IMPROVING THE OUTPUT OF ALGORITHMS FOR LARGE-SCALE APPROXIMATE GRAPH MATCHING

BY

JOSEPH LUBARS

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Adviser:

Professor R. Srikant

ABSTRACT

In approximate graph matching, the goal is to find the best correspondence between the labels of two correlated graphs. Recently, the problem has been applied to social network de-anonymization, and several efficient algorithms have been proposed for approximate graph matching in that domain. These algorithms employ seeds, or matches known before running the algorithm, as a catalyst to match the remaining nodes in the graph. We adapt the ideas from these seeded algorithms to develop a computationally efficient method for improving any given correspondence between two graphs. In our analysis of our algorithm, we show a new parallel between the seeded social network de-anonymization algorithms and existing optimization-based algorithms. When given a partially correct correspondence between two Erdos-Renyi graphs as input, we show that our algorithm can correct all errors with high probability. Furthermore, when applied to real-world social networks, we empirically demonstrate that our algorithm can perform graph matching accurately, even without using any seed matches.

To my family and friends, for their love and support.

ACKNOWLEDGMENTS

Thank you to R. Srikant, my advisor, all of my instructors, my family, and friends for giving me the knowledge and support needed to make it this far. Also, thank you to Sanjay Shakkottai and Matthias Grossglauser for helpful discussions and ideas about approximate graph matching.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

With the proliferation of publicly available social information, both released by companies and mined, privacy is becoming a large concern. Recently, after Netflix provided anonymized data to researchers on movie ratings for the Netflix Challenge, researchers were able to match the data to publicly available IMDB data to recover many identities [1]. Because of this, it is important to understand how to guarantee the privacy of data which is collected or released to the public.

Social network data, such as the Facebook or LinkedIn graphs, contains rich structural information in the form of friendships or connections between users. Recently, it has become apparent that this structural information is sufficient to recover node identities, even without additional labels such as movie ratings being attached to the anonymous nodes. This process of recovering node identities in a partially or completely unlabeled social network is known as social network de-anonymization. In the years since Narayanan and Shmatikov demonstrated a successful de-anonymization attack on real social networks, the problem of developing efficient de-anonymization algorithms has received much scrutiny [2].

Social network de-anonymization is a recent application of a more general graph mining problem called "approximate graph matching." The goal of approximate graph matching is to find the best correspondence between two given graphs, for some sense of "best." This goal is a generalization of that of graph isomorphism, where an exact correspondence is required. If a graph isomorphism is found, then this isomorphism will also solve the approximate graph matching problem. In addition to network security, approximate graph matching has seen various applications in many areas over the years, including computer vision [3] and bioinformatics [4, 5].

Finding an efficient, practical algorithm for graph isomorphism remains a celebrated problem, and by extension, there is also no known efficient exact

algorithm for approximate graph matching. In fact, one common setting for the approximate graph matching problem is known to be NP-hard [6]. Therefore, we turn to approximate algorithms. A broad range of approximate approaches exist for approximate graph matching, but they can be broadly divided into two categories: seedless and seeded matching algorithms.

Seedless algorithms attempt to solve the matching problem with no additional side information. Some algorithms use various relaxations of the original optimization problem. Examples include a convex relaxation approach called QCV and a convex-concave approach called PATH [7]. Other algorithms, such as the algorithm developed by Umeyama [8], use spectral techniques. Still other algorithms use techniques such as random walks [9] and Bayesian inference [10].

The other category of matching algorithms is the set of seeded algorithms. These require "seeds," which take the form of a set of matches which each identify the correct correspondence for one vertex in each graph. These initial matches can be used to efficiently recover the remaining matches across the entire graph. Many seedless algorithms can be adapted to perform seeded matching, for example, by encoding the constraints into the convex program used by the QCV (convex relaxation) algorithm. Doing so can significantly improve the graph matching results [11]. However, inspired by performing deanonymization on large-scale social networks, where some identities may not be anonymous and the network sizes can be enormous, new algorithms have been developed. The two primary algorithms are percolation graph matching, proposed by Pedarsani and Grossglauser [12], and a similar witness-based algorithm proposed by Korula and Lattanzi [13]. These and other similar algorithms have been shown to work on a fairly wide range of graph models [14].

We attempt to extend the ideas present in these seeded graph matching algorithms by asking a new question: Can we use them for correction? The output of a seedless algorithm can be interpreted as a set of seed matches, some of which are correct, and some of which are incorrect relative to an optimal solution. We develop a new algorithm, based on the algorithms in [12, 13], that will take in a partially correct graph matching and efficiently correct all of the errors. We will give a new interpretation of the matching mechanism for our algorithm, and show that it can theoretically correct all errors on a stochastic block model and, under certain assumptions, when used

as a post-processing step for other graph matching algorithms, can produce seedless graph matching with accuracy that is significantly better than the current state-of-the-art.

Our first contribution is to develop an algorithm which corrects errors in a given initial correspondence. In order to model seedless graph matching, which often produces results with some correct matches but also many unknown incorrect matches, we propose a new model where a correspondence is given between the node labels of two correlated graphs, but only a small fraction of the matches in this correspondence are correct. We show that under this model, our algorithm corrects all errors for stochastic block model graphs under certain, reasonably interpretable assumptions.

Our second contribution is to propose a general methodology for performing graph matching when there is no initial correspondence given, or only a very small number of seeds, where first an appropriate approximate graph matching algorithm is run, depending on the problem, and then our algorithm is run on the output. Using extensive numerical experiments on both random graph models (including models other than the stochastic block model) and real social network datasets, we show that using our algorithm as a post-processing step improves the results produced by state-of-the-art algorithms for approximate graph matching.

The rest of the thesis is organized as follows. Chapter 2 contains our mathematical model and problem statement. Chapters 3 and 4 contain our main result and its proof, respectively. Chapter 5 relates our work to prior work on the problem. Chapter 6 contains simulation results, and concluding remarks are presented in Chapter 7.

CHAPTER 2

MODEL AND PROBLEM STATEMENT

In order to generate two correlated graphs G_1 and G_2, we use the following subsampling model. We start with an underlying graph $G = (V, E)$ on n nodes, which can be interpreted, for example, as the set of all acquaintances among n people. Each edge of G is sampled independently with probability s for inclusion into a subgraph G_1, which could represent the relationships present between the individuals in one social network. Independently, the edges are sampled again with probability s to produce another graph G_2, which could represent the relationships between the same individuals in a different social network. However, the identities of the individuals in G_2 are anonymized, or unknown. Therefore, we permute the vertices of G_2 according to a permutation π, chosen uniformly at random. Given $G_1 = (V_1, E_1)$ and $\pi(G_2) = (V_2, \tilde{E}_2)$, the goal is to recover π. This model was introduced by Pedarsani and Grossglauser in the case that G is an Erdös-Rényi graph [15]. However, here, we will assume a more general stochastic block model for G: Partition V into k communities: $V = C_1 \sqcup C_2 \sqcup \ldots \sqcup C_k$. Then, given a symmetric $k \times k$ matrix P of community connection probabilities, connect vertices between communities C_i and C_j with probability P_{ij}. We will assume that $P_{ii} \geq P_{ij}$ for all i, j, so edges are more likely within the same community.

