Approximating the Number of Perfect Matchings in Bipartite Graphs

Abbas Mehrabian

amehrabi@uwaterloo.ca

University of Waterloo

April 6, 2010

What is The Problem?

Problem

Given a bipartite graph G with two parts of equal size, how many perfect matchings does G have?

Example

The following graph has two perfect matchings:

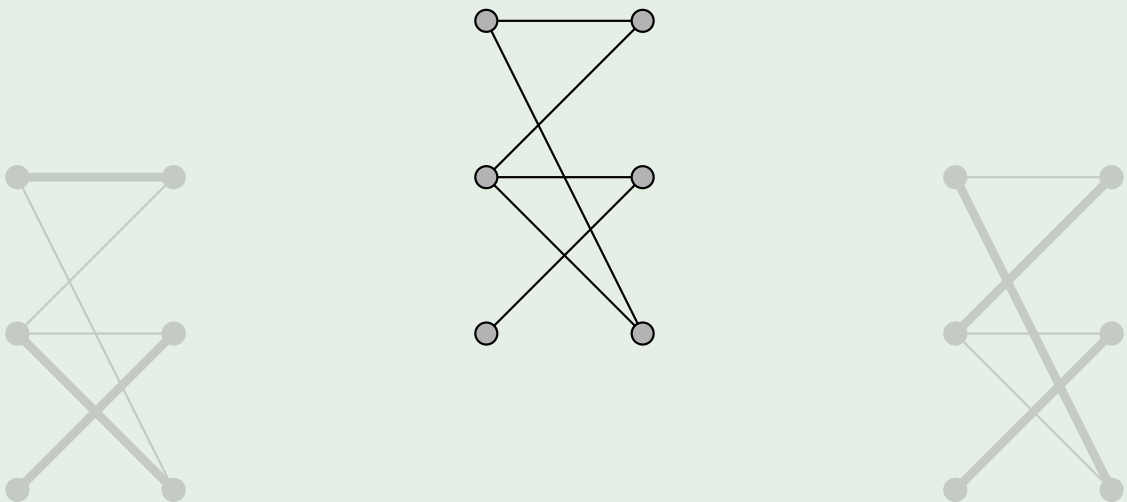
What is The Problem?

Problem

Given a bipartite graph G with two parts of equal size, how many perfect matchings does G have?

Example

The following graph has two perfect matchings:



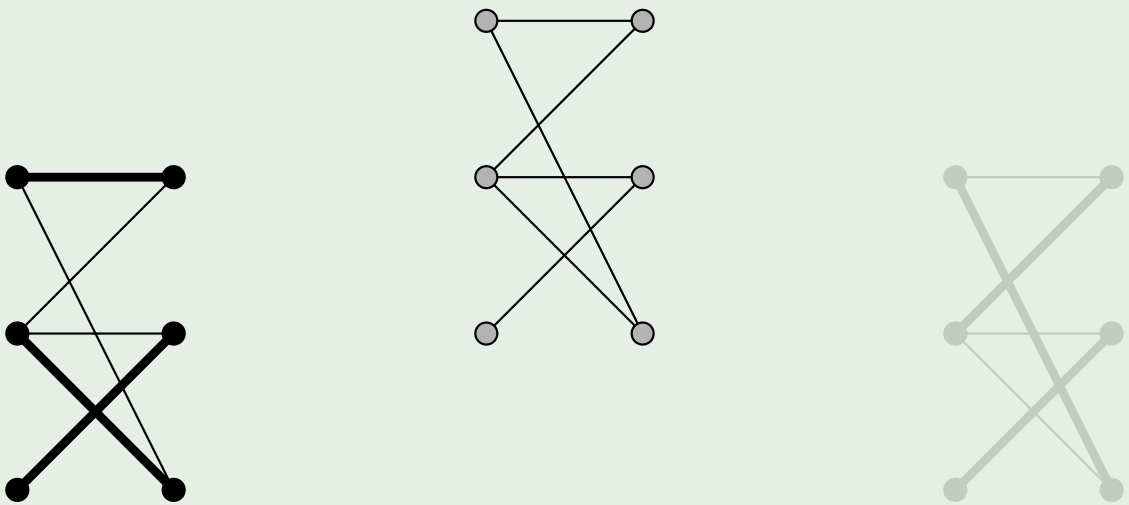
What is The Problem?

Problem

Given a bipartite graph G with two parts of equal size, how many perfect matchings does G have?

Example

The following graph has two perfect matchings:



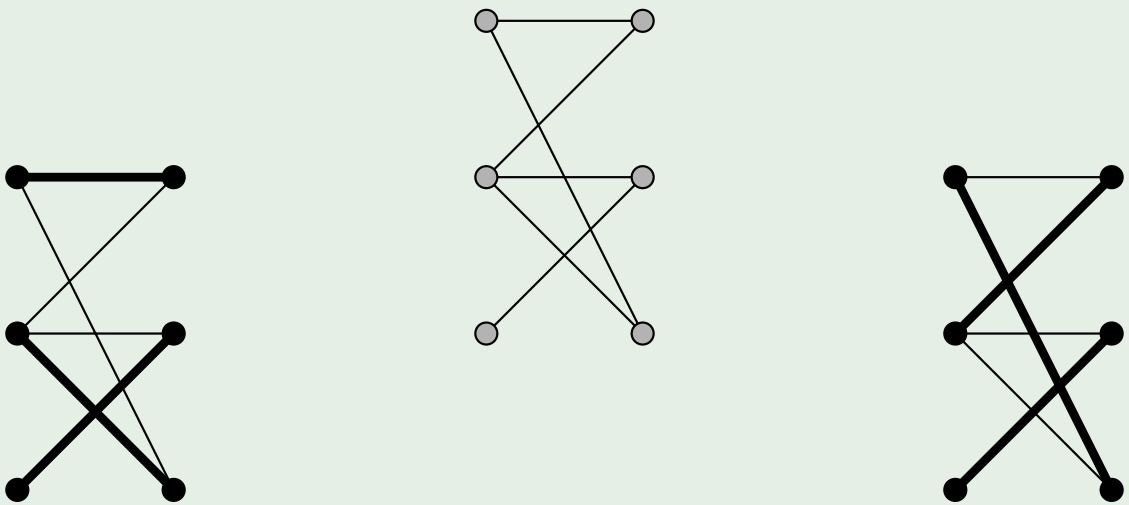
What is The Problem?

Problem

Given a bipartite graph G with two parts of equal size, how many perfect matchings does G have?

Example

The following graph has two perfect matchings:



Computational Complexity of The Problem

- Deciding if the graph has a perfect matching is in P (via reduction to max-flow).
- Counting the number of perfect matchings is $\#P$-complete.
  - Hence there is no polynomial-time exact algorithm (unless $P = NP$).
- But we can hope for polynomial-time approximation algorithms!
  - We will see a polynomial-time randomized approximation algorithm for dense graphs.

Computational Complexity of The Problem

- Deciding if the graph has a perfect matching is in P (via reduction to max-flow).
- Counting the number of perfect matchings is $\#P$-complete.
  - Hence there is no polynomial-time exact algorithm (unless $P = NP$).
- But we can hope for polynomial-time approximation algorithms!
  - We will see a polynomial-time randomized approximation algorithm for dense graphs.

Computational Complexity of The Problem

- Deciding if the graph has a perfect matching is in P (via reduction to max-flow).
- Counting the number of perfect matchings is $\#P$-complete.
  - Hence there is no polynomial-time exact algorithm (unless $P = NP$).
- But we can hope for polynomial-time approximation algorithms!
  - We will see a polynomial-time randomized approximation algorithm for dense graphs.

Some Notation

G	a bipartite graph with two parts of size n
dense	G is dense if all vertices have degree at least $n/2$.
k-matching	a matching having k edges
$M_k(G)$	the set of k-matchings of G
$m_k(G)$	the number of k-matchings of G e.g. $m_1(G) = \lvert E(G)\rvert$
$\text{poly}(x, y)$	the set of polynomially-bounded functions in x, y e.g. x^2y^3, $2x + \log y + 5 \in \text{poly}(x, y)$

Ratio of Approximation

Definition

For numbers $a, \widehat{a}, \epsilon > 0$, let's say $\widehat{a}$ approximates a within ratio $1+\epsilon$ if the following holds:

$$\frac{a}{1+\epsilon} < \widehat{a} < a \times (1+\epsilon).$$

We abbreviate this as: $\widehat{a} \simeq a$ within ratio $(1+\epsilon)$.

Why use this (unusual) definition?

If $\widehat{a}_1$ approximates a_1 within ratio $1+\epsilon_1$ and $\widehat{a}_2$ approximates a_2 within ratio $1+\epsilon_2$ then:

$\widehat{a}_1\widehat{a}_2$ approximates $a_1 a_2$ within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

$\widehat{a}_1/\widehat{a}_2$ approximates a_1/a_2 within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

Ratio of Approximation

Definition

For numbers $a, \widehat{a}, \epsilon > 0$, let's say $\widehat{a}$ approximates a within ratio $1+\epsilon$ if the following holds:

$$\frac{a}{1+\epsilon} < \widehat{a} < a \times (1+\epsilon).$$

We abbreviate this as: $\widehat{a} \simeq a$ within ratio $(1+\epsilon)$.

Why use this (unusual) definition?

If $\widehat{a}_1$ approximates a_1 within ratio $1+\epsilon_1$ and $\widehat{a}_2$ approximates a_2 within ratio $1+\epsilon_2$ then:

$\widehat{a}_1 \widehat{a}_2$ approximates $a_1 a_2$ within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

$\widehat{a}_1 / \widehat{a}_2$ approximates a_1 / a_2 within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

Ratio of Approximation

Definition

For numbers $a, \widehat{a}, \epsilon > 0$, let's say $\widehat{a}$ approximates a within ratio $1+\epsilon$ if the following holds:

$$\frac{a}{1+\epsilon} < \widehat{a} < a \times (1+\epsilon).$$

We abbreviate this as: $\widehat{a} \simeq a$ within ratio $(1+\epsilon)$.

Why use this (unusual) definition?

If $\widehat{a}_1$ approximates a_1 within ratio $1+\epsilon_1$ and

$\widehat{a}_2$ approximates a_2 within ratio $1+\epsilon_2$ then:

$\widehat{a}_1\widehat{a}_2$ approximates $a_1 a_2$ within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

$\widehat{a}_1/\widehat{a}_2$ approximates a_1/a_2 within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

Ratio of Approximation

Definition

For numbers $a, \widehat{a}, \epsilon > 0$, let's say $\widehat{a}$ approximates a within ratio $1+\epsilon$ if the following holds:

$$\frac{a}{1+\epsilon} < \widehat{a} < a \times (1+\epsilon).$$

We abbreviate this as: $\widehat{a} \simeq a$ within ratio $(1+\epsilon)$.

Why use this (unusual) definition?

If $\widehat{a}_1$ approximates a_1 within ratio $1+\epsilon_1$ and $\widehat{a}_2$ approximates a_2 within ratio $1+\epsilon_2$ then:

$\widehat{a}_1 \widehat{a}_2$ approximates $a_1 a_2$ within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

$\widehat{a}_1 / \widehat{a}_2$ approximates a_1 / a_2 within ratio $(1+\epsilon_1)(1+\epsilon_2)$.

Fully Polynomial Time Randomized Approximation Scheme

Definition

A fully polynomial time randomized approximation scheme (fpras) is a randomized algorithm that:

inputs: a dense bipartite graph G and a number $\epsilon > 0$

outputs: a number $\widehat{m}_n(G)$ that approximates $m_n(G)$ within ratio $1+\epsilon$ with probability more than 3/4:

$$Pr\left[\frac{m_n(G)}{1+\epsilon} < \widehat{m}_n(G) < m_n(G) \times (1+\epsilon)\right] > 3/4.$$

run-time: is in $\text{poly}(n, \epsilon^{-1})$.

Probability can be increased to $1-\delta$ by repeating $O(\lg \delta)$ times.

Fully Polynomial Time Randomized Approximation Scheme

Definition

A fully polynomial time randomized approximation scheme (fpras) is a randomized algorithm that:

inputs: a dense bipartite graph G and a number $\epsilon > 0$

outputs: a number $\widehat{m}_n(G)$ that approximates $m_n(G)$ within ratio $1+\epsilon$ with probability more than $3/4$:

$$Pr\left[\frac{m_n(G)}{1+\epsilon} < \widehat{m}_n(G) < m_n(G) \times (1+\epsilon)\right] > 3/4.$$

run-time: is in $\text{poly}(n, \epsilon^{-1})$.

Probability can be increased to $1-\delta$ by repeating $O(\lg \delta)$ times.

Fully Polynomial Time Randomized Approximation Scheme

Definition

A fully polynomial time randomized approximation scheme (fpras) is a randomized algorithm that:

inputs: a dense bipartite graph G and a number $\epsilon > 0$

outputs: a number $\widehat{m}_n(G)$ that approximates $m_n(G)$ within ratio $1+\epsilon$ with probability more than $3/4$:

$$Pr\left[\frac{m_n(G)}{1+\epsilon} < \widehat{m}_n(G) < m_n(G) \times (1+\epsilon)\right] > 3/4.$$

run-time: is in $\text{poly}(n, \epsilon^{-1})$.

Probability can be increased to $1-\delta$ by repeating $O(\lg \delta)$ times.

Fully Polynomial Time Randomized Approximation Scheme

Definition

A fully polynomial time randomized approximation scheme (fpras) is a randomized algorithm that:

inputs: a dense bipartite graph G and a number $\epsilon > 0$

outputs: a number $\widehat{m}_n(G)$ that approximates $m_n(G)$ within ratio $1+\epsilon$ with probability more than $3/4$:

$$Pr\left[\frac{m_n(G)}{1+\epsilon} < \widehat{m}_n(G) < m_n(G) \times (1+\epsilon)\right] > 3/4.$$

run-time: is in $\text{poly}(n, \epsilon^{-1})$.

