Approximating the Permanent

Eric Vigoda
Georgia Tech
KAIST (Spring '15)

Guest lecture for
KAIST CS 500
Graduate Algorithms
Wednesday, March 11, 2015

1. Permanent Definition
2. Random Matching
3. Random Perfect Matching

What is the Permanent?

3×3 *example:*

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Determinant of A:

$$\det(A) = (aei + bfg + cdh) - (ceg + bdi + afh).$$

Permanent of A:

$$\mathrm{per}(A) = aei + bfg + cdh + ceg + bdi + afh.$$

What is the Permanent?

3×3 *example:*

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Determinant of A:

$$\det(A) = (aei + bfg + cdh) - (ceg + bdi + afh).$$

Permanent of A:

$$\operatorname{per}(A) = aei + bfg + cdh + ceg + bdi + afh.$$

In general, for a $n \times n$ matrix A, the determinant of A is

$$\det(A) = \sum_{\pi} \operatorname{sgn}(\pi) \prod_{i} A(i, \pi(i)),$$

where π ranges over all permutations of $\{1, \ldots, n\}$.

The permanent of A is

$$\operatorname{per}(A) = \sum_{\pi} \prod_{i} A(i, \pi(i))$$

What is the significance of the Permanent?

$$\mathrm{per}(A) = \sum_{\pi} \prod_{i} A(i, \pi(i))$$

What is its significance?

For $0 - 1$ matrix, view A as adjacency matrix for bipartite graph.

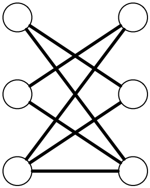
What is the significance of the Permanent?

$$\mathrm{per}(A) = \sum_{\pi} \prod_{i} A(i, \pi(i))$$

What is its significance?

For $0-1$ matrix, view A as adjacency matrix for bipartite graph.

Example: $A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$



$$\mathrm{per}(A) \quad = \quad aei + bfg + cdh + ceg + bdi + afh$$

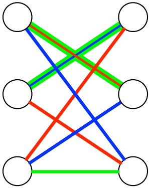
What is the significance of the Permanent?

$$\mathrm{per}(A) = \sum_{\pi} \prod_{i} A(i, \pi(i))$$

What is its significance?

For $0-1$ matrix, view A as adjacency matrix for bipartite graph.

Example: $A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$



$$\begin{aligned} \mathrm{per}(A) &= aei + bfg + cdh + ceg + bdi + afh \\ &= bfg + cdh + bdi = 3 \end{aligned}$$

Applications

Some applications of the Permanent:

- *Statistical Physics:*
  - Dimer model of adsorption of diatomic molecules,
  - Ice-type models of crystal lattices with hydrogen bonds,
- *Computer Vision:* Tracking objects
- Number of graphs with specified degree sequence

What is the Permanent?

Given a graph $G = (V, E)$ with $n = |V|$ vertices,

$$\text{let } \mathcal{P} = \text{perfect matchings of } G.$$

Can we compute $|\mathcal{P}|$ in time polynomial in n?

What is the Permanent?

Given a graph $G = (V, E)$ with $n = |V|$ vertices,

$$\text{let } \mathcal{P} = \text{perfect matchings of } G.$$

Can we compute $|\mathcal{P}|$ in time polynomial in n?

- Polynomial time algorithm for planar graphs [Kasteleyn '67]
- #P-complete for bipartite graphs [Valiant '79]
- FPRAS for counting *all* matchings [Jerrum-Sinclair '89]
- FPRAS for counting perfect matchings of bipartite [JSV '04]
  Fastest algorithm: $O^*(n^7)$ time [BSVV '09]

What is the Permanent?

Given a graph $G = (V, E)$ with $n = |V|$ vertices,

$$\text{let } \mathcal{P} = \text{perfect matchings of } G.$$

Can we compute $|\mathcal{P}|$ in time polynomial in n?

- Polynomial time algorithm for planar graphs [Kasteleyn '67]
- #P-complete for bipartite graphs [Valiant '79]
- FPRAS for counting *all* matchings [Jerrum-Sinclair '89]
- FPRAS for counting perfect matchings of bipartite [JSV '04]
  Fastest algorithm: $O^*(n^7)$ time [BSVV '09]

Agenda

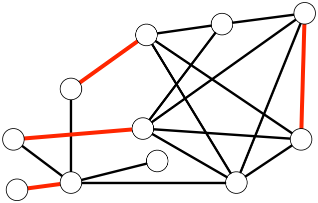
Main tasks:

1. Count all matchings or generate a random matching.
2. Count perfect matchings or generate a random perfect matching.

1. Permanent Definition
2. Random Matching
3. Random Perfect Matching

Random Matching

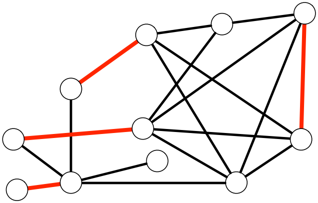
Undirected graph $G = (V, E)$:



Matching = subset of vertex disjoint edges.

Let Ω = collection of all matchings of G (of all sizes).

Undirected graph $G = (V, E)$:



Matching = subset of vertex disjoint edges.

Let $\Omega =$ collection of all matchings of G (of all sizes).

Can we generate a matching uniformly at random from Ω?
in time polynomial in $n = |V|$?

Markov Chain for Matchings

Consider an undirected graph $G = (V, E)$.

From a matching X_t the transition $X_t \to X_{t+1}$
is defined as follows:

1. Choose an edge $e = (v, w)$ uniformly at random from E.
2. *Remove:* If $e \in X_t$ then set $X_{t+1} = X_t \setminus \{e\}$.
3. *Add:* If v and w are unmatched in X_t then $X_{t+1} = X_t \bigcup \{e\}$.
4. Otherwise, set $X_{t+1} = X_t$.