Various algorithms exist for solving this approximate graph matching problem. However, they are, in general, imperfect, and may only successfully recover $\pi(v)$ for a subset of the vertices v in V. We wish to start with an estimate $\hat{\pi}$ which agrees with π on an *unknown* subset of V, and then use this estimate to correct these unknown errors and recover π exactly with high probability. In our model, we assume the initial estimate $\hat{\pi}$ is randomly generated with a constant fraction β of correct matches as follows:

First, a subset $W \subset V$ is generated by sampling each vertex in V with probability β. For each $v \in W$, set $\hat{\pi}(v) = \pi(v)$. Next, generate a permutation π_W of $V \setminus W$ uniformly at random. For each $v \in V \setminus W$, set

$\hat{\pi}(v) = \pi(\pi_W(V))$. Our algorithm is intended as a post-processing step for other algorithms which may produce such a $\hat{\pi}$. However, it is difficult to model exactly how such algorithms will produce their estimate $\hat{\pi}$. Our modeling assumption on $\hat{\pi}$ is used to obtain tractable theoretical results. However, the algorithm we obtain works remarkably well in practice, even though the pre-processing algorithms may or may not produce an estimate that agrees with our assumptions.

CHAPTER 3

MAIN RESULTS

3.1 Basic Algorithm

To correct the errors of $\hat{\pi}$, we propose an algorithm based on "witnesses," as defined in [13]: for each pair (v_1, v_2), where $v_1, v_2 \in V$, we count the number of *witnesses* for that pair, defined as the set of vertices x such that $(v_1, x) \in E_1$ and $(v_2, \hat{\pi}(x)) \in \tilde{E}_2$. Let $w(v_1, v_2)$ be this number of witnesses. Each of these witnesses encodes some evidence that $\pi(v_1) = v_2$. Therefore, we would like to find a maximum weight bipartite matching using the weights $w(v_1, v_2)$ in order to improve our estimate of $\hat{\pi}$ (see Figure 3.1). The complete algorithm is presented as Algorithm 1 below.

Algorithm 1: Basic Correction Algorithm

```
Input: G_1, π(G_2), π̂
Initialize W = zeros(n × n)
for u in V do
    for v in V do
        Calculate W_{u,v} = w(u, v)
Return MaximumWeightMatching(W)
```

Unfortunately, for large values of n, this procedure is inefficient. Naively iteratively computing $w(u, v)$ for every u and v requires at least n^2 computations, which is infeasible for large social networks which may contain thousands or even millions of nodes. Similarly, maximum weighted bipartite matching can be done in $n(|W| + n\log(n))$ time, where $|W|$ is the number of nonzero entries of W [16]. This is also too inefficient for large values of n.

Instead, we will replace the procedure for constructing W by a more efficient procedure featuring the "CountPaths" subroutine, introduced below, which can achieve a better complexity of $O(|E_1|\Delta_2)$, where Δ_2 is the largest

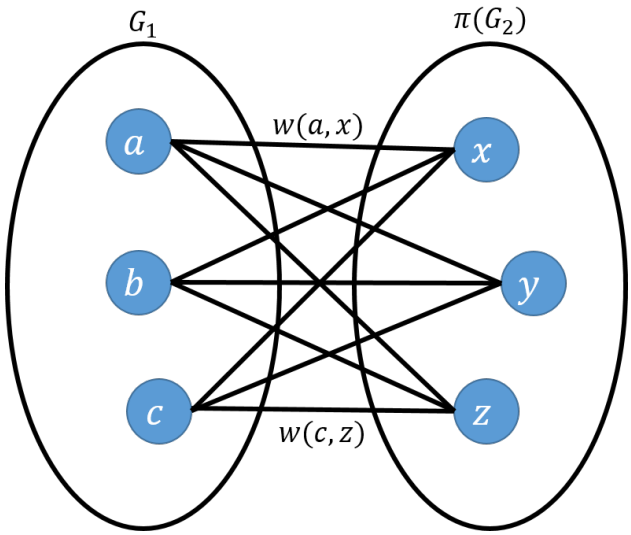

Figure 3.1: Maximizing the number of witnesses as a bipartite matching problem

degree of a vertex in G_2. In an Erdös-Rényi graph with $p = O(\log(n)/n)$, for example, at the threshold for connectivity, our this procedure runs in $O(n \log^2(n))$ time. We will also replace the maximum weight matching by a greedy matching (complexity $|W| \log(|W|)$). In practice, and theoretically in the case of the stochastic block model, the greedy matching is sufficient to recover π perfectly from $\hat{\pi}$.

Algorithm 2: Optimized Correction Algorithm

Input: G_1, $\pi(G_2)$, $\hat{\pi}$
for u *in* V **do**
  $W(u, \cdot) = \text{CountPaths}(G_1, \pi(G_2), \hat{\pi}, u)$
Return GreedyMaximumWeightMatching(W)

CountPaths relies on the interpretation that every vertex x which is a witness for (u, v) can be thought of as a "path" from u, to a, to $\pi(a)$, to v, where the first edge (u, a) is an edge in G_1, the second edge $(a, \pi(a))$ is along the mapping π, and the third edge $(\pi(a), v)$ is an edge in $\pi(G_2)$ (see Figure 3.2). Counting these paths can be implemented as a simple iterative process, reproduced below.

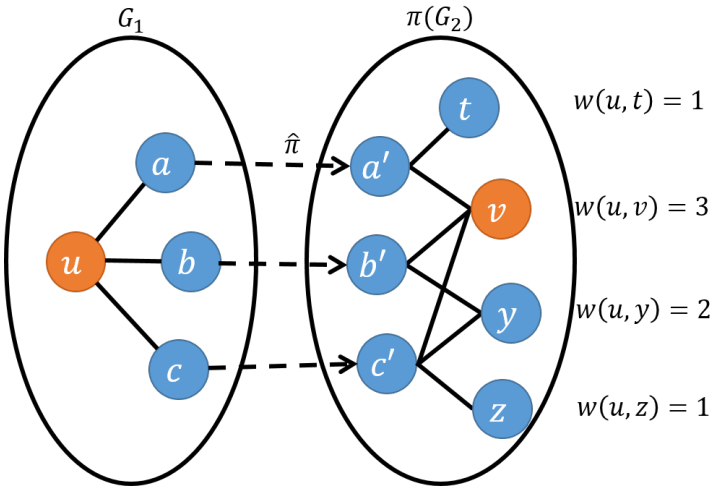

Figure 3.2: Counting paths to determine witnesses for (u, v)

Algorithm 3: CountPaths

Input: G_1, $\pi(G_2)$, $\hat{\pi}$, u
Initialize $W(u, \cdot) = 0$
for $x \in G_1.neighbors(u)$ **do**
  for $v \in \pi(G_2).neighbors(\hat{\pi}(x))$ **do**
    $W(u, v)$ += 1
Return $W(u, \cdot)$

3.2 Iterated Algorithm

One feature of our matching algorithm is its ability to take a mapping as input and output an improved mapping. It is then natural to take the new, improved mapping and feed it back into our algorithm, to further improve it. This is the intuition behind the iterated correction algorithm, presented as Algorithm 4:

Algorithm 4: Iterated Correction Algorithm

Input: G_1, $\pi(G_2)$, $\hat{\pi}$, k
for *iteration* $= 1, \ldots, k$ **do**
  $\hat{\pi} = OptimizedCorrection(G_1, \pi(G_2), \hat{\pi})$
Return $\hat{\pi}$

As long as the first iteration of our algorithm is able to improve the accuracy of the original input matching, further iterations should be able to further improve, until a local optimum is reached, where (hopefully) all nodes are matched correctly. An illustration of this phenomenon is available in Figures 3.3 and 3.4. As long as one iteration of our algorithm performs above the dotted line (corresponding to producing more correct matches than in the input matching), then the iterated algorithm is able to correct all errors in the input matching.

