Seeded Graph Matching via Large Neighborhood Statistics *

Elchanan Mossel† Jiaming Xu‡

October 31, 2018

Abstract

We study a well known noisy model of the graph isomorphism problem. In this model, the goal is to perfectly recover the vertex correspondence between two edge-correlated graphs, with an initial seed set of correctly matched vertex pairs revealed as side information. Specifically, the model first generates a parent graph G_0 from Erdős-Rényi random graph $\mathcal{G}(n,p)$ and then obtains two children graphs G_1 and G_2 by subsampling the edge set of G_0 twice independently with probability $s=\Theta(1)$. The vertex correspondence between G_1 and G_2 is obscured by randomly permuting the vertex labels of G_1 according to a latent permutation π^*. Finally, for each i, $\pi^*(i)$ is revealed independently with probability α as seeds.

In the sparse graph regime where $np \le n^\epsilon$ for any $\epsilon < 1/6$, we give a polynomial-time algorithm which perfectly recovers π^*, provided that $nps^2 - \log n \to +\infty$ and $\alpha \ge n^{-1+3\epsilon}$. This further leads to a sub-exponential-time, $\exp\left(n^{O(\epsilon)}\right)$, matching algorithm even without seeds. On the contrary, if $nps^2 - \log n = O(1)$, then perfect recovery is information-theoretically impossible as long as α is bounded away from 1.

In the dense graph regime, where $np = bn^a$, for fixed constants $a, b \in (0,1]$, we give a polynomial-time algorithm which succeeds when $b = O(s)$ and $\alpha = \Omega\left((np)^{-\lfloor 1/a \rfloor}\log n\right)$. In particular, when $a = 1/k$ for an integer $k \ge 1$, $\alpha = \Omega(\log n/n)$ suffices, yielding a quasi-polynomial-time $n^{O(\log n)}$ algorithm matching the best known algorithm by Barak et al. for the problem of graph matching without seeds when $k \ge 153$ and extending their result to new values of p for $k = 2, \ldots, 152$.

Unlike previous work on graph matching, which used small neighborhoods or small subgraphs with a logarithmic number of vertices in order to match vertices, our algorithms match vertices if their large neighborhoods have a significant overlap in the number of seeds.

*Supported by NSF grant CCF-1755960.

†Massachusetts Institute of Technology.

‡Duke University.

1 Introduction

In this paper, we study a well-known model of noisy graph isomorphism. Our main interest is in polynomial time algorithms for seeded problems where the matching between a small subset of the nodes is revealed. For seeded problems, our result provides a dramatic improvement over previously known results. Our results also shed light on the unseeded problem. In particular, we give (the first) sub-exponential time algorithms for sparse models and an $n^{O(\log n)}$ algorithm for dense models for some parameters, including some that are not covered by recent results of Barak et al. [BCL+18].

We recall that two graphs are isomorphic if there exists an edge-preserving bijection between their vertex sets. The Graph Isomorphism problem is not known to be solvable in polynomial time, except in special cases such as graphs of bounded degree [Luk80] and bounded eigenvalue multiplicity [BGM82]. However, a recent breakthrough of Babai [Bab16] gave a quasi-polynomial time algorithm.

In a number of applications including network security, systems biology, computer vision, and natural language processing, we are given two graphs as input which we believe have an underlying isomorphism between them. However, they are not exactly isomorphic because they have each been perturbed in some way, adding or deleting edges randomly. This suggests a noisy version of Graph Isomorphism also known as *graph matching* [CFSV04, LR13], where we seek a bijection that minimizes the number of edge disagreements.

Given two graphs with adjacency matrices G_1 and G_2, if our goal is to minimize the ℓ_2 distance between G_1 and some permuted version of G_2, then graph matching can be viewed as a special case of the *quadratic assignment problem* (QAP) [BCPP98]: namely,

$$\min_{\Pi} \|G_1 - \Pi G_2 \Pi^\top\|_F^2, \tag{1.1}$$

where Π ranges over all $n \times n$ permutation matrices, and $\|A\|_F^2 = \sum_{ij} A_{ij}^2$ denotes the Frobenius norm. QAP is NP-hard in the worst case. There are exact search methods for QAP based on branch-and-bound and cutting planes, as well as various approximation algorithms based on linearization schemes, and convex/semidefinite

programming relaxations (see [FQRM+16] and the references therein). However, approximating QAP within a factor $2^{\log^{1-\epsilon}(n)}$ for $\epsilon > 0$ is NP-hard [MMS10].

These hardness results only apply in the worst case, where the two graphs are designed by an adversary. However, in many aforementioned applications, we are not interested in worst-case instances, but rather in instances for which there is enough information in the data to recover the underlying isomorphism, i.e., when the amount of data or signal-to-noise ratio is above the information-theoretic limit. The key question is whether there exists an efficient algorithm that is successful all the way down to this limit. In this vein, we consider the following random graph model denoted by $\mathcal{G}(n, p; s)$ [PG11].

DEFINITION 1. (CORRELATED ERDŐS-RÉNYI MODEL) *Suppose we generate a parent graph G_0 from the Erdős-Rényi random graph model $\mathcal{G}(n, p)$. For a fixed realization of G_0, we generate two subgraphs G_1 and G_2 by subsampling the edges of G_0 twice. More specifically,*

- *We let G_1^* be a random subgraph of G_0 obtained by including every edge of G_0 with probability s independently.*
- *We repeat the above subsampling procedure, but independently to obtain another random subgraph of G_0, denoted by G_2.*

To further model the scenario that we do not know the vertex correspondence between G_1 and G_2 a prior, we sample a random permutation π^* over $[n]$ and let G_1 denote the graph obtained by relabeling every vertex i in G_1^* as $\pi^*(i)$.

The goal is to exactly recover π^* from the observation of G_1 and G_2 with high probability, i.e., to design an estimator $\widehat{\pi}$ based on G_1 and G_2 such that

$$\mathbb{P}\{\widehat{\pi}(G_1, G_2) = \pi^*\} \to 1, \quad \text{as } n \to \infty.$$

As a motivating example, we can model G_0 as some true underlying friendship network of n persons, G_1 is an anonymized Facebook network of the same set of persons, and G_2 is a Twitter network with known person identities. If we can recover the vertex correspondence between G_1 and G_2, then we can de-anonymize the Facebook network G_1 (this example ignores many important facts such as additional graph structures in real life networks).

Note that s is equal to the probability of $e \in E(G_2)$ conditional on $e \in E(G_1)$, and hence can be viewed as a measure of the edge correlations. Throughout this paper, without further specifications, we shall assume $s = \Theta(1)$.