Symmetric and ergodic, hence:

unique stationary distribution π is uniform over Ω.

Markov Chain for Matchings

Consider an undirected graph $G = (V, E)$.

From a matching X_t the transition $X_t \rightarrow X_{t+1}$
is defined as follows:

1. Choose an edge $e = (v, w)$ uniformly at random from E.
2. *Remove:* If $e \in X_t$ then set $X_{t+1} = X_t \setminus \{e\}$.
3. *Add:* If v and w are unmatched in X_t then $X_{t+1} = X_t \bigcup \{e\}$.
4. Otherwise, set $X_{t+1} = X_t$.

Symmetric and ergodic, hence:

unique stationary distribution π is uniform over Ω.

Mixing time $=$ How fast does it reach π?

Augmented Markov Chain for Matchings

Consider an undirected graph $G = (V, E)$.

From a matching X_t the transition $X_t \to X_{t+1}$ is defined by:

1. Choose an edge $e = (v, w)$ uniformly at random from E.
2. *Remove:* If $e \in X_t$ then set $X_{t+1} = X_t \setminus \{e\}$.
3. *Add:* If v and w are unmatched in X_t then $X_{t+1} = X_t \bigcup \{e\}$.
4. *Slide:* If v is unmatched and w is matched (or vice-versa):
    1. Let (w, z) denote the matched edge.
    2. Set $X_{t+1} = X_t \bigcup (v, w) \setminus (w, z)$.
5. Otherwise, set $X_{t+1} = X_t$.

Mixing Time

How fast does an ergodic MC reach its unique stationary π?

Mixing Time

How fast does an ergodic MC reach its unique stationary π?

Need to measure distance from π, use total variation distance.
For distributions μ and ν on set Ω:

$$\mathrm{d}_{\mathrm{TV}}(\mu, \nu) = \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)|.$$

Mixing Time

How fast does an ergodic MC reach its unique stationary π?

Need to measure distance from π, use total variation distance.
For distributions μ and ν on set Ω:

$$d_{TV}(\mu, \nu) = \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)|.$$

Example: $\Omega = \{1, 2, 3, 4\}$.
μ is uniform: $\mu(1) = \mu(2) = \mu(3) = \mu(4) = .25$.
And ν has: $\nu(1) = .5, \nu(2) = .1, \nu(3) = .15, \nu(4) = .25$.

$$d_{TV}(\mu, \nu) = \frac{1}{2}(.25 + .15 + .1 + 0) = .25$$

Mixing Time

Consider ergodic MC with states Ω, transition matrix P, and unique stationary distribution π.
For state $x \in \Omega$, time to mix from x:

$$T(x) = \min\{t : \mathrm{d}_{\mathrm{TV}}(\mathrm{P}^t(x, \cdot), \pi) \leq 1/4\}$$

Mixing Time

Consider ergodic MC with states Ω, transition matrix P, and unique stationary distribution π.
For state $x \in \Omega$, time to mix from x:

$$T(x) = \min\{t : \mathrm{d}_{\mathrm{TV}}(\mathrm{P}^t(x,\cdot),\pi) \le 1/4\}$$

Then, mixing time $\mathrm{T}_{\mathrm{mix}} = \max_x T(x)$.
Summarizing in words:

mixing time is time to get within distance $\le 1/4$ of π from the worst initial state X_0.

Mixing Time

Consider ergodic MC with states Ω, transition matrix P, and unique stationary distribution π.
For state $x \in \Omega$, time to mix from x:

$$T(x) = \min\{t : \mathrm{d_{TV}}(\mathrm{P}^t(x,\cdot),\pi) \le 1/4\}$$

Then, mixing time $\mathrm{T_{mix}} = \max_x T(x)$.
Summarizing in words:

> mixing time is time to get within distance $\le 1/4$ of π from the worst initial state X_0.

Choice of constant $1/4$ is somewhat arbitrary.
Can "boost" to distance $\le \epsilon$ in time $O(\mathrm{T_{mix}} \log(1/\epsilon))$.

Mixing Time

Consider ergodic MC with states Ω, transition matrix P, and unique stationary distribution π.
For state $x \in \Omega$, time to mix from x:

$$T(x) = \min\{t : \mathrm{d}_{\mathrm{TV}}(\mathrm{P}^t(x,\cdot),\pi) \leq 1/4\}$$

Then, mixing time $\mathrm{T}_{\mathrm{mix}} = \max_x T(x)$.
Summarizing in words:

mixing time is time to get within distance $\leq 1/4$ of π from the worst initial state X_0.

Choice of constant $1/4$ is somewhat arbitrary.
Can "boost" to distance $\leq \epsilon$ in time $O(\mathrm{T}_{\mathrm{mix}} \log(1/\epsilon))$.

Rapidly mixing if $\mathrm{T}_{\mathrm{mix}} = \mathrm{poly}(n)$.

Relaxation time $\mathrm{T}_{\mathrm{rel}}$ = mixing time from a nice initial μ_0.

How to Bound Convergence Time

Underlying directed graph $H = (\Omega, E_{\mathrm{P}})$ of the Markov chain:

Vertices $=$ states Ω

Edges $= \{M \to M' : M, M' \in \Omega, \mathrm{P}(M, M') > 0\}$.

How to Bound Convergence Time

Underlying directed graph $H = (\Omega, E_{\mathrm{P}})$ of the Markov chain:

Vertices $=$ states Ω

Edges $= \{M \to M' : M, M' \in \Omega, \mathrm{P}(M, M') > 0\}$.

