Matching is in quasi-NC

Jakub Tarnawski

joint work with Ola Svensson

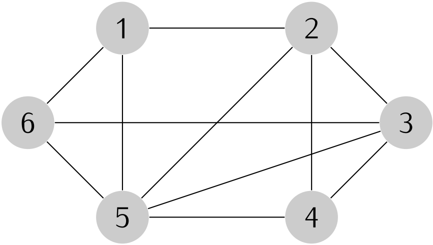
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

June 10, 2017

Perfect matching problem

- Basic question in computer science

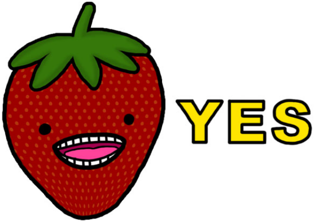
- Decision problem:
  Does given graph contain a perfect matching?

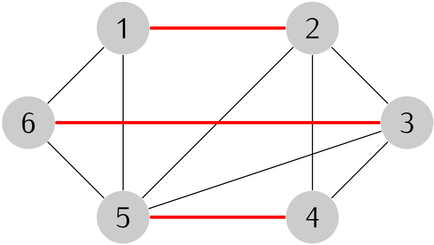


Perfect matching problem

- Basic question in computer science

- Decision problem:
  Does given graph contain a perfect matching?





Perfect matching problem

- Matching is in $\mathcal{P}$ (Edmonds 1965):
  has deterministic polytime algorithm

- Parallel algorithm?

Perfect matching problem

- Matching is in $\mathcal{P}$ (Edmonds 1965): has deterministic polytime algorithm
- It is also in RANDOMIZED $\mathcal{NC}$ (Lovász 1979): has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

Perfect matching problem

- Matching is in $\mathcal{P}$ (Edmonds 1965):
  has deterministic polytime algorithm

- It is also in RANDOMIZED $\mathcal{NC}$ (Lovász 1979):
  has randomized algorithm that uses:
  - polynomially many processors
  - polylog time

- Deterministic parallel complexity still not resolved:
  is matching in $\mathcal{NC}$?

Is matching in $\mathcal{NC}$?

Yes, for restricted graph classes:

- strongly chordal
- graphs with small number of perfect matchings
- dense
- P_4-tidy
- claw-free
- incomparability graphs
- bipartite planar
- bipartite regular
- bipartite convex

but not known for:

- bipartite
- planar (finding PM)

Is matching in $\mathcal{NC}$?

- Fenner, Gurjar and Thierauf (2015) showed:
  bipartite matching is in QUASI-$\mathcal{NC}$
  ($n^{\text{poly}\log n}$ processors, polylog time)

Is matching in $\mathcal{NC}$?

- Fenner, Gurjar and Thierauf (2015) showed:
  bipartite matching is in QUASI-$\mathcal{NC}$
  ($n^{\text{poly}\log n}$ processors, polylog time)

- We show: matching is in QUASI-$\mathcal{NC}$
  (for general graphs)

Isolating weight functions

Difficulty in parallelization:
how to coordinate machines to search for the same matching?

Difficulty in parallelization:
how to coordinate machines to search for the same matching?

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

Isolating weight functions

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** perfect matching M with minimum $w(M)$

Mulmuley, Vazirani and Vazirani (1987)

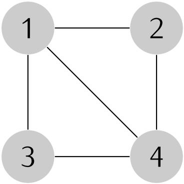
Given isolating w, can find perfect matching in $\mathcal{NC}$

Isolating weight functions

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** perfect matching M with minimum $w(M)$

Mulmuley, Vazirani and Vazirani (1987)

Given isolating w, can find perfect matching in $\mathcal{NC}$



$$T(G) = \begin{pmatrix} 0 & X_{12} & X_{13} & X_{14} \\ -X_{12} & 0 & 0 & X_{24} \\ -X_{13} & 0 & 0 & X_{34} \\ -X_{14} & -X_{24} & -X_{34} & 0 \end{pmatrix}$$

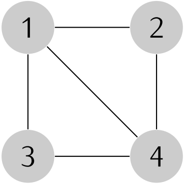
- build Tutte's matrix with entries X_{uv}
- $\det T(G) \neq 0$ (as polynomial) $\iff$ graph has perfect matching

Isolating weight functions

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** perfect matching M with minimum $w(M)$

Mulmuley, Vazirani and Vazirani (1987)

Given isolating w, can find perfect matching in $\mathcal{NC}$



$$T^w(G) = \begin{pmatrix} 0 & 2^{w(1,2)} & 2^{w(1,3)} & 2^{w(1,4)} \\ -2^{w(1,2)} & 0 & 0 & 2^{w(2,4)} \\ -2^{w(1,3)} & 0 & 0 & 2^{w(3,4)} \\ -2^{w(1,4)} & -2^{w(2,4)} & -2^{w(3,4)} & 0 \end{pmatrix}$$

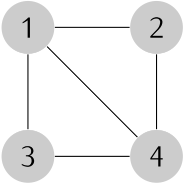
- build Tutte's matrix with entries $X_{uv} := 2^{w(u,v)}$
- $\det T^w(G) \neq 0$ (as scalar) $\iff$ graph has perfect matching

Isolating weight functions

Weight function $w : E \to \mathbb{Z}_+$ is **isolating**
if there is a **unique** perfect matching M with minimum $w(M)$

Mulmuley, Vazirani and Vazirani (1987)

Given isolating w, can find perfect matching in $\mathcal{NC}$



$$T^w(G) = \begin{pmatrix} 0 & 2^{w(1,2)} & 2^{w(1,3)} & 2^{w(1,4)} \\ -2^{w(1,2)} & 0 & 0 & 2^{w(2,4)} \\ -2^{w(1,3)} & 0 & 0 & 2^{w(3,4)} \\ -2^{w(1,4)} & -2^{w(2,4)} & -2^{w(3,4)} & 0 \end{pmatrix}$$

- build Tutte's matrix with entries $X_{uv} := 2^{w(u,v)}$
- $\det T^w(G) \neq 0$ (as scalar) $\iff$ graph has perfect matching
- we can compute determinant in $\mathcal{NC}$

Isolating weight functions

Isolation Lemma [MVV 1987]

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

Isolating weight functions

Isolation Lemma [MVV 1987]

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

Randomized $\mathcal{NC}$ algorithm [MVV 1987]

- Sample w (the only random component)
- Compute determinant (possible in $\mathcal{NC}$)
- Answer YES iff it is nonzero

Derandomize the Isolation Lemma

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

First step to derandomizing Polynomial Identity Testing?

(for polynomial being $\det T(G)$)

Derandomize the Isolation Lemma

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

First step to derandomizing Polynomial Identity Testing?

(for polynomial being $\det T(G)$)

Derandomize the Isolation Lemma

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

First step to derandomizing Polynomial Identity Testing?

(for polynomial being $\det T(G)$)

Future work

- go down to $\mathcal{NC}$
  (even for bipartite case)
- derandomize Isolation Lemma in other cases
  (totally unimodular polytopes?)
- derandomize EXACT MATCHING
  (is in RANDOMIZED $\mathcal{NC}$; is it in $\mathcal{P}$?)

Future work

- go down to $\mathcal{NC}$
  (even for bipartite case)
- derandomize Isolation Lemma in other cases
  (totally unimodular polytopes?)
- derandomize Exact Matching
  (is in Randomized $\mathcal{NC}$; is it in $\mathcal{P}$?)

Thank you!