Probability can be increased to $1-\delta$ by repeating $O(\lg \delta)$ times.

The First Idea

We have:

$$m_n = \frac{m_n}{m_{n-1}} \times \frac{m_{n-1}}{m_{n-2}} \times \cdots \times \frac{m_2}{m_1} \times m_1.$$

Focus on first ratio $\frac{m_n}{m_{n-1}}$ for now.

The First Idea

We have:

$$m_n = \frac{m_n}{m_{n-1}} \times \frac{m_{n-1}}{m_{n-2}} \times \cdots \times \frac{m_2}{m_1} \times m_1.$$

Focus on first ratio $\frac{m_n}{m_{n-1}}$ for now.

Approximating $\frac{m_n}{m_{n-1}}$

Large matchings

Definition

A large matching of G is a matching of size n or $n-1$,

$LM(G) = M_n(G) \cup M_{n-1}(G)$,

$lm(G) = |LM(G)| = m_n(G) + m_{n-1}(G)$.

Example

Approximating $\frac{m_n}{m_{n-1}}$

Large matchings

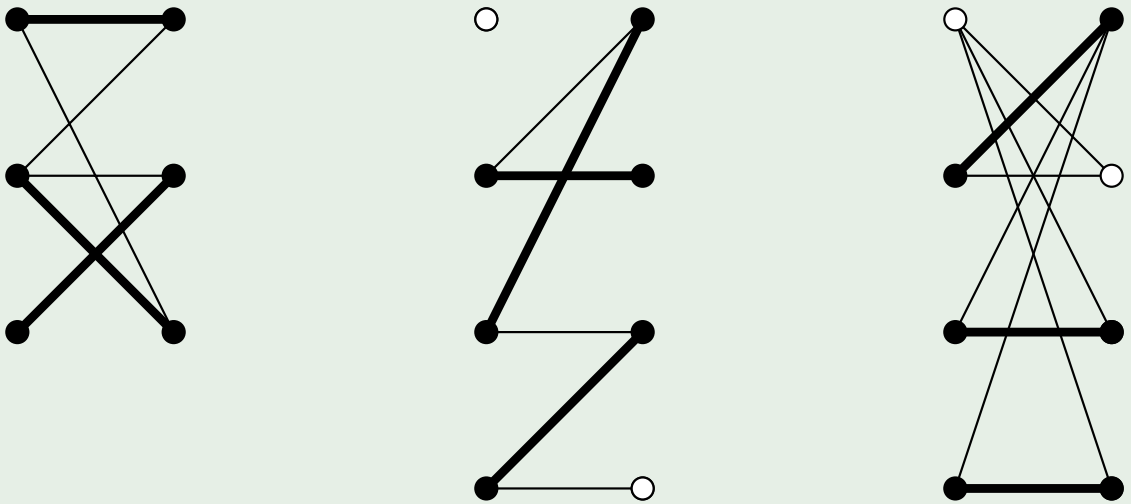
Definition

A large matching of G is a matching of size n or $n-1$,

$LM(G) = M_n(G) \cup M_{n-1}(G)$,

$lm(G) = |LM(G)| = m_n(G) + m_{n-1}(G)$.

Example



Approximating $\frac{m_n}{m_{n-1}}$

The idea of sampling

Definition

A large matching of G is a matching of size n or $n-1$,
$LM(G) = M_n(G) \cup M_{n-1}(G)$,
$lm(G) = |LM(G)| = m_n(G) + m_{n-1}(G)$.

The Sampling Method

1. Take X samples uniformly at random from LM.
2. Suppose that X_n of them have size n, and X_{n-1} of them have size $n-1$.
3. Then X_n/X_{n-1} is an estimate for m_n/m_{n-1}.

Question: How many samples are needed?

Approximating $\frac{m_n}{m_{n-1}}$

The idea of sampling

Definition

A large matching of G is a matching of size n or $n-1$,
$LM(G) = M_n(G) \cup M_{n-1}(G)$,
$lm(G) = |LM(G)| = m_n(G) + m_{n-1}(G)$.

The Sampling Method

1. Take X samples uniformly at random from LM.
2. Suppose that X_n of them have size n, and X_{n-1} of them have size $n-1$.
3. Then X_n/X_{n-1} is an estimate for m_n/m_{n-1}.

Question: How many samples are needed?

Approximating $\frac{m_n}{m_{n-1}}$

The idea of sampling

Definition

A large matching of G is a matching of size n or $n-1$,
$LM(G) = M_n(G) \cup M_{n-1}(G)$,
$lm(G) = |LM(G)| = m_n(G) + m_{n-1}(G)$.

The Sampling Method

1. Take X samples uniformly at random from LM.
2. Suppose that X_n of them have size n, and X_{n-1} of them have size $n-1$.
3. Then X_n/X_{n-1} is an estimate for m_n/m_{n-1}.

Question: How many samples are needed?

How Many Samples Are Needed?

The sampling lemma

Lemma (The Sampling Lemma)

Let $U \subseteq S$ and $p = |U|/|S|$. The number of samples needed to approximate p within ratio $1+\epsilon$ with probability at least $1-\delta$ is

$$\frac{675 \ln(2/\delta)}{p\epsilon^2} \in \text{poly}\left(p^{-1}, \epsilon^{-1}, \log(\delta^{-1})\right).$$

Proof.

Use Chernoff bounds. □

How Many Samples Are Needed?

Corollary

Let $p = \min\{m_n/lm, m_{n-1}/lm\}$. *We can approximate* m_n/m_{n-1} *within ratio* $1+\epsilon$ *with probability* $1-\delta$ *by taking* $\text{poly}\left(p^{-1}, \epsilon^{-1}, \log(\delta^{-1})\right)$ *samples from* LM.

Proof.

1. Take $X = \frac{10^4 \ln(4/\delta)}{p\epsilon^2}$ samples. Define X_n, X_{n-1} as before.
2. Then $X_n/X \simeq m_n/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$, $X_{n-1}/X \simeq m_{n-1}/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$,so

$$\frac{X_n}{X_{n-1}} = \frac{X_n/X}{X_{n-1}/X} \simeq \frac{m_n/lm}{m_{n-1}/lm} = \frac{m_n}{m_{n-1}}$$

within ratio $(1+\epsilon/3)^2 < 1+\epsilon$ with prob. $(1-\delta/2)^2 > 1-\delta$. □

How Many Samples Are Needed?

Corollary

Let $p = \min\{m_n/lm, m_{n-1}/lm\}$. *We can approximate* m_n/m_{n-1} *within ratio* $1+\epsilon$ *with probability* $1-\delta$ *by taking* poly $\left(p^{-1}, \epsilon^{-1}, \log(\delta^{-1})\right)$ *samples from* LM.

Proof.

1. Take $X = \frac{10^4 \ln(4/\delta)}{p\epsilon^2}$ samples. Define X_n, X_{n-1} as before.
2. Then $X_n/X \simeq m_n/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$, $X_{n-1}/X \simeq m_{n-1}/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$,so

$$\frac{X_n}{X_{n-1}} = \frac{X_n/X}{X_{n-1}/X} \simeq \frac{m_n/lm}{m_{n-1}/lm} = \frac{m_n}{m_{n-1}}$$

within ratio $(1+\epsilon/3)^2 < 1+\epsilon$ with prob. $(1-\delta/2)^2 > 1-\delta$. □

How Many Samples Are Needed?

Corollary

Let $p = \min\{m_n/lm, m_{n-1}/lm\}$. *We can approximate* m_n/m_{n-1} *within ratio* $1+\epsilon$ *with probability* $1-\delta$ *by taking* poly $\left(p^{-1}, \epsilon^{-1}, \log(\delta^{-1})\right)$ *samples from* LM.

Proof.

1. Take $X = \frac{10^4 \ln(4/\delta)}{p\epsilon^2}$ samples. Define X_n, X_{n-1} as before.
2. Then $X_n/X \simeq m_n/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$, $X_{n-1}/X \simeq m_{n-1}/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$,so

$$\frac{X_n}{X_{n-1}} = \frac{X_n/X}{X_{n-1}/X} \simeq \frac{m_n/lm}{m_{n-1}/lm} = \frac{m_n}{m_{n-1}}$$

within ratio $(1+\epsilon/3)^2 < 1+\epsilon$ with prob. $(1-\delta/2)^2 > 1-\delta$. □

How Many Samples Are Needed?

Corollary

Let $p = \min\{m_n/lm, m_{n-1}/lm\}$. *We can approximate* m_n/m_{n-1} *within ratio* $1+\epsilon$ *with probability* $1-\delta$ *by taking* poly $\left(p^{-1}, \epsilon^{-1}, \log(\delta^{-1})\right)$ *samples from* LM.

Proof.

1. Take $X = \frac{10^4 \ln(4/\delta)}{p\epsilon^2}$ samples. Define X_n, X_{n-1} as before.
2. Then $X_n/X \simeq m_n/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$, $X_{n-1}/X \simeq m_{n-1}/lm$ within ratio $(1+\epsilon/3)$ with prob. $1-\delta/2$,so

$$\frac{X_n}{X_{n-1}} = \frac{X_n/X}{X_{n-1}/X} \simeq \frac{m_n/lm}{m_{n-1}/lm} = \frac{m_n}{m_{n-1}}$$

within ratio $(1+\epsilon/3)^2 < 1+\epsilon$ with prob. $(1-\delta/2)^2 > 1-\delta$. □

A Technical Lemma

Lemma (The Technical Lemma)

If G is dense then we have

$$1/n^2 \leq \frac{m_n}{m_{n-1}} \leq n^2.$$

Corollary

If we can sample from the large matchings in polynomial time then we can approximate $\frac{m_n}{m_{n-1}}$.

Proof (of Corollary).

Recall that $lm = m_n + m_{n-1}$. So

$$p = \min\{m_n/lm, m_{n-1}/lm\} > n^{-3},$$

i.e. p^{-1} is polynomially bounded in n. □

A Technical Lemma

Lemma (The Technical Lemma)

If G is dense then we have

$$1/n^2 \leq \frac{m_n}{m_{n-1}} \leq n^2.$$

Corollary

If we can sample from the large matchings in polynomial time then we can approximate $\frac{m_n}{m_{n-1}}$.

Proof (of Corollary).

Recall that $lm = m_n + m_{n-1}$. So

$$p = \min\{m_n/lm, m_{n-1}/lm\} > n^{-3},$$

i.e. p^{-1} is polynomially bounded in n. □

A Technical Lemma

Lemma (The Technical Lemma)

If G is dense then we have

$$1/n^2 \leq \frac{m_n}{m_{n-1}} \leq n^2.$$

Corollary

If we can sample from the large matchings in polynomial time then we can approximate $\frac{m_n}{m_{n-1}}$.

Proof (of Corollary).

Recall that $lm = m_n + m_{n-1}$. So

$$p = \min\{m_n/lm, m_{n-1}/lm\} > n^{-3},$$

i.e. p^{-1} is polynomially bounded in n. □

Proof of the Technical Lemma

upper bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (Upper Bound).

1. Any k-matching can be built by adding an edge to some $(k-1)$-matching.
2. Each $(k-1)$-matching has $(n-k+1)$ unmatched vertices in each part, thus at most $(n-k+1)^2$ edges can be added to build a k-matching.
3. Therefore, $m_k \leq (n-k+1)^2 m_{k-1}$. □

Proof of the Technical Lemma

upper bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (Upper Bound).

1. Any k-matching can be built by adding an edge to some $(k-1)$-matching.
2. Each $(k-1)$-matching has $(n-k+1)$ unmatched vertices in each part, thus at most $(n-k+1)^2$ edges can be added to build a k-matching.
3. Therefore, $m_k \leq (n-k+1)^2 m_{k-1}$. □

Proof of the Technical Lemma

upper bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (Upper Bound).

1. Any k-matching can be built by adding an edge to some $(k-1)$-matching.
2. Each $(k-1)$-matching has $(n-k+1)$ unmatched vertices in each part, thus at most $(n-k+1)^2$ edges can be added to build a k-matching.
3. Therefore, $m_k \leq (n-k+1)^2 m_{k-1}$. □

Proof of the Technical Lemma

upper bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (Upper Bound).

1. Any k-matching can be built by adding an edge to some $(k-1)$-matching.
2. Each $(k-1)$-matching has $(n-k+1)$ unmatched vertices in each part, thus at most $(n-k+1)^2$ edges can be added to build a k-matching.
3. Therefore, $m_k \leq (n-k+1)^2 m_{k-1}$. □

Proof of the Technical Lemma

lower bound

Definition

De-augmenting a matching:
removing two edges from the matching
and adding a cross-edge.

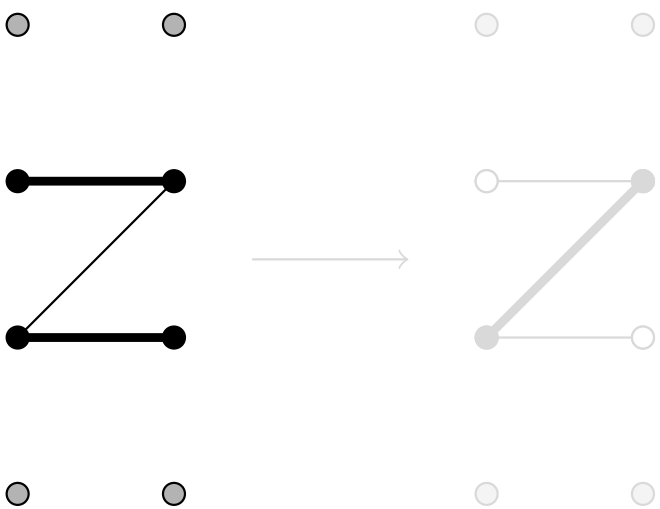
There are $\leq 2\binom{k}{2} = k^2 - k$ ways to de-augment a k-matching.