3.3 Intuition

One metric sometimes used for quantifying the correctness of an approximate graph matching is the "overlap metric," which counts the number of edges on which two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ agree. The larger this metric, the more evidence we have that G_1 and G_2 are the same. In our setting where G_1 and G_2 are random objects, this can be conveniently expressed using indicator variables $\mathbb{I}$:

$$\Delta(G_1, G_2) = \sum_{u \in V} \sum_{v \in V} \mathbb{I}\{(u, v) \in E_1, (u, v) \in E_2\} \tag{3.1}$$

If we are given a random permutation $\pi(G_2) = (V, \tilde{E}_2)$, we can use this metric to try to recover π, by solving a maximization problem over all per-

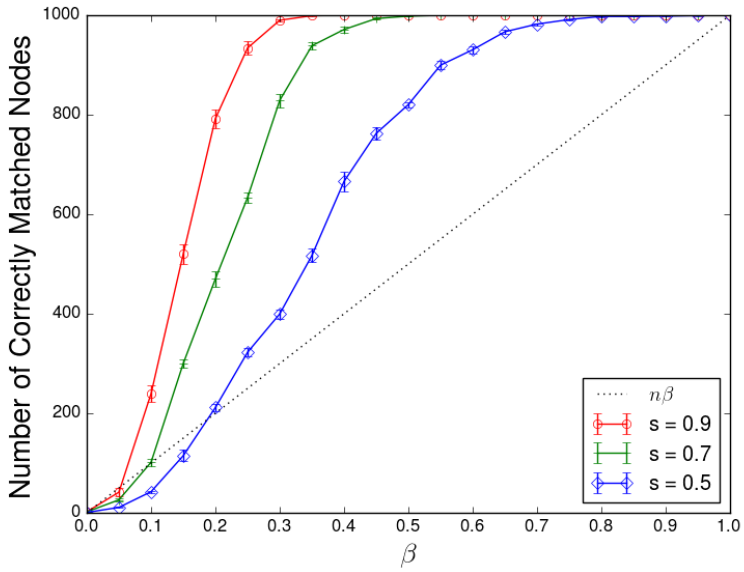

Figure 3.3: Erdös-Rényi graphs for varying values of β. Algorithm 2 is used.

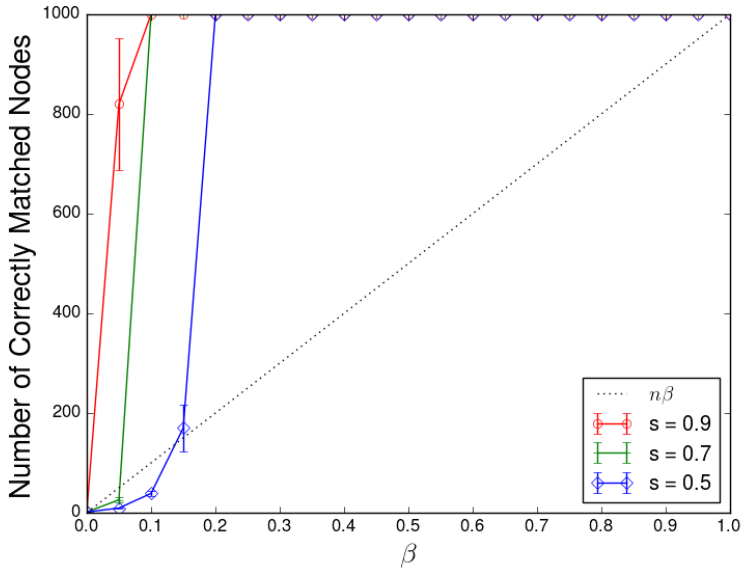

Figure 3.4: Erdös-Rényi graphs for varying values of β. Algorithm 4 is used.

mutations, attempting to recover π:

$$\max_{\pi'} \sum_{u \in V} \sum_{v \in V} \mathbb{I}\{(u,v) \in E_1, (\pi'(u), \pi'(v)) \in \tilde{E}_2\} \tag{3.2}$$

This natural approach is the maximum likelihood estimator in the case that G is an Erdös-Rényi graph and G_1 and G_2 are sampled according to our model [17]. However, it is very difficult to solve exactly. Our approach, given an estimate matching $\hat{\pi}$, is to solve the maximization problem using our estimate in place of π' in one spot:

$$\max_{\pi'} \sum_{u \in V} \sum_{v \in V} \mathbb{I}\{(u,v) \in E_1, (\pi'(u), \hat{\pi}(v)) \in \tilde{E}_2\} \tag{3.3}$$

Note that the inside sum is merely the number of witnesses for u and $\pi'(u)$:

$$\max_{\pi'} \sum_{u \in V_1} w(u, \pi'(u)) \tag{3.4}$$

This one change transforms the problem from an extremely difficult combinatorial optimization problem into a much simpler maximum weighted bipartite matching problem, which can now be solved efficiently. The assumption is that if $\hat{\pi}$ is close enough to π, the solutions to (3.2) and (3.3) will also be close.

In order to provide intuition as to why this is true, we will consider the simple case where G is an Erdös-Rényi graph with edge probability p. Proceeding in a similar manner to Yartseva and Grossglauser in [12], look at the indicator $\mathbb{I}\{(u,v) \in E_1, (\hat{\pi}(u), \pi'(v)) \in \tilde{E}_2\}$ in (3.3). Suppose $\hat{\pi}(v) = \pi(v)$. Then if $\pi'(u) = \pi(u)$, it takes the value 1 if $(u,v) \in E$ and the edge is sampled twice, for a total probability of ps^2. However, in any other case, (u,v) in E_1 and $(\hat{\pi(u)}, \pi'(v))$ in E_2 no longer are sampled from the same edge in the underlying graph E. Therefore, the indicator takes the value 1 with probability only p^2s^2. Because p is assumed to be small, $p^2s^2 << ps^2$.

