A Polynomial Time Approximation Scheme for the Maximal Overlap of Two Independent Erdös-Rényi Graphs

Shuyang Gong

School of Mathematical Sciences, Peking University

Shandong University, November 2022

Based on a joint work with Jian Ding(PKU) and Hang Du(PKU)

Motivations

- **Random Graph Matching** is an extensively studied topic in recent years, which lies in the intersection of **probability, statistics and computer science**.

Motivations

- **Random Graph Matching** is an extensively studied topic in recent years, which lies in the intersection of **probability, statistics and computer science**.
- The motivations and prototypes of the problem come from various applied fields.

Mathematical Model

- Erdös-Rényi graph $G(n, p)$: Each edge in K_n is preserved with probability p independently.

Mathematical Model

- Erdös-Rényi graph $\mathrm{G}(n,p)$: Each edge in K_n is preserved with probability p independently.
- Sample two independent Erdös-Rényi graphs $\mathrm{G}_1(n,p)$ and $\mathrm{G}_2(n,p)$, where $p = n^{-\alpha}$.

Mathematical Model

- Erdös-Rényi graph $G(n, p)$: Each edge in K_n is preserved with probability p independently.
- Sample two independent Erdös-Rényi graphs $G_1(n, p)$ and $G_2(n, p)$, where $p = n^{-\alpha}$.
- Find a **bijection** π between the two vertex set such that the number of **common edges** under π is maximized. Let $C(\pi)$ be the number of common edges under π. Formally,

$$C(\pi) := \sum_{i,j=1}^{n} G^{(1)}_{i,j} G^{(2)}_{\pi(i),\pi(j)}.$$

where $G^{(i)}$ is the adjacency matrix for G_i.

Our problem

- Goal: Find π^* such that $\pi^* = \arg\max_\pi \mathsf{C}(\pi)$

Our problem

- Goal: Find π^* such that $\pi^* = \arg\max_\pi \mathsf{C}(\pi)$
- $\mathbb{P}$ the joint law of G_1 and G_2

Our problem

- Goal: Find π^* such that $\pi^* = \arg\max_\pi \mathsf{C}(\pi)$
- $\mathbb{P}$ the joint law of G_1 and G_2
- What is the typical behavior of $\mathsf{C}(\pi^*)$ under different α?

Our problem

- Goal: Find π^* such that $\pi^* = \arg\max_\pi \mathrm{C}(\pi)$
- $\mathbb{P}$ the joint law of G_1 and G_2
- What is the typical behavior of $\mathrm{C}(\pi^*)$ under different α?
- A union bound on $\mathrm{C}(\pi^*)$ yields an **upper bound**,

$$\mathbb{P}\left[\mathrm{C}(\pi^*) > \gamma(n)\right] \le n!\mathbb{P}[\mathbf{B} > \gamma(n)] \text{ where } \mathbf{B} \sim \mathbf{Bin}\left(\binom{n}{2}, p^2\right).$$

But is this $\gamma(n)$ the right asymptotic?

Upper bound estimation

- We focus on the regime $0 \leq \alpha \leq 1$ and $p = n^{-\alpha + o(1)}$.

Upper bound estimation

- We focus on the regime $0 \le \alpha \le 1$ and $p = n^{-\alpha+o(1)}$.
- For $0 < \alpha < 1/2$, then by a Chernoff bound

$$\mathbb{P}\left[\exists \pi, \text{ s.t. } \left|\frac{C(\pi)}{\binom{n}{2}p^2} - 1\right| > \varepsilon\right] = o(1).$$

which is highly concentrated.

Upper bound estimation

- We focus on the regime $0 \leq \alpha \leq 1$ and $p = n^{-\alpha + o(1)}$.
- For $0 < \alpha < 1/2$, then by a Chernoff bound

$n! \exp\left(-n^{2-2\alpha}\right)$

$$\mathbb{P}\left[\exists \pi, \text{ s.t. } \left|\frac{C(\pi)}{\binom{n}{2}p^2} - 1\right| > \varepsilon\right] = o(1).$$

which is highly concentrated.

