Matching is in QUASI-NC

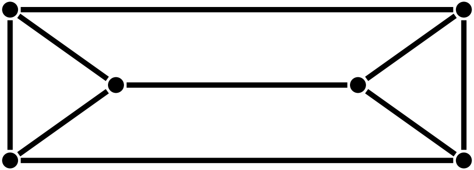
Jakub Tarnawski

joint work with Ola Svensson

September 28, 2017 am Mittag

Perfect matching problem

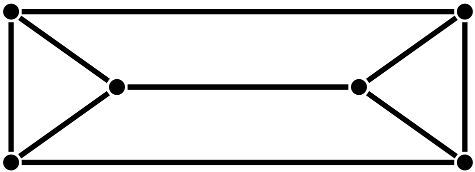
Given a graph, can we pair up all vertices using edges?



Perfect matching problem

Given a graph, can we pair up all vertices using edges?

very tough instance:
graph is non-bipartite!



Perfect matching problem

Given a graph, can we pair up all vertices using edges?

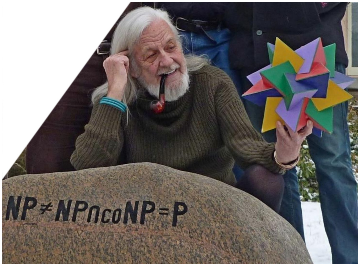
very tough instance: graph is non-bipartite!

Perfect matching problem

Benchmark problem in computer science

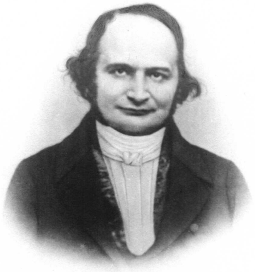
Algorithms:

- bipartite: Jacobi [XIX century, weighted!]
- general: Edmonds [1965]
  - polynomial-time = efficient
- since then, tons of research and still active
- many models of computation: monotone circuits, extended formulations, parallel, distributed, streaming/sublinear, ...



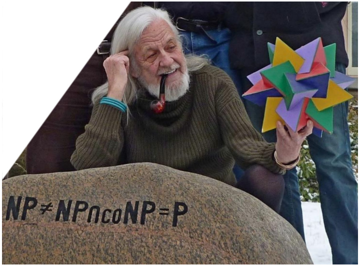
Perfect matching problem

Benchmark problem in computer science



Algorithms:

- bipartite: Jacobi [XIX century, weighted!]
- general: Edmonds [1965]
  - polynomial-time = efficient
- since then, tons of research and still active
- many models of computation: monotone circuits, extended formulations, **parallel**, distributed, streaming/sublinear, ...



Parallel complexity

Class $\mathcal{NC}$: problems that paralellize completely

poly n processors

poly log n time

Parallel complexity

Class $\mathcal{NC}$: problems that paralellize completely

poly n processors

poly log n time

Main open question: is matching in $\mathcal{NC}$?

Parallel complexity

Class $\mathcal{NC}$: problems that paralellize completely

poly n processors

it's in **Randomized** $\mathcal{NC}$

poly log n time

Main open question: is matching in $\mathcal{NC}$?

Parallel complexity

- Matching is in RANDOMIZED $\mathcal{NC}$ [Lovász 1979]:
  has randomized algorithm that uses:
  
  - polynomially many processors
  - polylog time
- Search version is in RANDOMIZED $\mathcal{NC}$:
  
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

Parallel complexity

- Matching is in RANDOMIZED $\mathcal{NC}$ [Lovász 1979]: has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Search version is in RANDOMIZED $\mathcal{NC}$:
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

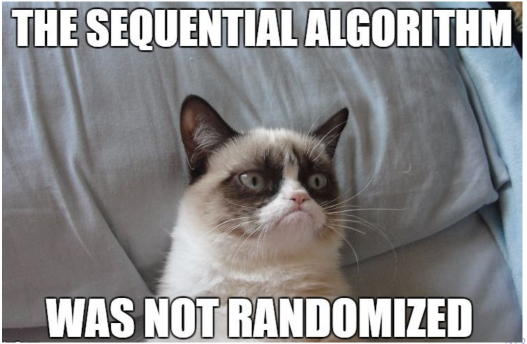
first matching algorithm
to use Tutte's matrix
and Zippel–Schwartz Lemma

led to understanding of
computational relationship between
search and decision problems

introduced
the Isolation Lemma

Parallel complexity

- Matching is in Randomized $\mathcal{NC}$ [Lovász 1979]:
  has randomized algorithm that uses:
  
  - polynomially many processors
  - polylog time
- Search version is in Randomized $\mathcal{NC}$:
  
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

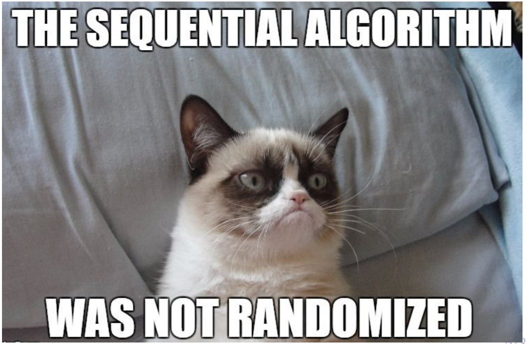


Parallel complexity

- Matching is in RANDOMIZED $\mathcal{NC}$ [Lovász 1979]:
  has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Search version is in RANDOMIZED $\mathcal{NC}$:
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

Can we derandomize
all efficient computation?

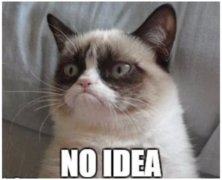


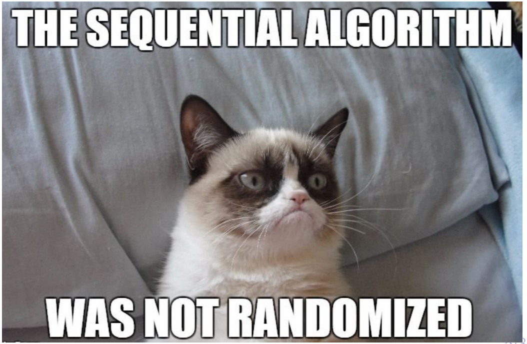
Parallel complexity

- Matching is in RANDOMIZED $\mathcal{NC}$ [Lovász 1979]:
  has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Search version is in RANDOMIZED $\mathcal{NC}$:
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

Can we derandomize
all efficient computation?

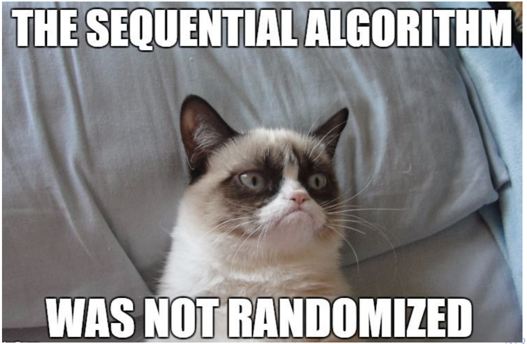




- Matching is in RANDOMIZED $\mathcal{NC}$ [Lovász 1979]:
  has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Search version is in RANDOMIZED $\mathcal{NC}$:
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

Can we derandomize ~~all efficient computation~~?

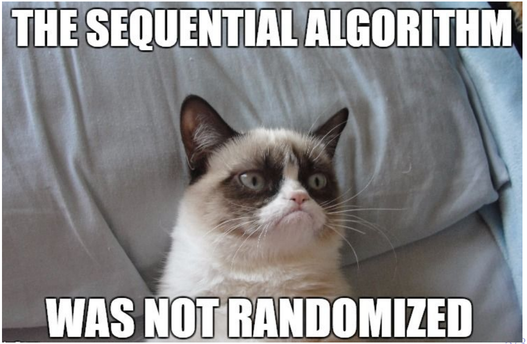


Parallel complexity

- Matching is in Randomized $\mathcal{NC}$ [Lovász 1979]: has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Search version is in Randomized $\mathcal{NC}$:
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

Can we derandomize one of these algorithms?



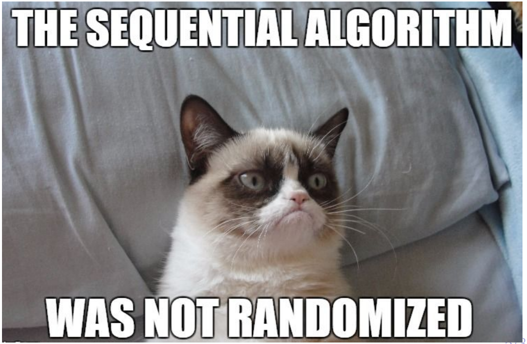
Parallel complexity

- Matching is in RANDOMIZED $\mathcal{NC}$ [Lovász 1979]:
  has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Search version is in RANDOMIZED $\mathcal{NC}$:
  - [Karp, Upfal, Wigderson 1986]
  - [Mulmuley, Vazirani, Vazirani 1987]

Can we derandomize one of these algorithms?

Is matching in $\mathcal{NC}$?



Is matching in $\mathcal{NC}$?

Yes, for restricted graph classes:

- bipartite regular [Lev, Pippenger, Valiant 1981]
- bipartite convex [Dekel, Sahni 1984]
- incomparability graphs [Kozen, Vazirani, Vazirani 1985]
- bipartite graphs with small number of perfect matchings [Grigoriev, Karpinski 1987]
- claw-free [Chrobak, Naor, Novick 1989]
- $K_{3,3}$-free (decision version) [Vazirani 1989]
- planar bipartite [Miller, Naor 1989]
- dense [Dahlhaus, Hajnal, Karpinski 1993]
- strongly chordal [Dahlhaus, Karpinski 1998]
- P_4-tidy [Parfenoff 1998]
- bipartite small genus [Mahajan, Varadarajan 2000]
- graphs with small number of perfect matchings [Agrawal, Hoang, Thierauf 2006]
- planar (search version) [Anari, Vazirani 2017]

Is matching in $\mathcal{NC}$?