We expect approximately $n(\beta ps^2 + (1-\beta)p^2s^2)$ witnesses for a correct match $(u, \pi(u))$, and only approximately np^2s^2 witnesses for an incorrect match. Because $n(\beta ps^2 + (1-\beta)p^2s^2) >> np^2s^2$ if β is large enough, we expect greedy matching to recover correct matches with high probability. As an example, see Figure 3.5. Already, for $n = 40$, one entry corresponding to the correct match dominates each row and column. We will make a similar

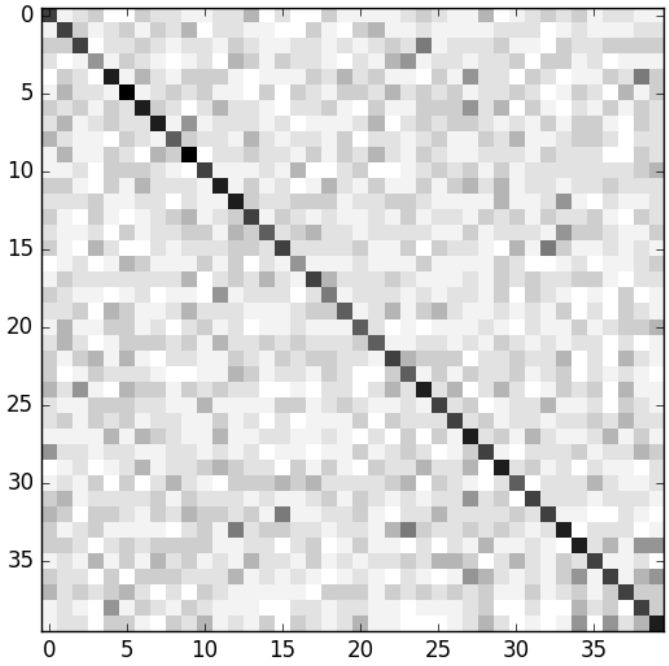

Figure 3.5: $w(u, v)$ for $u, v \in G_1, G_2$. Here, G is Erdös-Rényi(40, 0.25), $s = 0.9$, and $\hat{\pi}$ agrees with π=identity on 20 vertices.

argument formal for the stochastic block model below.

3.4 Connection to Optimization-Based Methods

There is another way to represent the witness matrix W which is enlightening. Letting A_1 be the adjacency matrix of G_1 and A_2 the adjacency matrix of $\pi(G_2)$, we note that $w(u, v)$ is the inner product of the u-th row of A_1 and the $\hat{\pi}(v)$-th row of A_2. Therefore, if $\hat{P}$ is the permutation matrix corresponding to $\hat{\pi}$, we can express W succinctly as:

$$W = A_1^T \hat{P} A_2 \tag{3.5}$$

One way to write the overlap metric in matrix form (defined in the previous section) is as follows:

$$f(\hat{P}) = \Delta(G_1, \hat{\pi}(G_2)) = Tr(\hat{P} A_1^T \hat{P}^T A_2) \tag{3.6}$$

It is easy to verify that, in fact, our witness matrix W is proportional to the gradient of f:

$$\nabla f(\hat{P}) = A_1^T \hat{P} A_2 + A_1 \hat{P} A_2^T = 2A_1^T \hat{P} A_2 \tag{3.7}$$

This implies a connection between witness-based methods like ours and optimization-based methods based on the overlap metric. The closest connection is to the Frank-Wolfe algorithm. Following is the Frank-Wolfe algorithm for maximizing f over the set of doubly stochastic matrices (Π_n here is the set of $n \times n$ permutation matrices):

Algorithm 5: Frank-Wolfe Algorithm for Approximate Graph Matching

Input: Initial Guess P_0, A_1, A_2, k
for $i = 0, \ldots, k-1$ **do**
  Compute $W = 2A_1 P_i^T A_2$
  Use the Hungarian algorithm to maximize $\langle Q, W \rangle$ subject to $Q \in \Pi_n$
  Maximize $\langle ((1-\gamma)P_i + \gamma Q_i)^T A_1 ((1-\gamma)P_i + \gamma Q_i), A_2 \rangle$ over $\gamma \in [0, 1]$
  Set $P_{i+1} \Leftarrow P_i + \gamma(Q - P_i)$
Find $\hat{P}$ to maximize $\langle P_k, P \rangle$ for $P \in \Pi$ using the Hungarian Algorithm

For comparison, following is our iterated algorithm:

Algorithm 6: Our Iterated Correction Algorithm (Again)

Input: Initial Guess P_0, A_1, A_2, k
for $i = 0, \ldots, k-1$ **do**
  Compute $W = A_1 P_i^T A_2$
  Use the greedy matching to approximately maximize $\langle Q, W \rangle$ subject to $Q \in \Pi_n$
  $\gamma = 1$
  Set $P_{i+1} \Leftarrow P_i + \gamma(Q - P_i)$
Return P_k

The only difference here is that we are approximately solving the linear assignment problem to compute Q, and we are always using a step size γ of 1 instead of computing the optimal step size at every iteration. We have already explained why using a greedy matching makes sense, but why does setting $\gamma = 1$ make sense? Computationally, it restricts P_i to a permutation

matrix at every step, ensuring that computing W always remains efficient. In practice, it seems to work well, often still converging to a good solution.

3.5 Performance Guarantees

Given n, P, $C_1, \ldots, C_k$, and β, let G, G_1, G_2, π, and $\hat{\pi}$ be generated according to our model above (so G is generated from a stochastic block model). We define d_i as follows for each community $i \in [k]$, with the interpretation that d_i is the average degree in G for vertices in community i:

$$d_i := P_{ii}(|C_i| - 1) + \sum_{j \in [k] \setminus \{i\}} P_{ij}|C_j| \tag{3.8}$$

Then, as long as the minimum d_i is large enough, but not so large as to make the graph too densely connected, the optimized correction algorithm above will perfectly recover π with high probability (in the asymptotic limit as n increases):

Theorem 1 *Suppose $s^2\beta \min_{i \in [k]} d_i > 16 \log(n)$ and $P_{ij} = o(1)$ for all i, j. Then the optimized correction algorithm recovers π from $\hat{\pi}$ with probability $1 - o(1)$.*

As a corollary, by setting every entry of P to be equal to p, we recover a result similar to that of Korula and Lattanzi [13]:

Corollary 1 *Suppose G is Erdös-Rényi(n, p), where $(n-1)ps^2\beta > 16 \log(n)$ and $p = o(1)$. Then the optimized correction algorithm recovers π from $\hat{\pi}$ with probability $1 - o(1)$.*

These results can be interpreted as follows: First, the expected degree of every node should be high enough that the intersection of G_1 and G_2 is connected. For Erdös-Rényi graphs, it is known that if $nps^2 < \log(n)$, then no algorithm can recover π, given no side information, and if $nps^2 > 2\log(n)$, then the maximum likelihood estimator succeeds with high probability [17]. Our lower bound on the average degree is therefore within a constant factor

of optimal for Erdös-Rényi graphs. Secondly, the graphs should not be too densely connected. Some limit on the density of the graphs is required, as matching two graphs is equivalent to matching their complements, and we have already established that the graphs cannot be too sparse. Furthermore, most real-life networks have relatively small degrees or even constant average degrees, so forcing the degree of each node to be $o(n)$ is a reasonable assumption in this light.