- For $1/2 < \alpha \leq 1$,

$$\mathbb{P}\left[C(\pi^*) > \frac{1+\varepsilon}{2\alpha - 1} n\right] = o(1).$$

Upper bound estimation

- We focus on the regime $0 \le \alpha \le 1$ and $p = n^{-\alpha+o(1)}$.
- For $0 < \alpha < 1/2$, then by a Chernoff bound

$$\mathbb{P}\left[\exists\pi, \text{ s.t. } \left|\frac{C(\pi)}{\binom{n}{2}p^2} - 1\right| > \varepsilon\right] = o(1).$$

  which is highly concentrated.
- For $1/2 < \alpha \le 1$,

$$\mathbb{P}\left[C(\pi^*) > \frac{1+\varepsilon}{2\alpha - 1}n\right] = o(1).$$

- For the critical case $\alpha = 1/2$, things become more subtle.

A PTAS for the matching problem

- We focus on the case $1/2 < \alpha \leq 1$ and show that the bound $\frac{1}{2\alpha-1}n$ is indeed tight.

A PTAS for the matching problem

- We focus on the case $1/2 < \alpha \leq 1$ and show that the bound $\frac{1}{2\alpha-1}n$ is indeed tight.
- What we show is further: this random optimization problem enjoys a **polynomial-time approximation scheme**(PTAS).

Theorem (Ding-Du-G.' 22+)

For any constant $\epsilon > 0$, there exists $C = C(\epsilon)$ together with an $O(n^C)$-time algorithm, which takes G_1, G_2 as input and outputs some π^*, such that

$$\mathbb{P}\left[\mathsf{C}(\pi^*) \geq \frac{1-\epsilon}{2\alpha-1}n\right] = 1 - o(1).$$

A PTAS for the matching problem

- We focus on the case $1/2 < \alpha \leq 1$ and show that the bound $\frac{1}{2\alpha-1}n$ is indeed tight.
- What we show is further: this random optimization problem enjoys a **polynomial-time approximation scheme**(PTAS).

Theorem (Ding-Du-G.' 22+)

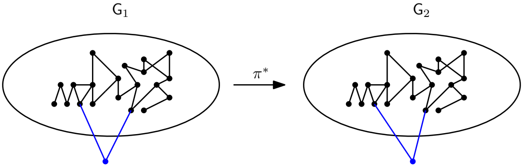
For any constant $\epsilon > 0$, there exists $C = C(\epsilon)$ together with an $O(n^C)$-time algorithm, which takes G_1, G_2 as input and outputs some π^*, such that

$$\mathbb{P}\left[\mathsf{C}(\pi^*) \geq \frac{1-\epsilon}{2\alpha-1}n\right] = 1 - o(1).$$

- This proves that the asymptotic for $\mathsf{C}(\pi^*)$ is typically $\frac{n}{2\alpha-1}$.

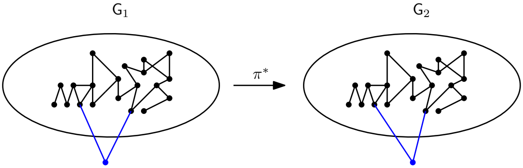
Heuristic Idea I

- Consider the case $\alpha = 3/4 - \delta$ for some small δ, under which $\frac{1}{2\alpha-1}n = \frac{2}{1-4\delta}n \approx 2n$.



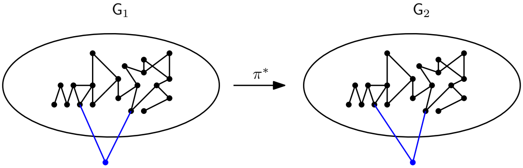
Heuristic Idea I

- Consider the case $\alpha = 3/4 - \delta$ for some small δ, under which $\frac{1}{2\alpha-1}n = \frac{2}{1-4\delta}n \approx 2n$.
- Match the first $\varepsilon n/3$ vertices arbitrarily.