For a set $S \subset \Omega$ where $\pi(S) \leq 1/2$ denote its conductance by:

$$\Phi(S) = \mathbf{Pr}\left(X_{t+1} \notin S \mid X_t \in S, X_t \sim \pi\right) = \sum_{M \in S, M' \in \overline{S}} \frac{\pi(M)\mathrm{P}(M, M')}{\pi(S)}$$

$$\text{Let } \Phi = \min_S \Phi(S)$$

How to Bound Convergence Time

Underlying directed graph $H = (\Omega, E_{\mathrm{P}})$ of the Markov chain:

Vertices = states Ω

Edges = $\{M \to M' : M, M' \in \Omega, \mathrm{P}(M, M') > 0\}$.

For a set $S \subset \Omega$ where $\pi(S) \leq 1/2$ denote its conductance by:

$$\Phi(S) = \mathbf{Pr}\left(X_{t+1} \notin S \mid X_t \in S, X_t \sim \pi\right) = \sum_{M \in S, M' \in \overline{S}} \frac{\pi(M)\mathrm{P}(M, M')}{\pi(S)}$$

For our chain since π is uniform simplifies to:

$$\Phi(S) = \frac{1}{m} \frac{\#\{\text{of edges from } S \text{ to } \overline{S}\}}{|S|}$$

$$\text{Let } \Phi = \min_S \Phi(S)$$

How to Bound Convergence Time

Underlying directed graph $H = (\Omega, E_{\mathrm{P}})$ of the Markov chain:

Vertices $=$ states Ω

Edges $= \{M \to M' : M, M' \in \Omega, \mathrm{P}(M, M') > 0\}$.

For a set $S \subset \Omega$ where $\pi(S) \leq 1/2$ denote its conductance by:

$$\Phi(S) = \mathbf{Pr}\left(X_{t+1} \notin S \mid X_t \in S, X_t \sim \pi\right) = \sum_{M \in S, M' \in \overline{S}} \frac{\pi(M)\mathrm{P}(M, M')}{\pi(S)}$$

For our chain since π is uniform simplifies to:

$$\Phi(S) = \frac{1}{m} \frac{\#\{\text{of edges from } S \text{ to } \overline{S}\}}{|S|}$$

$$\text{Let } \Phi = \min_S \Phi(S)$$

$$\Omega(1/\Phi) = \mathrm{T}_{\mathrm{rel}} = O(1/\Phi^2).$$

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.
For edge $T = M \to M' \in E_{\mathrm{P}}$, define its *congestion*:

$$\mathrm{cp}(T) = \{(I, F) : T \in \gamma_{I,F}\}$$

$$\text{Let } \rho = \max_{T \in E_{\mathrm{P}}} \frac{|\mathrm{cp}(T)|}{|\Omega|}$$

$$\Phi \geq \frac{1}{2m\rho}$$

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.
For edge $T = M \to M' \in E_{\mathrm{P}}$, define its *congestion*:

$$\mathrm{cp}(T) = \{(I, F) : T \in \gamma_{I,F}\}$$

$$\text{Let } \rho = \max_{T \in E_{\mathrm{P}}} \frac{|\mathrm{cp}(T)|}{|\Omega|}$$

$$\Phi \geq \frac{1}{2m\rho}$$

Corollary: $\mathrm{T}_{\mathrm{rel}} = O((m\rho)^2)$.

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.
For edge $T = M \to M' \in E_{\mathrm{P}}$, define its *congestion*:

$$\mathrm{cp}(T) = \{(I, F) : T \in \gamma_{I,F}\}$$

$$\text{Let } \rho = \max_{T \in E_{\mathrm{P}}} \frac{|\mathrm{cp}(T)|}{|\Omega|}$$

$$\Phi \geq \frac{1}{2m\rho}$$

Proof: For $S \subset \Omega$ where $|S| \leq |\overline{S}|$:

Corollary: $\mathrm{T}_{\mathrm{rel}} = O((m\rho)^2)$.

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.
For edge $T = M \to M' \in E_{\mathrm{P}}$, define its *congestion*:

$$\mathrm{cp}(T) = \{(I, F) : T \in \gamma_{I,F}\}$$

$$\text{Let } \rho = \max_{T \in E_{\mathrm{P}}} \frac{|\mathrm{cp}(T)|}{|\Omega|}$$

$$\Phi \geq \frac{1}{2m\rho}$$

Proof: For $S \subset \Omega$ where $|S| \leq |\overline{S}|$:
$|S| \times |\overline{S}|$ (I, F) pairs where $I \in S$ and $F \in \overline{S}$.

Corollary: $\mathrm{T}_{\mathrm{rel}} = O((m\rho)^2)$.

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.
For edge $T = M \to M' \in E_{\mathrm{P}}$, define its *congestion*:

$$\mathrm{cp}(T) = \{(I, F) : T \in \gamma_{I,F}\}$$

$$\text{Let } \rho = \max_{T \in E_{\mathrm{P}}} \frac{|\mathrm{cp}(T)|}{|\Omega|}$$

$$\Phi \geq \frac{1}{2m\rho}$$

Proof: For $S \subset \Omega$ where $|S| \leq |\overline{S}|$:
$|S| \times |\overline{S}|$ (I, F) pairs where $I \in S$ and $F \in \overline{S}$.
Each $T \in E_{\mathrm{P}}$ has at most $\rho|\Omega|$ paths thru it.

Corollary: $\mathrm{T}_{\mathrm{rel}} = O((m\rho)^2)$.

How to Bound Conductance

For every pair $I, F \in \Omega$ define a path $\gamma_{I,F}$ along edges of H.
For edge $T = M \to M' \in E_{\mathrm{P}}$, define its *congestion*:

$$\mathrm{cp}(T) = \{(I, F) : T \in \gamma_{I,F}\}$$

$$\text{Let } \rho = \max_{T \in E_{\mathrm{P}}} \frac{|\mathrm{cp}(T)|}{|\Omega|}$$

$$\Phi \geq \frac{1}{2m\rho}$$

Proof: For $S \subset \Omega$ where $|S| \leq |\overline{S}|$:
$|S| \times |\overline{S}|$ (I, F) pairs where $I \in S$ and $F \in \overline{S}$.
Each $T \in E_{\mathrm{P}}$ has at most $\rho|\Omega|$ paths thru it.
Hence, $\geq \frac{|S||\overline{S}|}{\rho\Omega} \geq \frac{|S|}{2\rho}$ transitions from S to $\overline{S}$. □

Corollary: $\mathrm{T}_{\mathrm{rel}} = O((m\rho)^2)$.