Interestingly, none of our proofs depend on the number of communities k. Therefore, we can let k grow with n as fast as we like, or even set $k = n$ to independently designate each edge probability in our graph G. This allows us to extend our result to other models utilizing independent Bernoulli edges such as the Chung-Lu model (see [18]), as long as the expected degree d_i of each node is large enough to satisfy the same constraint $s^2\beta d_i > 16\log(n)$:

Corollary 2 *Let P be a symmetric $n \times n$ matrix, and let $G = (V, E)$ be the undirected graph with each edge (i, j) independently present with probability P_{ij}. Suppose $P_{ij} = o(1)$ for all i, j. Furthermore, assume that for every vertex i, the expected degree d_i satisfies $s^2\beta d_i > 16\log(n)$. Then, the optimized correction algorithm recovers π from $\hat{\pi}$ with probability $1 - o(1)$.*

CHAPTER 4

PROOF

The general strategy will be to show that with high probability, $w(u, \pi(u)) > w(u, v)$ for all u and all $v \neq u$. Then, a greedy maximum weight matching will match u with $\pi(u)$ for each u. In order to show that $w(u, \pi(u))$ dominates other numbers of witnesses for a given w, we will show the following results:

Lemma 1 *For each $u \in V$, suppose $u \in C_i$. Then, if $s^2\beta d_i > 16\log(n)$:*

$$\mathbb{P}\left(w(u, \pi(u)) < \frac{3}{8}s^2\beta d_i\right) = O(n^{-\frac{3}{2}}) \tag{4.1}$$

Lemma 2 *For each $u \in V$ and $v \in V \setminus \{\pi(u)\}$, suppose $u \in C_i$. Then, if $s^2\beta d_i > 16\log(n)$ and $P_{jj} = o(1)$ for every j:*

$$\mathbb{P}\left(w(u, v) > \frac{3}{8}s^2\beta d_i\right) = O(n^{-4}) \tag{4.2}$$

Lemma 1 says that the number of witnesses obtained for a correct match is at least a constant times the average number of witnesses expected from correct matches. Independently, we also show with Lemma 2 that no incorrect match ever reaches this same number of witnesses. By the union bound, these thresholds are respected with high probability for every u and v. Therefore, $w(u, \pi(u)) > w(u, v)$ for all u and for all $v \neq u$. By the reasoning given at the beginning of the section, therefore the theorem is proved.

To show the second corollary, note that the graph generation process is equivalent to the stochastic block model with n communities. Setting the diagonal entries of P to be equal to the maximum of their rows gives us the condition $P_{jj} = o(1)$ for every j. Therefore, Lemma 1 and 2 still apply to this case.

4.1 Proofs of Lemmas

In order to prove lemmas 2 and 3, it will be convenient to express $w(u, v)$ as a sum of indicators as follows:

$$\sum_{x \in V_1 \setminus \{u\}} \mathbb{I}\{(u, x) \in E_1, (v, \hat{\pi}(x)) \in \tilde{E}_2\} \tag{4.3}$$

We will also find the following two Chernoff bounds useful (see, for example, [19]):

Bound 1 *If $X = X_1 + \ldots + X_n$, where the $\{X_i\}$ are independent random variables taking values in $\{0, 1\}$, then for $\delta \in (0, 1)$ we have:*

$$\mathbb{P}\left(X < (1 - \delta)\mathbb{E}[X]\right) \leq \exp(-\frac{\delta^2 \mathbb{E}[X]}{2}) \tag{4.4}$$

Bound 2 *If $X = X_1 + \ldots + X_n$, where the $\{X_i\}$ are independent random variables taking values in $\{0, 1\}$, then for $\delta > 1$ we have:*

$$\mathbb{P}\left(X > (1 + \delta)\mathbb{E}[X]\right) \leq \exp(-\frac{\delta \mathbb{E}[X]}{3}) \tag{4.5}$$

4.1.1 Proof of Lemma 1

Recall that in our generation of $\hat{\pi}$, first every vertex is sampled with probability β to form a subset $W \subset V$, and those vertices are matched correctly in $\hat{\pi}$. Let $A(u, x)$ be the indicator that $(u, x) \in E_1$, $(\pi(u), \pi(x)) \in E_2$, and $x \in W$. Clearly, we have:

$$w(u, \pi(u)) \geq \sum_{x \in (F_1 \cup \ldots \cup F_k) \setminus \{u\}} A(u, x)$$

After fixing u, each $A(u, x)$ is independent for all $x \neq u$. The indicators all take the value 1 with probability $\beta P_{ij} s^2$, for appropriate P_{ij}. Therefore:

$$\mathbb{E}\left[\sum_{x \in (F_1 \cup \ldots \cup F_k) \setminus \{u\}} A(u, x)\right] = s^2 \beta d_i$$

Applying Bound 2 to this, we get:

$$\mathbb{P}\left(w(u,\pi(u)) < \frac{1}{2}s^2\beta d_i\right) \leq \exp(-\frac{s^2\beta d_i}{8}) \leq \exp(-2\log(n)) = \frac{1}{n^2}$$

where we used the assumption that $s^2\beta d_i \geq 16\log(n)$.

4.1.2 Proof of Lemma 2

For this proof, assume the following procedure is used to produce $G = (V, E)$: First, create an Erdös-Rényi graph $G_{ER} = (V, E_{ER})$ with edge probability $p_{max} := \max_j P_{jj}$. Then, independently, for every edge $(u, v) \in E_{ER}$ with $u \in C_i$ and $v \in C_j$, sample the edge (u, v) with probability $P_{ij}/(p_{max})$ to obtain E. Clearly, this process is stochastically identical to the original creation process for the stochastic block model.

For this proof, we always assume $v \neq \pi(u)$ and $u \in C_i$. For now, choose an arbitrary fixed $\hat{\pi}$. This time, we decompose the sum (4.3) as

$$w(u,v) = \sum_{j\in[k]} \sum_{x\in C_j\setminus\{u\}} \mathbb{I}\left\{(u,x) \in E_1, (v,\hat{\pi}(x)) \in \tilde{E}_2\right\}$$

The indicator in the summand above can be stochastically dominated by an indicator (denoted by $B(u, v, x)$) for the following event: $(u, x) \in E$, $(v, \hat{\pi}(x)) \in E_{ER}$, (u, x) is sampled for inclusion into E_1 (with probability s), and $(v, \hat{\pi}(x))$ is sampled for inclusion into $\tilde{E}_2$ (with probability s). This indicator $B(u, v, x)$ is independent for every $x \neq u$, is 0 if $\hat{\pi}(x) = v$, and is $Ber(s^2 P_{ij}(p_{max}))$, otherwise, where $x \in C_j$. We will bound the case when $B(u, v, x) = 0$ by a Bernoulli random variable as well. We can therefore bound the sum, conditioned on our choice of $\hat{\pi}$:

$$\mathbb{P}\left(w(u,v) > (1+\delta)s^2(p_{max})d_i\right) < \exp(-\frac{\delta s^2(p_{max})d_i}{3})$$

Letting $1 + \delta = (3\beta)/(8(p_{max}))$, we get for large enough n:

$$\mathbb{P}\left(w(u,v) > \frac{3}{8}\beta s^2 d_i\right) < \exp\left(-\frac{(\frac{3\beta}{8(p_{max})} - 1)s^2(p_{max})d_i}{3}\right)$$

But $(\frac{3\beta}{8(p_{max})} - 1)(p_{max}) < \beta$, so combining that with $s^2\beta d_i > 16\log(n)$ gives us the result conditioned on $\hat{\pi}$. However, since the bound holds for every choice of $\hat{\pi}$, it also holds without conditioning on $\hat{\pi}$, and the result is proved.

CHAPTER 5

RELATIONSHIP TO PRIOR WORK

Our algorithm for approximate graph matching correction uses the concept of witnesses presented by Korula and Lattanzi in [13]. This paper first presents the idea of counting witnesses, and we use similar terminology. However, in their theoretical analysis, the seed set is assumed to be accurate. This also precludes the case where the seed set encompasses the entire graph. When these assumptions are relaxed, we showed that the algorithm can then be viewed as a sort of bipartite matching problem. Their degree bucketing approach can be viewed as a sort of greedy matching algorithm in that light. Furthermore, we give the first analysis of a witness method with highly noisy seeds.