In the fully sampling case $s = 1$, graph matching under $\mathcal{G}(n, p; 1)$ reduces to the Graph Automorphism problem for Erdős-Rényi graphs. In this case, a celebrated result [Wri71] shows that if $\log n + \omega(1) \le np \le n - \log n - \omega(1)$, then with probability $1 - o(1)$, the size of the automorphism group of G_0 is 1 and hence the underlying permutation π^* can be exactly recovered; otherwise, with probability $1 - o(1)$, the size of the automorphism group of G_0 is strictly bigger than 1 and hence exact recovery of the underlying permutation is information-theoretically impossible. Recent work [CK16, CK17][1] has extended this result to the partially sampling case $s = \Theta(1)$ and $p \le 1/2$, showing that the Maximum Likelihood Estimator, or equivalently the optimum of QAP (1.1), coincides with the ground truth π^* with high probability, provided that $nps^2 \ge \log n + \omega(1)$; on the contrary, any estimator is correct with probability at most $o(1)$, if $nps^2 \le \log n - \omega(1)$.

From a computational perspective, in the fully sampling case $s = 1$, there exist linear-time algorithms which attain the recovery threshold, in the sense that they exactly recover the underlying permutation with high probability whenever $np = \log n + \omega(1)$ [Bol82, CP08]. However, in the partially sampling case, it is still open whether any efficient algorithm can succeed close to the threshold. A recent breakthrough result [BCL+18] obtains a quasi-polynomial-time ($n^{O(\log n)}$) algorithm which succeeds when $np \ge n^{o(1)}$ and $s \ge (\log n)^{-o(1)}$. However, this is still far away from the information-theoretic limit $nps^2 \ge \log n + \omega(1)$.

Another line of work [PG11, YG13, KL14, KHG15, LFP13, FAP18, SGE17] in this area considers a relaxed version of the graph matching problem, where an initial seed set of correctly matched vertex pairs is revealed as side information. This is motivated by the fact that in many real applications, some side information on the vertex identities are available and have been successfully utilized to match many real-world networks [NS09, NS08]. Formally, in this paper, we assume the seed set is randomly generated as follows.

DEFINITION 2. (SEEDED GRAPH MATCHING) *In addition to G_1, G_2 that are generated under $\mathcal{G}(n, p; s)$ with a latent permutation π^*, we have access to π_0 such that $\pi_0(i) = \pi^*(i)$ with probability α and $\pi_0(i) =?$ with probability $1 - \alpha$ independently across different i. The goal is to recover π^* based on G_1, G_2, and π_0.*

The vertex i such that $\pi_0(i) = \pi^*(i)$ is called seeded vertices and the set of seed vertices is denoted by I_0. Note that according to our model, the number of seeds

[1] In fact, a more general correlated Erdős-Rényi random graph model is considered in [CK16, CK17], where $\mathbb{P}\{G_1(i, j) = a, G_2(i, j) = b\} = p_{a,b}$ for $a, b \in \{0, 1\}$.

$|I_0|$ is distributed as $\text{Binom}(n, \alpha)$. For a given size K, we could also consider a deterministic size model where I_0 is chosen uniformly at random from all possible subsets of $[n]$ with size K. The main results of this paper readily extend to this deterministic size model with $K = \lfloor n\alpha \rfloor$.

The results of the seeded graph matching turn out to be useful for designing graph matching without seeds. On the one hand, when a seed set of size K is not given, we could obtain it in $n^{O(K)}$ steps by randomly choosing a set of K vertices and then enumerating all the possible mapping. This is known as the beacon set approach to graph isomorphism [Lip78]. On the other hand, we could first apply a seedless graph matching algorithm and then apply a seeded graph matching algorithm to boost its accuracy. This two-step algorithms have been successful both theoretically [BES80] [Bol01, Section 3.5] and empirically [LFP13].

In the sparse graph regime $np = \Theta(\log n)$, it is shown in [YG13] that if $\alpha = \Omega(1/\log^{4/3} n)$, or equivalently, the size of the seed set is $\Omega(n/\log^{4/3} n)$, then a percolation-based graph matching algorithm correctly matches $n - o(n)$ vertices in polynomial-time with high probability. In the dense graph regime $np = n^{\delta}$ for some constant $\delta \in (0, 1)$, a seed set of size $\Theta(n^{1-\delta})$ suffices as shown in [YG13]. Another work [KL14] shows that if $nps^2\alpha \geq 24 \log n$, then one can match all vertices correctly in polynomial-time with high probability based on counting the number of "common" seeded vertices. Note that this exact recovery result requires the seed set size to be linear in n in the sparse graph regime $np = \Theta(\log n)$.

In summary, despite a significant amount of previous work on seedless and seeded graph matching, the following two fundamental questions remain elusive:

QUESTION 1. *In terms of graph sparsity, can we achieve the information-theoretic limit $nps^2 - \log n \to +\infty$ in sub-exponential, or polynomial time?*

QUESTION 2. *In terms of seed set, what is the minimum number of seeds required for exact recovery in sub-exponential, or polynomial time?*

Our main results shed light on these two questions by improving the state-of-the-art of seeded graph matching. First, we show that it is possible to achieve the information theoretic limit $nps^2 \geq \log n + \omega(1)$ of graph sparsity in polynomial-time. Then, we show the number of seeds needed for exact recovery in polynomial-time can be as low as n^{ϵ} in the sparse graph regime ($np \leq n^{\epsilon}$) and $\Omega(\log n)$ in the dense graph regime.

1.1 Main Results

We first consider the sparse graph regime.

THEOREM 1.1. *Suppose $np \leq n^{1/2-\epsilon}$ for a fixed constant $\epsilon > 0$ and $s = \Theta(1)$. Assume*

$$nps^2 - \log n \to +\infty \tag{1.2}$$

$$\alpha \geq n^{-1/2+3\epsilon}. \tag{1.3}$$

Then there exists a polynomial-time algorithm, namely Algorithm 1, which outputs $\widehat{\pi} = \pi^*$ with probability at least $1 - o(1)$ under the seeded $\mathcal{G}(n, p; s, \alpha)$ model.

Notice that (1.2) is the information-theoretic limit for graph matching under the seedless $\mathcal{G}(n, p; s)$ model. In fact, Theorem 1.2 shows that (1.2) is necessary for seeded graph matching as long as α is bounded away from 1. Its proof is standard and can be found in the full paper [MX18].

THEOREM 1.2. *If*

$$nps^2 - \log n = O(1),$$

then any algorithm outputs $\widehat{\pi} \neq \pi^*$ with at least a probability of $\Omega\left((1-\alpha)^2\right)$ under the seeded $\mathcal{G}(n, p; s, \alpha)$ model.

Also, the condition (1.3) requires that the size of the seed set is $n^{1/2+3\epsilon}$ compared to the best previously known results that required the seed set to be almost linear in n.

It is natural to ask if $n^{1/2}$ seeded nodes are required for polynomial time algorithm. While from the proof of Theorem 1.1, it might look that $n^{1/2}$ is optimal due to the birthday paradox effect, it turns out we can do better!