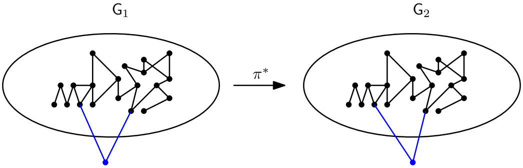
Heuristic Idea I

- Consider the case $\alpha = 3/4 - \delta$ for some small δ, under which $\frac{1}{2\alpha-1}n = \frac{2}{1-4\delta}n \approx 2n$.
- Match the first $\varepsilon n/3$ vertices arbitrarily.
- In each step follows, try to match one vertex and two edges.



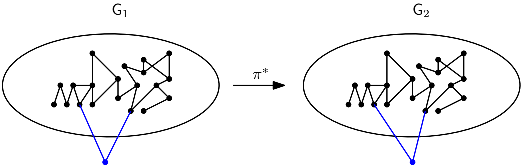
Heuristic Idea I

- Consider the case $\alpha = 3/4 - \delta$ for some small δ, under which $\frac{1}{2\alpha-1}n = \frac{2}{1-4\delta}n \approx 2n$.
- Match the first $\varepsilon n/3$ vertices arbitrarily.
- In each step follows, try to match one vertex and two edges.
- Suppose we have matched $k-1$ steps, for step k, we find a unmatched vertex u_k in G_1(w.r.t. some $\prec$). See the neighbors of u_k in M_k, where M_k is the matched area.



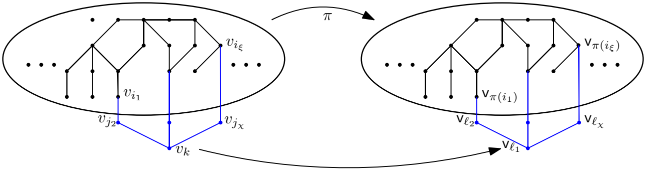
Heuristic Idea I

- Consider the case $\alpha = 3/4 - \delta$ for some small δ, under which $\frac{1}{2\alpha-1}n = \frac{2}{1-4\delta}n \approx 2n$.
- Match the first $\varepsilon n/3$ vertices arbitrarily.
- In each step follows, try to match one vertex and two edges.
- Suppose we have matched $k-1$ steps, for step k, we find a unmatched vertex u_k in G_1(w.r.t. some $\prec$). See the neighbors of u_k in M_k, where M_k is the matched area.
- For each pair (u, u') in $N_{u_k}(M_k)$, we try to find a $v \in \mathsf{G}_2$ s.t. $(\pi^*(u), v)$, $(\pi^*(u'), v) \in \mathsf{G}_2$.



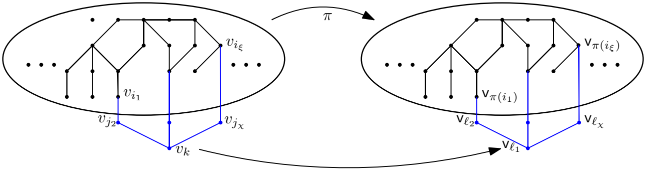
Heuristic idea II

- What if $\frac{1}{2\alpha-1}$ is not an integer ?



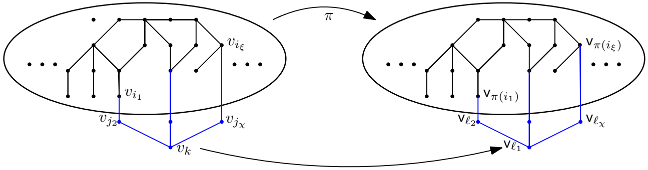
Heuristic idea II

- What if $\frac{1}{2\alpha-1}$ is not an integer ?
- Consider the case $\alpha = \frac{5}{6} - \delta$ which means in each step we add 4 and 6 vertices, i.e. we add a tree **T**.