Noisy seeds have been considered in the past for similar algorithms by Kazemi, Hassani, and Grossglauser [20]. Here, however, the algorithm that is analyzed, when modified for our use case, does not perform well on general graph models, due to the thresholding that they use for the percolation model. Their more applicable algorithm is heuristic in nature and cannot be applied directly to graph matching correction. Furthermore, unlike other papers covering similar graph matching approaches, we apply our analysis to the stochastic block model [21, 12, 14].

CHAPTER 6

SIMULATION RESULTS

We examine the performance of our graph matching correction algorithm, and present experiments to attempt to answer the following questions:

1. How does our algorithm perform on our model?
2. How does our algorithm perform for correcting seedless graph matching?
3. How does our algorithm perform for correcting with a small number of seeds?

6.1 Results for Subsampling Model

Recall that in our subsampling model, there is an initial graph G, whose edges are sampled twice with probability s to create two correlated graphs to match. Then, we are given a permutation $\hat{\pi}$ to correct, which matches the two graphs correctly on a fraction β of vertices. We apply this model to a number of graphs G, both synthetic and real-world, and evaluate the performance of our correction algorithm.

6.1.1 Synthetic Graphs

For our first experiment, we attempt to use our algorithm to correct matching errors, assuming a $\hat{\pi}$ randomly generated according to our model and assuming that G is an Erdös-Rényi graph with $n = 1000$ and $(n-1)p = 40$, randomly generated and sampled according to various values of s for each trial. The results were presented in Figure 3.3.

We also run our algorithm on a simple stochastic block model, with two communities, and edge probabilities chosen such that the average degree of

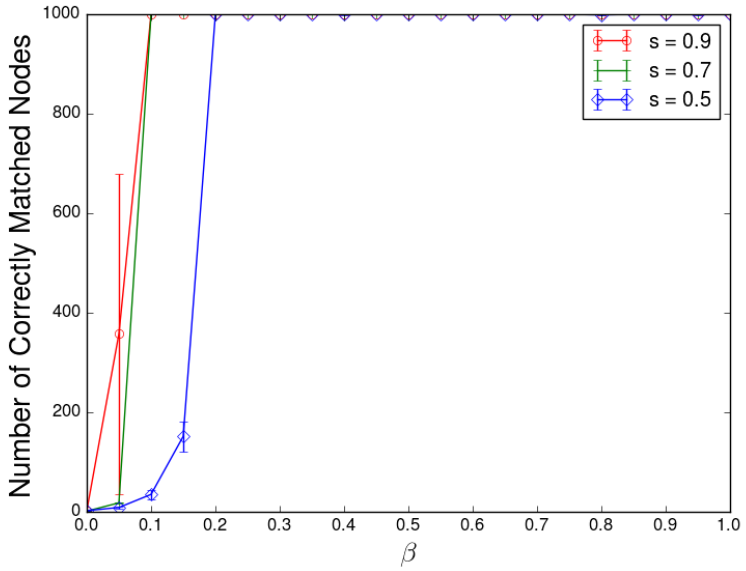

Figure 6.1: Stochastic block model graphs for varying values of β. Iterated algorithm used.

the graph is 40, and $P_{11} = P_{22} = 2P_{12}$. The performance, plotted in Figure 6.1, is similar to that of the Erdös-Rényi graph. Finally, we run our algorithm on Barabási-Albert graphs, a more realistic model for social networks, again chosen with an average degree of 40. The performance is similar, as shown in Figure 6.2. Note that when $s = 0.5$, some nodes are isolated in either G_1 or G_2 and therefore cannot be matched at all by our algorithm.

6.1.2 Real-World Graphs

All of the graphs used so far have been synthetic, but our algorithm also performs well when the subsampling model is applied to real-world graphs. For this experiment, we used a snapshot of the Slashdot social network (Figure 6.3) and a snapshot of the Epinions social network (Figure 6.4) with 77360 and 75888 nodes, respectively [22]. We sampled each edge twice with probabilities 0.5, 0.7, and 0.9, and evaluated the performance of our algorithm on our model, for various values of β.

Interestingly, for $s = 0.9$, our algorithm could sometimes match a large fraction of nodes with no seed information, making it an efficient seedless graph matching algorithm. Again, large fractions of nodes were isolated in

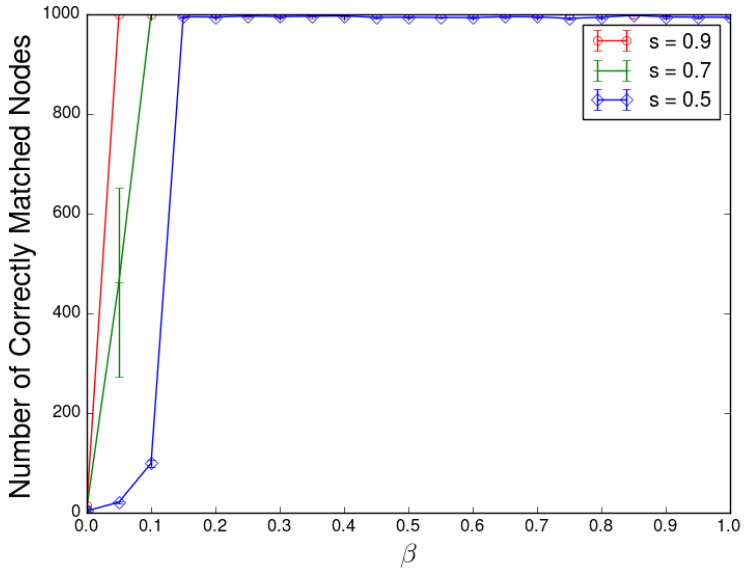

Figure 6.2: Barabási-Albert graphs for varying values of β. Iterated algorithm used.

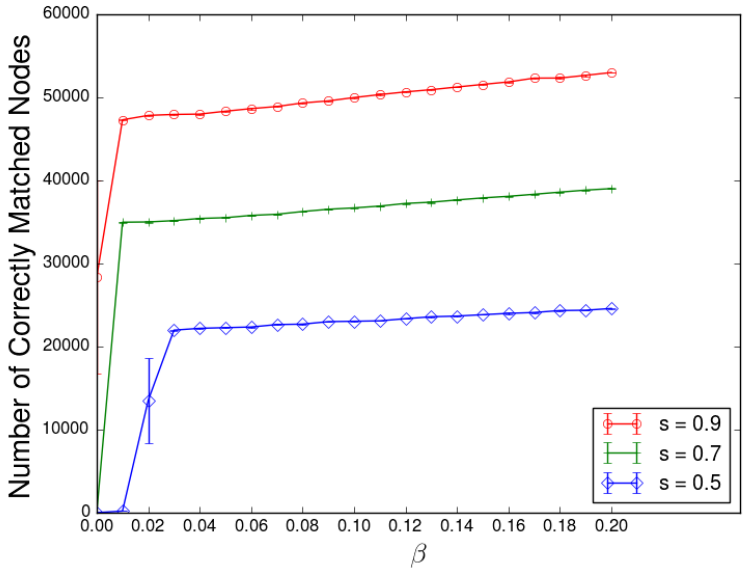

Figure 6.3: Error correction on the Slashdot social network graph for varying values of β. Iterated algorithm used.