Proof of the Technical Lemma

lower bound

Definition

De-augmenting a matching:
removing two edges from the matching
and adding a cross-edge.



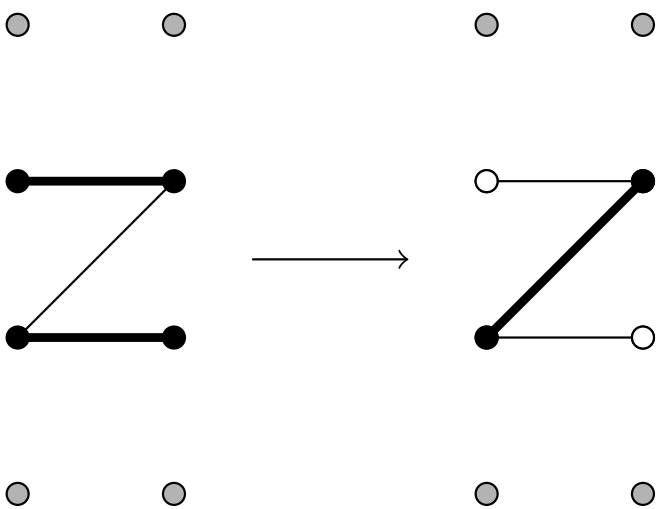
There are $\leq 2\binom{k}{2} = k^2 - k$ ways to de-augment a k-matching.

Proof of the Technical Lemma

lower bound

Definition

De-augmenting a matching:
removing two edges from the matching
and adding a cross-edge.



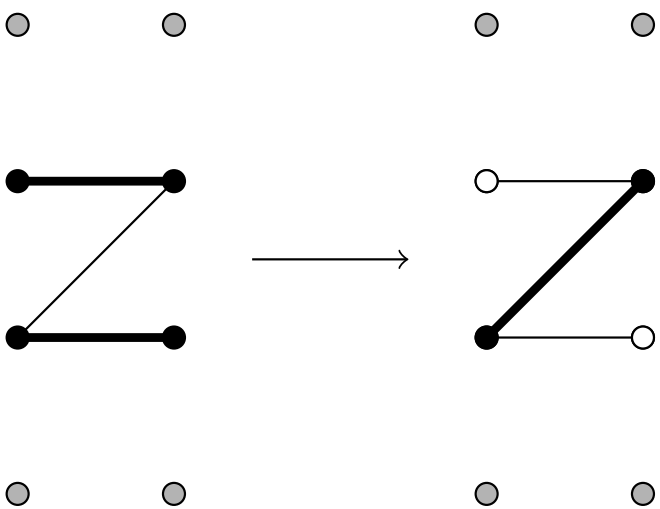
There are $\leq 2\binom{k}{2} = k^2 - k$ ways to de-augment a k-matching.

Proof of the Technical Lemma

lower bound

Definition

De-augmenting a matching:
removing two edges from the matching
and adding a cross-edge.



There are $\leq 2\binom{k}{2} = k^2 - k$ ways to de-augment a k-matching.

Proof of the Technical Lemma

lower bound

Remark.

If a $(k-1)$-matching has an alternating path of length 3, then it can be built by de-augmenting some k-matching.

Proof.

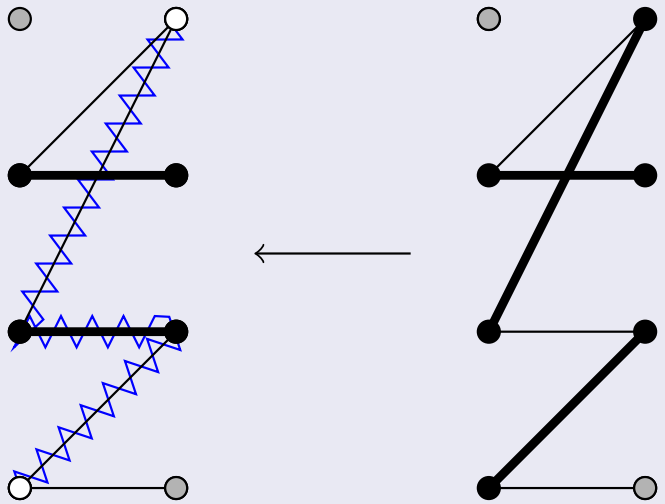
Proof of the Technical Lemma

lower bound

Remark.

If a $(k-1)$-matching has an alternating path of length 3, then it can be built by de-augmenting some k-matching.

Proof.



Proof of the Technical Lemma

lower bound

Claim.

If G is dense and $2 \leq k \leq n$, then any $(k-1)$-matching has an alternating path of length at most 3.

Proof.

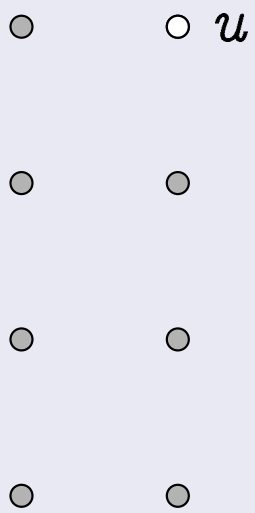
Proof of the Technical Lemma

lower bound

Claim.

If G is dense and $2 \leq k \leq n$, then any $(k-1)$-matching has an alternating path of length at most 3.

Proof.



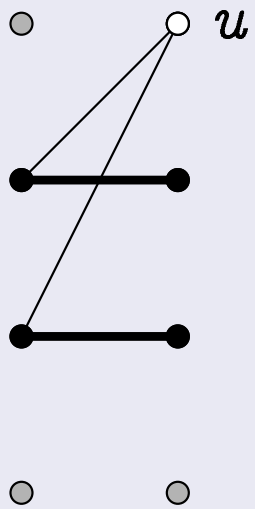
Proof of the Technical Lemma

lower bound

Claim.

If G is dense and $2 \leq k \leq n$, then any $(k-1)$-matching has an alternating path of length at most 3.

Proof.



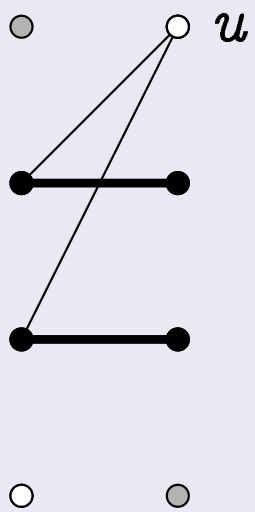
Proof of the Technical Lemma

lower bound

Claim.

If G is dense and $2 \leq k \leq n$, then any $(k-1)$-matching has an alternating path of length at most 3.

Proof.



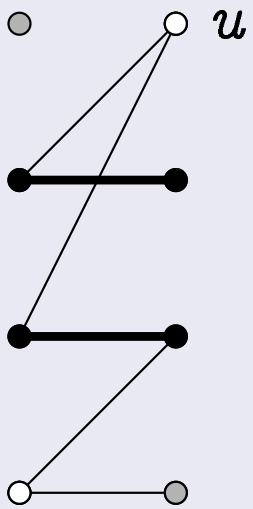
Proof of the Technical Lemma

lower bound

Claim.

If G is dense and $2 \leq k \leq n$, then any $(k-1)$-matching has an alternating path of length at most 3.

Proof.



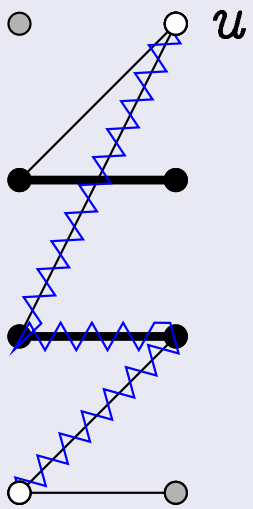
Proof of the Technical Lemma

lower bound

Claim.

If G is dense and $2 \leq k \leq n$, then any $(k-1)$-matching has an alternating path of length at most 3.

Proof.



Proof of the Technical Lemma

lower bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (lower bound).

1. Any $(k-1)$-matching can be built by one of these operations:
   - removing an edge from some k-matching.
   - de-augmenting some k-matching.
2. There are k ways to remove an edge from a k-matching.
3. There are $k^2 - k$ ways to de-augment a k-matching.
4. Therefore, $m_{k-1} \leq k^2 m_k$. □

Proof of the Technical Lemma

lower bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (lower bound).

1. Any $(k-1)$-matching can be built by one of these operations:
   - removing an edge from some k-matching.
   - de-augmenting some k-matching.
2. There are k ways to remove an edge from a k-matching.
3. There are $k^2 - k$ ways to de-augment a k-matching.
4. Therefore, $m_{k-1} \leq k^2 m_k$. □

Proof of the Technical Lemma

lower bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (lower bound).

1. Any $(k-1)$-matching can be built by one of these operations:
   - removing an edge from some k-matching.
   - de-augmenting some k-matching.
2. There are k ways to remove an edge from a k-matching.
3. There are $k^2 - k$ ways to de-augment a k-matching.
4. Therefore, $m_{k-1} \leq k^2 m_k$. □

Proof of the Technical Lemma

lower bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (lower bound).

1. Any $(k-1)$-matching can be built by one of these operations:
   - removing an edge from some k-matching.
   - de-augmenting some k-matching.
2. There are k ways to remove an edge from a k-matching.
3. There are $k^2 - k$ ways to de-augment a k-matching.
4. Therefore, $m_{k-1} \leq k^2 m_k$. □

Proof of the Technical Lemma

lower bound

Lemma (The Technical Lemma (generalized))

If G is dense then for all $2 \leq k \leq n$ we have

$$1/n^2 \leq \frac{m_k}{m_{k-1}} \leq n^2.$$

Proof (lower bound).

1. Any $(k-1)$-matching can be built by one of these operations:
   - removing an edge from some k-matching.
   - de-augmenting some k-matching.
2. There are k ways to remove an edge from a k-matching.
3. There are $k^2 - k$ ways to de-augment a k-matching.
4. Therefore, $m_{k-1} \leq k^2 m_k$. □

Recall: The Technical Lemma

Lemma (The Technical Lemma)

If G is dense then we have

$$1/n^2 \leq \frac{m_n}{m_{n-1}} \leq n^2.$$

Corollary

If we can sample from the large matchings in polynomial time then we can approximate $\frac{m_n}{m_{n-1}}$.

Now, let's focus on the sampling.

Recall: The Technical Lemma

Lemma (The Technical Lemma)

If G is dense then we have

$$1/n^2 \leq \frac{m_n}{m_{n-1}} \leq n^2.$$

Corollary

If we can sample from the large matchings in polynomial time then we can approximate $\frac{m_n}{m_{n-1}}$.

Now, let's focus on the sampling.