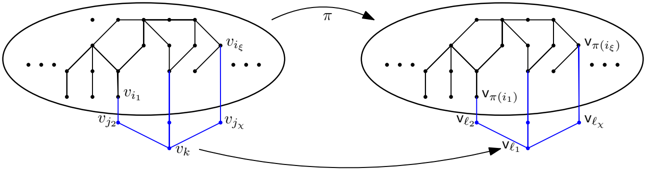
Heuristic idea II

- What if $\frac{1}{2\alpha-1}$ is not an integer ?
- Consider the case $\alpha = \frac{5}{6} - \delta$ which means in each step we add 4 and 6 vertices, i.e. we add a tree **T**.
- In each step, we find a vertex v_k in G_1. Find all the 3-tuples in M_k which forms a tree **T** with root v_k.



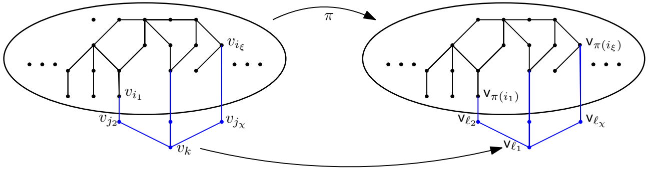
Heuristic idea II

- What if $\frac{1}{2\alpha-1}$ is not an integer ?
- Consider the case $\alpha = \frac{5}{6} - \delta$ which means in each step we add 4 and 6 vertices, i.e. we add a tree **T**.
- In each step, we find a vertex v_k in G_1. Find all the 3-tuples in M_k which forms a tree **T** with root v_k.
- Map these tuples to G_2 by π^*, for each tuple, we check if there is a $v \in G_2$ such that there is a tree **T** in G_2 with root v and leaves being the tuple.



Heuristic idea II

- What if $\frac{1}{2\alpha-1}$ is not an integer ?
- Consider the case $\alpha = \frac{5}{6} - \delta$ which means in each step we add 4 and 6 vertices, i.e. we add a tree **T**.
- In each step, we find a vertex v_k in G_1. Find all the 3-tuples in M_k which forms a tree **T** with root v_k.
- Map these tuples to G_2 by π^*, for each tuple, we check if there is a $v \in G_2$ such that there is a tree **T** in G_2 with root v and leaves being the tuple.
- If success, we add this tree to M_k(blue).



The General Case-The tree structure

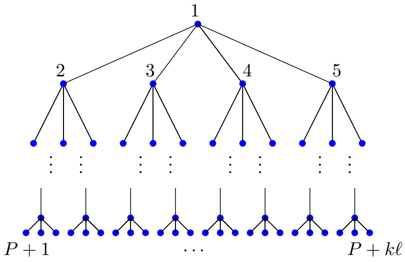
- $\frac{1}{2\alpha-1} \approx \frac{Q}{P}$, which means in each step, we need to add P vertices and Q edges.

The General Case-The tree structure

- $\frac{1}{2\alpha-1} \approx \frac{Q}{P}$, which means in each step, we need to add P vertices and Q edges.
- To do this, we need to **carefully design** a tree $\mathbf{T}$ with P non-leaf nodes and Q edges. The tree satisfies some "balanced" conditions for technical reasons.

The General Case-The tree structure

- $\frac{1}{2\alpha-1} \approx \frac{Q}{P}$, which means in each step, we need to add P vertices and Q edges.
- To do this, we need to **carefully design** a tree **T** with P non-leaf nodes and Q edges. The tree satisfies some "balanced" conditions for technical reasons.
- The structure of the tree **T**,



The General Case-Algorithm

- In each step, we find a vertex v_k in G_1. Find all the $(Q+1-P)$-tuples L in M_k such that there exists a tree **T** with root v_k and leaves L in G_1.