Yes, for restricted graph classes:

- bipartite regular [Lev, Pippenger, Valiant 1981]
- bipartite convex [Dekel, Sahni 1984]
- incomparability graphs [Kozen, Vazirani, Vazirani 1985]
- bipartite graphs with small number of perfect matchings [Grigoriev, Karpinski 1987]
- claw-free [Chrobak, Naor, Novick 1989]
- $K_{3,3}$-free (decision version) [Vazirani 1989]
- planar bipartite [Miller, Naor 1989]
- dense [Dahlhaus, Hajnal, Karpinski 1993]
- strongly chordal [Dahlhaus, Karpinski 1998]
- P_4-tidy [Parfenoff 1998]
- bipartite small genus [Mahajan, Varadarajan 2000]
- graphs with small number of perfect matchings [Agrawal, Hoang, Thierauf 2006]
- planar (search version) [Anari, Vazirani 2017]

but not known for:

- general

Is matching in $\mathcal{NC}$?

Yes, for restricted graph classes:

- bipartite regular [Lev, Pippenger, Valiant 1981]
- bipartite convex [Dekel, Sahni 1984]
- incomparability graphs [Kozen, Vazirani, Vazirani 1985]
- bipartite graphs with small number of perfect matchings [Grigoriev, Karpinski 1987]
- claw-free [Chrobak, Naor, Novick 1989]
- $K_{3,3}$-free (decision version) [Vazirani 1989]
- planar bipartite [Miller, Naor 1989]
- dense [Dahlhaus, Hajnal, Karpinski 1993]
- strongly chordal [Dahlhaus, Karpinski 1998]
- P_4-tidy [Parfenoff 1998]
- bipartite small genus [Mahajan, Varadarajan 2000]
- graphs with small number of perfect matchings [Agrawal, Hoang, Thierauf 2006]
- planar (search version) [Anari, Vazirani 2017]

but not known for:

- general
- bipartite

Is matching in $\mathcal{NC}$?

Fenner, Gurjar and Thierauf [2015] showed:

- Bipartite matching is in QUASI-$\mathcal{NC}$
  ($n^{\text{poly}\log n}$ processors, $\text{poly}\log n$ time, deterministic)



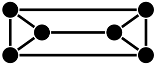
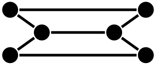
Is matching in $\mathcal{NC}$?

Fenner, Gurjar and Thierauf [2015] showed:

- Bipartite matching is in QUASI-$\mathcal{NC}$
  ($n^{\text{poly}\log n}$ processors, $\text{poly}\log n$ time, deterministic)



- Approach fails for non-bipartite graphs

 much harder than 

Our result

We show: general matching is in QUASI-$\mathcal{NC}$:

- $n^{\text{poly}\log n}$ processors
- $\text{poly}\log n$ time
- deterministic

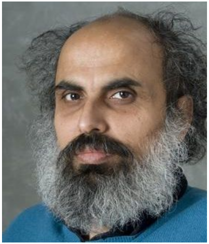
Outline

1. Isolating weight functions
   [Mulmuley, Vazirani, Vazirani 1987]

2. Bipartite case
   [Fenner, Gurjar, Thierauf 2015]

3. Difficulties of general case
   & our approach

1. Isolating weight functions [Mulmuley, Vazirani, Vazirani 1987]



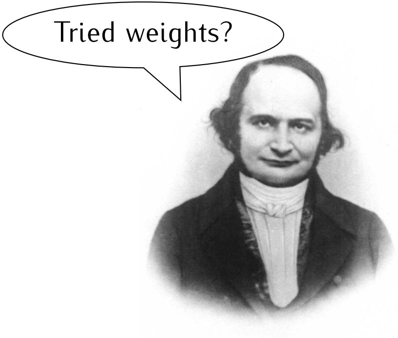
Isolating weight functions

Difficulty:
too many possible perfect matchings

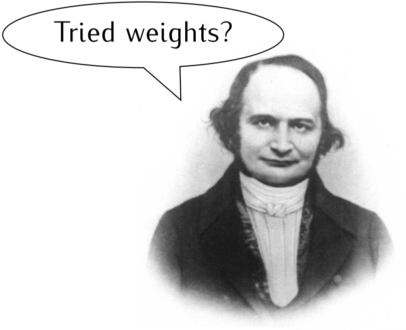
Isolating weight functions

Difficulty:
too many possible perfect matchings

Isolating weight functions



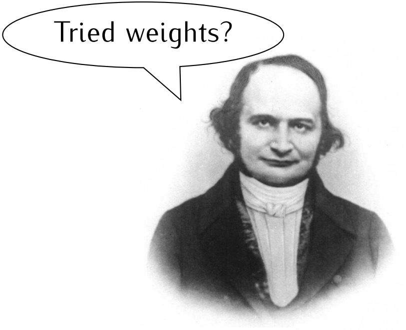
Difficulty:
too many possible perfect matchings



Difficulty:
too many possible perfect matchings

Solution: look for a **min-weight** perfect matching

Isolating weight functions

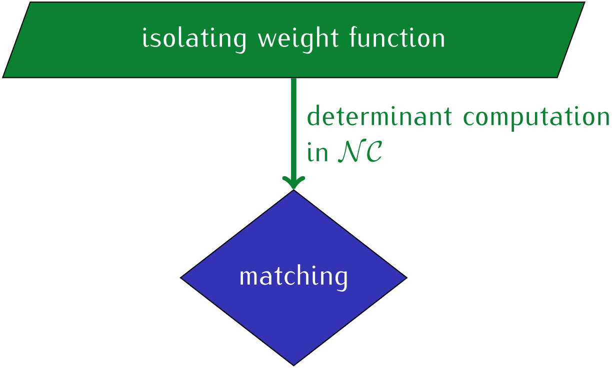


Difficulty:
too many possible perfect matchings

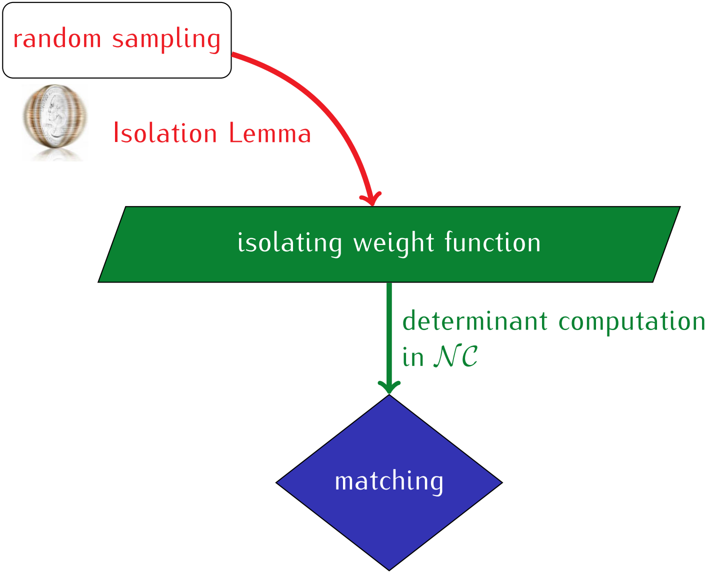
Solution: look for a **min-weight** perfect matching

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** min-weight perfect matching

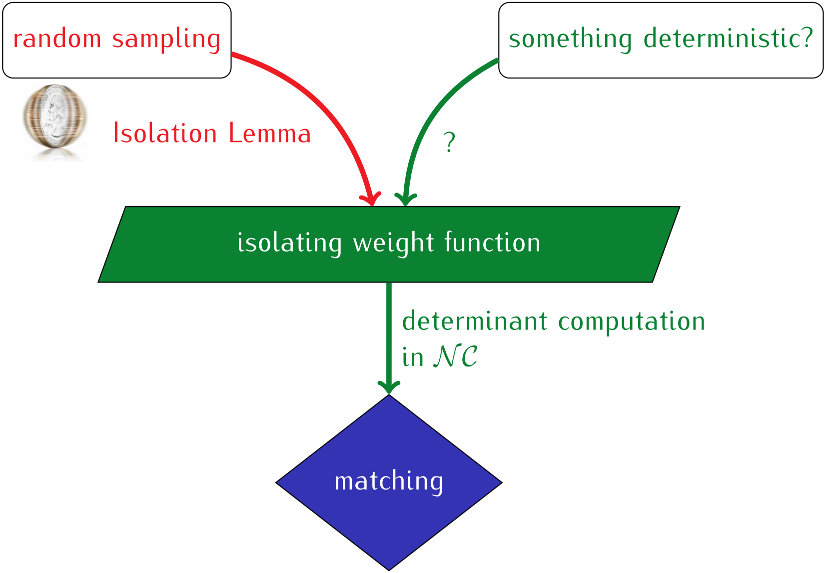
[Mulmuley, Vazirani, Vazirani 1987]



[Mulmuley, Vazirani, Vazirani 1987]



[Mulmuley, Vazirani, Vazirani 1987]



Isolation Lemma

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** min-weight perfect matching

Isolation Lemma [MVV 1987]

If each $w(e)$ picked randomly from $\{1, 2, \ldots, n^3\}$,
then $\mathrm{P}[w \text{ isolating}] \geq 1 - \frac{1}{n}$

Isolation Lemma

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** min-weight perfect matching

Isolation Lemma [MVV 1987]

If each $w(e)$ picked randomly from $\{1, 2, ..., n^3\}$,
then $P[w \text{ isolating}] \geq 1 - \frac{1}{n}$

- holds more generally,
  for any set family in place of matchings!
- many applications in complexity theory
- related to Polynomial Identity Testing

Derandomize the Isolation Lemma

- **Challenge**:
  get an isolating weight function
  deterministically in $\mathcal{NC}$

- **We prove**:
  can construct $n^{O(\log^2 n)}$ weight functions in QUASI-$\mathcal{NC}$
  such that one of them is isolating

- We do it without looking at the graph

- **Implies: matching is in QUASI-$\mathcal{NC}$**

Special case of derandomizing Polynomial Identity Testing
– for the polynomial being $\det T(G)$

Derandomize the Isolation Lemma

- **Challenge**:
  get an isolating weight function
  deterministically in $\mathcal{NC}$
- **We prove**:
  can construct $n^{O(\log^2 n)}$ weight functions in QUASI-$\mathcal{NC}$
  such that one of them is isolating
- We do it without looking at the graph
- **Implies: matching is in QUASI-$\mathcal{NC}$**

Special case of derandomizing Polynomial Identity Testing
– for the polynomial being $\det T(G)$

Derandomize the Isolation Lemma

- **Challenge**:
  get an isolating weight function
  deterministically in $\mathcal{NC}$

- **We prove**:
  can construct $n^{O(\log^2 n)}$ weight functions in QUASI-$\mathcal{NC}$
  such that one of them is isolating

- We do it without looking at the graph

- **Implies: matching is in QUASI-$\mathcal{NC}$**

Special case of derandomizing Polynomial Identity Testing
– for the polynomial being $\det T(G)$

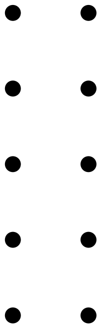
2. Bipartite case [Fenner, Gurjar, Thierauf 2015]

Goal: how to construct $n^{O(\log n)}$ weight functions such that one of them is isolating?