Elementary Operations on Matchings

Definition

insert

delete

rotate

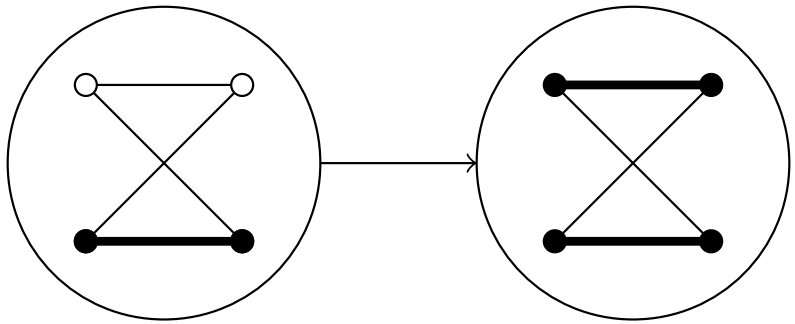
Elementary Operations on Matchings

Definition

insert

delete

rotate



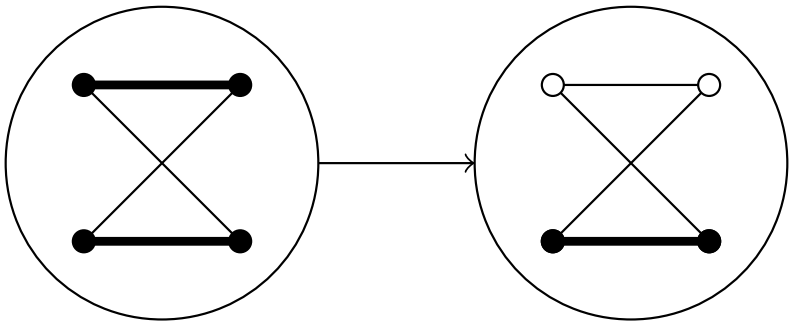
Elementary Operations on Matchings

Definition

insert

delete

rotate



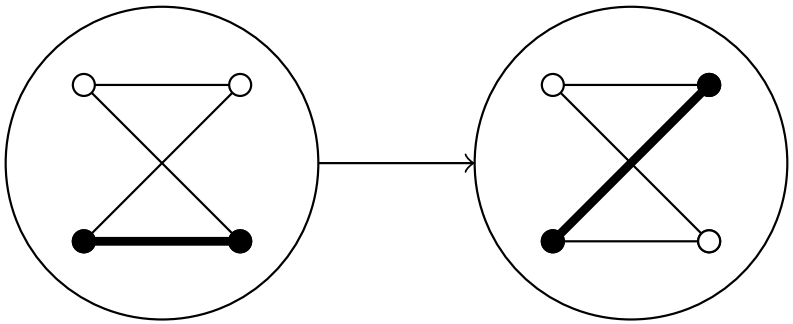
Elementary Operations on Matchings

Definition

insert

delete

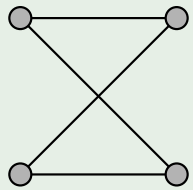
rotate



Sampling From the Large Matchings

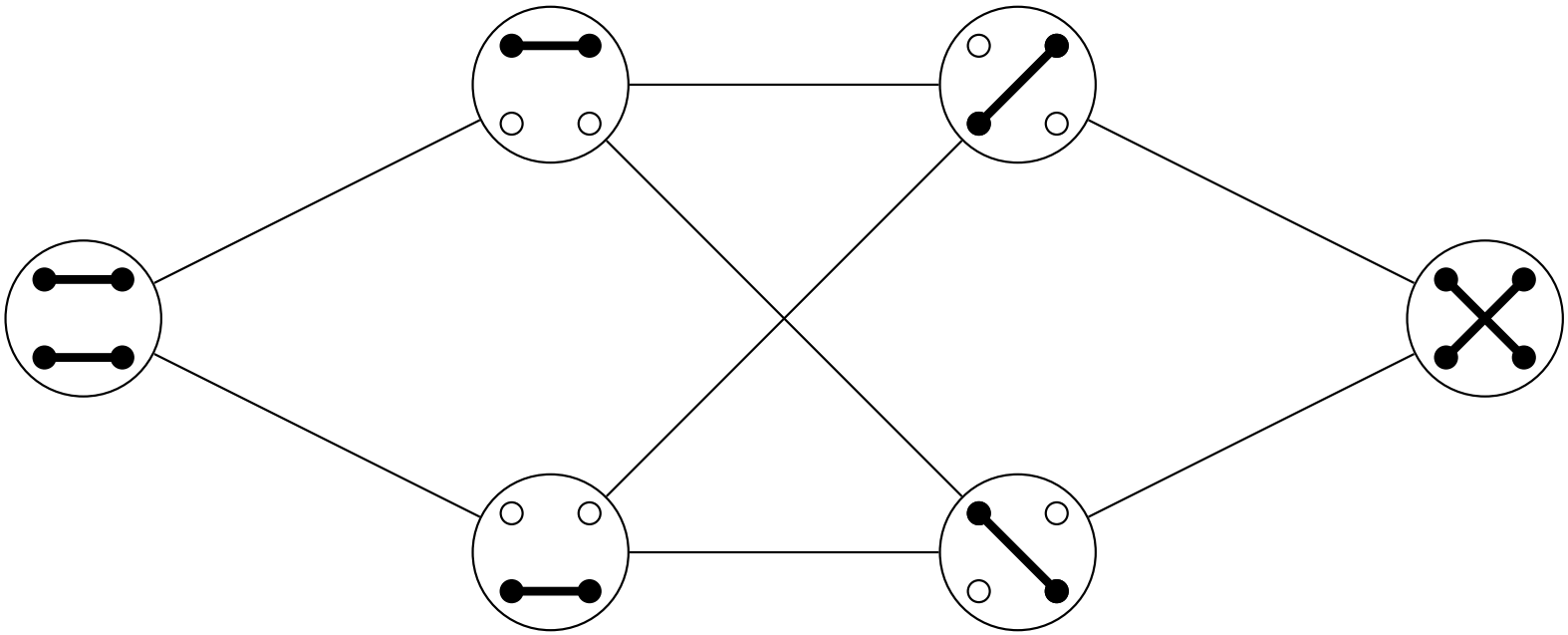
Example

Suppose that we are trying to sample from the set of large matchings of the following graph:



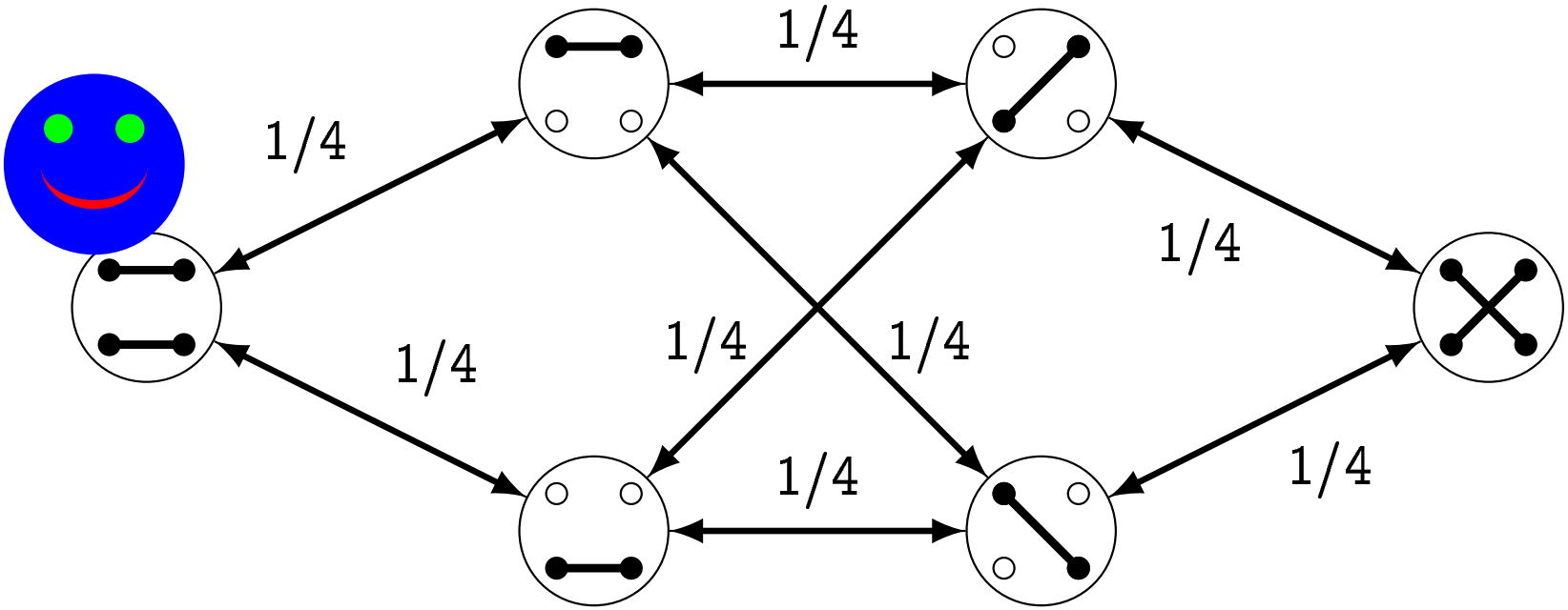
Sampling From the Large Matchings

The random walk



Sampling From the Large Matchings

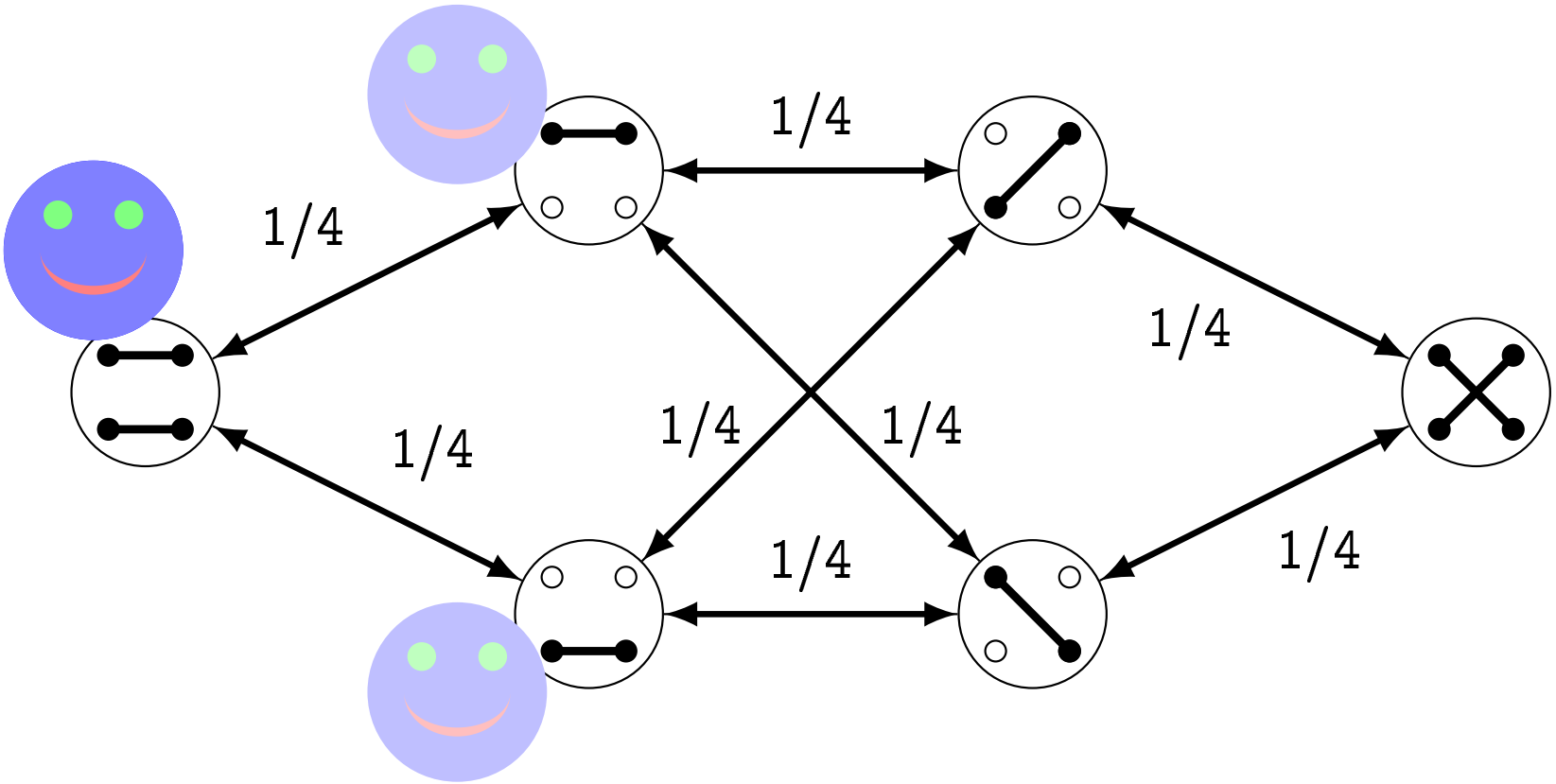
The random walk



Sampling From the Large Matchings

The random walk

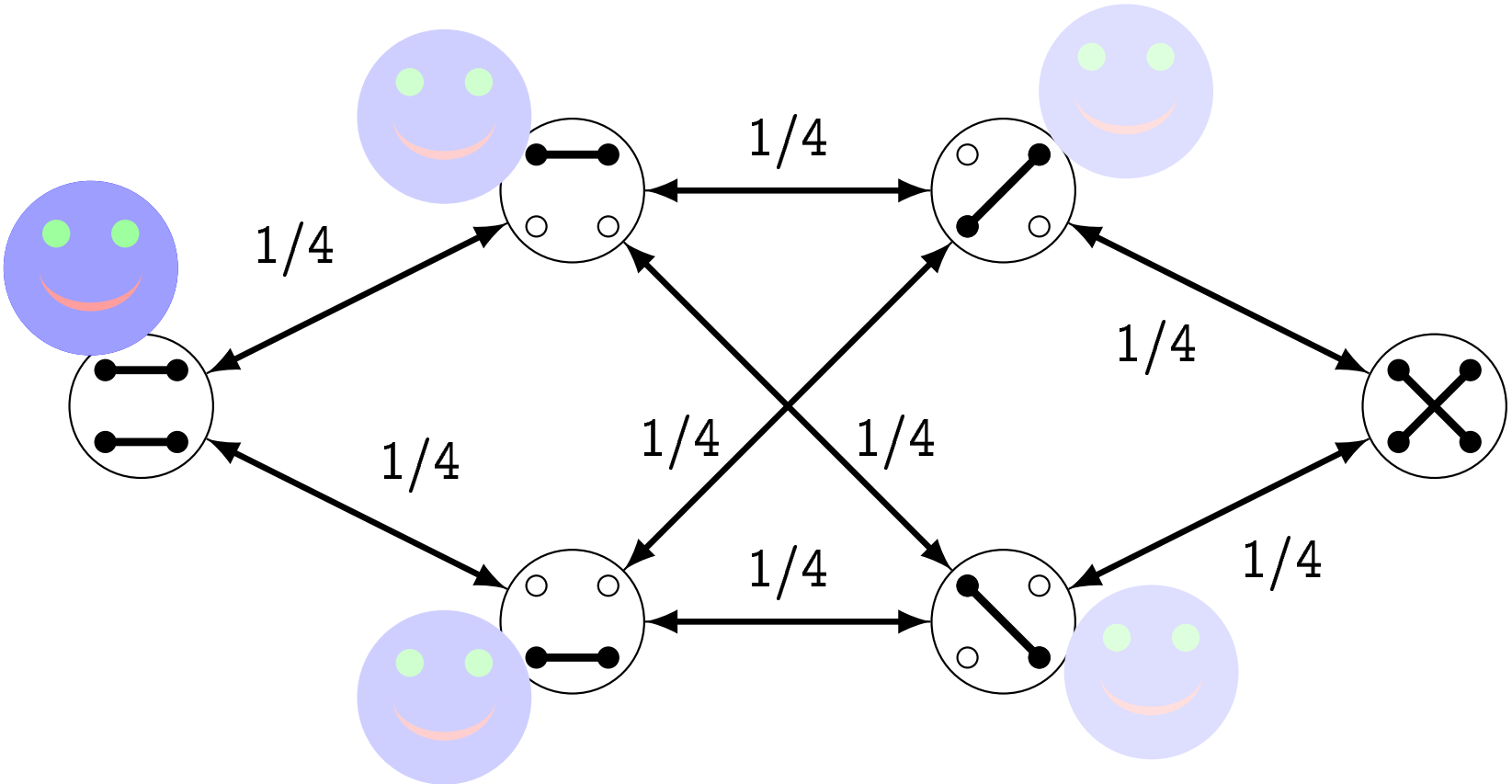
after first step ...