The General Case-Algorithm

- In each step, we find a vertex v_k in G_1. Find all the $(Q+1-P)$-tuples L in M_k such that there exists a tree **T** with root v_k and leaves L in G_1.
- Denote the set of such tuples by CAND_k and we assign an uniform order for CAND_k.

The General Case-Algorithm

- In each step, we find a vertex v_k in G_1. Find all the $(Q+1-P)$-tuples L in M_k such that there exists a tree **T** with root v_k and leaves L in G_1.
- Denote the set of such tuples by CAND_k and we assign an uniform order for CAND_k.
- Check the tuples in CAND_k according to this order.

The General Case-Algorithm

- In each step, we find a vertex v_k in G_1. Find all the $(Q+1-P)$-tuples L in M_k such that there exists a tree **T** with root v_k and leaves L in G_1.
- Denote the set of such tuples by CAND_k and we assign an uniform order for CAND_k.
- Check the tuples in CAND_k according to this order.
- For each tuple L, if there is a tree of structure **T** with leaves being $\pi^*(\mathrm{L})$ in G_2, we add this tree into M_k and go to the next step, otherwise, we check the next tuple.

The General Case-Algorithm

- In each step, we find a vertex v_k in G_1. Find all the $(Q+1-P)$-tuples L in M_k such that there exists a tree **T** with root v_k and leaves L in G_1.
- Denote the set of such tuples by CAND_k and we assign an uniform order for CAND_k.
- Check the tuples in CAND_k according to this order.
- For each tuple L, if there is a tree of structure **T** with leaves being $\pi^*(\mathrm{L})$ in G_2, we add this tree into M_k and go to the next step, otherwise, we check the next tuple.
- Our goal is to show that in most$(n-o(n))$ steps, we can find a tuple in CAND_k which matches successfully.

The General Case-Proof

- A key ingredient is the **second moment method**, i.e.

$$\mathbb{P}[X > 0] \geq \frac{(\mathbb{E}[X])^2}{\mathbb{E}[X^2]} \text{ for every integer valued } X.$$

The General Case-Proof

- A key ingredient is the **second moment method**, i.e.

$$\mathbb{P}[X > 0] \geq \frac{(\mathbb{E}[X])^2}{\mathbb{E}[X^2]} \text{ for every integer valued } X.$$

- What X shall we take?

The General Case-Proof

- A key ingredient is the **second moment method**, i.e.

$$\mathbb{P}[X > 0] \geq \frac{(\mathbb{E}[X])^2}{\mathbb{E}[X^2]} \text{ for every integer valued } X.$$

- What X shall we take?
- Let

$$X_k = \sum_{\mathsf{L} \in \mathsf{CAND}_k} \mathbf{1}_{\{\pi^*(\mathsf{L}) \bowtie \mathsf{R}_k\}}.$$

The General Case-Proof

- A key ingredient is the **second moment method**, i.e.

$$\mathbb{P}[X > 0] \geq \frac{(\mathbb{E}[X])^2}{\mathbb{E}[X^2]} \text{ for every integer valued } X.$$

- What X shall we take?
- Let

$$X_k = \sum_{\mathsf{L} \in \mathsf{CAND}_k} \mathbf{1}_{\{\pi^*(\mathsf{L}) \bowtie \mathsf{R}_k\}}.$$

- We want to show that $\mathbb{P}[X_k > 0 | \mathcal{F}_{k-0.5}] \geq 1 - o(1)$ using second moment method.

The General Case-Proof

- A key ingredient is the **second moment method**, i.e.

$$\mathbb{P}[X > 0] \geq \frac{(\mathbb{E}[X])^2}{\mathbb{E}[X^2]} \text{ for every integer valued } X.$$

- What X shall we take?
- Let

$$X_k = \sum_{\mathsf{L} \in \mathsf{CAND}_k} \mathbf{1}_{\{\pi^*(\mathsf{L}) \bowtie \mathsf{R}_k\}}.$$

- We want to show that $\mathbb{P}[X_k > 0 | \mathcal{F}_{k-0.5}] \geq 1 - o(1)$ using second moment method.
- We need to compute the first moment and the second moment separately.