The following result relaxes the size of seed set needed to $n^{3\epsilon}$ when the average degree is smaller.

THEOREM 1.3. *Suppose $np \leq n^{\epsilon}$ for a fixed constant $\epsilon < 1/6$ and $s = \Theta(1)$. Assume*

$$nps^2 - \log n \to +\infty$$

$$\alpha \geq n^{-1+3\epsilon}.$$

Then there exists a polynomial-time algorithm, namely Algorithm 3, which outputs $\widehat{\pi} = \pi^*$ with probability at least $1 - o(1)$ under the seeded $\mathcal{G}(n, p; s, \alpha)$ model.

We next consider the dense graph regime, where we assume the average degree np is parameterized as:

$$np = bn^a \tag{1.4}$$

for some fixed constants $a, b \in (0, 1]$. Let

$$d = \left\lfloor \frac{1}{a} \right\rfloor + 1. \tag{1.5}$$

THEOREM 1.4. *Consider the dense graph regime (1.4). Assume*

$$b \le \frac{s}{16(2-s)^2}, \tag{1.6}$$

and

$$\alpha \ge \frac{300 \log n}{(nps^2)^{d-1}}, \tag{1.7}$$

where d is given in (1.5). Then there exists an polynomial-time algorithm, namely Algorithm 2, which outputs $\widehat{\pi} = \pi^*$ with probability $1 - 4n^{-1}$ under the seeded $\mathcal{G}(n, p; s, \alpha)$ model.

Note that d is the diameter of $\mathcal{G}(n,p)$ in the regime (1.4) [Bol01, Corollary 10.12]. The seed set size condition (1.7) stems from the fact that we match two vertices based on the number of "common" seeded vertices in their neighborhoods of radius $d-1$.

Our results for seeded graph matching also imply the results for graph matching without seeds.

THEOREM 1.5. *Suppose a Seeded Graph Matching algorithm outputs $\widehat{\pi} = \pi^*$ with high probability under the seeded graph matching model $\mathcal{G}(n, p; s, \alpha)$. Assume $nps^2 - \log n \to +\infty$ and $\alpha n \to +\infty$. Then there exists an algorithm, namely Algorithm 4, which calls the Seeded Graph Matching algorithm $n^{O(\alpha n)}$ times and outputs $\widehat{\pi} = \pi^*$ under the seedless model $\mathcal{G}(n, p; s)$ with high probability.*

REMARK 1. *Consider the dense regime (1.4) with $a = 1/k$ for an integer $k \ge 1$. Then $d = k+1$ and $(np)^{d-1} = b^k n$. Hence, as shown by Theorem 1.4, $\alpha n \ge 300 \log n (bs^2)^{-k}$, or equivalently $\Omega(\log n)$ number of seeds, suffice for exact recovery in polynomial-time. Since we can enumerate over all possible matchings for $\log n$ seeds in quasi-polynomial $n^{O(\log n)}$ time, this implies a quasi-polynomial time matching algorithm even without seeds, as shown by Theorem 1.5. The previous work [BCL+18] gives a quasi-polynomial time matching algorithm in the range*

$$np \in \left[n^{o(1)}, n^{1/153}\right] \cup \left[n^{2/3}, n^{1-\epsilon}\right].$$

Our results complement their results by filling in gaps in the above range with points $np \in \{bn^{1/k} : 1 \le k \le 152\}$.

1.2 Key Algorithmic Ideas and Analysis Techniques Most previous work [PG11, YG13, KL14, LFP13, FAP18, SGE17] on seeded graph matching exploits the seeded information by looking at the number of seeded vertices that are *direct* neighbors of a given vertex. Since the average degree of a vertex is np, $np\alpha \gg 1$ is needed so that there are sufficiently many seeded vertices that are direct neighbors of a given vertex.

Our idea is to explore much bigger ("global") neighborhoods of a given vertex up to radius ℓ for a suitably chosen ℓ, and match two vertices by comparing the set of seeded vertices in their ℓ-th local neighborhoods. This idea was used before in the noiseless and seedless case, in [Bol82, CP08] but to the best of our knowledge was not used in the noisy and seeded case. Since we are looking at global neighborhoods, we can only perform very simple tests. Indeed, the test we perform to check if two vertices are matched is just to count how many seeded vertices do the two neighborhoods have in common. Thus, our algorithms are very simple.

The main challenge in the analysis is to control the size of neighborhoods of the coupled graphs G_0, G_1 and G_2. In this regard, we draw on a number of tools from the literature on studying subgraph counts [JLR11] and the diameter in random graphs [Bol01]. See the full paper [MX18] for details.

2 Our Algorithms

Before presenting our algorithms, we first explain why (1.2) is needed for graph matching under $\mathcal{G}(n,p,s)$. Denote the intersection graph and the union graph by $G_1^* \wedge G_2$ and $G_1^* \vee G_2$. Then

$$G_1^* \wedge G_2 \sim \mathcal{G}(n, ps^2) \quad \text{and} \quad G_1^* \vee G_2 \sim \mathcal{G}(n, ps(2-s)).$$

Notice that $G_1^* \wedge G_2$ contains the statistical signature for matching vertices, as a subgraph in $G_1^* \wedge G_2$ will appear in both G_1 and G_2. If $nps^2 - \log n = O(1)$, then classical random graph theory implies that with high probability, $G_1^* \wedge G_2$ contains isolated vertices. The underlying true vertex correspondence of these isolated vertices cannot be correctly matched; hence the impossibility of exact recovery. See the full paper [MX18] for a precise argument.

In contrast, if $nps^2 - \log n \to +\infty$, then $G_1^* \wedge G_2$ is connected with high probability. Moreover, for a high-degree vertex i in $G_1^* \wedge G_2$, its local neighbhorhood grows like a branching process. In particular, the number of vertices at distance ℓ from i is approximately $(nps^2)^\ell$. Furthermore, for a pair of two vertices i, j chosen at random in $G_1^* \vee G_2$, the intersection of the local neighborhoods of i and j is typically of size $O\left((nps)^{2\ell} n^{-1}\right)$. Therefore, if $(nps^2)^\ell \gg (nps)^{2\ell} n^{-1}$ and $\alpha (nps^2)^\ell \gg 1$, a large number of vertices can be distinguished with high probability based on the set of seeded vertices in their ℓ-th local neighborhoods. This is the key idea underlying our algorithms.

We shall use the following notations of local neighborhoods. For a given graph G, we denote by $\Gamma_k^G(u)$ the

set of vertices at distance k from v in G:

$$\Gamma_k^G(u) = \{v \in V(G) : d(u,v) = k\} \tag{2.8}$$

and write $N_k^G(u)$ for the set of vertices within distance k from u:

$$N_k^G(u) = \cup_{i=0}^k \Gamma_i(u). \tag{2.9}$$

When the context is clear, we abbreviate $\Gamma_k^G(u)$ and $N_k^G(u)$ as $\Gamma_k(u)$ and $N_k(u)$ for simplicity.