Sampling From the Large Matchings

The random walk

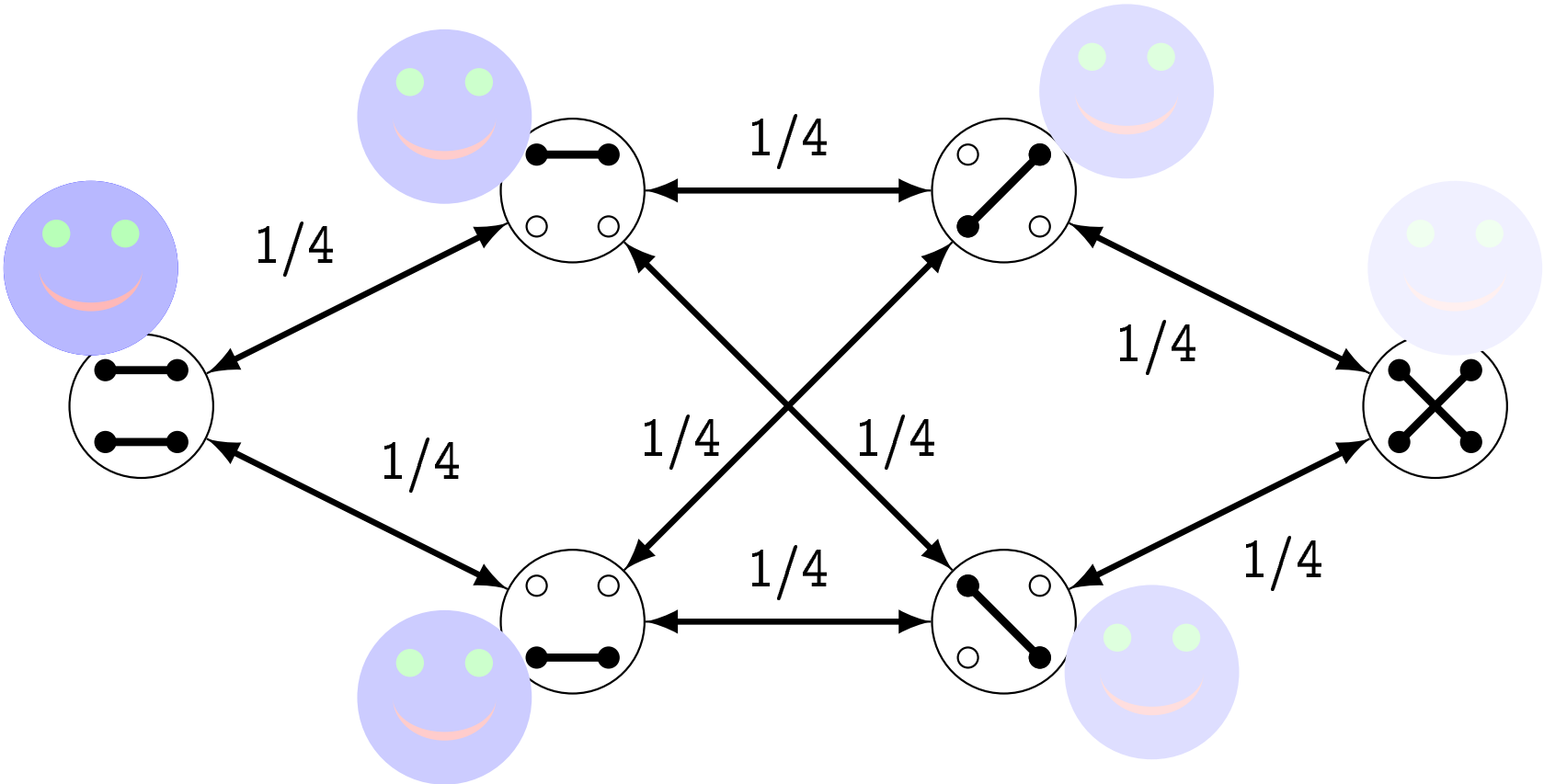
after second step ...



Sampling From the Large Matchings

The random walk

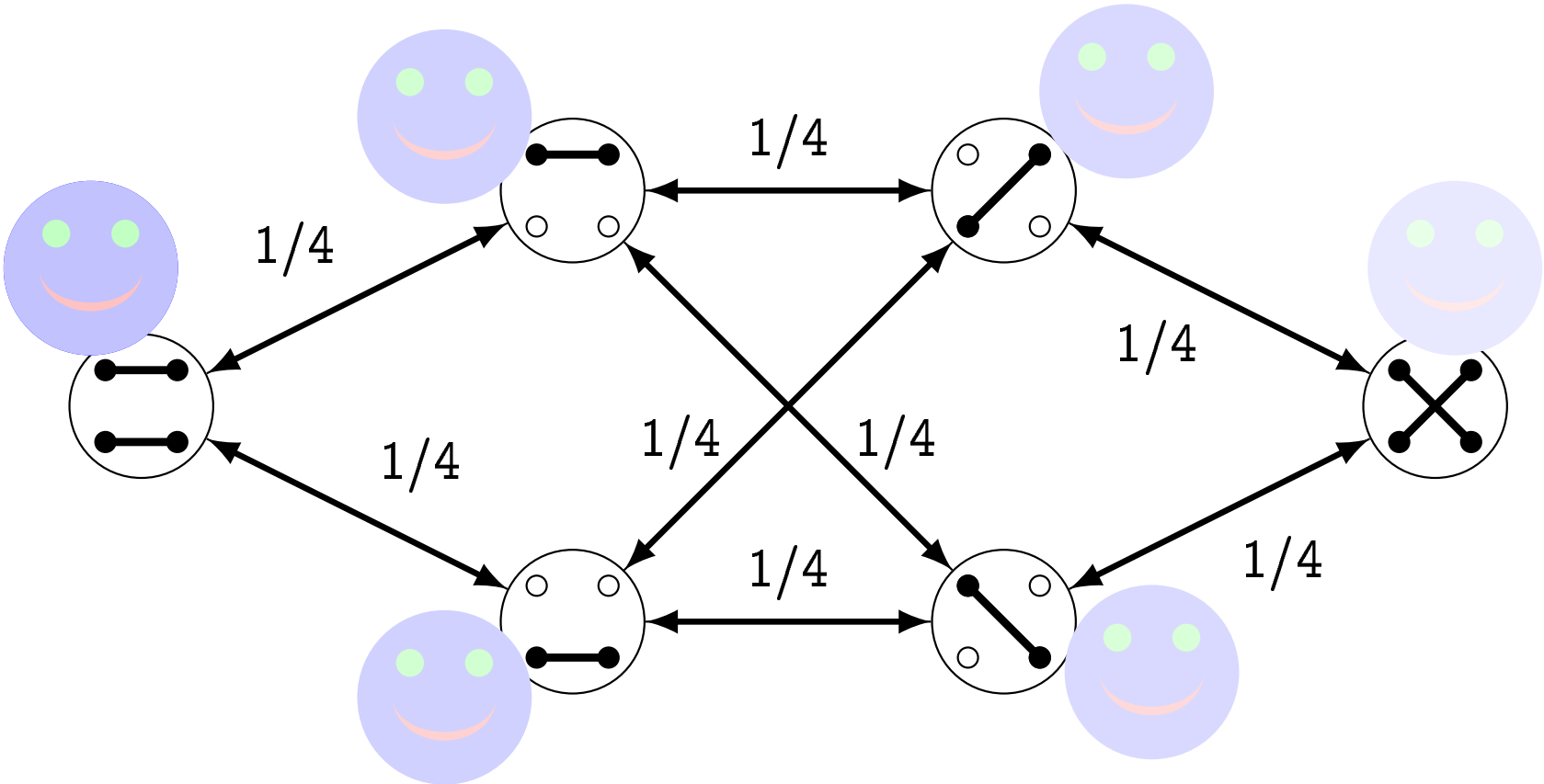
after third step ...



Sampling From the Large Matchings

The random walk

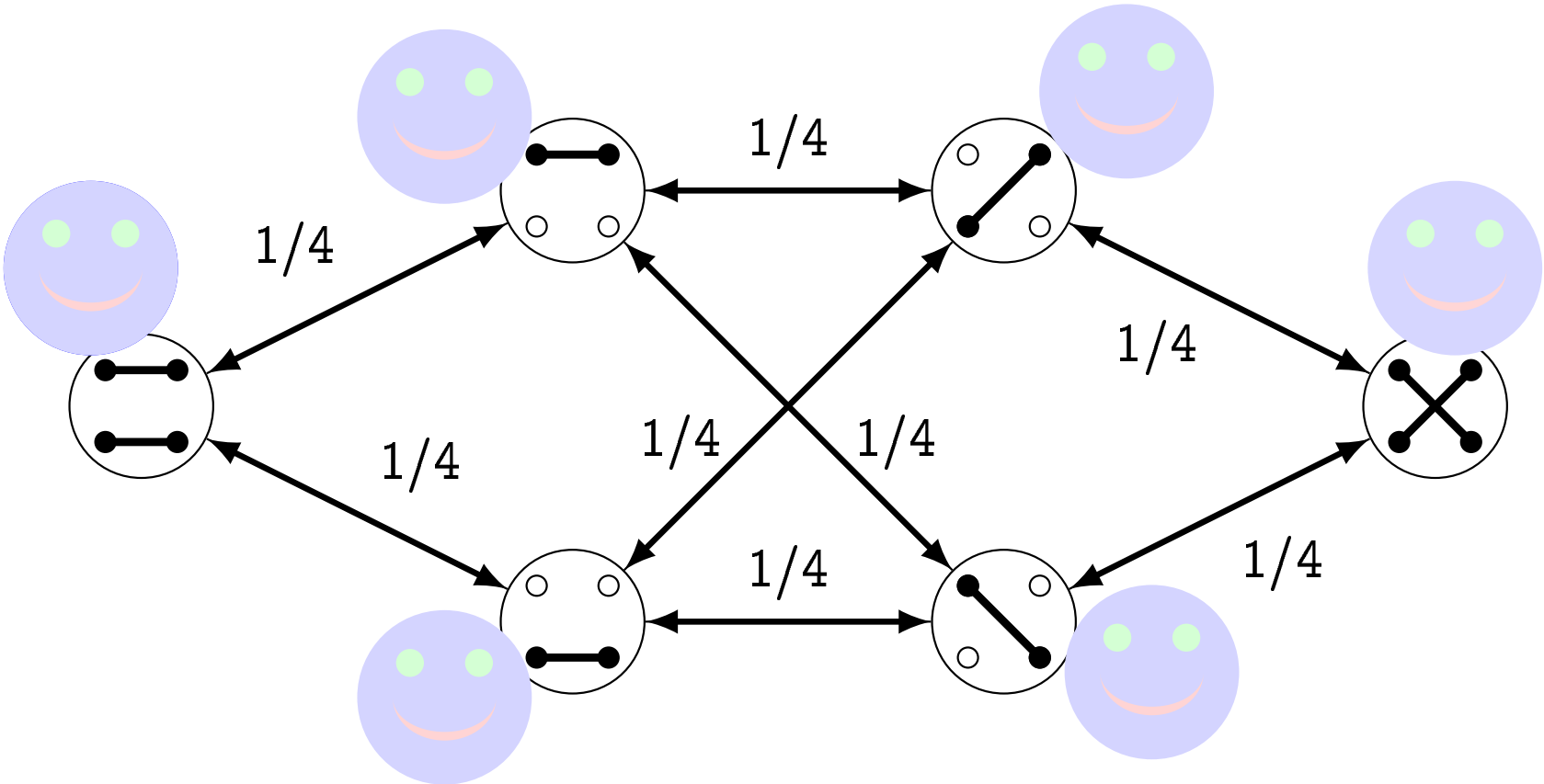
after fourth step ...



Sampling From the Large Matchings

The random walk

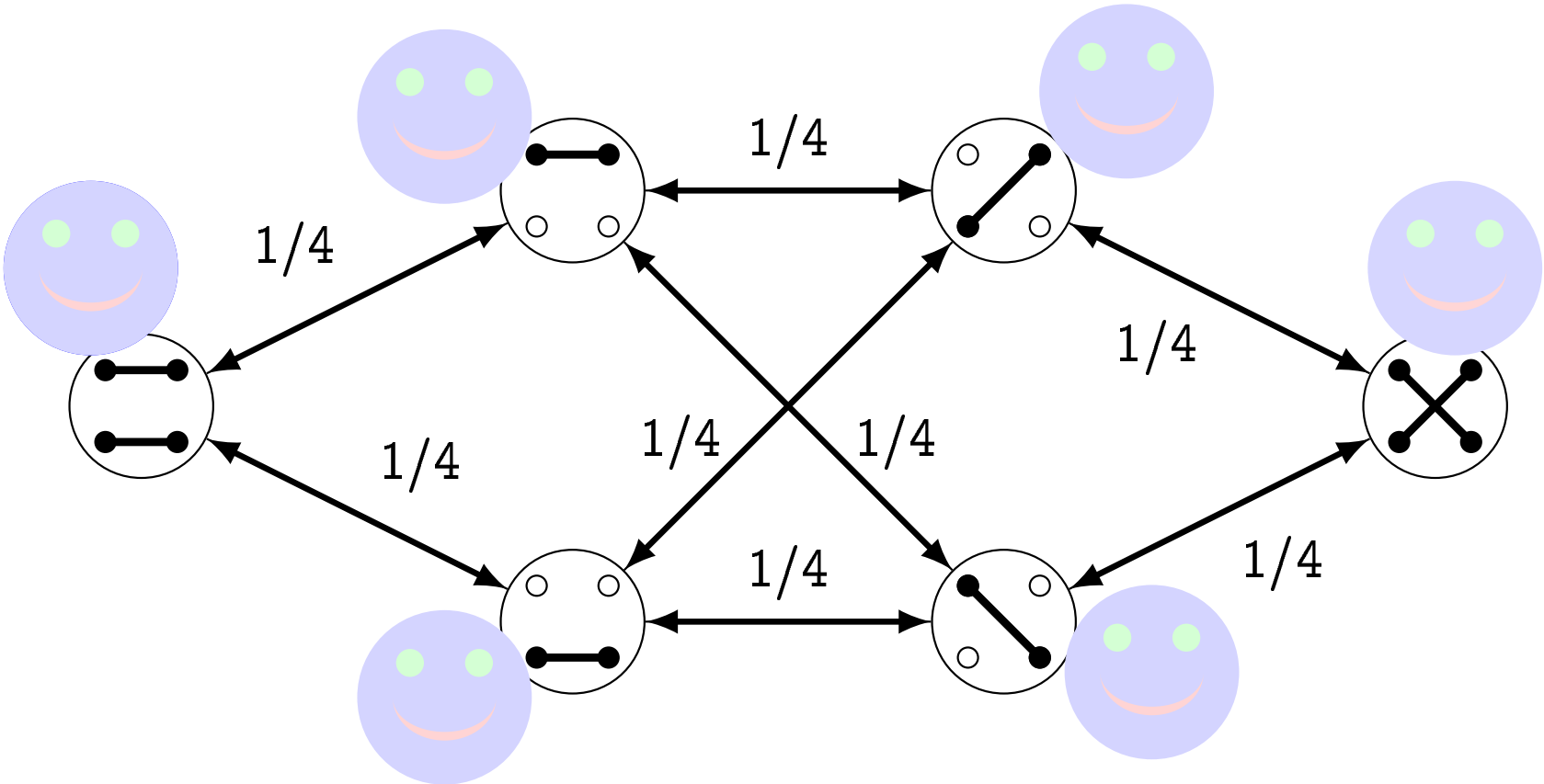
after tenth step ...