The General Case-Proof

- A key ingredient is the **second moment method**, i.e.

$$\mathbb{P}[X > 0] \geq \frac{(\mathbb{E}[X])^2}{\mathbb{E}[X^2]} \text{ for every integer valued } X.$$

- What X shall we take?
- Let

$$X_k = \sum_{\mathsf{L} \in \mathsf{CAND}_k} \mathbf{1}_{\{\pi^*(\mathsf{L}) \bowtie \mathsf{R}_k\}}.$$

- We want to show that $\mathbb{P}[X_k > 0 | \mathcal{F}_{k-0.5}] \geq 1 - o(1)$ using second moment method.
- We need to compute the first moment and the second moment separately.
- It's fair to say most of our work is to compute the moments, but it is not as obvious as it may seem.

The General Case-Proof

- If the previous conditioning(positive and negative) does not have devastating effects, then our idea works. Given this,

Proposition

For $1 \leq k \leq (1 - \varepsilon/3)n$, *for any "good" realization* $\mathcal{F}_{k-0.5}$ *and any tuple in* CAND_k, *it holds uniformly for all* k *that,*

$$(1 - o(1))\mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k \mid \mathcal{F}_{k-0.5}] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k].$$

The General Case-Proof

- If the previous conditioning(positive and negative) does not have devastating effects, then our idea works. Given this,

Proposition

For $1 \leq k \leq (1-\varepsilon/3)n$, for any "good" realization $\mathcal{F}_{k-0.5}$ and any tuple in CAND_k, it holds uniformly for all k that,

$$(1-o(1))\mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k \mid \mathcal{F}_{k-0.5}] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k].$$

- The first moment $\mathbb{E}[X_k|\mathcal{F}_{k-0.5}] \to \infty$.

The General Case-Proof

- If the previous conditioning(positive and negative) does not have devastating effects, then our idea works. Given this,

Proposition

For $1 \leq k \leq (1-\varepsilon/3)n$*, for any "good" realization* $\mathcal{F}_{k-0.5}$ *and any tuple in* CAND_k*, it holds uniformly for all* k *that,*

$$(1-o(1))\mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k \mid \mathcal{F}_{k-0.5}] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k].$$

- The first moment $\mathbb{E}[X_k|\mathcal{F}_{k-0.5}] \to \infty$.
- It then remains for us to show that,

$$\mathbb{E}[X_k^2|\mathcal{F}_{k-0.5}] \leq (1+o(1))\left(\mathbb{E}[X_k|\mathcal{F}_{k-0.5}]\right)^2.$$

The General Case-Proof

- If the previous conditioning(positive and negative) does not have devastating effects, then our idea works. Given this,

Proposition

For $1 \leq k \leq (1-\varepsilon/3)n$, for any "good" realization $\mathcal{F}_{k-0.5}$ and any tuple in CAND_k, it holds uniformly for all k that,

$$(1-o(1))\mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k \mid \mathcal{F}_{k-0.5}] \leq \mathbb{P}[\pi^*(\mathrm{L}) \bowtie \mathrm{R}_k].$$

- The first moment $\mathbb{E}[X_k|\mathcal{F}_{k-0.5}] \to \infty$.
- It then remains for us to show that,

$$\mathbb{E}[X_k^2|\mathcal{F}_{k-0.5}] \leq (1+o(1))\left(\mathbb{E}[X_k|\mathcal{F}_{k-0.5}]\right)^2.$$

- The claim $\mathbb{P}[X_k > 0|\mathcal{F}_{k-0.5}] \geq 1-o(1)$ follows.