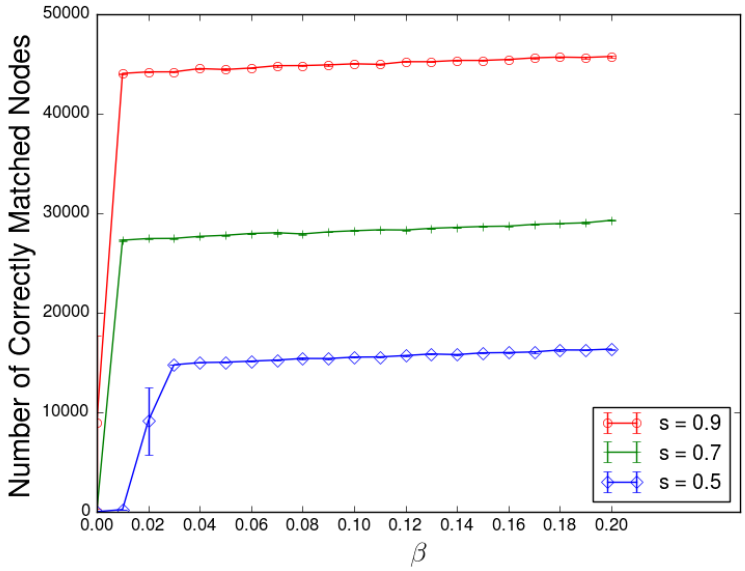

Figure 6.4: Error correction on the Epinions social network graph for varying values of β. Iterated algorithm used.

either G_1 or G_2 after edge sampling, making those nodes impossible to match using our algorithm.

6.1.3 Beyond the Subsampling Model

Naturally, real-world examples of G_1 and G_2 may not be generated by sampling edges independently from a ground-truth graph G. To show that our algorithm works even in this case, we examine the two-hop neighborhood of the article for "Earth" in both the French and German Wikipedia, as they appeared on June 20, 2017. Wikipedia maintains inter-language links, which we use as the canonical correspondence between the two graphs. We treat links between different articles as undirected edges in the graphs. For this experiment, we let G_1 and G_2 be the subgraph in each respective language containing articles which are present in both two-hop neighborhoods of "Earth." The results can be found in Figure 6.5.

We note that in many real-life graphs, it may be impossible to match all of the nodes. Generally, if there exists an automorphism (i.e., a permutation of the nodes that leaves the adjacency graph unchanged) of either G_1 or G_2 that changes the labels of some of the nodes, then such node labels cannot be

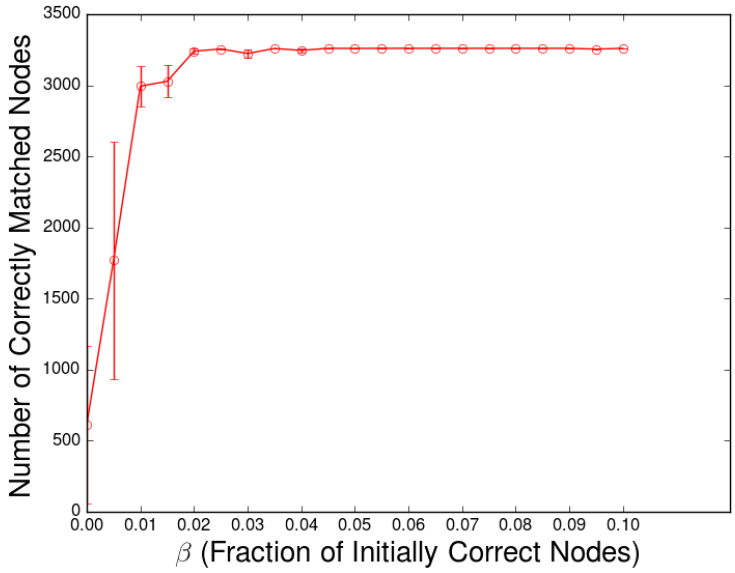

Figure 6.5: Error correction on a subgraph of the French and German Wikipedia graphs (containing 8418 articles), for varying values of β. Iterated algorithm used.

uniquely determined. For example, there could be multiple isolated nodes, or multiple degree-1 nodes with the same neighbor. In the French and German Wikipedia subgraphs, each subgraph has 8418 nodes, but we can only recover approximately 3260 of the correct correspondences. We examined one $\hat{\pi}$ obtained for $\beta = 0.1$, and verified that it is correct up to an automorphism of G_1 and G_2, so we succeeded in correctly identifying all correspondences between the two graphs which were possible to discern uniquely.

6.2 Seedless Graph Matching

One primary application of our algorithm is in correction of seedless graph matching algorithms. Such algorithms include QCV [7], PATH [7], and a Bayesian approach [10]. We test the performance of our algorithm in correcting initial matchings made by each of these algorithms on Barabási-Albert graphs. For these experiments, we use $n = 500$, as some of these algorithms are not scalable for extremely large graphs. In Table 6.1 we record the precision, defined as the fraction of matched nodes which are matched correctly. In every case, all 500 nodes are present in each matching. For QCV

Table 6.1: Average precision after various seedless matching techniques

	QCV	PATH	Bayesian	QCV + Correction
$s = 1.0$	1.00	1.00	1.00	**1.00**
$s = 0.9$	1.00	1.00	0.06	**1.00**
$s = 0.8$	0.71	0.00	0.03	**1.00**
$s = 0.7$	0.23	0.00	0.02	**1.00**
$s = 0.6$	**0.07**	0.00	0.02	**0.07**
$s = 0.5$	**0.03**	0.00	0.02	0.02

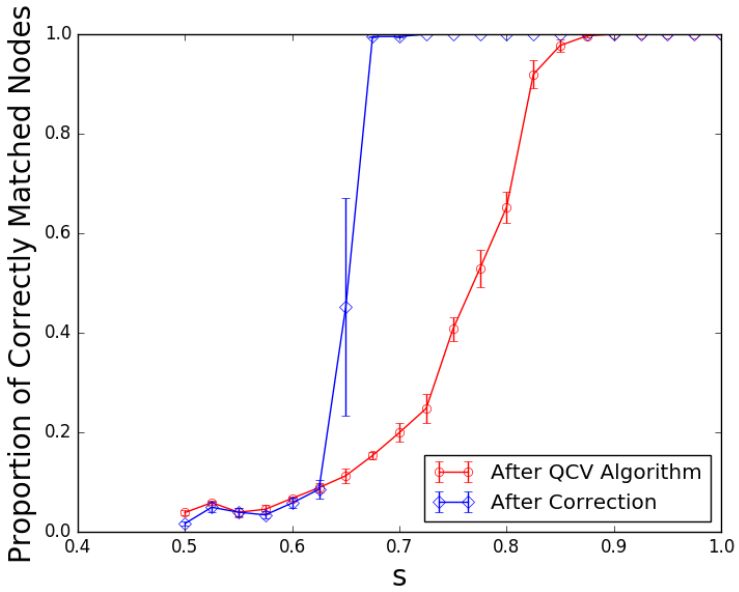

Figure 6.6: Error correction on the QCV algorithm using our algorithm

and PATH, we use the implementation from the publicly available GraphM package [7]. For the Bayesian method developed by Pedarsani et al., we use the implementation provided by SecGraph [10, 23].