Sampling From the Large Matchings

The random walk

How many steps are needed to achieve a uniform sample?



How Many Steps Are Needed?

Lemma (The Rapid Convergence Lemma)

Let G be a dense graph. If we take $24n^7$ steps in the random walk associated with G, then the final large matching is an almost uniform sample from $LM(G)$.

Corollary

For a dense G and positive numbers ϵ, δ, it is possible to approximate $\frac{m_n}{m_{n-1}}$ within ratio $(1+\epsilon)$ with probability $1-\delta$ in time poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$.

How Many Steps Are Needed?

Lemma (The Rapid Convergence Lemma)

Let G be a dense graph. If we take $24n^7$ steps in the random walk associated with G, then the final large matching is an almost uniform sample from $LM(G)$.

Corollary

For a dense G and positive numbers ϵ, δ, it is possible to approximate $\frac{m_n}{m_{n-1}}$ within ratio $(1+\epsilon)$ with probability $1-\delta$ in time poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$.

What Next?

Corollary

For a dense G and positive numbers ϵ, δ, it is possible to approximate $\frac{m_n}{m_{n-1}}$ within ratio $(1+\epsilon)$ with probability $1-\delta$ in time poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$.

Recall.

We have:

$$m_n = \frac{m_n}{m_{n-1}} \times \frac{m_{n-1}}{m_{n-2}} \times \cdots \times \frac{m_2}{m_1} \times m_1.$$

We are done with $\frac{m_n}{m_{n-1}}$.
Let us consider rest of the ratios now!

What Next?

Corollary

For a dense G and positive numbers ϵ, δ, it is possible to approximate $\frac{m_n}{m_{n-1}}$ within ratio $(1+\epsilon)$ with probability $1-\delta$ in time poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$.

Recall.

We have:

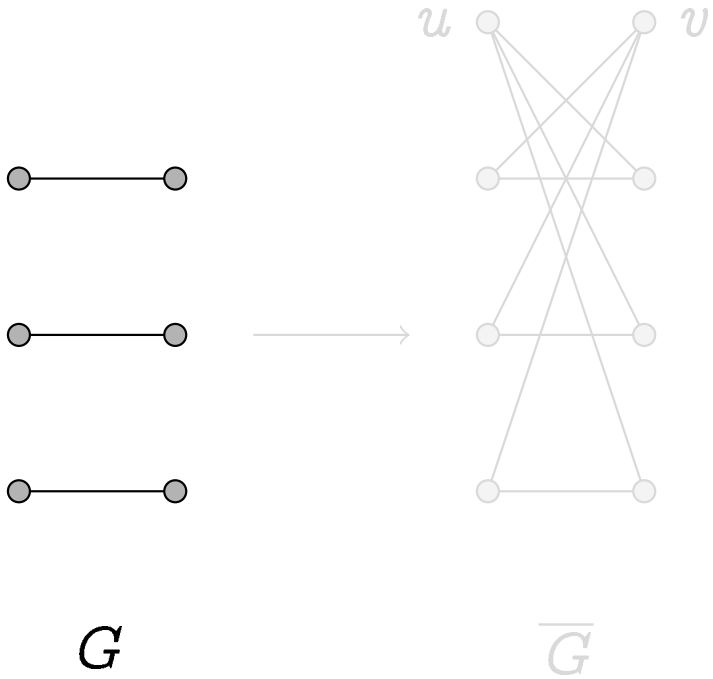
$$m_n = \frac{m_n}{m_{n-1}} \times \frac{m_{n-1}}{m_{n-2}} \times \cdots \times \frac{m_2}{m_1} \times m_1.$$

We are done with $\frac{m_n}{m_{n-1}}$.
Let us consider rest of the ratios now!

The Auxiliary Graph $\overline{G}$

We know how to approximate m_3/m_2.

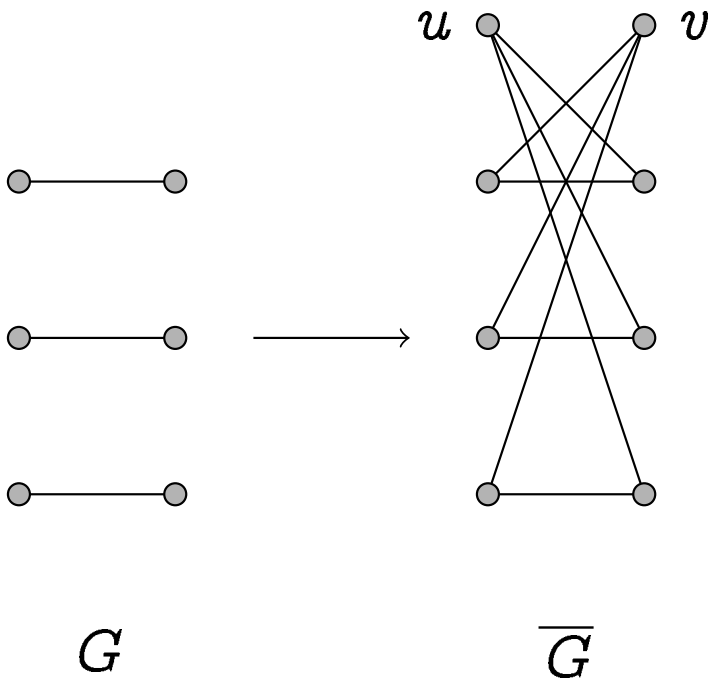
Suppose we want to approximate m_2/m_1:



The Auxiliary Graph $\overline{G}$

We know how to approximate m_3/m_2.

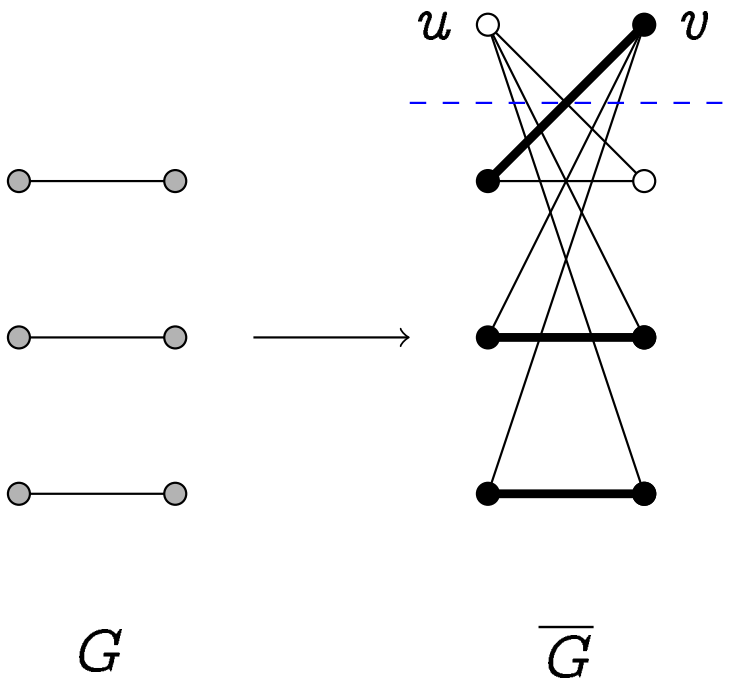
Suppose we want to approximate m_2/m_1:



The Auxiliary Graph $\overline{G}$

We know how to approximate m_3/m_2.

Suppose we want to approximate m_2/m_1:

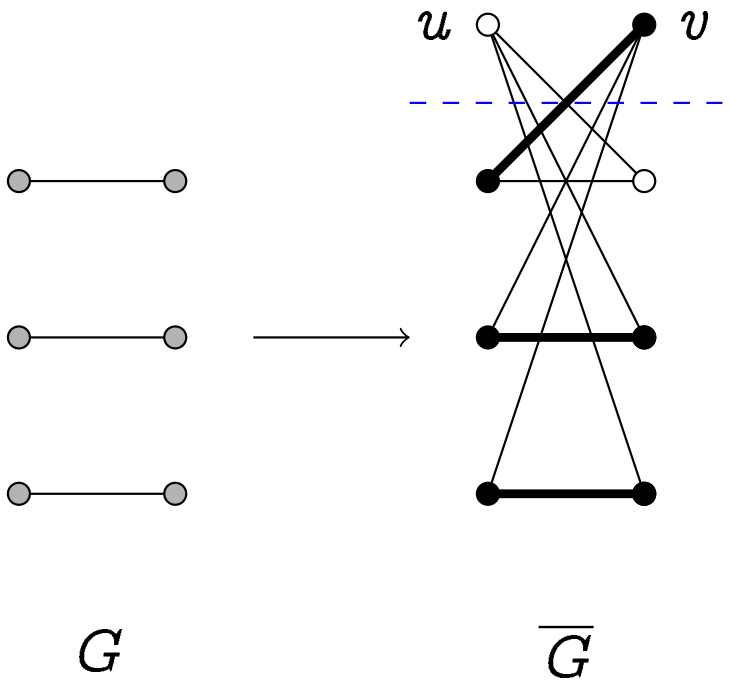


There are four types of large matchings of $\overline{G}$.

The Auxiliary Graph $\overline{G}$

We know how to approximate m_3/m_2.

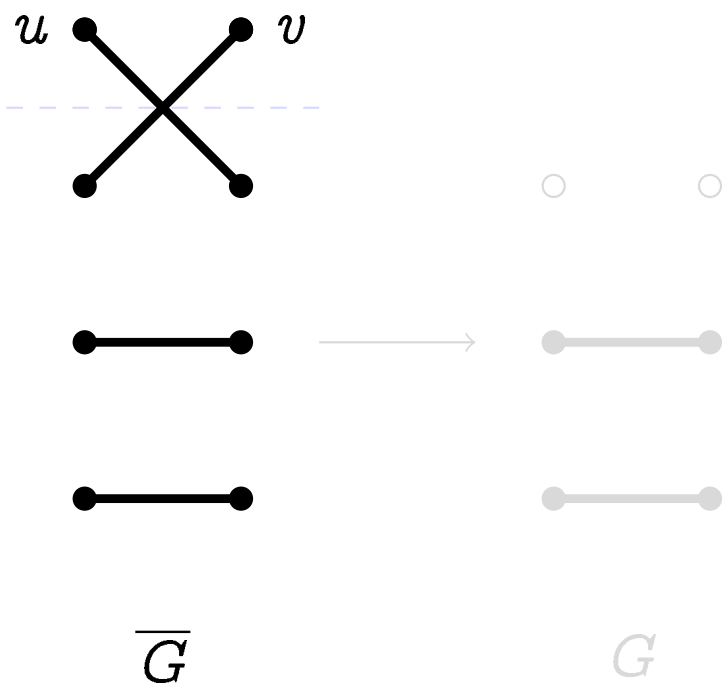
Suppose we want to approximate m_2/m_1:



There are four types of large matchings of $\overline{G}$.