2.1 A Simple Algorithm in Sparse Graph Regime We first present a simple seeded graph matching algorithm which succeeds up to the information-theoretic limit in terms of graph sparsity when the initial seed set is of size $n^{1/2+3\epsilon}$.

The idea of the algorithm is based on matching ℓ-th local neighborhoods of two vertices by finding independent paths (vertex-disjoint except for the starting vertex) to seeded vertices. The ℓ is chosen such that $(np)^\ell \approx n^{1/2-\epsilon}$. In this setting, we expect that if i in G_1 and j in G_2 are true matches, then their local neighborhoods intersect a lot; if i and j are wrong matches, then their local neighborhoods barely intersect. Hence, if $\alpha(nps^2)^\ell \gg 1$, then we can find a sufficiently large number of, say m, independent (vertex-disjoint except for i) paths of length ℓ from i to m seeded vertices in $\Gamma_\ell^{G_1^* \wedge G_2}(i)$. Such m paths of length ℓ form a starlike tree T in $G_1^* \wedge G_2$ with root vertex i and a set of m seeded leaves, denoted by L (See Fig. 1 for an example of $m = 3$ and $\ell = 2$). Note that T will appear in G_2 with root vertex i and the set of seeded leaves L; it will also appear in G_1 with root vertex $\pi^*(i)$ and the corresponding set of seeded leaves $\pi^*(L)$. However, since the ℓ-th local neighborhoods of two distinct vertices barely intersect, T will *not* appear in $G_1^* \vee G_2$ with a root vertex other than i. Therefore, we can correctly match the vertex $\pi^*(i)$ in G_1 with the high-degree vertex i in G_2 by finding such a starlike tree T, or equivalently m independent ℓ-paths to a set of m common seeded vertices.

There are two tuning parameters ℓ and m in Algorithm 1. Later in our analysis, we will optimally choose

$$\ell = \left\lfloor \left(\frac{1}{2} - \epsilon\right) \frac{\log n}{\log(nps^2)} \right\rfloor \geq 1 \tag{2.10}$$

and

$$m = \left\lceil \frac{2}{\epsilon} \right\rceil. \tag{2.11}$$

Note that when $nps^2 - \log n \to +\infty$, there may exist vertices with small degrees. In fact, classical random graph results say that the minimum degree of $\mathcal{G}(n,p)$ is

Algorithm 1 Graph matching based on counting independent ℓ-paths to seeded vertices

1: **Input:** G_1, G_2, π_0, $m, \ell \in \mathbb{Z}$
2: **Output:** $\widehat{\pi}$.
3: **Match high-degree vertices:** For each pair of unseeded vertices $i_1 \in V(G_1)$ and $i_2 \in V(G_2)$, if there are m independent ℓ-paths in G_2 from i_2 to a set of m seeded vertices $L \subset \Gamma_\ell^{G_2}(i_2)$, and there are m independent ℓ-paths in G_1 from i_1 to the corresponding set of m seeded vertices $\pi_0(L) \subset \Gamma_\ell^{G_1}(i_1)$, then set $\widehat{\pi}(i_2) = i_1$. Declare failure if there is any conflict.
4: **Match low-degree vertices:** For every $i_2 \in I_0$, set $\widehat{\pi}(i_2) = \pi_0(i_2)$. For all the pairs of unmatched vertices (i_1, i_2), if i_1 is adjacent to a matched vertex j_1 in G_1 and i_2 is adjacent to vertex $\widehat{\pi}(j_1)$ in G_2, set $\widehat{\pi}(i_2) = i_1$. Declare failure if there is any conflict.
5: Output $\widehat{\pi}$ to be a random permutation when failure is declared or there is any vertex unmatched.

k with high probability for a fixed integer k, provided that

$$(k-1)\log\log n + \omega(1) \leq nps^2 - \log n \leq k\log\log n - \omega(1),$$

see, e.g., [FK15, Section 4.2]. Hence, due to the existence of low-degree vertices, we may not be able to match all vertices correctly at one time based on the number of independent paths to seeded vertices. Our idea is to first match high-degree vertices and then match the remaining low-degree vertices with the aid of high-degree vertices matched in the first step. In particular, we let

$$\tau = \frac{nps^2}{\log(nps^2)}. \tag{2.12}$$

We say a vertex i high-degree, if its degree $d_i \geq \tau$ in $G_1^* \wedge G_2$; otherwise, we say it is a low-degree vertex. As we will see in Section 3, conditioning on that $G_1^* \wedge G_2$ and $G_1^* \vee G_2$ satisfy some typical graph properties, all low-degree vertices can be easily matched correctly given a correct matching of high-degree vertices.

In passing, we remark on the time complexity of Algorithm 1. Note that for ease of presentation, in Algorithm 1, we do not specify how to efficiently find out whether there exist m independent ℓ-paths in G_2 from i_2 to seed set $L \subset \Gamma_\ell^{G_2}(i_2)$, and m independent ℓ-paths in G_1 from i_1 to the corresponding seed set $\pi_0(L) \subset \Gamma_\ell^{G_1}(i_1)$. It turns out for a given pair of vertices i_1, i_2, this task can be reduced to a maximum flow problem in a directed graph, which can be solved via Ford–Fulkerson algorithm [FF56] in $O(n\alpha)$ time steps

(see the full paper [MX18] for details). Since there are at most n^2 pairs of vertices i_1, i_2 to consider, Step 3 of Algorithm 1 taks at most $O(n^3\alpha)$. The Step 4 of matching low-degree vertices in Algorithm 1 takes at most $O(n^3 p)$ time steps. Hence, in total Algorithm 1 takes at most $O\left(n^3(\alpha+p)\right)$ time steps.

2.2 A Simple Algorithm in Dense Graph Regime In this subsection, we consider the dense graph regime given in (1.4), where $np = bn^a$ and $d = \lfloor 1/a \rfloor + 1$. In this setting, since $p^d n^{d-1} - 2\log n \to +\infty$ and $p^{d-1}n^{d-2} - 2\log n \le -\infty$, it follows from [Bol01, Corollary 10.12] that $\mathcal{G}(n,p)$ has diameter d with high probability. Thus, when $s = \Theta(1)$, both $G_1^* \wedge G_2$ and $G_1^* \vee G_2$ have diameter d with high probability. Therefore, we present an algorithm based on matching the $d-1$-th local neighborhood of two vertices. More specifically, our algorithm matches $i_1 \in V(G_1)$ and vertex $i_2 \in V(G_2)$ based on the number of seeded vertices *within* distance $d-1$ from i_1 in G_1 and *within* distance $d-1$ from i_2 in G_2.

