t}$ from the blue edges and adding $\frac{1}{t}$ to the red edges.
- Then $\vec{y} \in \mathrm{PM}(G)$. Moreover,

$$w(\vec{y}) = w(\vec{x}) - \frac{c_w(C)}{t} < q \ .$$

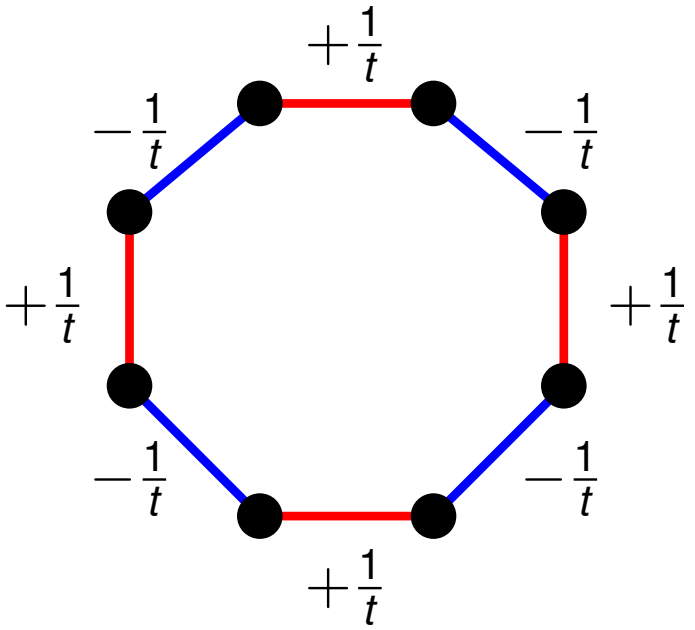
- But then there must be a p.m. of G with weight $< q$. Contradiction.



- Suppose some cycle C in the derived graph G' has nonzero circulation. W.l.o.g., the blue edges outweigh the red edges.
- Let $\vec{y} = (y_e)_e$ be the vector obtained from $\vec{x}$ by subtracting $\frac{1}{t}$ from the blue edges and adding $\frac{1}{t}$ to the red edges.
- Then $\vec{y} \in \mathrm{PM}(G)$. Moreover,

$$w(\vec{y}) = w(\vec{x}) - \frac{c_w(C)}{t} < q\,.$$

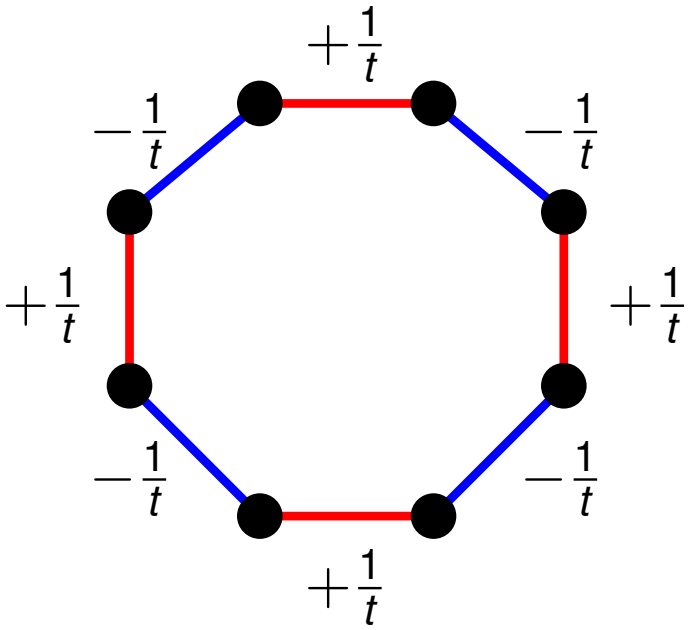
- But then there must be a p.m. of G with weight $< q$. Contradiction.



- Suppose some cycle C in the derived graph G' has nonzero circulation. W.l.o.g., the blue edges outweigh the red edges.
- Let $\vec{y} = (y_e)_e$ be the vector obtained from $\vec{x}$ by subtracting $\frac{1}{t}$ from the blue edges and adding $\frac{1}{t}$ to the red edges.
- Then $\vec{y} \in \mathrm{PM}(G)$. Moreover,

$$w(\vec{y}) = w(\vec{x}) - \frac{c_w(C)}{t} < q\,.$$

- But then there must be a p.m. of G with weight $< q$. Contradiction.