Markov Chain for Matchings

Consider an undirected graph $G = (V, E)$.

From a matching X_t the transition $X_t \to X_{t+1}$ is defined by:

1. Choose an edge $e = (v, w)$ uniformly at random from E.
2. *Remove:* If $e \in X_t$ then set $X_{t+1} = X_t \setminus \{e\}$.
3. *Add:* If v and w are unmatched in X_t then $X_{t+1} = X_t \bigcup \{e\}$.
4. *Slide:* If v is unmatched and w is matched (or vice-versa):
    1. Let (w, z) denote the matched edge.
    2. Set $X_{t+1} = X_t \bigcup (v, w) \setminus (w, z)$.
5. Otherwise, set $X_{t+1} = X_t$.

Canonical Paths Definition

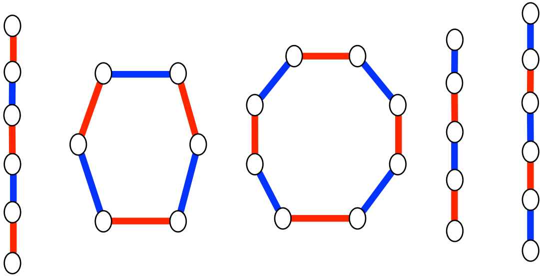
Consider a pair of matchings I and F.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:

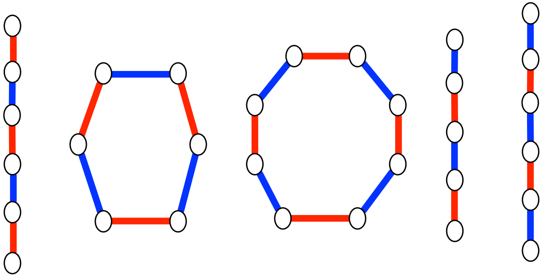


Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

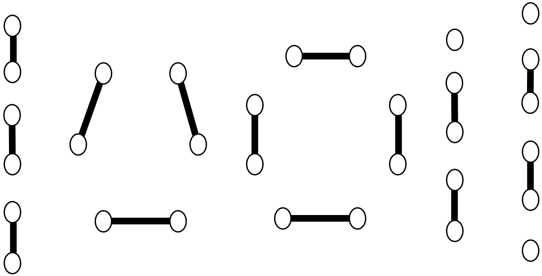
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

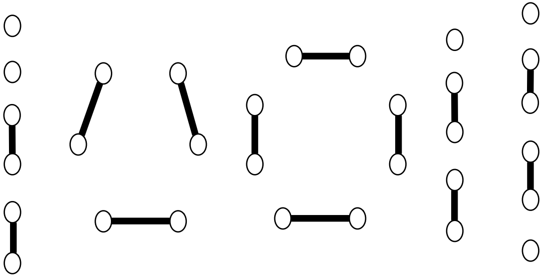
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

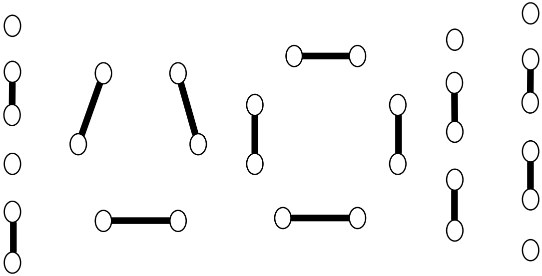
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

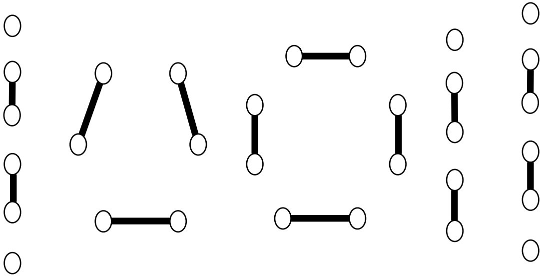
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

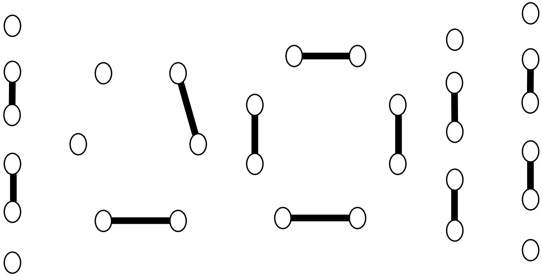
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

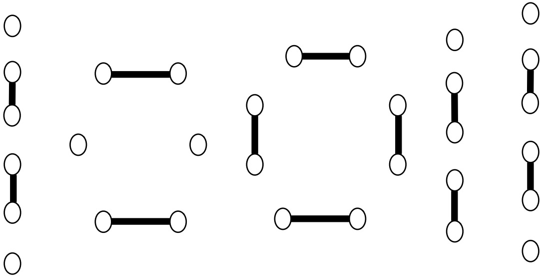
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

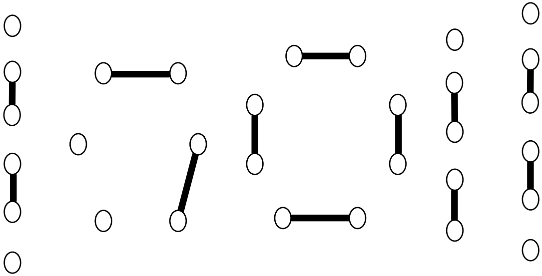
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