Large Matchings of the Auxiliary Graph

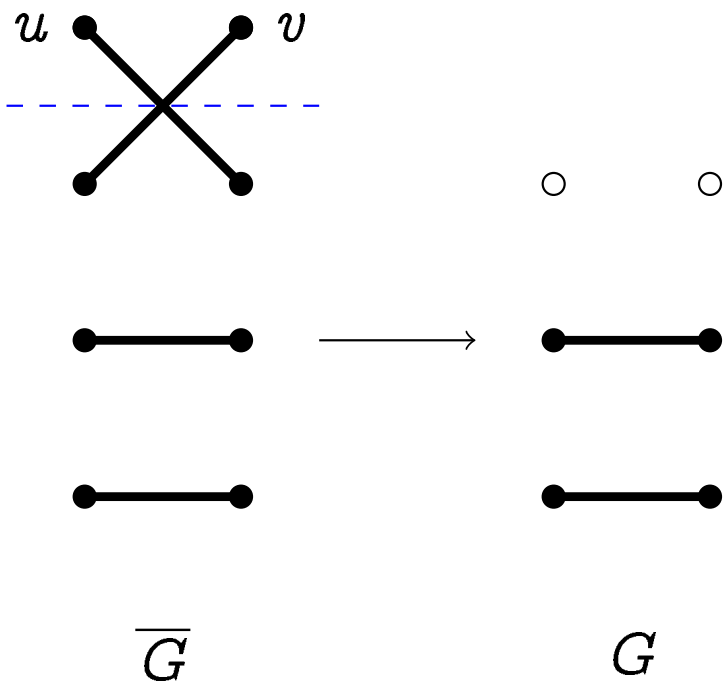
First type: a 4-matching



There are $m_2(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

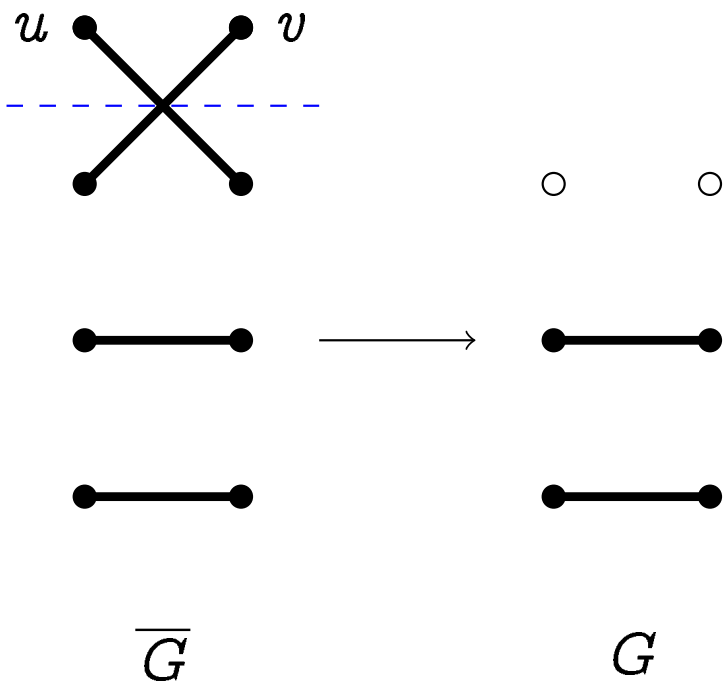
First type: a 4-matching



There are $m_2(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

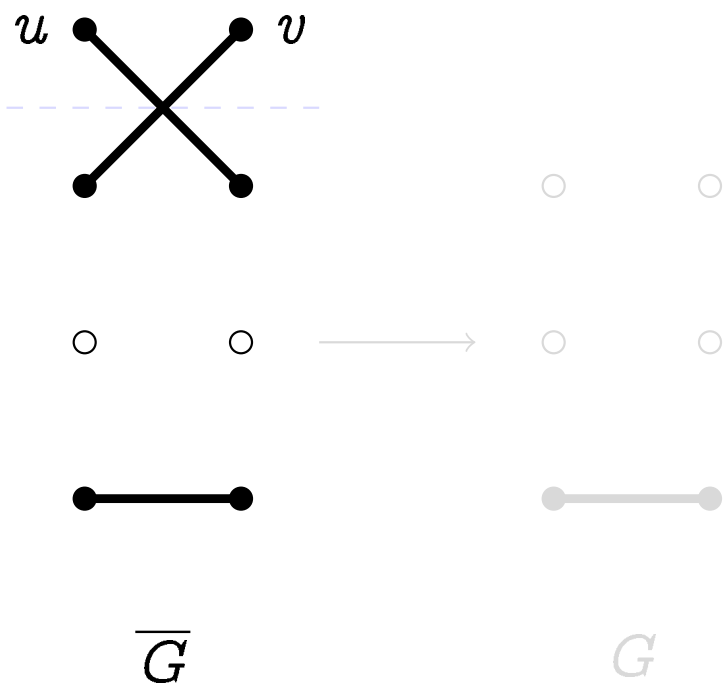
First type: a 4-matching



There are $m_2(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

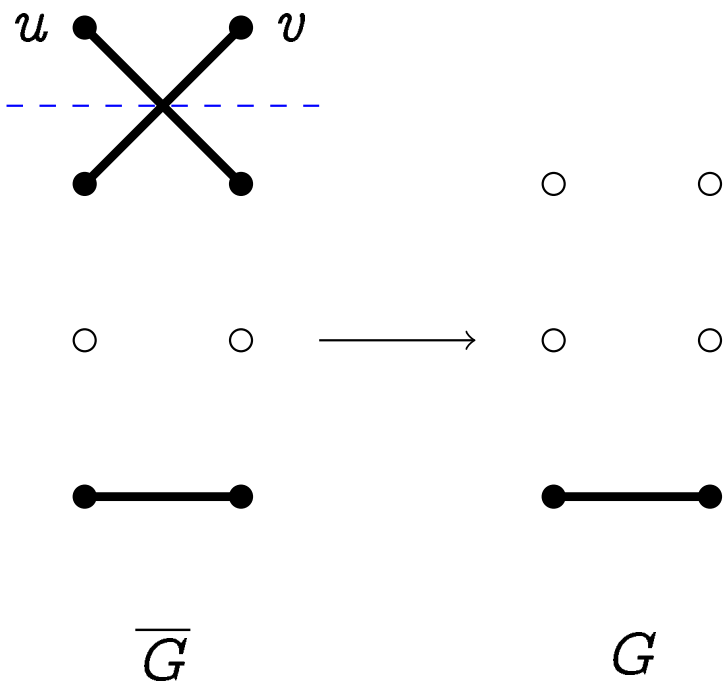
Second type: a 3-matching that covers both of u and v



There are $4m_1(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

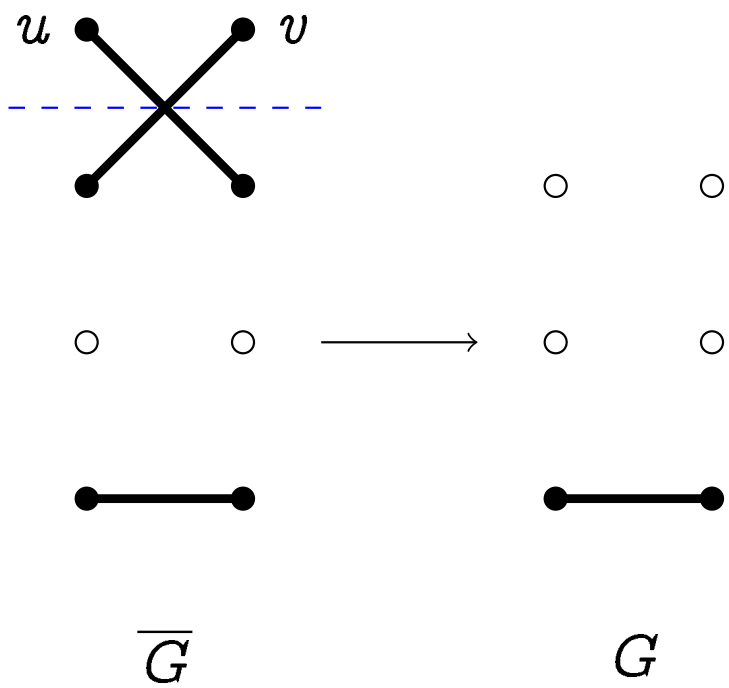
Second type: a 3-matching that covers both of u and v



There are $4m_1(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

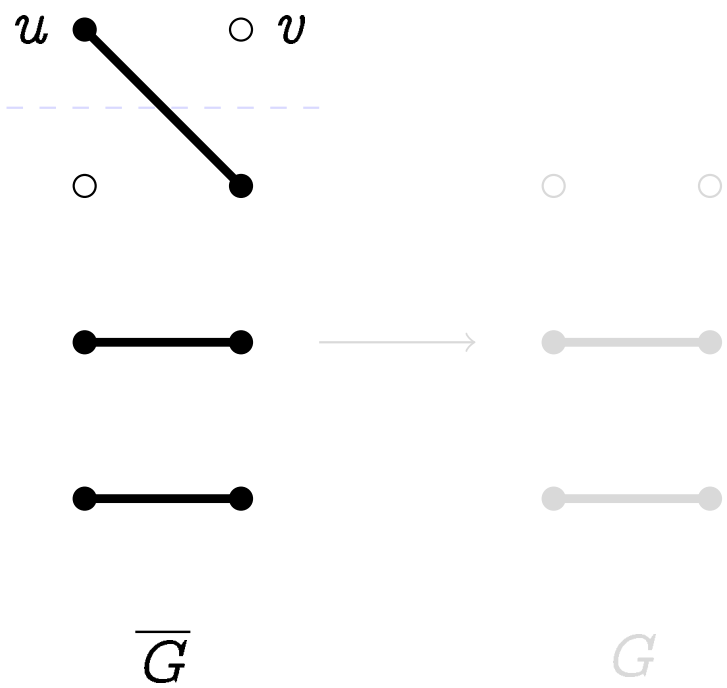
Second type: a 3-matching that covers both of u and v



There are $4m_1(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

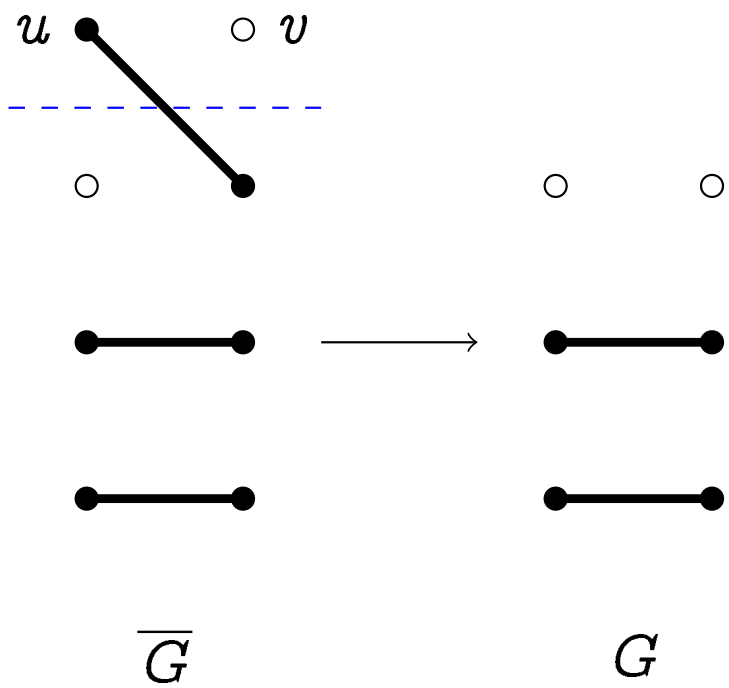
Third type: a 3-matching that covers one of u and v



There are $2m_2(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

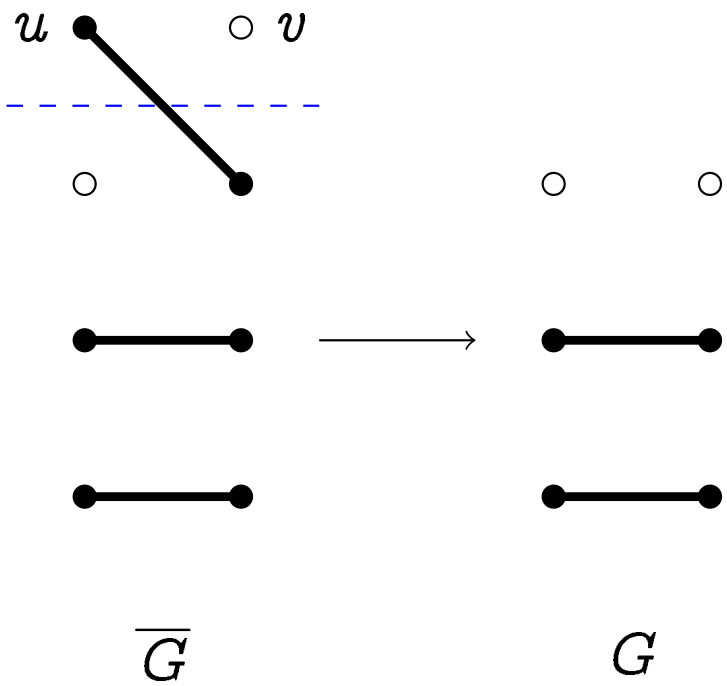
Third type: a 3-matching that covers one of u and v



There are $2m_2(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

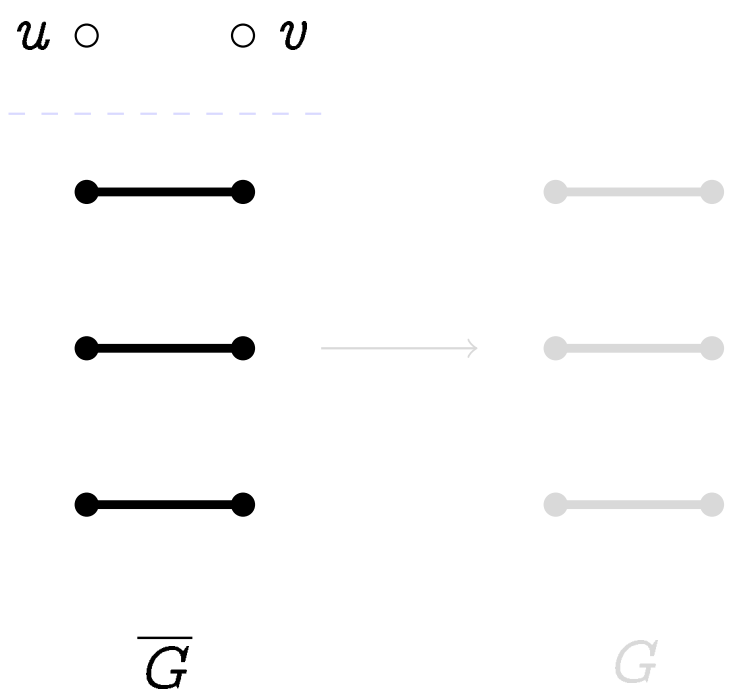
Third type: a 3-matching that covers one of u and v