Isolating matchings

What if w is **not** isolating?

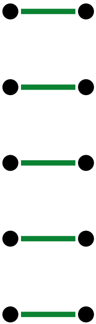
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum



Isolating matchings

What if w is **not** isolating?

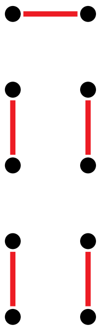
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum



Isolating matchings

What if w is **not** isolating?

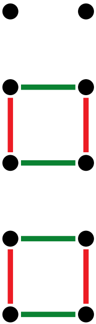
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum



Isolating matchings

What if w is **not** isolating?

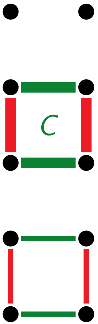
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum
- symmetric difference $=$ alternating cycles



Isolating matchings

What if w is **not** isolating?

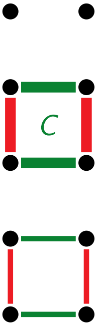
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum
- symmetric difference $=$ alternating cycles
- in each cycle C,
  $w(\text{GREEN}) = w(\text{RED})$
  (otherwise could get lighter matching)



Isolating matchings

What if w is **not** isolating?

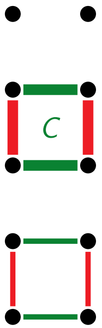
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum
- symmetric difference = alternating cycles
- in each cycle C, $w(\text{GREEN}) = w(\text{RED})$ (otherwise could get lighter matching)
- define **discrepancy** of a cycle: $d_w(C) := w(\text{GREEN}) - w(\text{RED})$



Isolating matchings

What if w is **not** isolating?

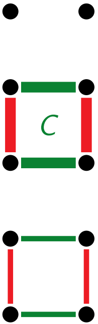
- there are perfect matchings M, M' with $w(M) = w(M')$ minimum
- symmetric difference = alternating cycles
- in each cycle C, $w(\text{GREEN}) = w(\text{RED})$ (otherwise could get lighter matching)
- define **discrepancy** of a cycle: $d_w(C) := w(\text{GREEN}) - w(\text{RED})$
- $d_w(C) = 0$



Isolating matchings

What if w is **not** isolating?

- there are perfect matchings M, M' with $w(M) = w(M')$ minimum
- symmetric difference = alternating cycles
- in each cycle C, $w(\text{GREEN}) = w(\text{RED})$ (otherwise could get lighter matching)
- define **discrepancy** of a cycle: $d_w(C) := w(\text{GREEN}) - w(\text{RED})$
- $d_w(C) = 0$

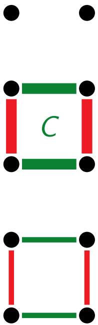


If $(\forall C)\ d_w(C) \neq 0$, then w isolating!

Isolating matchings

What if w is **not** isolating?

- there are perfect matchings M, M' with $w(M) = w(M')$ minimum
- symmetric difference $=$ alternating cycles
- in each cycle C, $w(\text{GREEN}) = w(\text{RED})$ (otherwise could get lighter matching)
- define **discrepancy** of a cycle: $d_w(C) := w(\text{GREEN}) - w(\text{RED})$
- $d_w(C) = 0$



If $(\forall C)\ d_w(C) \neq 0$, then w isolating!

New objective: assign $\neq 0$ discrepancy to every cycle

Removing cycles

New objective: assign $\neq 0$ discrepancy to every cycle

Removing cycles

New objective: assign $\neq 0$ discrepancy to every cycle

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
assigns **all of them** $\neq 0$ discrepancy.

Removing cycles

New objective: assign $\neq 0$ discrepancy to every cycle

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
assigns **all of them** $\neq 0$ discrepancy.

Removing cycles

New objective: assign $\neq 0$ discrepancy to every cycle

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
assigns **all of them** $\neq 0$ discrepancy.

If $\leq n^4$ cycles in the graph: done!

Removing cycles

New objective: assign $\neq 0$ discrepancy to every cycle

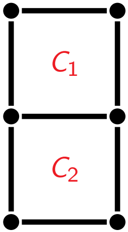
Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
assigns **all of them** $\neq 0$ discrepancy.

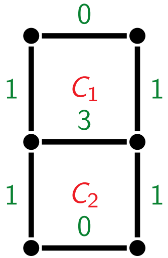
If $\leq n^4$ cycles in the graph: done!

Not so easy, but we can cope with all **4-cycles**.

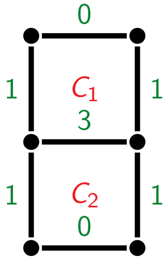
Removing cycles



Removing cycles

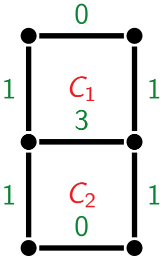


Removing cycles



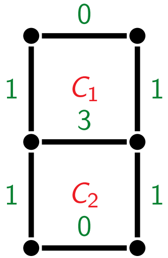
$d_w(C_1) = 1 \neq 0$
$d_w(C_2) = 1 \neq 0$

Active subgraph:
those edges that are in a min-weight perfect matching

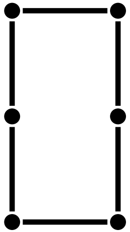


$d_w(C_1) = 1 \neq 0$
$d_w(C_2) = 1 \neq 0$
$\Longrightarrow$

Active subgraph:
those edges that are in a min-weight perfect matching



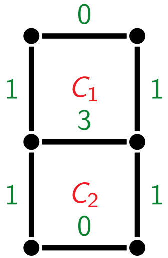
$d_w(C_1) = 1 \neq 0$
$d_w(C_2) = 1 \neq 0$
$\Longrightarrow$



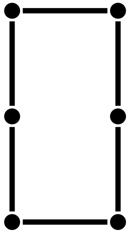
Active subgraph:
those edges that are in a min-weight perfect matching

Bipartite key property

Once we assign a cycle $\neq 0$ discrepancy,
it will disappear from the active subgraph.



$$d_w(C_1) = 1 \neq 0$$
$$d_w(C_2) = 1 \neq 0$$
$$\Longrightarrow$$



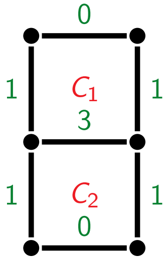
Removing cycles

Active subgraph:
those edges that are in a min-weight perfect matching

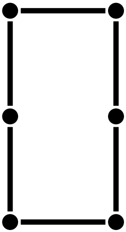
Bipartite key property

Once we assign a cycle $\neq 0$ discrepancy,
it will disappear from the active subgraph.

That is, any perfect matching in the active subgraph is min-weight.



$d_w(C_1) = 1 \neq 0$
$d_w(C_2) = 1 \neq 0$
$\Longrightarrow$



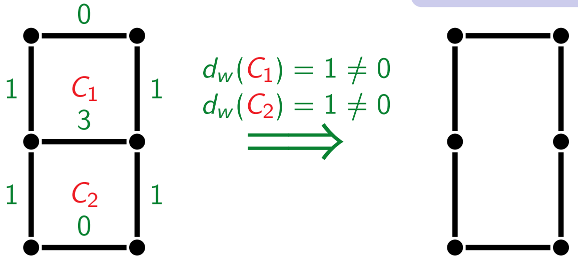
Removing cycles

Active subgraph:
those edges that are in a min-weight perfect matching

Bipartite key property

Once we assign a cycle $\neq 0$ discrepancy,
it will disappear from the active subgraph.

That is, any perfect matching in the active subgraph is min-weight.



By assigning $\neq 0$ discrepancy to 4-cycles, we can remove them.
Then continue restricted to the smaller active subgraph!

Isolating in stages

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
assigns **all of them** $\neq 0$ discrepancy.

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$w = w_1$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$w = w_1$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$w = w_1$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$w = w_1$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$ $\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$w = \langle w_1, w_2 \rangle$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$ $\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2 \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2 \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2, w_3 \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$ $\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2, w_3 \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2, w_3, \dots \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\Longrightarrow$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \left\langle w_1, w_2, w_3, \ldots, w_{\log n} \right\rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$ $\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2, w_3, \ldots, w_{\log n} \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \langle w_1, w_2, w_3, \dots, w_{\log n} \rangle$$

Lemma

There is a poly-sized set $\mathcal{W}$ of weight functions such that:
for any n^4 cycles,
some $w \in \mathcal{W}$
removes **all of them**.

Counting argument

No cycles of length $\leq r$
$\implies$ only n^4 cycles of length $\leq 2r$

- active subgraph has $\leq n^4$ 4-cycles
  - apply $w_1 \in \mathcal{W}$
    - active subgraph has no 4-cycles
- active subgraph has $\leq n^4$ 8-cycles
  - apply $w_2 \in \mathcal{W}$
    - active subgraph has no 8-cycles
- active subgraph has $\leq n^4$ 16-cycles
  - apply $w_3 \in \mathcal{W}$
    - active subgraph has no 16-cycles
- ...
  - apply $w_{\log n} \in \mathcal{W}$
    - active subgraph has no cycles whatsoever
    - success!