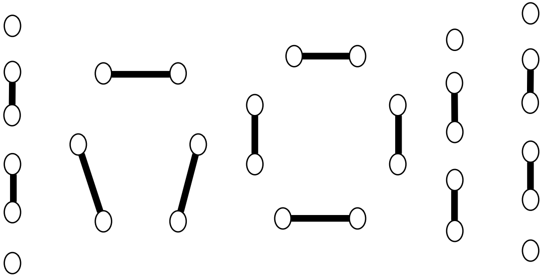
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

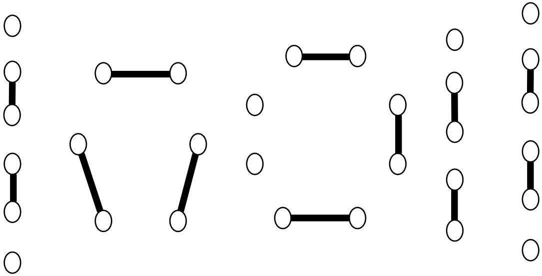
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

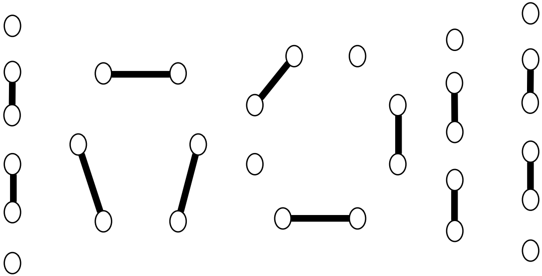
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

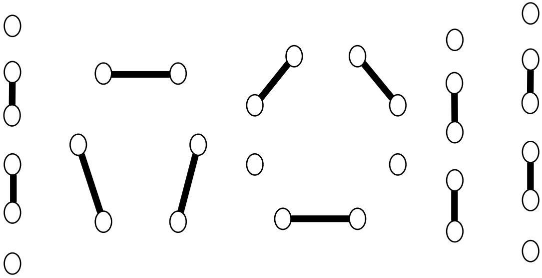
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

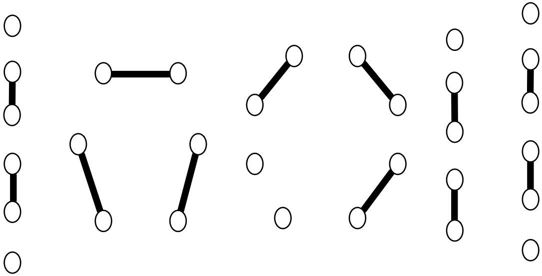
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

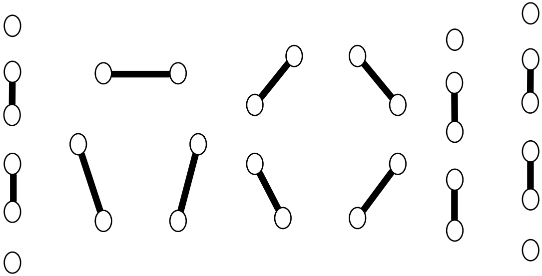
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

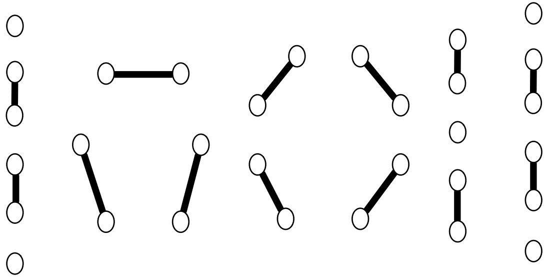
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

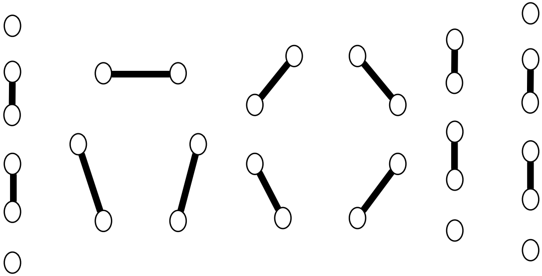
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

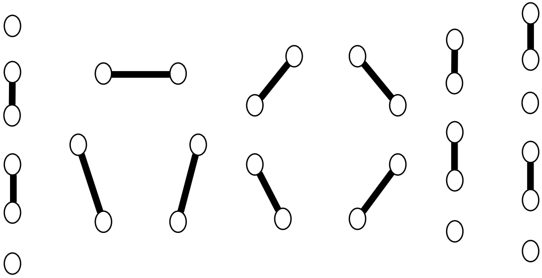
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

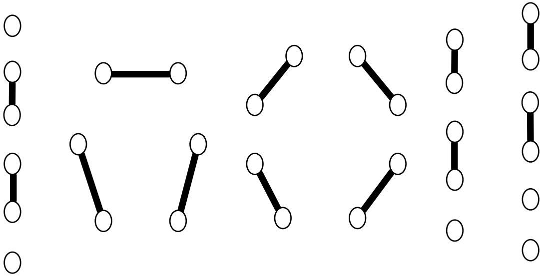
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

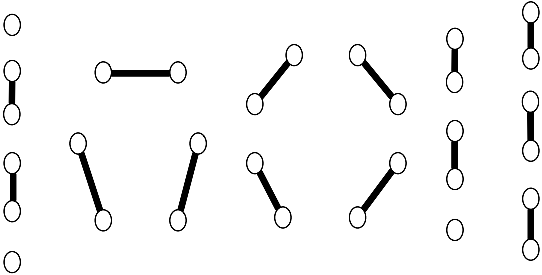
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

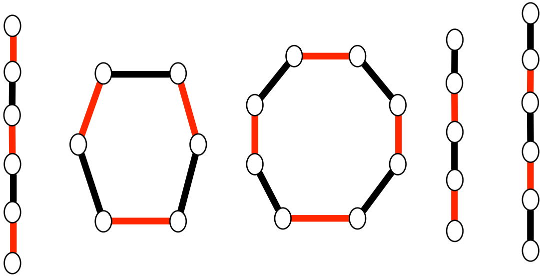
1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Canonical Paths Definition

Consider a pair of matchings I and F.