Algorithm 2 Graph matching based on $(d-1)$-hop witnesses in dense regime

1: **Input:** G_1, G_2, π_0, $d \in \mathbb{Z}$.
2: **Output:** $\widehat{\pi}$.
3: **Match all vertices:** For each pair of unseeded vertices $i_1 \in V(G_1)$ and $i_2 \in V(G_2)$, compute

(2.13)
$$w_{i_1,i_2} = \left|\left\{ j \in I_0 : \pi_0(j) \in N_{d-1}^{G_1}(i_1),\ j \in N_{d-1}^{G_2}(i_2) \right\}\right|.$$

Set $\widehat{\pi}(i_2) \in \arg\max_{i_1} w_{i_1,i_2}$. Set $\widehat{\pi}(i_2) = \pi_0(i_2)$ for each seeded vertex $i_2 \in I_0$. Declare failure if there is any conflict.

Algorithm 2 runs in polynomial-time. The precise running time depends on the data structures for storing and processing graphs. To be specific, let us assume it takes one time step to fetch the set of direct neighbors of a given vertex. Then fetching the set $N_\ell^G(i)$ of all vertices within distance ℓ from a given vertex i takes a total of $O(|N_\ell^G(i)|) = O(n)$ time steps. Thus computing w_{i_1,i_2} in (2.13) for a given pair of vertices i_1, i_2 takes at most $O(n)$ time steps. Hence, in total Algorithm 2 takes $O(n^3)$ time steps. One could possibly obtain a better running time via a more careful analysis or a better data structure.

The difference in the analysis compared to the first algorithm is that the $(d-1)$-th local neighborhoods are not tree-like anymore. Instead, we have to analyze the exposure process of the two neighborhoods, for which we use a previous result of [Bol01, Lemma 10.9] in studying the diameter of random graphs.

2.3 An Improved Algorithm in Sparse Graph Regime In the sparse regime where np is poly-logarithmic, Algorithm 2 does not perform well. This is because for two distinct vertices u, v that are close by, their ℓ-th local neighborhoods have a large overlap, i.e., $|N_\ell^G(u) \cap N_\ell^G(v)|$ is not much smaller than $|N_\ell^G(u)|$ or $|N_\ell^G(v)|$, rendering w_{i_1,i_2} given in (2.13) ineffective to distinguish u from v.

However, in the sparse regime, distinct vertices u, v only have very few common neighbors. Moreover, if we remove vertices u, v, the non-common neighbors become far apart, and for distinct vertices far apart, their local neighborhoods only have a small overlap. Therefore, we expect most of u, v's neighbor's ℓ-th local neighborhoods (after removing vertices u, v) do not have large intersections for a suitably chosen ℓ. This gives rise to Algorithm 3.

Algorithm 3 Graph matching based on neighbors' ℓ-hop witnesses in sparse regime

1: **Input:** G_1, G_2, π_0, $\ell \in \mathbb{Z}$, $\eta \in \mathbb{R}_+$.
2: **Output:** $\widehat{\pi}$.
3: **Match high-degree vertices:** For all the pairs of unseeded vertices (u, v) and for each pair of their neighbors (i, j) with $i \in \Gamma_1^{G_1}(u)$ and $j \in \Gamma_1^{G_2}(v)$, compute

(2.14)
$$w_{i,j}^{u,v} = \min_{x \in V(G_1), y \in V(G_2)} \left|\left\{ k \in I_0 : \pi_0(k) \in N_\ell^{G_1 \setminus \{u,x\}}(i),\ k \in N_\ell^{G_2 \setminus \{v,y\}}(j) \right\}\right|,$$

where $G \setminus S$ denotes G with set of vertices S removed. Let

(2.15)
$$Z_{u,v} = \sum_{i \in \Gamma_1^{G_1}(u)} \sum_{j \in \Gamma_1^{G_2}(v)} \mathbf{1}_{\{w_{i,j}^{u,v} \ge \eta\}}.$$

If $Z_{u,v} \ge \log n / \log\log n - 1$, set $\widehat{\pi}(v) = u$. Declare failure if there is any conflict.
4: The remaining two steps are the same as Algorithm 1.

Note that in computing the number of seeded vertices within distance ℓ from both vertex i in G_1 and vertex j in G_2 in (2.14), we remove vertices u, x in G_1 and vertices v, y in G_2, and take the minimum over all

possible choices of x and y. As a result,

(2.16)
$$w_{i,j}^{u,v} \le \left|\left\{k \in I_0 : \pi_0(k) \in N_\ell^{G_1\setminus\{u,v\}}(i),\ k \in N_\ell^{G_2\setminus\{u,v\}}(j)\right\}\right|,$$

where the right hand side becomes independent from the edges incident to u and v in $G_1^* \vee G_2$. This independence is crucial in our analysis to ensure that $Z_{u,v}$ is small for $u \neq \pi^*(v)$ via concentration inequalities of multivariate polynomials [Vu02].

There are two tuning parameters ℓ and η in Algorithm 3. In our analysis later, we will optimally choose

$$\ell = \left\lfloor \frac{(1-\epsilon)\log n}{\log(np)} \right\rfloor, \tag{2.17}$$

and

$$\eta = 4^{2\ell+2} n^{1-2\epsilon}\alpha. \tag{2.18}$$

As for time complexity, Algorithm 3 takes at most $O(n^{5+2\epsilon})$ time steps. To see this, similar to Algorithm 2, if we assume one unit time to fetch a set of direct neighbors of a given vertex, then it takes at most $O(n^3)$ time steps to compute (2.16) for given pairs of vertices (u,v) and (i,j). There are at most $n^{2+2\epsilon}$ such pairs. The step of matching low-degree vertices as specified in Algorithm 1 takes $O(n^3 p)$ time steps in total. Thus in total Algorithm 3 takes at most $O(n^{5+2\epsilon})$ time steps.

2.4 Graph Matching without Seeds Even without an initial seed set revealed as side information, we can select a random subset of vertices I_0 in G_1 and enumerate all the possible mappings $f : I_0 \to [n]$ from I_0 to vertices in G_2 in at most $n^{|I_0|}$ steps. Each of the possible mappings can be viewed as seeds; thus we can apply our seeded graph matching algorithm. Among all possible $n^{|I_0|}$ mappings, we finally output the best matching which minimizes the edge disagreements. See Algorithm 4 for details.

Since one of the possible mapping f will correspond to the underlying true matches of vertices in I_0, it follows that if our seeded graph matching succeeds with high probability and we are above the information-theoretic limit (so that the true matching minimizes the edge disagreements with high probability), the final output will coincide with the true matching with high probability, as stated in Theorem 1.5. More specifically, the proof is sketched below.

Proof. [Proof of Theorem 1.5] If $f : I_0 \to [n]$ is such that $f(i) = \pi^*(i)$ for all $i \in I_0$, then since our seeded graph matching succeeds with high probability, it follows that $\pi_f = \pi^*$ with high probability.