Isolating in stages

$$w = \left\langle w_1, w_2, \ldots, w_{\log n} \right\rangle$$

- For each stage i, some $w_i \in \mathcal{W}$ removes the wanted cycles
- So some concatenation $\left\langle w_1, w_2, \ldots, w_{\log n} \right\rangle$ is isolating
- But not sure how to check in $\mathcal{NC}$ if given w_i is good...

The oblivious algorithm checks all concatenations:

$$|\mathcal{W}|^{\log n} = n^{O(\log n)}$$

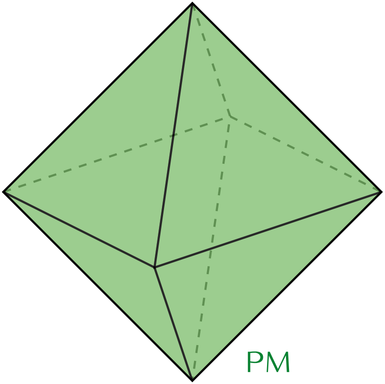
3. Difficulties of general case & our approach

Bipartite key property fails

~~**Bipartite key property**~~

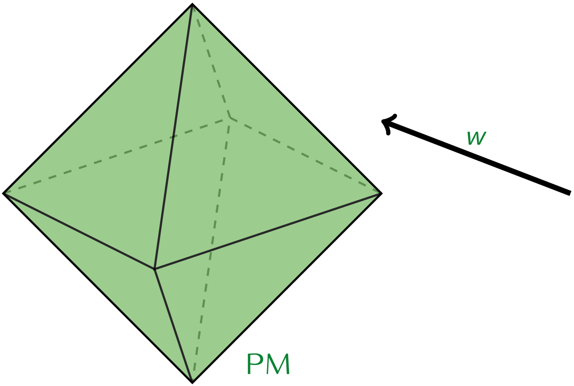
~~Once we assign a cycle $\neq 0$ discrepancy, it will disappear from the active subgraph.~~

- PM: perfect matching polytope
  (convex hull of all perfect matchings)



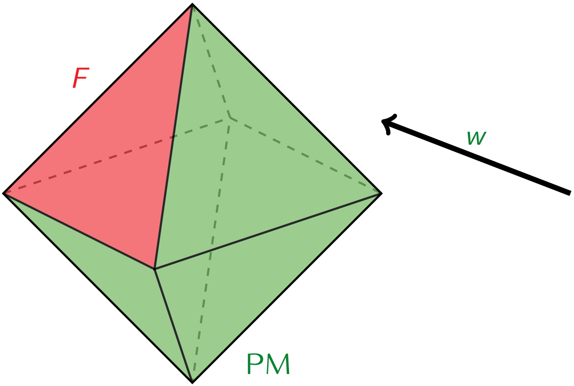
Polyhedral perspective

- PM: perfect matching polytope
  (convex hull of all perfect matchings)



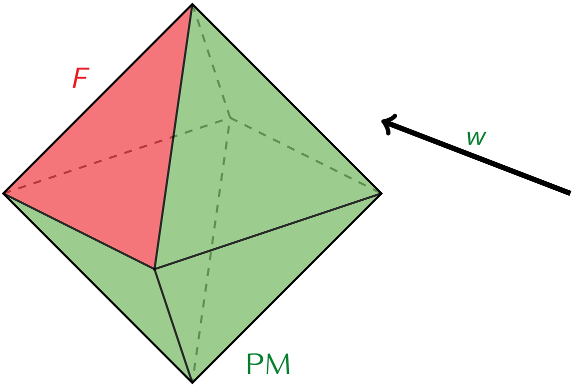
Polyhedral perspective

- PM: perfect matching polytope (convex hull of all perfect matchings)
- F: set of points in PM that minimize w
  - F is a face of PM



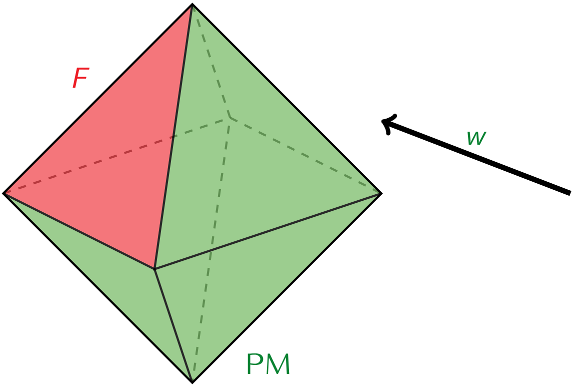
Polyhedral perspective

- PM: perfect matching polytope (convex hull of all perfect matchings)
- F: set of points in PM that minimize w
  - F is a face of PM
- w isolating $\iff |F| = 1$ (F is a vertex)



Polyhedral perspective

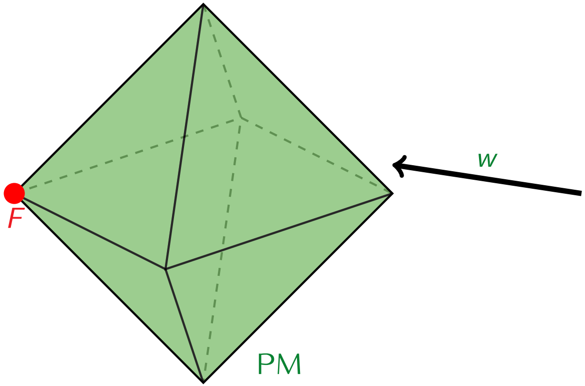
- PM: perfect matching polytope (convex hull of all perfect matchings)
- F: set of points in PM that minimize w
  - F is a face of PM
- w isolating $\iff |F| = 1$ (F is a vertex)



w not isolating

Polyhedral perspective

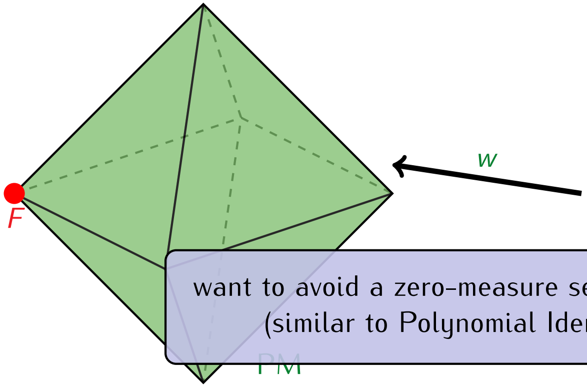
- PM: perfect matching polytope (convex hull of all perfect matchings)
- F: set of points in PM that minimize w
  - F is a face of PM
- w isolating $\iff |F| = 1$ (F is a vertex)



w isolating

Polyhedral perspective

- PM: perfect matching polytope (convex hull of all perfect matchings)
- F: set of points in PM that minimize w
  - F is a face of PM
- w isolating $\iff |F| = 1$ (F is a vertex)

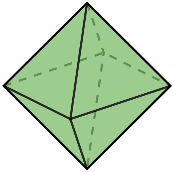


w isolating

want to avoid a zero-measure set deterministically
(similar to Polynomial Identity Testing)

Polyhedral perspective

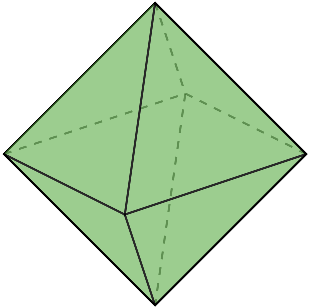
1



isolating in stages

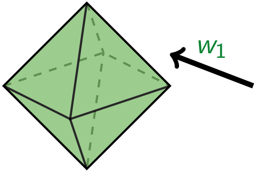
=

decreasing sequence of faces

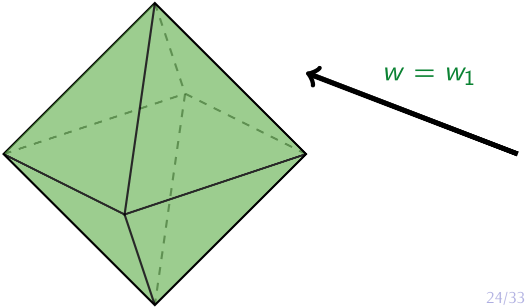


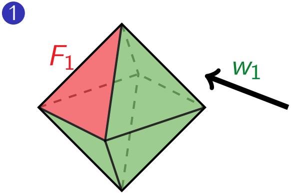
Polyhedral perspective

1

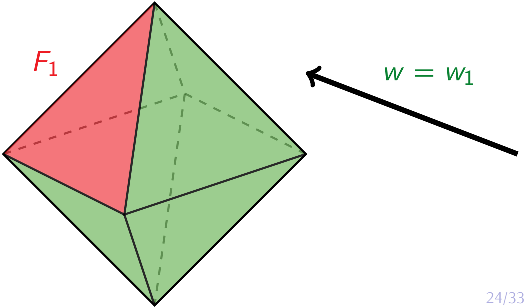


isolating in stages
=
decreasing sequence of faces



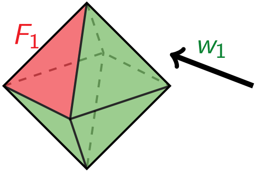


isolating in stages
=
decreasing sequence of faces

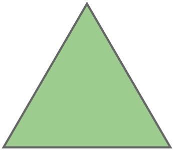


Polyhedral perspective

1



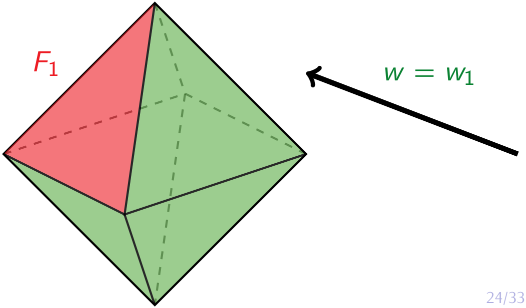
2



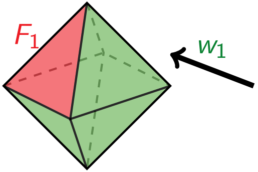
isolating in stages

=

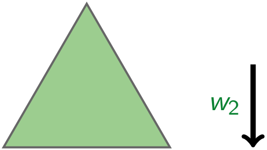
decreasing sequence of faces



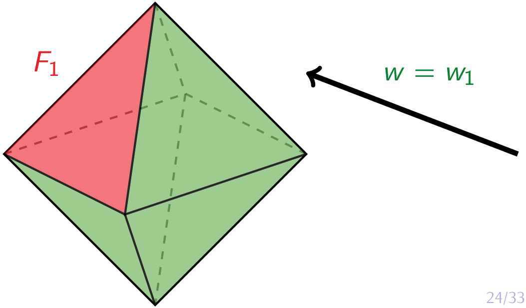
1



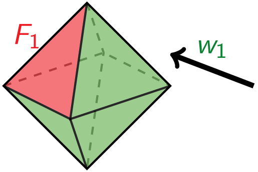
2



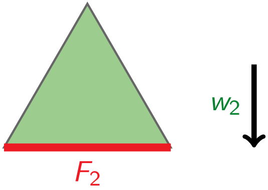
isolating in stages
=
decreasing sequence of faces