- Suppose some cycle C in the derived graph G' has nonzero circulation. W.l.o.g., the blue edges outweigh the red edges.
- Let $\vec{y} = (y_e)_e$ be the vector obtained from $\vec{x}$ by subtracting $\frac{1}{t}$ from the blue edges and adding $\frac{1}{t}$ to the red edges.
- Then $\vec{y} \in \mathrm{PM}(G)$. Moreover,

$$w(\vec{y}) = w(\vec{x}) - \frac{c_w(C)}{t} < q\,.$$

- But then there must be a p.m. of G with weight $< q$. Contradiction.

The RNC algorithm

Recall $w_{\bmod j}(e_i) = 2^i \bmod j$ for each edge $e_i \in E$ and $2 \leq j \leq t$.
Instead of trying all of these weight functions, we let j be a random prime.
Any set of s many cycles has nonzero circulation with high probability
Doing this k times gives random $w_1, \ldots, w_k$.

The RNC algorithm

Recall $w_{\bmod j}(e_i) = 2^i \bmod j$ for each edge $e_i \in E$ and $2 \leq j \leq t$.

Instead of trying all of these weight functions, we let j be a random prime.

Any set of s many cycles has nonzero circulation with high probability

Doing this k times gives random $w_1, \ldots, w_k$.

The RNC algorithm

Recall $w_{\bmod j}(e_i) = 2^i \bmod j$ for each edge $e_i \in E$ and $2 \leq j \leq t$.
Instead of trying all of these weight functions, we let j be a random prime.

Any set of s many cycles has nonzero circulation with high probability

Doing this k times gives random $w_1, \ldots, w_k$.

The RNC algorithm

Recall $w_{\bmod j}(e_i) = 2^i \bmod j$ for each edge $e_i \in E$ and $2 \leq j \leq t$.
Instead of trying all of these weight functions, we let j be a random prime.
Any set of s many cycles has nonzero circulation with high probability
Doing this k times gives random $w_1, \ldots, w_k$.

The RNC algorithm

Recall $w_{\bmod j}(e_i) = 2^i \bmod j$ for each edge $e_i \in E$ and $2 \leq j \leq t$.
Instead of trying all of these weight functions, we let j be a random prime.
Any set of s many cycles has nonzero circulation with high probability
Doing this k times gives random $w_1, \ldots, w_k$.

Other results

The following are all in quasi-NC:

- bipartite weighted PM with quasi-polynomially bounded integer weights
- maximum bipartite matching
- cycle cover with polynomially bounded integer weights
- subtree isomorphism
- max flow with polynomially bounded integer capacities
- constructing a depth-first search tree

Acknowledgments

I would like to thank Ran Raz and the rest of the IAS faculty for inviting me to give this talk.
We would also like to thank Manindra Agrawal and Nitin Saxena for the encouragement and very helpful discussions.
We thank Arpita Korwar for discussions on some techniques used for our RNC algorithm.

Acknowledgments

I would like to thank Ran Raz and the rest of the IAS faculty for inviting me to give this talk.
We would also like to thank Manindra Agrawal and Nitin Saxena for the encouragement and very helpful discussions.
We thank Arpita Korwar for discussions on some techniques used for our RNC algorithm.

Acknowledgments

I would like to thank Ran Raz and the rest of the IAS faculty for inviting me to give this talk.
We would also like to thank Manindra Agrawal and Nitin Saxena for the encouragement and very helpful discussions.
We thank Arpita Korwar for discussions on some techniques used for our RNC algorithm.

References


Manindra Agrawal, Thanh Minh Hoang, and Thomas Thierauf.
The polynomially bounded perfect matching problem is in NC^2.
In *24th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 4393 of *Lecture Notes in Computer Science*, pages 489–499. Springer Berlin Heidelberg, 2007.


Stuart J. Berkowitz.
On computing the determinant in small parallel time using a small number of processors.
Information Processing Letters, 18(3):147–150, 1984.


Suresh Chari, Pankaj Rohatgi, and Aravind Srinivasan.
Randomness-optimal unique element isolation with applications to perfect matching and related problems.
SIAM Journal on Computing, 24(5):1036–1050, 1995.


Elias Dahlhaus and Marek Karpinski.
Matching and multidimensional matching in chordal and strongly