Algorithm 4 Seedless Graph matching via Seeded Graph Matching

1: **Input:** G_1, G_2
2: **Output:** $\widehat{\pi}$.
3: Select a random subset I_0 of $V(G_1)$ by including each vertex with probability α.
4: For every possible mapping $f : I_0 \to [n]$, run Seeded Graph Matching Algorithm with a seed set I_0, and output π_f.
5: Output

$$\widehat{\pi} \in \arg\min_{\pi_f} \|G_1 - \Pi_f G_2 \Pi^\top\|_F^2,$$

where Π_f is the permutation matrix corresponding to π_f.

Moreover, since we are above the information-theoretic limit, it follows from [CK17, Theorem 1] that with high probability,

$$\pi^* \in \arg\min_{\pi} \|G_1 - \Pi G_2 \Pi^\top\|_F^2,$$

where Π is the permutation matrix corresponding to π.

Therefore, $\widehat{\pi} = \pi^*$ with high probability. Finally, since $\alpha n \to \infty$, it follows that $|I_0|$ is at most $2\alpha n$ with high probability. Hence, Algorithm 4 calls the Seeded Graph Matching algorithm at most $n^{O(\alpha n)}$ times with high probability.

3 Analysis of Algorithm 1 in Sparse Regime

In this and next two sections, we sketch the proofs of our main results. The excluded details can be found in the full paper [MX18]. For the sake of analysis, we assume $\pi^* = id$, i.e., $\pi^*(i) = i$ for all $i \in [n]$, without loss of generality.

Our analysis of Algorithm 1 uses the technique for analyzing small subgraph containment [JLR11]. Let T denote a starlike tree formed by m independent (vertex-disjoint except for the root vertex) paths of length ℓ from root vertex to m distinct leaves for $\ell, m \ge 1$. Note that T consists of $m\ell + 1$ vertices and $m\ell$ edges (See Fig. 1 for an example of $m = 3$ and $\ell = 2$). Let $r(T)$ denote the root vertex of T and $L(T)$ denote the set of leaves of T. We say T is a subgraph of G, denoted by $T \subset G$, if $V(T) \subset V(G)$ and $E(T) \subset E(G)$. The key of our proof is to show that under certain conditions with high probability:

1. For every vertex i, there exists a copy of T rooted at i with all leaves seeded in the intersection graph $G_1^* \wedge G_2$;
2. There is no copy of $T_1 \cup T_2$ in the union graph $G_1^* \vee G_2$.

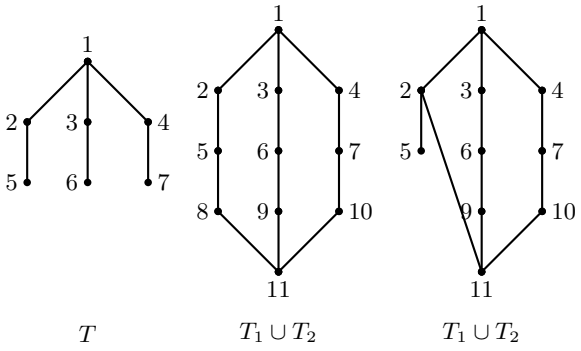

Figure 1: Left: T is a starlike tree with $m = 3$, $\ell = 2$, $r(T) = 1$ and $L(T) = \{5, 6, 7\}$. Middle and Right: Two examples of $T_1 \cup T_2$ such that T_1, T_2 are isomorphic to T, $r(T_1) \neq r(T_2)$, and $L(T_1) = L(T_2) = \{5, 6, 7\}$. For the middle, $V(T_1) \cap V(T_2) = \{5, 6, 7\}$; for the right, $V(T_1) \cap V(T_2) = \{2, 5, 6, 7\}$.

3.1 Success of Algorithm 1 on the Intersection of Good Events We first introduce a sequence of "good" events on whose intersection, Algorithm 1 correctly matches all vertices. We need the following graph properties:

(i) there is no isolated vertex;

(ii) for any two adjacent vertices, there are at least τ vertices adjacent to at least one of them;

(iii) For all vertices i with $d_i \geq \tau$, there are at least $2m$ independent ℓ-paths from i to $2m$ distinct vertices in I_0;

(iv) There is no pairs of subgraphs $T_1, T_2 \subset G$ that are isomorphic to T such that $r(T_1) \neq r(T_2)$, and $L(T_1) = L(T_2)$ (See Fig. 1 for an illustration).

(v) For every vertex i, there exist at most $m - 1$ independent ℓ-paths from i to $m - 1$ distinct vertices in $N_{\ell-1}^{G}(i)$.

Let

- $\mathcal{E}_1$ denote the event such that $G_1^* \wedge G_2$ satisfy properties (i)–(iii);
- $\mathcal{E}_2$ denote the event such that $G_1^* \vee G_2$ satisfy properties (iv) and (v);
- $\mathcal{E}_3$ denote the event such that for any two vertices i, j that are connected by a 2-path in $G_1^* \vee G_2$, at least one of the two vertices i, j must be a high-degree vertex in $G_1^* \wedge G_2$.

We claim that on event $\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3$, Algorithm 1 correctly matches all vertices. Recall that we can assume $\pi^* = id$ and thus $G_1 = G_1^*$ without loss of generality.

First, since $G_1^* \wedge G_2$ satisfy graph property (iii), it follows that in $G_1^* \wedge G_2$, for all high-degree vertices i, there exist $2m$ independent ℓ-paths to a set $S \subset \Gamma_\ell^{G_1^* \wedge G_2}(i)$ of $2m$ seeded vertices. Let $\widetilde{S} = S \backslash N_{\ell-1}^{G_1^* \vee G_2}(i)$. Since $G_1^* \vee G_2$ satisfy graph property (v), and $G_1^* \wedge G_2 \subset G_1^* \vee G_2$, it follows that

$$\left| S \cap N_{\ell-1}^{G_1^* \vee G_2}(i) \right| \leq m - 1$$

and thus $|\widetilde{S}| \geq m+1$. Moreover, since $G_1^* \wedge G_2 \subset G_1, G_2$, it follows that

$$\widetilde{S} \subset \Gamma_\ell^{G_1^* \wedge G_2}(i) \backslash N_{\ell-1}^{G_1^* \vee G_2} \subset \Gamma_\ell^{G_1^*}(i) \cap \Gamma_\ell^{G_2}(i).$$

Therefore, in both G_1 and G_2, there are at least $m + 1$ independent ℓ-paths from i to $\Gamma_\ell^{G_1^*}(i) \cap \Gamma_\ell^{G_2}(i)$.

Second, note that on event $\mathcal{E}_2$, $G_1^* \vee G_2$ satisfy graph property (iv). For the sake of contradiction, suppose there exist a pair of distinct vertices i, j and a set L of m seeded vertices such that there exist m independent ℓ-paths from i to set L in G_1 and m independent ℓ-paths from j to set L in G_2. Let T_k denote the starlike tree formed by the m independent ℓ-paths in G_k for $k = 1, 2$. Then $T_1, T_2 \subset G_1^* \vee G_2$ are isomorphic to T such that $r(T_1) = i, r(T_2) = j$ and $L(T_1) = L(T_2) = L$. This is in contradiction with the fact that $G_1^* \vee G_2$ satisfy graph property (iv).