1



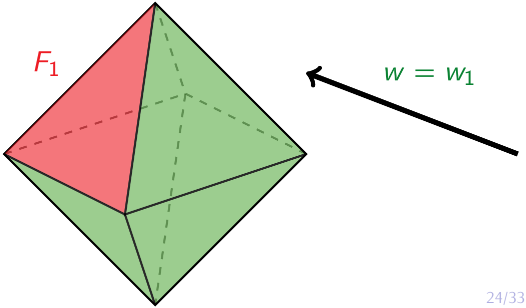
2



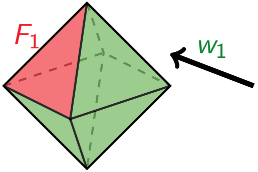
isolating in stages

=

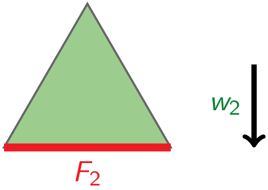
decreasing sequence of faces



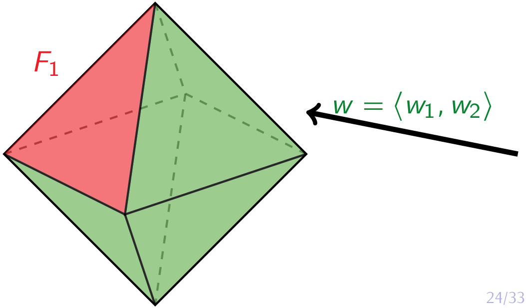
1



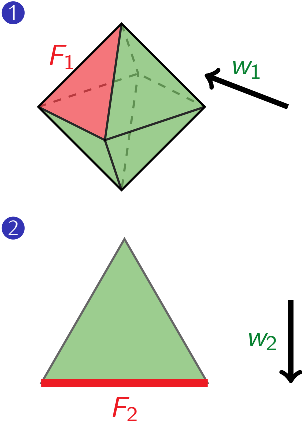
2



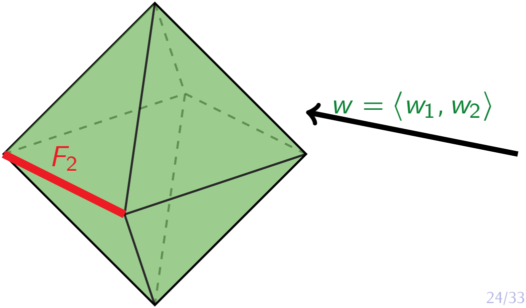
isolating in stages
=
decreasing sequence of faces



Polyhedral perspective

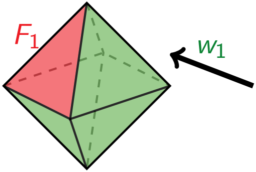


isolating in stages
=
decreasing sequence of faces

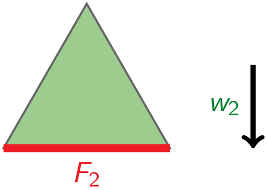


Polyhedral perspective

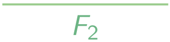
1



2



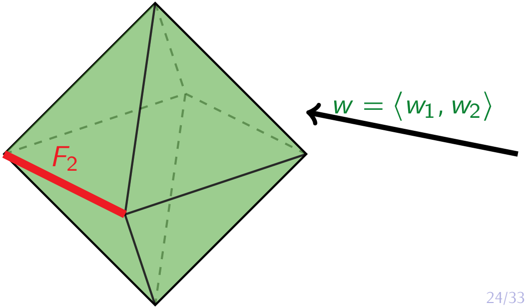
3



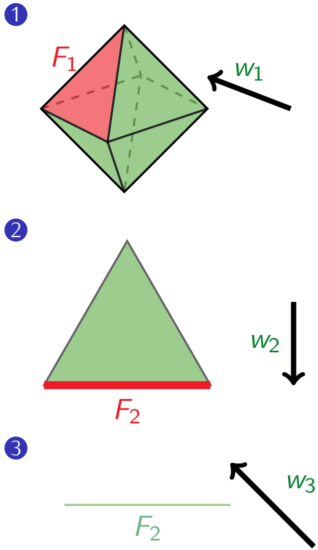
isolating in stages

=

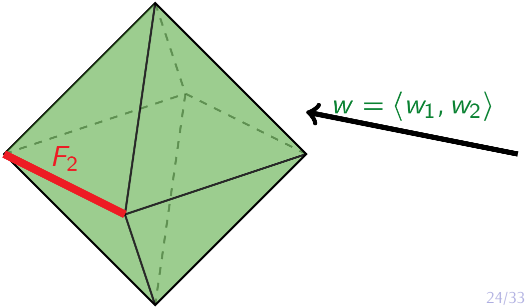
decreasing sequence of faces



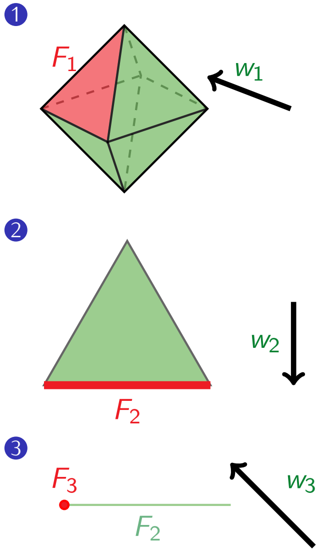
Polyhedral perspective



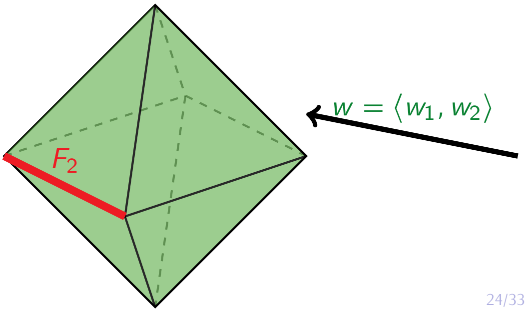
isolating in stages
=
decreasing sequence of faces



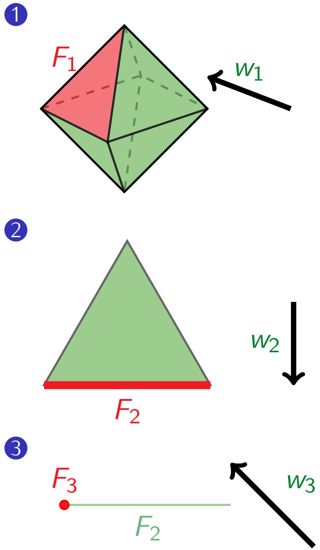
Polyhedral perspective



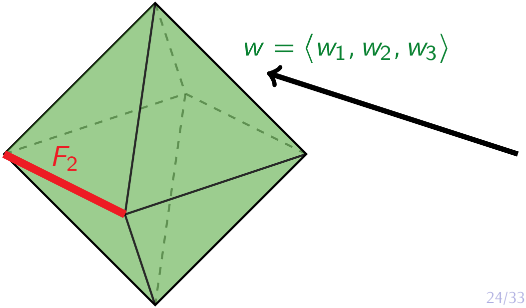
isolating in stages
=
decreasing sequence of faces



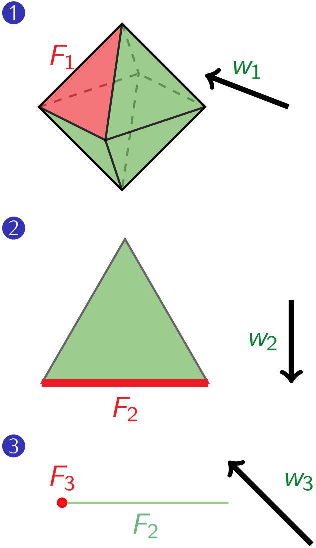
Polyhedral perspective



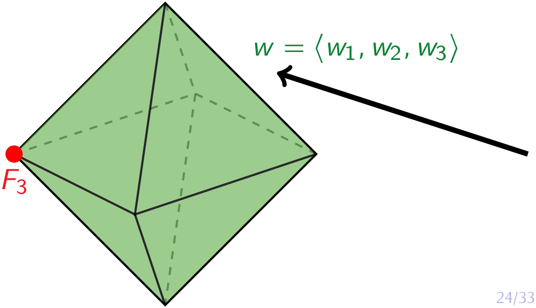
isolating in stages
=
decreasing sequence of faces



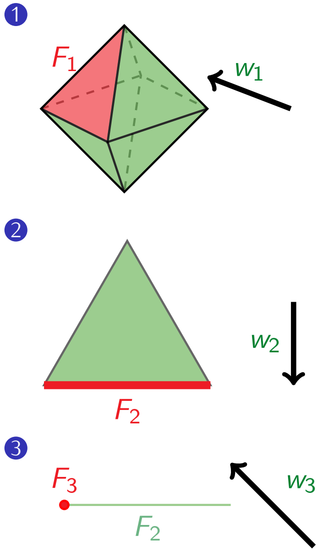
Polyhedral perspective



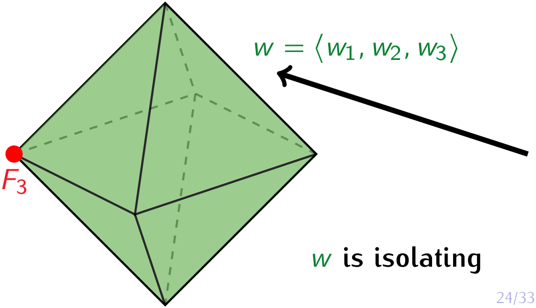
isolating in stages
=
decreasing sequence of faces