Look at their difference: $I \oplus F$.

Consists of alternating/augmenting paths and alternating cycles:



Definition of $\gamma_{I,F}$:

1. Order components of $I \oplus F$ by smallest vertex in each.
2. "Unwind" components in order.

Bounding Congestion

Consider a transition $T = M \to M'$.

Recall, $\mathrm{cp}(T) = \{(I, F) : \gamma_{I,F} \ni T\}$.

Bounding Congestion

Consider a transition $T = M \to M'$.

Recall, $\mathrm{cp}(T) = \{(I, F) : \gamma_{I,F} \ni T\}$.

We'll define a map $\eta_T : \mathrm{cp}(T) \to \Omega \times E$.

and η is one-to-one.

Bounding Congestion

Consider a transition $T = M \to M'$.

Recall, $\mathrm{cp}(T) = \{(I, F) : \gamma_{I,F} \ni T\}$.

We'll define a map $\eta_T : \mathrm{cp}(T) \to \Omega \times E$.

and η is one-to-one.

Hence, $|\mathrm{cp}(T)| \leq |\Omega| \times |E|$ so congestion satisfies:

$$\rho = \frac{\max_T |\mathrm{cp}(T)|}{|\Omega|} \leq m.$$

Bounding Congestion

Consider a transition $T = M \to M'$.

Recall, $\mathrm{cp}(T) = \{(I, F) : \gamma_{I,F} \ni T\}$.

We'll define a map $\eta_T : \mathrm{cp}(T) \to \Omega \times E$.

and η is one-to-one.

Hence, $|\mathrm{cp}(T)| \leq |\Omega| \times |E|$ so congestion satisfies:

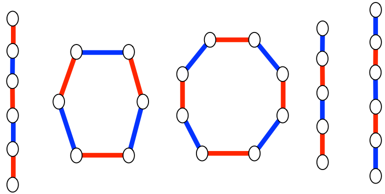
$$\rho = \frac{\max_T |\mathrm{cp}(T)|}{|\Omega|} \leq m.$$

Easy to define η:

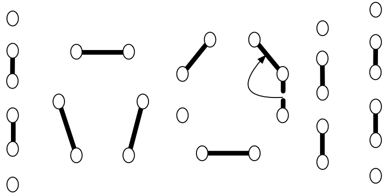
$$\eta_T(I, F) = (I \cap F) \bigcup (I \oplus F \setminus (M \cup M'))$$

Encoding

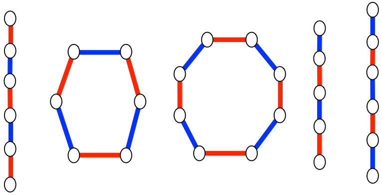
Example I and F:



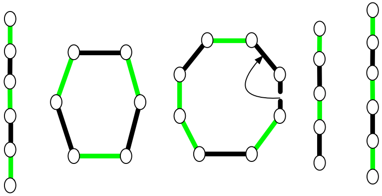
Transition $T = M \to M'$:



Example I and F:

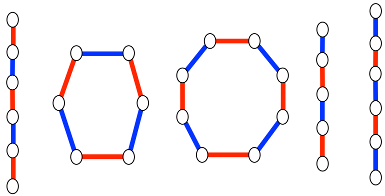


Transition $T = M \rightarrow M'$:

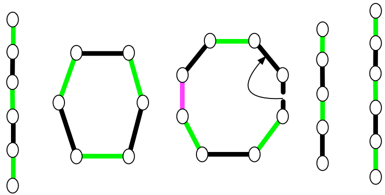


$$\eta_T(I, F) = (I \cap F) \bigcup (I \oplus F \setminus (M \cup M'))$$

Example I and F:



Transition $T = M \to M'$:



$$\eta_T(I,F) = (I \cap F) \bigcup (I \oplus F \setminus (M \cup M' \cup e_0))$$

where e_0 is the first edge of I in the current cycle.

1. Permanent Definition
2. Random Matching
3. Random Perfect Matching

First Idea for Markov Chain

For bipartite graph $G = (V, E)$ with $n + n$ vertices,

$$\text{let } \mathcal{P} = \text{ perfect matchings of } G.$$

Can we design a Markov chain only on $\mathcal{P}$?

What are the transitions?

First Idea for Markov Chain

For bipartite graph $G = (V, E)$ with $n + n$ vertices,

$$\text{let } \mathcal{P} = \text{ perfect matchings of } G.$$

Can we design a Markov chain only on $\mathcal{P}$?

What are the transitions?

Enlarge the states: Near-perfect matchings:

$$\text{let } \mathcal{N} = \text{ matchings of } G \text{ with exactly 2 unmatched vertices.}$$

$$\text{Let } \Omega = \mathcal{P} \bigcup \mathcal{N}.$$

Run earlier Markov chain restricted to Ω.

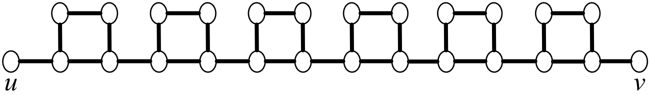
Markov Chain for Perfect Matchings

Consider an undirected bipartite graph $G = (V, E)$.
Let $\Omega = \mathcal{P} \bigcup \mathcal{N}$.