It follows from the above two points that Algorithm 1 correctly matches all high-degree vertices i in $G_1^* \wedge G_2$, i.e., $\widehat{\pi}(i) = i$.

Next, we show that all low-degree vertices are matched correctly. Fix a low-degree vertex i. Since $G_1^* \wedge G_2$ satisfy graph properties (i) and (ii), it must have a high-degree neighbor j in $G_1^* \wedge G_2$. Since the high-degree vertex j has been matched correctly, i is adjacent to j in G_1 and i is also adjacent to $\widehat{\pi}(j) = j$ in G_2. Moreover, for the sake of contradiction, suppose there exists a pair of two distinct low-degree vertices i_1 and i_2 such that i_1 is adjacent to a matched vertex j_1 in G_1 and i_2 is adjacent to vertex $\widehat{\pi}(j_1)$ in G_2. Since $\widehat{\pi}(j_1) = j_1$, it follows that (i_1, j_1, i_2) form a 2-path in $G_1^* \vee G_2$. However, on event $\mathcal{E}_3$, i_1 and i_2 cannot be low-degree vertices simultaneously in $G_1^* \wedge G_2$, which leads to a contradiction. As a consequence, $\widehat{\pi}(i) = i$ for every low-degree vertex i.

Finally, to prove Theorem 1.1, it remains to show that under the theorem assumptions, $\mathbb{P}\{\mathcal{E}_i\} \to 1$ for all $i = 1, 2, 3$, which can be found in the full paper [MX18].

4 Analysis of Algorithm 2 in Dense Regime

Recall that $\Gamma_G^k(u)$ and $N_k^G(u)$ denotes the set of vertices *at* and *within* distance k from u in graph G, respectively, as defined in (2.8) and (2.9). The key is to show that $|N_{d-1}^{G_1^* \wedge G_2}(u)|$ is larger than $|N_{d-1}^{G_1^* \vee G_2}(u) \cap N_{d-1}^{G_1^* \vee G_2}(v)|$ for $u \neq v$ by a constant factor, so that we can matches two vertices correctly based on the number of common seeded vertices in their two large neighborhoods.

Proof. [Proof of Theorem 1.4] Define event

$$\mathcal{A} = \cap_u \left\{ \left| N_{d-1}^{G_1^* \wedge G_2}(u) \right| \geq \frac{3}{4}(nps^2)^{d-1} \right\}.$$

and

$$\mathcal{B} = \cap_{u \neq v} \left\{ \left| N_{d-1}^{G_1^* \vee G_2}(u) \cap N_{d-1}^{G_1^* \vee G_2}(v) \right| \leq \frac{1}{2}(nps^2)^{d-1} \right\}.$$

Using [Bol01, Lemma 10.9], one can show that $\mathbb{P}\{\mathcal{A}\} \geq 1 - n^{-10}$ and $\mathbb{P}\{\mathcal{B}\} \geq 1 - n^{-10}$ (see the full paper [MX18] for details).

Recall that I_0 is the initial set of seeded vertices. Define event

$$\mathcal{C} = \cap_u \left\{ \left| N_{d-1}^{G_1^* \wedge G_2}(u) \cap I_0 \right| > \frac{3}{5}(nps^2)^{d-1}\alpha \right\}.$$

Since each vertex is seeded independently with probability α, using binomial concentration inequalities, it can be shown that $\mathbb{P}\{\mathcal{C}^c\} \leq 2n^{-1}$. Similarly, define event

$$\mathcal{D} = \cap_{u \neq v} \left\{ \left| N_{d-1}^{G_1^* \vee G_2}(u) \cap N_{d-1}^{G_1^* \vee G_2}(v) \cap I_0 \right| < \frac{3}{5}(nps^2)^{d-1}\alpha \right\}.$$

It can be shown that $\mathbb{P}\{\mathcal{D}^c\} \leq 2n^{-1}$. Hence, $\mathbb{P}\{\mathcal{C} \cap \mathcal{D}\} \geq 1 - 4n^{-1}$.

Finally, since $G_1^* \wedge G_2$ is a subgraph of both G_1^* and G_2, it follows that

$$N_{d-1}^{G_1^* \wedge G_2}(i_2) \subset \left\{ j \in I_0 : \pi_0(j) \in N_{d-1}^{G_1}(\pi^*(i_2)), j \in N_{d-1}^{G_2}(i_2) \right\}.$$

Similarly, both G_1^* and G_2 are subgraphs of $G_1^* \vee G_2$, it follows that

$$\left\{ j \in I_0 : \pi_0(j) \in N_{d-1}^{G_1}(i_1), j \in N_{d-1}^{G_2}(i_2) \right\} \subset N_{d-1}^{G_1^* \vee G_2}\left((\pi^*)^{-1}(i_1)\right) \cap N_{d-1}^{G_1^* \vee G_2}(i_2).$$

Thus, on event $\mathcal{C} \cap \mathcal{D}$, for every vertex $i_2 \in V(G_2) \setminus I_0$,

$$w_{i_1,i_2} \begin{cases} > \frac{3}{5}(nps^2)^{d-1}\alpha & \text{if } i_1 = \pi^*(i_2) \\ < \frac{3}{5}(nps^2)^{d-1}\alpha & \text{o.w. .} \end{cases}$$

Hence, Algorithm 2 outputs $\widehat{\pi} = \pi^*$ on event $\mathcal{C} \cap \mathcal{D}$.

5 Analysis of Algorithm 3 in Sparse Regime

Before proving Theorem 1.3, we present two key lemmas without proofs.

The first lemma will be used later to conclude that the test statistic $Z_{u,u}$ given in (2.15) is large for all high degree vertices u.

LEMMA 5.1. *Suppose $G \sim \mathcal{G}(n,p)$ with $\log n \leq np \leq n^\epsilon$, and each vertex is included in I_0 with probability α. Recall that ℓ and η are given in (2.17) and (2.18), respectively. Assume $\eta \geq 4\log n$. Let $G \backslash S$ denote the graph G with set of vertices S removed. Then with probability at least $1 - n^{-1+o(1)}$,*

$$\sum_{j \in \Gamma_1^G(i)} \mathbf{1}_{\left\{ |\Gamma_\ell^{G \backslash S}(j) \cap I_0| \geq \eta \right\}} \geq d_i - 1, \ \forall S \text{ s.t. } \in S, |S| \leq 3.$$

The second lemma is useful to conclude that the test statistic $Z_{u,v}$ given in (2.15) is small for all distinct vertices u, v.