Polyhedral perspective

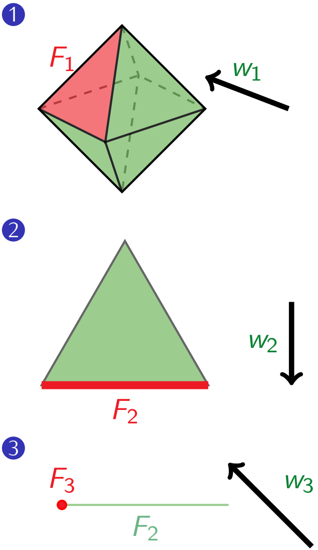


isolating in stages
=
decreasing sequence of faces



w **is isolating**

Polyhedral perspective



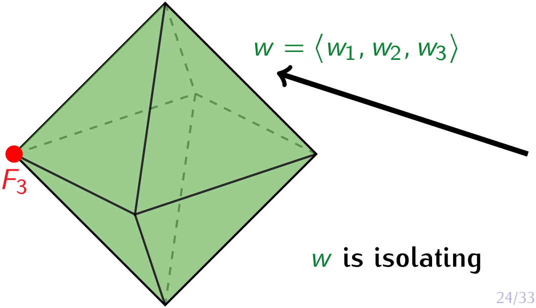
isolating in stages

=

decreasing sequence of faces

decreasing fast due to the bipartite matching polytope:

- bipartite key property: every face is a subgraph
- so girth doubles at every step



w **is isolating**

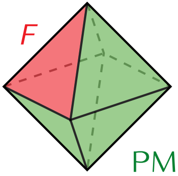
LP formulation

Edmonds [1965]

PM described as set of $x \in \mathbb{R}^E$ such that:

- $x_e \geq 0$ for every edge e
- $x(\delta(v)) = 1$ for every vertex v
- $x(\delta(S)) \geq 1$ for every odd set S of vertices

($\delta(S)$ = edges crossing S)



Edmonds [1965]

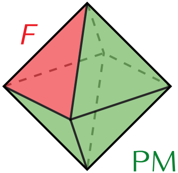
PM described as set of $x \in \mathbb{R}^E$ such that:

- $x_e \geq 0$ for every edge e
- $x(\delta(v)) = 1$ for every vertex v
- $x(\delta(S)) \geq 1$ for every odd set S of vertices

($\delta(S)$ = edges crossing S)

So every face F is given as:

$$F = \{x \in \mathrm{PM} : \begin{array}{ll} x_e = 0 & \text{for some edges } e, \\ x(\delta(S)) = 1 & \text{for some odd sets } S\} \end{array}$$



LP formulation

Edmonds [1965]

PM described as set of $x \in \mathbb{R}^E$ such that:

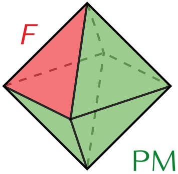
- $x_e \geq 0$ for every edge e
- $x(\delta(v)) = 1$ for every vertex v
- $x(\delta(S)) \geq 1$ for every odd set S of vertices

($\delta(S)$ = edges crossing S)

So every face F is given as:

$$F = \{x \in \text{PM} : x_e = 0 \quad \text{for some edges } e, \quad x(\delta(S)) = 1 \quad \text{for some odd sets } S\}$$

- In bipartite case:
  $F = \{x \in \text{PM} : \; x_e = 0 \;$ for some edges $e\}$
  (F given by the active subgraph)
- Now, faces are exponentially harder
- Need $2^{\Omega(n)}$ inequalities [Rothvoss 2013]



LP formulation

Edmonds [1965]

PM described as set of $x \in \mathbb{R}^E$ such that:

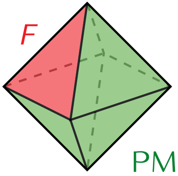
- $x_e \geq 0$ for every edge e
- $x(\delta(v)) = 1$ for every vertex v
- $x(\delta(S)) \geq 1$ for every odd set S of vertices

($\delta(S)$ = edges crossing S)

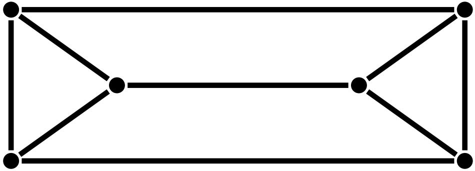
Bipartite key property fails!

$x(\delta(S)) = 1$ for some odd sets $S\}$

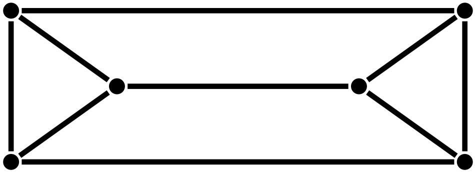
- In bipartite case:
  $F = \{x \in \mathrm{PM} : \ x_e = 0 \ \text{for some edges } e\}$
  (F given by the active subgraph)
- Now, faces are exponentially harder
- Need $2^{\Omega(n)}$ inequalities [Rothvoss 2013]



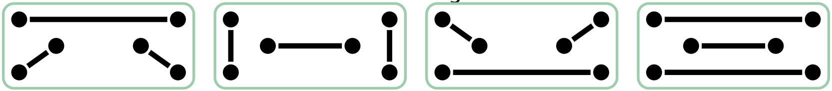
How bipartite key property fails



How bipartite key property fails



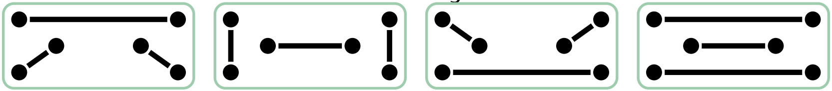
PM: convex hull of all four matchings:



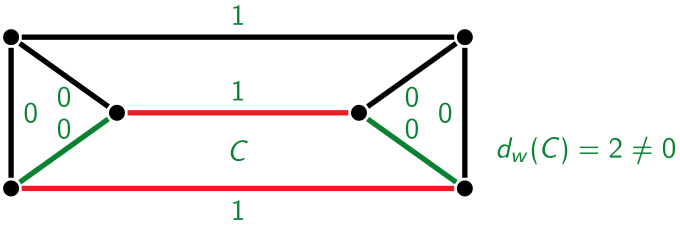
How bipartite key property fails

want:
$d_w(C) \neq 0$

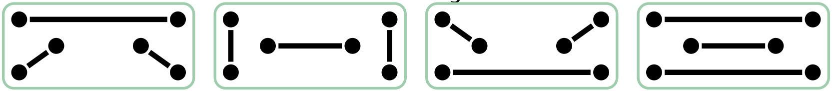
PM: convex hull of all four matchings:



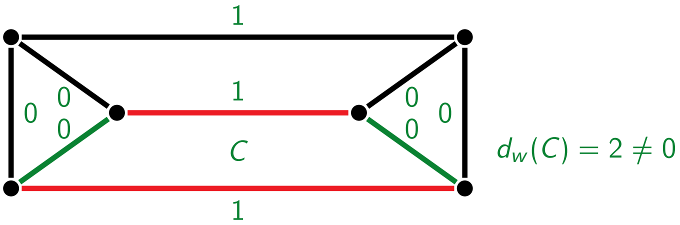
How bipartite key property fails



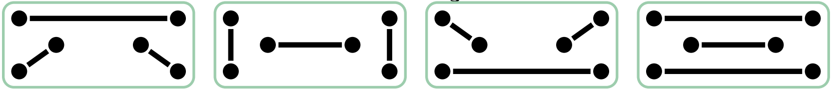
PM: convex hull of all four matchings:



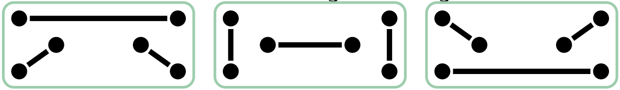
How bipartite key property fails



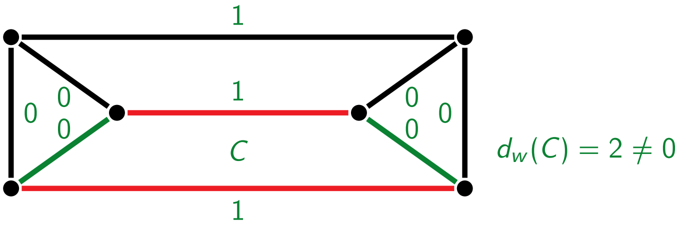
PM: convex hull of all four matchings:



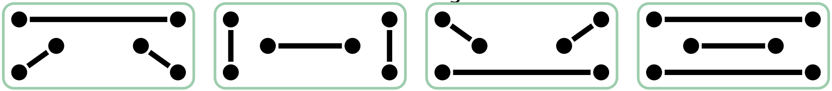
F: convex hull of matchings of weight 1:



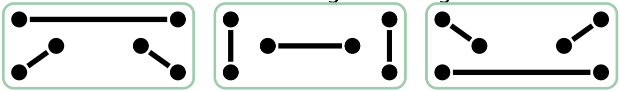
How bipartite key property fails



PM: convex hull of all four matchings:

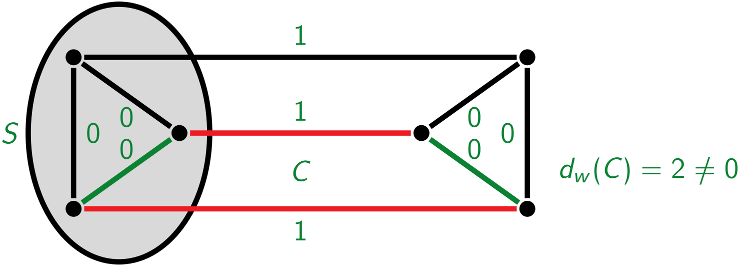


F: convex hull of matchings of weight 1:

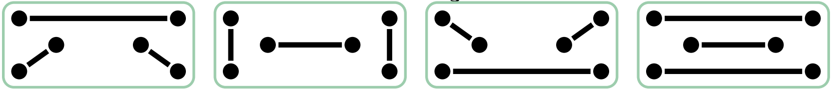


$F \subsetneq PM$ but still has all edges... 😢

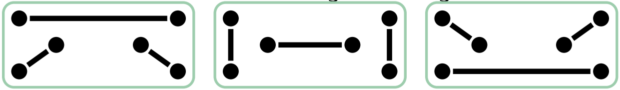
How bipartite key property fails



PM: convex hull of all four matchings:



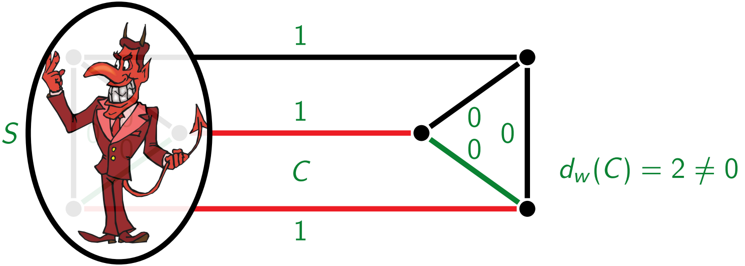
F: convex hull of matchings of weight 1:



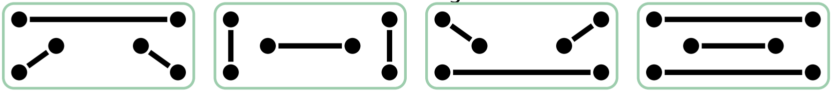
$F \subsetneq PM$ but still has all edges...

$F = \{x \in PM : x(\delta(S)) = 1\}$

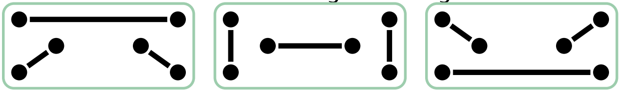
How bipartite key property fails



PM: convex hull of all four matchings:



F: convex hull of matchings of weight 1:



$F \subsetneq PM$ but still has all edges... 

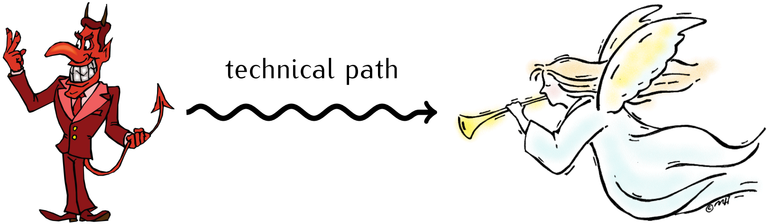
$F = \{x \in PM : x(\delta(S)) = 1\}$

How we cope

How we cope

technical path

How we cope



Main ingredients:

- Laminar family of tight cut constraints
- Tight cut constraints decompose the instance
  - $\Rightarrow$ divide-and-conquer approach

Every face F is given as:

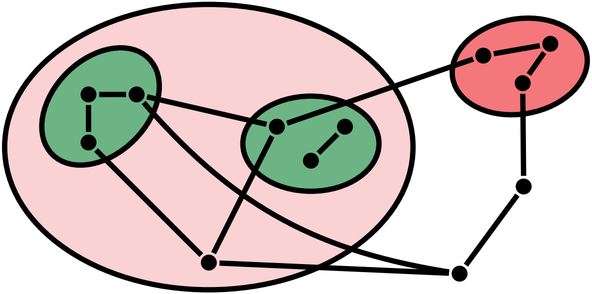
$$F = \{x \in \mathrm{PM} : \begin{array}{ll} x_e = 0 & \text{for some edges } e, \\ x(\delta(S)) = 1 & \text{for some odd sets } S\} \end{array}$$

Every face F is given as:

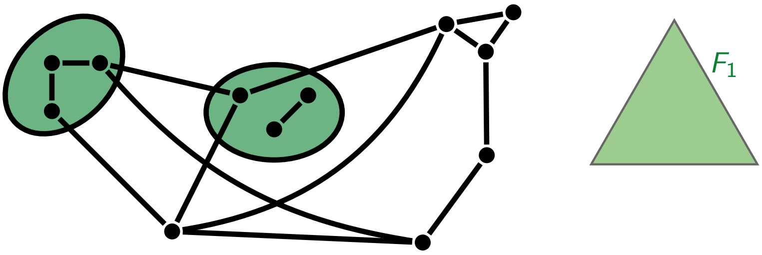
$$F = \{x \in \mathrm{PM} : x_e = 0 \quad \text{for some edges } e, \quad x(\delta(S)) = 1 \quad \text{for some odd sets } S\}$$

Great news: “some” can be chosen to be a laminar family!

(at most $n/2$ constraints instead of exponentially many to describe a face)

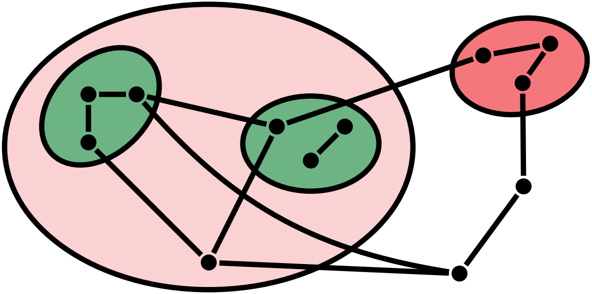


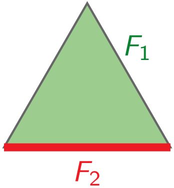
Laminarity



face $\sim$ (edge subset, laminar family)

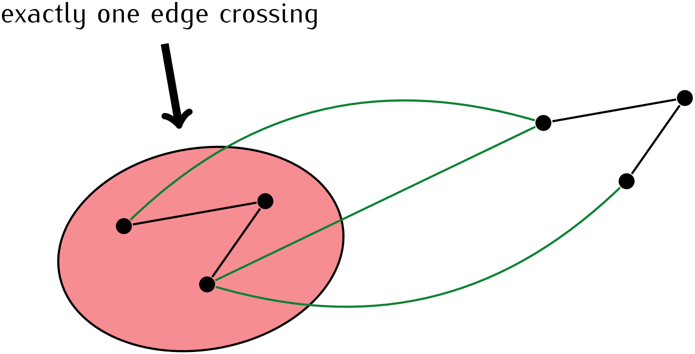
Laminarity





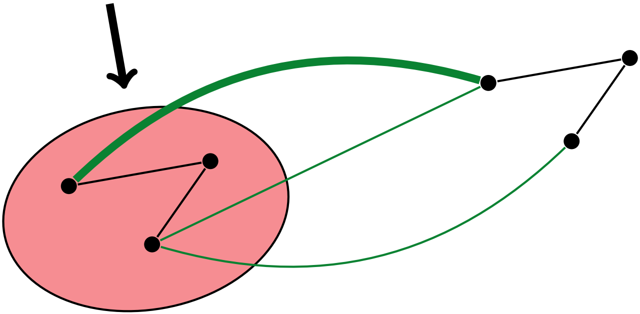
face $\sim$ (edge subset, laminar family)

Tight odd cuts are not all bad



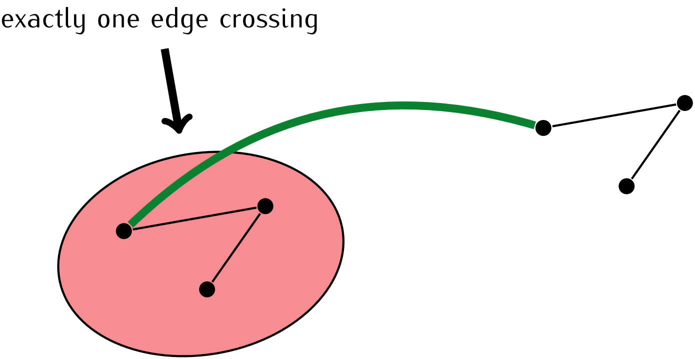
- once we fix a boundary edge...

Tight odd cuts are not all bad



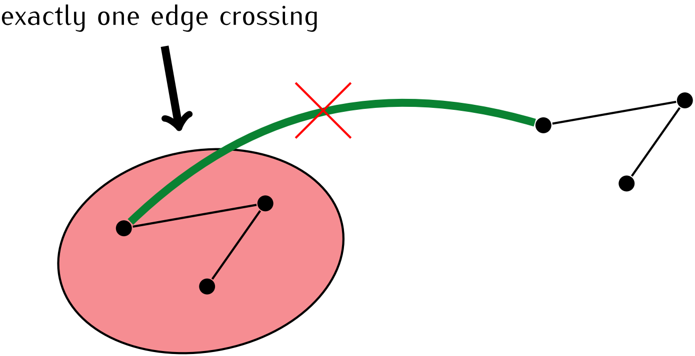
- once we fix a boundary edge...

Tight odd cuts are not all bad



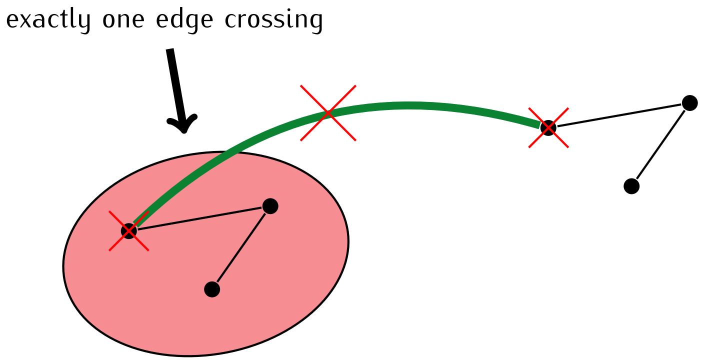
- once we fix a boundary edge...