A key input-Extensions

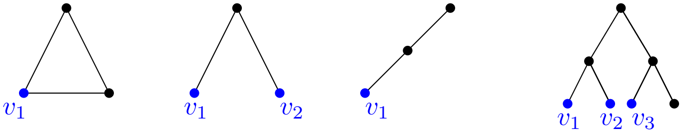
- Now, we introduce a notion which is closely related to our proof, namely, the extensions.

A key input-Extensions

- Now, we introduce a notion which is closely related to our proof, namely, the extensions.
- Given k vertexes in an Erdös-Rényi Graph, denoted by $v_1, v_2, \cdots, v_k$.

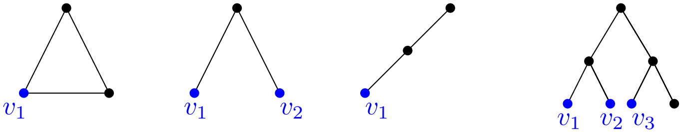
A key input-Extensions

- Now, we introduce a notion which is closely related to our proof, namely, the extensions.
- Given k vertexes in an Erdös-Rényi Graph, denoted by $v_1, v_2, \cdots, v_k$.
- We want to know some specific extension using the set of vertices in $\{v_1, v_2, \cdots, v_k\}$ and the position of this set of vertices in extension is fixed.



A key input-Extensions

- Now, we introduce a notion which is closely related to our proof, namely, the extensions.
- Given k vertexes in an Erdös-Rényi Graph, denoted by $v_1, v_2, \cdots, v_k$.
- We want to know some specific extension using the set of vertices in $\{v_1, v_2, \cdots, v_k\}$ and the position of this set of vertices in extension is fixed.



- The expectation may not be able to reflect the actual enumerations. It depends on the structure. e.g. For the second structure above, we have,

$$\max_{v_1, v_2} \mathrm{Ext}(v_1, v_2, \mathrm{Structure}) = O(1) \neq np^2 = o(1)$$

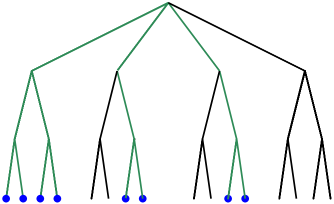
while for the third structure the expectation reflects the structures.

A key input-Tree extensions

- We now focus on counting the extensions of a tree-like structure **T** given some leaves $v_1, v_2, \cdots, v_k$.

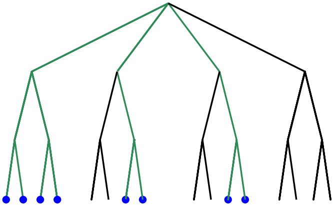
A key input-Tree extensions

- We now focus on counting the extensions of a tree-like structure **T** given some leaves $v_1, v_2, \cdots, v_k$.
- We take a tree for example to give you a flavor of this method.



A key input-Tree extensions

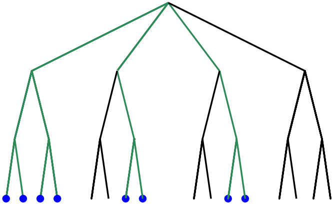
- We now focus on counting the extensions of a tree-like structure **T** given some leaves $v_1, v_2, \cdots, v_k$.
- We take a tree for example to give you a flavor of this method.



- We first count the green tree spanned by the fixed vertices. To compute this enumeration, we **decompose** the tree(green) into dense parts and sparse parts.

A key input-Tree extensions

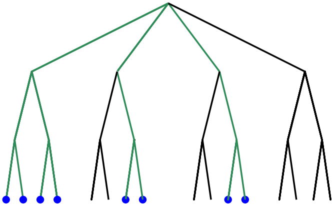
- We now focus on counting the extensions of a tree-like structure **T** given some leaves $v_1, v_2, \cdots, v_k$.
- We take a tree for example to give you a flavor of this method.