LEMMA 5.2. *Assume the same setup as Lemma 5.1. With probability at least $1 - 4/n$, for all distinct u, v, there exists a constant C depending only on ϵ such that*

$$\sum_{i \in \Gamma_1^G(u)} \sum_{j \in \Gamma_1^G(v)} \mathbf{1}_{\left\{ |N_\ell^{G \backslash \{u,v\}}(i) \cap N_\ell^{G \backslash \{u,v\}}(j) \cap I_0| \geq \eta \right\}} \leq C.$$

With Lemma 5.1 and Lemma 5.2, we are ready to finish the proof of Theorem 1.3.

Proof. [Proof of Theorem 1.3] Recall that τ is given in (2.12) and the definition of high-degree vertices. We first prove that Algorithm 2 correctly matches the high-degree vertices in $G_1^* \wedge G_2$ with high probability.

Recall the definition of Z give in (2.15). Applying Lemma 5.1 with $G = G_1^* \wedge G_2$, we get that with high probability, for all high-degree vertices u,

$$Z_{u,u} \geq \tau - 1 = \frac{nps^2}{\log(nps^2)} - 1.$$

Moreover, by definition,

$$w_{i,j}^{u,v} \leq \left| \left\{ k \in I_0 : \pi_0(k) \in N_\ell^{G_1 \backslash \{u,v\}}(i), \ k \in N_\ell^{G_2 \backslash \{u,v\}}(j) \right\} \right| \leq \left| N_\ell^{G_1^* \vee G_2 \backslash \{u,v\}}(i) \cap N_\ell^{G_1^* \vee G_2 \backslash \{u,v\}}(j) \cap I_0 \right|.$$

Applying Lemma 5.2 with $G = G_1^* \vee G_2$, we get that with high probability, $Z_{u,v} \leq C$ for all $u \neq v$ for a constant $C > 0$ only depending on ϵ. Since for sufficiently large n, $\tau \geq C + 1$, it follows that Algorithm 2 correctly matches all high-degree vertices with high probability.

The proof of correctness for matching low-degree vertices is the same as Algorithm 1 and thus omitted.

References

[Bab16] László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 684–697, New York, NY, USA, 2016. ACM.

[BCL+18] Boaz Barak, Chi-Ning Chou, Zhixian Lei, Tselil Schramm, and Yueqi Sheng. (nearly) efficient algorithms for the graph matching problem on correlated random graphs. *arXiv preprint arXiv:1805.02349*, 2018.

[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.

[BES80] László Babai, Paul Erdos, and Stanley M Selkow. Random graph isomorphism. *SIaM Journal on computing*, 9(3):628–635, 1980.

[BGM82] László Babai, D Yu Grigoryev, and David M Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 310–324. ACM, 1982.

[Bol82] Béla Bollobás. Distinguishing vertices of random graphs. *North-Holland Mathematics Studies*, 62:33–49, 1982.

[Bol01] Béla Bollobás. *Random Graphs (2nd Edition)*. Cambridge Studies in Advanced Mathematics, 2001.

[CFSV04] Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Thirty years of graph matching in pattern recognition. *International journal of pattern recognition and artificial intelligence*, 18(03):265–298, 2004.

[CK16] Daniel Cullina and Negar Kiyavash. Improved achievability and converse bounds for erdos-rényi graph matching. In *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, pages 63–72. ACM, 2016.

[CK17] Daniel Cullina and Negar Kiyavash. Exact alignment recovery for correlated erdos renyi graphs. *arXiv preprint arXiv:1711.06783*, 2017.

[CP08] Tomek Czajka and Gopal Pandurangan. Improved random graph isomorphism. *Journal of Discrete Algorithms*, 6(1):85–92, 2008.

[FAP18] Donniell E. Fishkind, Sancar Adali, and Carey E. Priebe. Seeded graph matching. *arXiv preprint arXiv:1209.0367*, 2018.

[FF56] Lester R Ford and Delbert R Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8(3):399–404, 1956.

[FK15] Alan Frieze and Micha Karoski. *Introduction to Random Graphs*. Cambridge University Press, 2015.

[FQRM+16] Soheil Feizi, Gerald Quon, Mariana Recamonde-Mendoza, Muriel Médard, Manolis Kellis, and Ali Jadbabaie. Spectral alignment of networks. *arXiv preprint arXiv:1602.04181*, 2016.

[JLR11] Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random Graphs*. John Wiley& Sons, Inc., 2011.

[KHG15] Ehsan Kazemi, S Hamed Hassani, and Matthias Grossglauser. Growing a graph matching from a handful of seeds. *Proceedings of the VLDB Endowment*, 8(10):1010–1021, 2015.

[KL14] Nitish Korula and Silvio Lattanzi. An efficient reconciliation algorithm for social networks. *Proceedings of the VLDB Endowment*, 7(5):377–388, 2014.

[LFP13] Vince Lyzinski, Donniell E. Fishkind, and Carey E. Priebe. Seeded graph matching for correlated Erdős-Rényi graphs. *Journal of Machine Learning Research*, 15, 2013.

[Lip78] R. J. Lipton. The beacon set approach to graph isomorphism. Technical report, Yale University, 1978.

[LR13] Lorenzo Livi and Antonello Rizzi. The graph matching problem. *Pattern Analysis & Applications*, 16(3):253–283, 2013.

[Luk80] Eugene M Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. In *21st Annual Symposium on Foundations of Computer Science*, pages 42–49. IEEE, 1980.

[MMS10] Konstantin Makarychev, Rajsekar Manokaran, and Maxim Sviridenko. Maximum quadratic assignment problem: Reduction from maximum label cover and lp-based approximation algorithm. *Automata, Languages and Programming*, pages 594–604, 2010.

[MX18] Elchanan Mossel and Jiaming Xu. Seeded graph matching via large neighborhood statistics. arXiv preprint arXiv:1807.10262, to appear in 2019 ACM-SIAM Symposium on Discrete Algorithms (SODA),, 2018.

[NS08] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125. IEEE, 2008.

[NS09] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *Security and Privacy, 2009 30th IEEE Symposium on*, pages 173–187. IEEE, 2009.

[PG11] Pedram Pedarsani and Matthias Grossglauser. On the privacy of anonymized networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1243, 2011.

[SGE17] F Shirani, S Garg, and E Erkip. Seeded graph matching: Efficient algorithms and theoretical guarantees. *arXiv preprint arXiv:1805.02349*, 2017.

[Vu02] V. H. Vu. Concentration of non-lipschitz functions and applications. *Random Struct. Algorithms*, 20(3):262–316, May 2002.

[Wri71] Edward M Wright. Graphs on unlabelled nodes with a given number of edges. *Acta Mathematica*, 126(1):1–9, 1971.

[YG13] Lyudmila Yartseva and Matthias Grossglauser. On the performance of percolation graph matching. In *Proceedings of the first ACM conference on Online social networks*, pages 119–130. ACM, 2013.