From a matching $X_t \in \Omega$ the transition $X_t \to X_{t+1}$ is defined by:

1. Choose an edge $e = (v, w)$ uniformly at random from E.
2. *Remove:* If $e \in X_t$ and $X_t \in \mathcal{P}$ then set $X_{t+1} = X_t \setminus \{e\}$.
3. *Add:* If v and w are unmatched in X_t then $X_{t+1} = X_t \bigcup \{e\}$.
4. *Slide:* If v is unmatched and w is matched (or vice-versa):
    1. Let (w, z) denote the matched edge.
    2. Set $X_{t+1} = X_t \bigcup (v, w) \setminus (w, z)$.
5. Otherwise, set $X_{t+1} = X_t$.

Bad Example



Key properties:

- $|\mathcal{P}| = 1$: Only 1 perfect matching
- $|\mathcal{N}| \geq 2^{n/4}$: if u and v unmatched then 2^s ways to complete where s is $\#$ of squares.

Conclusion:
Sampling from $\Omega = \mathcal{P} \bigcup \mathcal{N}$ may not help for sampling from $\mathcal{P}$.

Weights on Matchings

Assign matching $M \in \Omega$ a weight $w(M)$.

Add "Metropolis filter" to the Markov chain so that:

Stationary distribution $\pi(M) \propto w(M)$.

Choose weights so that:

1. $\pi(\mathcal{P}) = 1/\text{poly}(n)$ and every $P \in \mathcal{P}$ has the same weight.
2. Markov chain has mixing time poly(n).

Revised Markov Chain

Consider an undirected bipartite graph $G = (V, E)$.
Let $\Omega = \mathcal{P} \bigcup \mathcal{N}$.

From a matching $X_t \in \Omega$ the transition $X_t \to X_{t+1}$ is defined by:

1. Choose an edge $e = (v, y)$ uniformly at random from E.
2. *Remove:* If $e \in X_t$ and $X_t \in \mathcal{P}$ then set $X' = X_t \setminus \{e\}$.
3. *Add:* If v and y are unmatched in X_t then $X' = X_t \bigcup \{e\}$.
4. *Slide:* If v is unmatched and y is matched (or vice-versa):
    1. Let (y, z) denote the matched edge.
    2. Set $X' = X_t \bigcup (v, y) \setminus (y, z)$.
5. If X' is defined then:
   set $X_{t+1} = X'$ with probability $\min\{1, w(X')/w(X_t)\}$
6. Otherwise, set $X_{t+1} = X_t$.

Choice of Weights

Weight of matching $M \in \mathcal{P} \cup \mathcal{N}$ depends on unmatched vertices.

If $M \in \mathcal{P}$ then $w(M) = 1$.

Choice of Weights

Weight of matching $M \in \mathcal{P} \cup \mathcal{N}$ depends on unmatched vertices.

If $M \in \mathcal{P}$ then $w(M) = 1$.

Let $\mathcal{N}(u, v) = \{M \in \mathcal{N} : u \text{ and } v \text{ are unmatched}\}$.

If $M \in \mathcal{N}(u, v)$ then $w(M) = w(u, v)$ where:

$$w(u, v) = \frac{|\mathcal{P}|}{|\mathcal{N}(u, v)|}$$

Choice of Weights

Weight of matching $M \in \mathcal{P} \cup \mathcal{N}$ depends on unmatched vertices.

If $M \in \mathcal{P}$ then $w(M) = 1$.

Let $\mathcal{N}(u, v) = \{M \in \mathcal{N} : u \text{ and } v \text{ are unmatched}\}$.

If $M \in \mathcal{N}(u, v)$ then $w(M) = w(u, v)$ where:

$$w(u, v) = \frac{|\mathcal{P}|}{|\mathcal{N}(u, v)|}$$

$$\text{Note: } \sum_{P \in \mathcal{P}} w(P) = \sum_{N \in \mathcal{N}(u,v)} w(N) = |\mathcal{P}|$$

Hence: $\pi(\mathcal{P}) = \pi(\mathcal{N}(u, v)) = 1/(n^2 + 1)$.

Rapid Mixing

Key: for perfect matchings I, F, for $T = M \to M' \in \gamma_{I,F}$,

$$w(I)w(F) \geq w(M)w(\eta_T(I,F)).$$

Yields that Markov chain is rapidly mixing for these weights.

Rapid Mixing

Key: for perfect matchings I, F, for $T = M \to M' \in \gamma_{I,F}$,

$$w(I)w(F) \geq w(M)w(\eta_T(I,F)).$$

Yields that Markov chain is rapidly mixing for these weights.

Also, if use weights $w(u,v)$ where:

$$\frac{1}{2}\frac{|\mathcal{P}|}{|\mathcal{N}(u,v)|} \leq w(u,v) \leq 2\frac{|\mathcal{P}|}{|\mathcal{N}(u,v)|}$$

then mixing time slows by a factor of 16.

Rapid Mixing

Key: for perfect matchings I, F, for $T = M \to M' \in \gamma_{I,F}$,

$$w(I)w(F) \geq w(M)w(\eta_T(I,F)).$$

Yields that Markov chain is rapidly mixing for these weights.

Also, if use weights $w(u,v)$ where:

$$\frac{1}{2}\frac{|\mathcal{P}|}{|\mathcal{N}(u,v)|} \leq w(u,v) \leq 2\frac{|\mathcal{P}|}{|\mathcal{N}(u,v)|}$$

then mixing time slows by a factor of 16.

Key: Can correct slightly wrong weights:

If $w(u,v) = \alpha\frac{|\mathcal{P}|}{|\mathcal{N}(u,v)|}$ then $\pi(\mathcal{N}(u,v)) = \alpha\pi(\mathcal{P})$ so:

- Generate many samples from π, and then
- Correct the weights $w(u,v)$.