- We first count the green tree spanned by the fixed vertices. To compute this enumeration, we **decompose** the tree(green) into dense parts and sparse parts.
- We then count the enumeration of the black parts according to its expectation.

A key input-Tree extensions

- We now focus on counting the extensions of a tree-like structure **T** given some leaves $v_1, v_2, \cdots, v_k$.
- We take a tree for example to give you a flavor of this method.



- We first count the green tree spanned by the fixed vertices. To compute this enumeration, we **decompose** the tree(green) into dense parts and sparse parts.
- We then count the enumeration of the black parts according to its expectation.
- This finishes the counting of tree extension. To learn more about the extensions(of other shapes), see Spencer's paper in the reference.

Some discussions

- Our result gives an **algorithmic lower bound** for this problem, which matches the theoretical upper bound. Before this, the lower bound is unknown.

Some discussions

- Our result gives an **algorithmic lower bound** for this problem, which matches the theoretical upper bound. Before this, the lower bound is unknown.
- Attempts for establishing this through **second moment method** have been made by Ding, Wu and Xu. And it is fair to say that they have been very close to the answer.

Some discussions

- Our result gives an **algorithmic lower bound** for this problem, which matches the theoretical upper bound. Before this, the lower bound is unknown.
- Attempts for establishing this through **second moment method** have been made by Ding, Wu and Xu. And it is fair to say that they have been very close to the answer.
- Based on the belief of the **information computation gap**, our original goal is to establish some negative results on the performance of efficient algorithm. But later things turned out to evolve in an unexpected way.

Some discussions

- Our result gives an **algorithmic lower bound** for this problem, which matches the theoretical upper bound. Before this, the lower bound is unknown.
- Attempts for establishing this through **second moment method** have been made by Ding, Wu and Xu. And it is fair to say that they have been very close to the answer.
- Based on the belief of the **information computation gap**, our original goal is to establish some negative results on the performance of efficient algorithm. But later things turned out to evolve in an unexpected way.
- Our result contributes an example for which approximation algorithms were discovered for random instances whereas the worst-case of the problem is known as NP-hard.

Future works

- For the correlated graph model, prove or fix the (still quite large) information-computation gap for the maximal overlap.

Future works

- For the correlated graph model, prove or fix the (still quite large) information-computation gap for the maximal overlap.
- For the matching problem between independent graphs, is there a polynomial-time algorithm with fixed power that finds (near) optimal matchings?

Future works

- For the correlated graph model, prove or fix the (still quite large) information-computation gap for the maximal overlap.
- For the matching problem between independent graphs, is there a polynomial-time algorithm with fixed power that finds (near) optimal matchings?
- For the case $\alpha = 1/2$, determine the asymptotic of the maximal overlap.(in progress with Hang Du and Rundong Huang).

Future works

- For the correlated graph model, prove or fix the (still quite large) information-computation gap for the maximal overlap.
- For the matching problem between independent graphs, is there a polynomial-time algorithm with fixed power that finds (near) optimal matchings?
- For the case $\alpha = 1/2$, determine the asymptotic of the maximal overlap.(in progress with Hang Du and Rundong Huang).
- For the case $0 < \alpha < 1/2$, determine the second order of the maximal overlap.

References

[S90] J. Spencer. Counting extensions. J. Combin. Theory Ser. A, 55(2):247-255, 1990.
[PRW94] Panos M. Pardalos, Franz Rendl, and Henry Wolkowicz. The quadratic assignment problem: A survey and recent developments.
[BCPP98] Rainer E Burkard, Eranda Cela, Panos M Pardalos, and Leonidas S Pitsoulis. The quadratic assignment problem. In handbook of combinatorial optimization, pages 1713-1809. Springer, 1998.
[DDG22] Jian Ding, Hang Du and Shuyang Gong, A Polynomial-time Approximation Scheme for the Maximal Overlap Between Two Independent Erdös-Rényi Graphs, Arxiv:2210.07823.

Thanks!