Tight odd cuts are not all bad



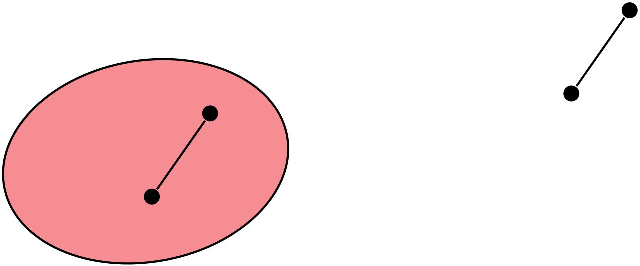
- once we fix a boundary edge...

Tight odd cuts are not all bad



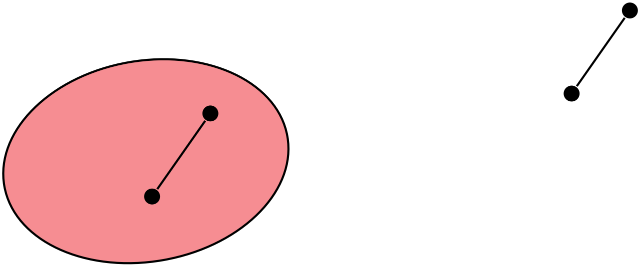
- once we fix a boundary edge...

Tight odd cuts are not all bad



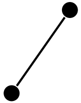
- once we fix a boundary edge...

Tight odd cuts are not all bad



- once we fix a boundary edge...
- ... the instance decomposes into two **independent** ones

Tight odd cuts are not all bad

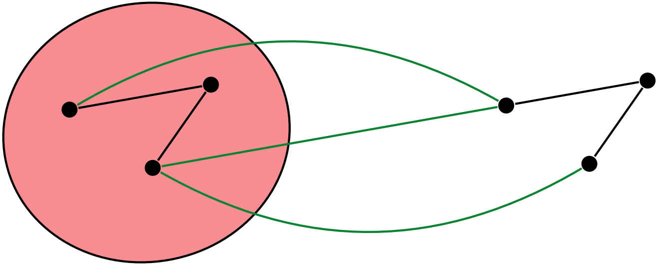


- once we fix a boundary edge...
- ... the instance decomposes into two **independent** ones

Divide & conquer

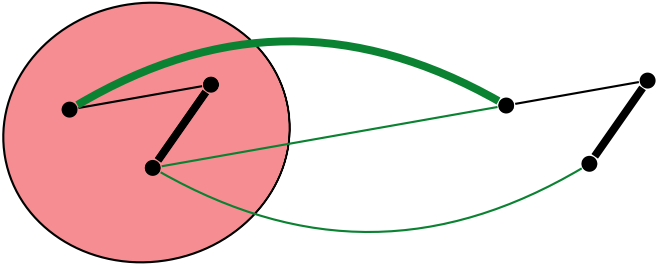
Simplest case of laminar family: only one tight odd set

Between friends: cycles that do not cross tight odd sets behave like in the bipartite case and can thus be removed



Simplest case of laminar family: only one tight odd set

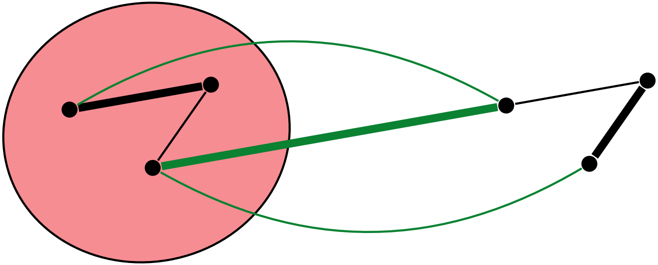
Between friends: cycles that do not cross tight odd sets behave like in the bipartite case and can thus be removed



- then every boundary edge determines entire matching

Simplest case of laminar family: only one tight odd set

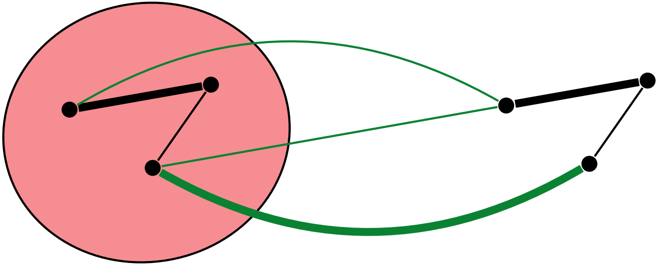
Between friends: cycles that do not cross tight odd sets behave like in the bipartite case and can thus be removed



- then every boundary edge determines entire matching

Simplest case of laminar family: only one tight odd set

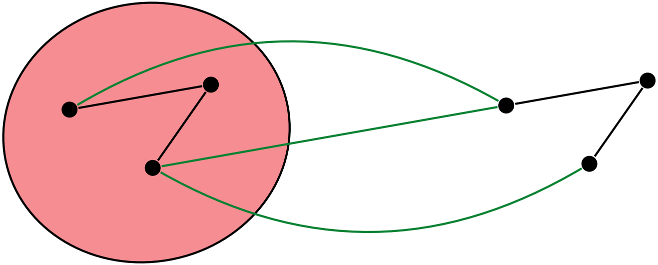
Between friends: cycles that do not cross tight odd sets behave like in the bipartite case and can thus be removed



- then every boundary edge determines entire matching

Simplest case of laminar family: only one tight odd set

Between friends: cycles that do not cross tight odd sets behave like in the bipartite case and can thus be removed

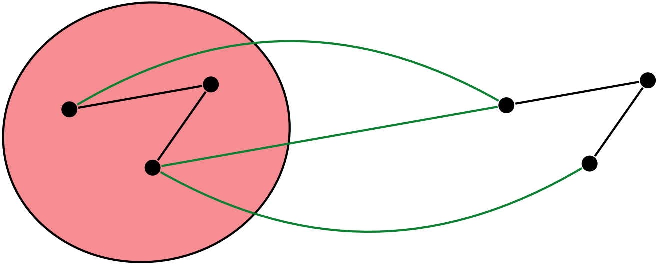


- then every boundary edge determines entire matching
- so: at most n^2 perfect matchings

Divide & conquer

Simplest case of laminar family: only one tight odd set

Between friends: cycles that do not cross tight odd sets behave like in the bipartite case and can thus be removed



- then every boundary edge determines entire matching
- so: at most n^2 perfect matchings
- some $w \in \mathcal{W}$ will give them different weights

Our dichotomy

Dichotomy:

- remove cycles not crossing tight odd-sets
- use tight odd-sets to decompose problem (divide & conquer)

Our dichotomy

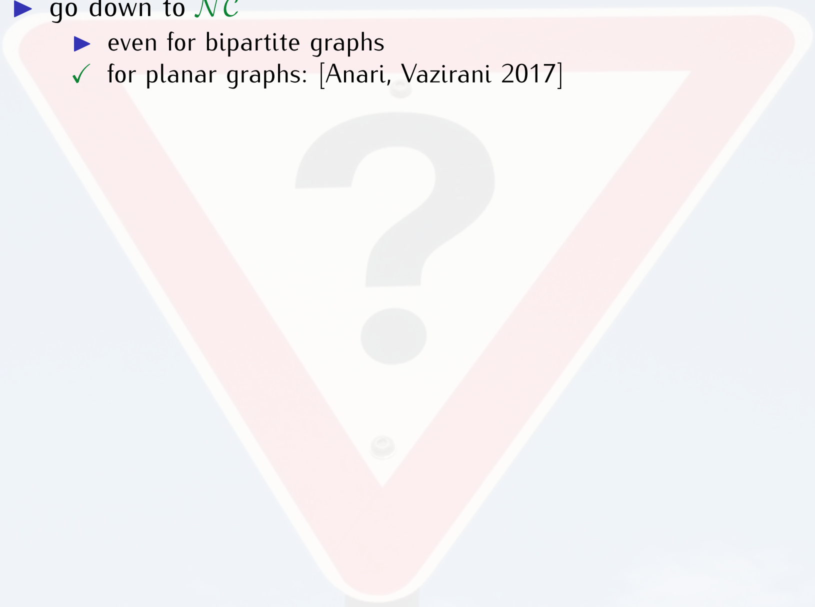
Dichotomy:

- remove cycles not crossing tight odd-sets
- use tight odd-sets to decompose problem (divide & conquer)

Details: see paper or talk to me :)

Future work

- go down to $\mathcal{NC}$
  - even for bipartite graphs
  - ✓ for planar graphs: [Anari, Vazirani 2017]



Future work

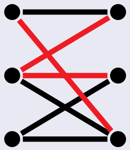
- go down to $\mathcal{NC}$
  - even for bipartite graphs
  - ✓ for planar graphs: [Anari, Vazirani 2017]

- derandomize Isolation Lemma in other cases
  - ✓ matroid intersection: [Gurjar, Thierauf 2017]
  - ✓ totally unimodular polytopes: [Gurjar, Thierauf, Vishnoi 2017]
  - any efficiently solvable 0/1-polytope?

Future work

- go down to $\mathcal{NC}$
  - even for bipartite graphs
  - ✓ for planar graphs: [Anari, Vazirani 2017]
- derandomize Isolation Lemma in other cases
  - ✓ matroid intersection: [Gurjar, Thierauf 2017]
  - ✓ totally unimodular polytopes: [Gurjar, Thierauf, Vishnoi 2017]
  - any efficiently solvable 0/1-polytope?

EXACT MATCHING



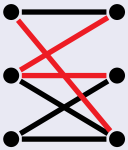
Given: graph with some edges red, number k.
Is there a perfect matching with exactly k red edges?

- randomized complexity: even RANDOMIZED $\mathcal{NC}$
- deterministic complexity: is it in $\mathcal{P}$?

Future work

- go down to $\mathcal{NC}$
  - even for bipartite graphs
  - ✓ for planar graphs: [Anari, Vazirani 2017]
- derandomize Isolation Lemma in other cases
  - ✓ matroid intersection: [Gurjar, Thierauf 2017]
  - ✓ totally unimodular polytopes: [Gurjar, Thierauf, Vishnoi 2017]
  - any efficiently solvable 0/1-polytope?

EXACT MATCHING



Given: graph with some edges red, number k.
Is there a perfect matching with exactly k red edges?

- randomized complexity: even RANDOMIZED $\mathcal{NC}$
- deterministic complexity: is it in $\mathcal{P}$?

Thank you!