Simulated Annealing Approach

Input bipartite graph $G = (L \cup R, E)$ captured by:
complete bipartite $K_{n,n}$ with edge activities for $y \in L, z \in R$:

$$\lambda(y,z) = \begin{cases} \lambda & \text{if } (y,z) \notin E \\ 1 & \text{if } (y,z) \in E \end{cases}$$

Slowly go from $\lambda = 1$ to $\lambda \approx 0$.

Simulated Annealing Approach

Input bipartite graph $G = (L \cup R, E)$ captured by:
complete bipartite $K_{n,n}$ with edge activities for $y \in L, z \in R$:

$$\lambda(y,z) = \begin{cases} \lambda & \text{if } (y,z) \notin E \\ 1 & \text{if } (y,z) \in E \end{cases}$$

Slowly go from $\lambda = 1$ to $\lambda \approx 0$.

Matching M of $K_{n,n}$ has activity: $\lambda(M) = \prod_{(y,z)\in M} \lambda(y,z)$.

$$\text{Redefine } w(u,v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}.$$

Simulated Annealing Approach

Input bipartite graph $G = (L \cup R, E)$ captured by:
complete bipartite $K_{n,n}$ with edge activities for $y \in L, z \in R$:

$$\lambda(y,z) = \begin{cases} \lambda & \text{if } (y,z) \notin E \\ 1 & \text{if } (y,z) \in E \end{cases}$$

Slowly go from $\lambda = 1$ to $\lambda \approx 0$.

Matching M of $K_{n,n}$ has activity: $\lambda(M) = \prod_{(y,z)\in M} \lambda(y,z)$.

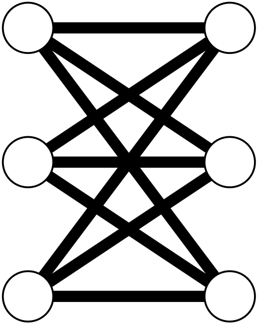
$$\text{Redefine } w(u,v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}.$$

Algorithm:
Start with $\lambda = 1$ and $w(u,v) = n$ for all $u \in L, v \in R$.
Repeat until $\lambda < 1/n!$:

1. Set $\lambda = (1 - \frac{1}{2n})\lambda$.
2. Generate $O(n^2 \log n)$ samples from π.
3. Correct the weights $w(u,v)$ for all u, v.

Simulated Annealing Algorithm

Illustration of the algorithm:

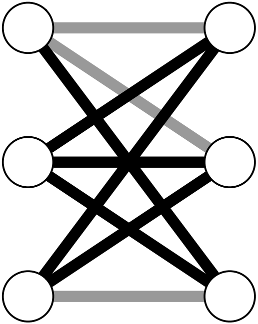


$$weights = \begin{bmatrix} 3 & 3 & 3 \\ 3 & 3 & 3 \\ 3 & 3 & 3 \end{bmatrix}$$

1. Start at the complete bipartite graph
2. Slowly remove non-edges:
   - Generate many samples from π, and
   - Recalibrate the weights $w(u, v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}$.

Simulated Annealing Algorithm

Illustration of the algorithm:

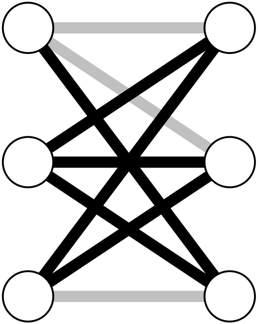


$$weights = \begin{bmatrix} 2.33 & 2.33 & 1.75 \\ 2.8 & 2.8 & 3.5 \\ 2.33 & 2.33 & 3.5 \end{bmatrix}$$

1. Start at the complete bipartite graph
2. Slowly remove non-edges:
   - Generate many samples from π, and
   - Recalibrate the weights $w(u, v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}$.

Simulated Annealing Algorithm

Illustration of the algorithm:

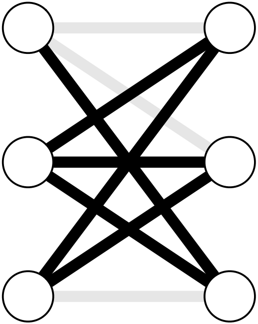


$$weights = \begin{bmatrix} 2.1 & 2.1 & 1.31 \\ 2.47 & 2.47 & 5.25 \\ 2.1 & 2.1 & 5.25 \end{bmatrix}$$

1. Start at the complete bipartite graph
2. Slowly remove non-edges:
   - Generate many samples from π, and
   - Recalibrate the weights $w(u,v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}$.

Simulated Annealing Algorithm

Illustration of the algorithm:

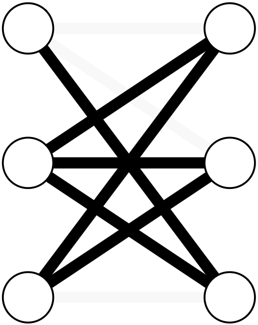


$$weights = \begin{bmatrix} 2.03 & 2.03 & 1.14 \\ 2.15 & 2.15 & 9.125 \\ 2.03 & 2.03 & 9.125 \end{bmatrix}$$

1. Start at the complete bipartite graph
2. Slowly remove non-edges:
   - Generate many samples from π, and
   - Recalibrate the weights $w(u,v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}$.

Simulated Annealing Algorithm

Illustration of the algorithm:



$$weights = \begin{bmatrix} 2.007 & 2.007 & 1.066 \\ 2.124 & 2.124 & 17.06 \\ 2.007 & 2.007 & 17.06 \end{bmatrix}$$

1. Start at the complete bipartite graph
2. Slowly remove non-edges:
    - Generate many samples from π, and
    - Recalibrate the weights $w(u,v) = \frac{\lambda(\mathcal{P})}{\lambda(\mathcal{N}(u,v))}$.

The end

Thank you!