There are $2m_2(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

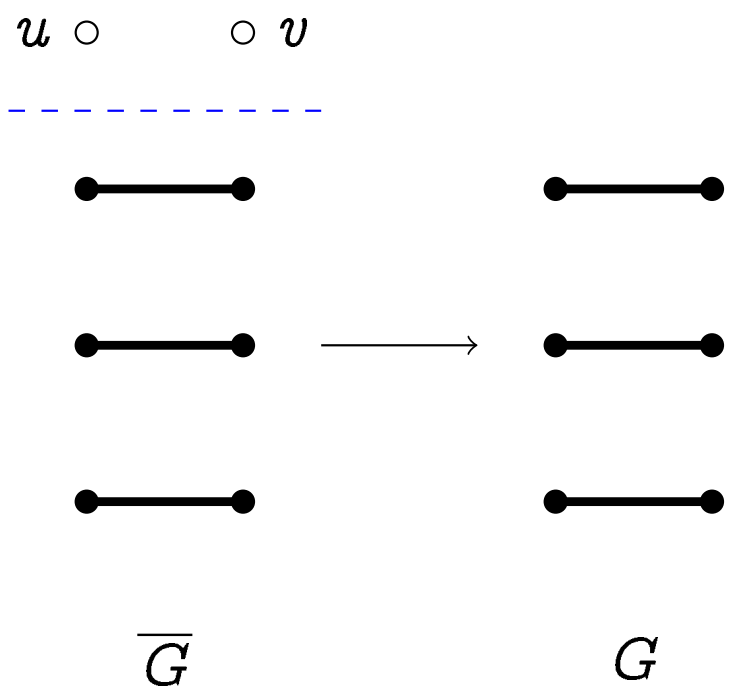
Fourth type: a 3-matching that covers none of u and v



There are $m_3(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

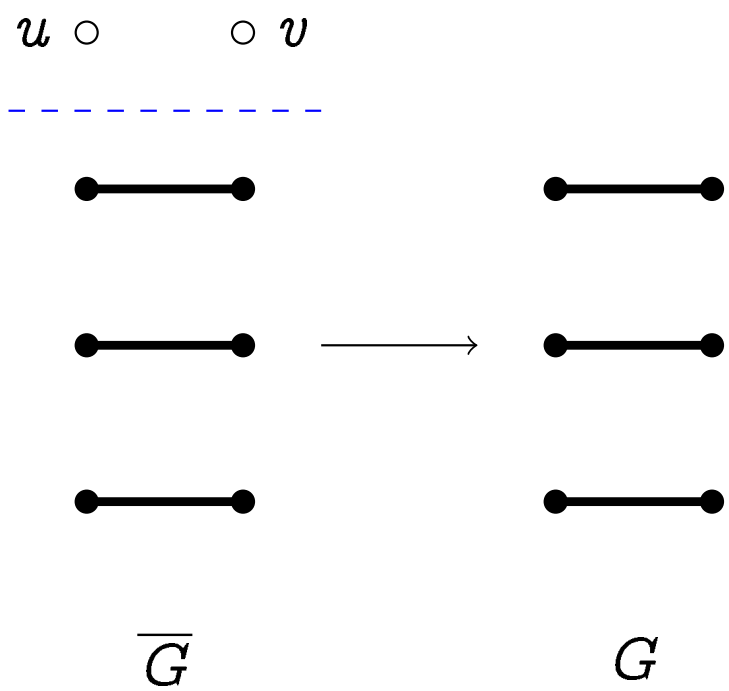
Fourth type: a 3-matching that covers none of u and v



There are $m_3(G)$ matchings of this type.

Large Matchings of the Auxiliary Graph

Fourth type: a 3-matching that covers none of u and v



There are $m_3(G)$ matchings of this type.

Rest of the Ratios

Therefore, the set $LM(\overline{G})$ can be partitioned into four sets T_1, T_2, T_3, T_4, where
$|T_1| = m_2, |T_2| = 4m_1, |T_3| = 2m_2, |T_4| = m_3$.

To approximate m_2/m_1

1. Take X samples from $LM(\overline{G})$.
2. Let X_i be the number of samples of type i, for $i = 1, \ldots, 4$.
3. Then $\frac{X_1+X_3}{X_2}$ is an approximation for $\frac{3m_2}{4m_1}$.
4. Using Sampling Lemma and Technical Lemma, it can be proved that just poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$ samples are needed to approximate m_2/m_1 within ratio $(1+\epsilon)$ with probability $(1-\delta)$.

Rest of the Ratios

Therefore, the set $LM(\overline{G})$ can be partitioned into four sets T_1, T_2, T_3, T_4, where
$|T_1| = m_2, |T_2| = 4m_1, |T_3| = 2m_2, |T_4| = m_3$.

To approximate m_2/m_1

1. Take X samples from $LM(\overline{G})$.
2. Let X_i be the number of samples of type i, for $i = 1, \ldots, 4$.
3. Then $\frac{X_1+X_3}{X_2}$ is an approximation for $\frac{3m_2}{4m_1}$.
4. Using Sampling Lemma and Technical Lemma, it can be proved that just poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$ samples are needed to approximate m_2/m_1 within ratio $(1+\epsilon)$ with probability $(1-\delta)$.

Rest of the Ratios

Therefore, the set $LM(\overline{G})$ can be partitioned into four sets T_1, T_2, T_3, T_4, where
$|T_1| = m_2, |T_2| = 4m_1, |T_3| = 2m_2, |T_4| = m_3$.

To approximate m_2/m_1

1. Take X samples from $LM(\overline{G})$.
2. Let X_i be the number of samples of type i, for $i = 1, \ldots, 4$.
3. Then $\frac{X_1+X_3}{X_2}$ is an approximation for $\frac{3m_2}{4m_1}$.
4. Using Sampling Lemma and Technical Lemma, it can be proved that just poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$ samples are needed to approximate m_2/m_1 within ratio $(1+\epsilon)$ with probability $(1-\delta)$.

Rest of the Ratios

Therefore, the set $LM(\overline{G})$ can be partitioned into four sets T_1, T_2, T_3, T_4, where
$|T_1| = m_2, |T_2| = 4m_1, |T_3| = 2m_2, |T_4| = m_3$.

To approximate m_2/m_1

1. Take X samples from $LM(\overline{G})$.
2. Let X_i be the number of samples of type i, for $i = 1, \ldots, 4$.
3. Then $\frac{X_1+X_3}{X_2}$ is an approximation for $\frac{3m_2}{4m_1}$.
4. Using Sampling Lemma and Technical Lemma, it can be proved that just poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$ samples are needed to approximate m_2/m_1 within ratio $(1+\epsilon)$ with probability $(1-\delta)$.

Rest of the Ratios

Therefore, the set $LM(\overline{G})$ can be partitioned into four sets T_1, T_2, T_3, T_4, where
$|T_1| = m_2, |T_2| = 4m_1, |T_3| = 2m_2, |T_4| = m_3$.

To approximate m_2/m_1

1. Take X samples from $LM(\overline{G})$.
2. Let X_i be the number of samples of type i, for $i = 1, \ldots, 4$.
3. Then $\frac{X_1+X_3}{X_2}$ is an approximation for $\frac{3m_2}{4m_1}$.
4. Using Sampling Lemma and Technical Lemma, it can be proved that just poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$ samples are needed to approximate m_2/m_1 within ratio $(1+\epsilon)$ with probability $(1-\delta)$.

Rest of the Ratios

Therefore, the set $LM(\overline{G})$ can be partitioned into four sets T_1, T_2, T_3, T_4, where
$|T_1| = m_2, |T_2| = 4m_1, |T_3| = 2m_2, |T_4| = m_3$.

To approximate m_2/m_1

1. Take X samples from $LM(\overline{G})$.
2. Let X_i be the number of samples of type i, for $i = 1, \ldots, 4$.
3. Then $\frac{X_1+X_3}{X_2}$ is an approximation for $\frac{3m_2}{4m_1}$.
4. Using Sampling Lemma and Technical Lemma, it can be proved that just poly $\left(n, \epsilon^{-1}, \log(\delta^{-1})\right)$ samples are needed to approximate m_2/m_1 within ratio $(1+\epsilon)$ with probability $(1-\delta)$.

Wrapping Things Up

Outline of the Algorithm

1. For each $k = 2, \ldots, n$, suppose that $r_k \simeq m_k/m_{k-1}$ within ratio $(1 + \epsilon/4n)^2$ with probability $(1 - 1/8n)^2$.
2. Return $\widehat{m}_n = m_1 \times r_2 \times r_3 \times \cdots \times r_n$.

Analysis

- Each r_k can be calculated in time $\text{poly}(n, \epsilon^{-1})$.
- The answer $\widehat{m}_n$ approximates m_n within ratio $(1 + \epsilon/4n)^{2n} < 1 + \epsilon$ with probability at least $(1 - 1/8n)^{2n} > 3/4$.

Wrapping Things Up

Outline of the Algorithm

1. For each $k = 2, \ldots, n$, suppose that $r_k \simeq m_k/m_{k-1}$ within ratio $(1 + \epsilon/4n)^2$ with probability $(1 - 1/8n)^2$.
2. Return $\widehat{m}_n = m_1 \times r_2 \times r_3 \times \cdots \times r_n$.

Analysis

- Each r_k can be calculated in time poly(n, ϵ^{-1}).
- The answer $\widehat{m}_n$ approximates m_n within ratio $(1 + \epsilon/4n)^{2n} < 1 + \epsilon$ with probability at least $(1 - 1/8n)^{2n} > 3/4$.

Wrapping Things Up

Outline of the Algorithm

1. For each $k = 2, \ldots, n$, suppose that $r_k \simeq m_k/m_{k-1}$ within ratio $(1 + \epsilon/4n)^2$ with probability $(1 - 1/8n)^2$.
2. Return $\widehat{m}_n = m_1 \times r_2 \times r_3 \times \cdots \times r_n$.

Analysis

- Each r_k can be calculated in time $\text{poly}(n, \epsilon^{-1})$.
- The answer $\widehat{m}_n$ approximates m_n within ratio $(1 + \epsilon/4n)^{2n} < 1 + \epsilon$ with probability at least $(1 - 1/8n)^{2n} > 3/4$.

Wrapping Things Up

Outline of the Algorithm

1. For each $k = 2, \ldots, n$, suppose that $r_k \simeq m_k/m_{k-1}$ within ratio $(1 + \epsilon/4n)^2$ with probability $(1 - 1/8n)^2$.
2. Return $\widehat{m}_n = m_1 \times r_2 \times r_3 \times \cdots \times r_n$.

Analysis

- Each r_k can be calculated in time $\text{poly}(n, \epsilon^{-1})$.
- The answer $\widehat{m}_n$ approximates m_n within ratio $(1 + \epsilon/4n)^{2n} < 1 + \epsilon$ with probability at least $(1 - 1/8n)^{2n} > 3/4$.

Notes

- This algorithm was proposed in 1986 by Broder.
- The Rapid Convergence Lemma (for dense bipartite graphs) was proved in 1988 by Jerrum and Sinclair.
- An fpras for nondense bipartite graphs was found in 2001 by Jerrum, Sinclair and Vigoda, which uses a more complicated random walk.
- The problem of finding an fpras for non-bipartite graphs is open.

What is The Permanent?

Definition

The permanent of an $n \times n$ matrix $A = [a_{i,j}]$ is defined as

$$\text{per}(A) = \sum_{\sigma} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

where the sum is over all permutations σ of $\{1, 2, \ldots, n\}$.

Example

$$A = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

What is The Permanent?

Definition

The permanent of an $n \times n$ matrix $A = [a_{i,j}]$ is defined as

$$\text{per}(A) = \sum_{\sigma} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

where the sum is over all permutations σ of $\{1, 2, \ldots, n\}$.

Example

$$A = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

What is The Permanent?

Definition

The permanent of an $n \times n$ matrix $A = [a_{i,j}]$ is defined as

$$\text{per}(A) = \sum_{\sigma} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

where the sum is over all permutations σ of $\{1, 2, \ldots, n\}$.

Example

$$\text{per}(A) = 2: \quad \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Estimating the Permanent of a 0,1-Matrix

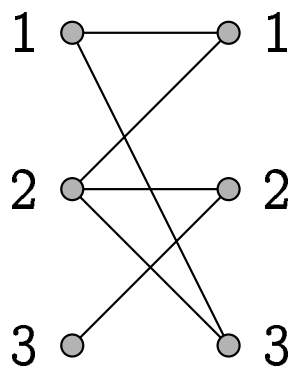
The permanent of a 0,1-matrix is equal to the number of perfect matchings of a bipartite graph:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \end{array}$$

Estimating the Permanent of a 0,1-Matrix

The permanent of a 0,1-matrix is equal to the number of perfect matchings of a bipartite graph:

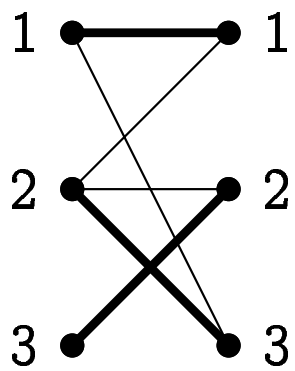
$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \end{array}$$



Estimating the Permanent of a 0,1-Matrix

The permanent of a 0,1-matrix is equal to the number of perfect matchings of a bipartite graph:

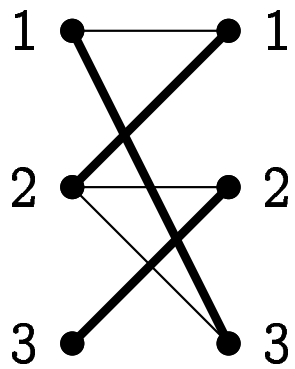
$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \end{array}$$



Estimating the Permanent of a 0,1-Matrix

The permanent of a 0,1-matrix is equal to the number of perfect matchings of a bipartite graph:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \end{array}$$



Computational Complexity of Estimating Permanent

- The discussed algorithm gives an fpras for permanent of a 0,1-matrix in which each row/column sums to at least $n/2$.
- Jerrum et al. gave an fpras for the permanent of any matrix with nonnegative entries in 2001.
- They showed that there is no fpras for the permanent of a general matrix unless $P = NP$.