Although the performance degrades rapidly for all of the seedless algorithms as s decreases, QCV still manages to match a fraction of nodes correctly, allowing our algorithm to correct the remaining errors and outperform every seedless algorithm run in isolation. Figure 6.6 shows this phenomenon for QCV on various values of s.

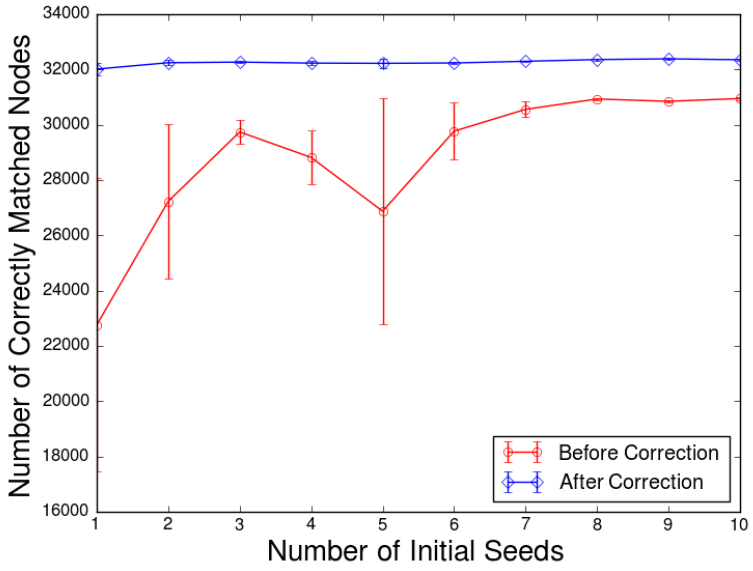

Figure 6.7: Number of correctly matched nodes on the Epinions graph (sampled with $s = 0.9$), after ExpandWhenStuck from [20] and after correction

6.3 Graph Matching with a Small Number of Seeds

In addition to seedless graph matching algorithms, we can boost the performance of some seeded algorithms. In [20], Kazemi and Grossglauser present an algorithm called ExpandWhenStuck that can perform approximate graph matching on very large graphs with just a handful of seeds. Their algorithm matches a large fraction of nodes correctly on the Epinions social networks when starting with even just one seed. However, with such a small number of seeds, the accuracy of the matching suffers. We apply our algorithm as a post-processing step, and universally improve the results when starting with ten or fewer seeds, as seen in Figure 6.7. For a fair comparison, we have restricted ourselves to matching only nodes with two or more witnesses for this experiment, as ExpandWhenStuck uses a similar technique to increase precision.

CHAPTER 7

CONCLUSIONS

Approximate graph matching is a hard problem in general, but efficient algorithms exist to solve the problem on a wide range of graphs, in practice. Sometimes, these algorithms produce sub-optimal results. However, using ideas learned from seeded graph matching algorithms, we can improve these results to produce state-of-the-art graph matching results.

Although we have only proven that our algorithm can correct a random initial distribution of errors on stochastic block model graphs, in practice, the effect seems much more general. We suspect there is a general thresholding effect: if enough initial seeds are present, all correct matches can be recovered, regardless of the initial condition, but if not enough are present, then no matches can be recovered. This implies that the greatest difficulty for approximate graph matching is to match just a few vertices. Quantifying this difficulty is an interesting direction for future research.

REFERENCES

[1] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on.* IEEE, 2008, pp. 111–125.

[2] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Security and Privacy, 2009 30th IEEE Symposium on.* IEEE, 2009, pp. 173–187.

[3] A. Toshev, J. Shi, and K. Daniilidis, "Image matching via saliency region correspondences," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on.* IEEE, 2007, pp. 1–8.

[4] G. W. Klau, "A new graph-based method for pairwise global network alignment," *BMC Bioinformatics*, vol. 10, no. 1, p. S59, 2009.

[5] E. Kazemi, H. Hassani, M. Grossglauser, and H. P. Modarres, "Proper: Global protein interaction network alignment through percolation matching," *BMC Bioinformatics*, vol. 17, no. 1, p. 527, 2016.

[6] J. W. Raymond and P. Willett, "Maximum common subgraph isomorphism algorithms for the matching of chemical structures," *Journal of Computer-Aided Molecular Design*, vol. 16, no. 7, pp. 521–533, 2002.

[7] M. Zaslavskiy, F. Bach, and J.-P. Vert, "A path following algorithm for the graph matching problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2227–2242, 2009.

[8] S. Umeyama, "An eigendecomposition approach to weighted graph matching problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 695–703, 1988.

[9] M. Gori, M. Maggini, and L. Sarti, "Exact and approximate graph matching using random walks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 7, pp. 1100–1111, 2005.

[10] P. Pedarsani, D. R. Figueiredo, and M. Grossglauser, "A Bayesian method for matching two similar graphs without seeds," in *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on.* IEEE, 2013, pp. 1598–1607.

[11] V. Lyzinski, D. E. Fishkind, and C. E. Priebe, "Seeded graph matching for correlated Erdös-Rényi graphs." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3513–3540, 2014.

[12] L. Yartseva and M. Grossglauser, "On the performance of percolation graph matching," in *Proceedings of the first ACM conference on Online social networks*. ACM, 2013, pp. 119–130.

[13] N. Korula and S. Lattanzi, "An efficient reconciliation algorithm for social networks," *Proceedings of the VLDB Endowment*, vol. 7, no. 5, pp. 377–388, 2014.

[14] C. Fabiana, M. Garetto, and E. Leonardi, "De-anonymizing scale-free social networks by percolation graph matching," in *Computer Communications (INFOCOM), 2015 IEEE Conference on*. IEEE, 2015, pp. 1571–1579.

[15] P. Pedarsani and M. Grossglauser, "On the privacy of anonymized networks," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2011, pp. 1235–1243.

[16] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *Journal of the ACM (JACM)*, vol. 34, no. 3, pp. 596–615, 1987.

[17] D. Cullina and N. Kiyavash, "Improved achievability and converse bounds for Erdös-Rényi graph matching," in *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*. ACM, 2016, pp. 63–72.

[18] W. Aiello, F. Chung, and L. Lu, "A random graph model for massive graphs," in *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*. Acm, 2000, pp. 171–180.

[19] D. P. Dubhashi and A. Panconesi, *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[20] E. Kazemi, S. H. Hassani, and M. Grossglauser, "Growing a graph matching from a handful of seeds," *Proceedings of the VLDB Endowment*, vol. 8, no. 10, pp. 1010–1021, 2015.

[21] C.-F. Chiasserini, M. Garetto, and E. Leonardi, "Impact of clustering on the performance of network de-anonymization," in *Proceedings of the 2015 ACM on Conference on Online Social Networks*. ACM, 2015, pp. 83–94.

[22] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, June 2014.

[23] S. Ji, W. Li, P. Mittal, X. Hu, and R. A. Beyah, "Secgraph: A uniform and open-source evaluation system for graph data anonymization and de-anonymization," in *USENIX Security Symposium*, 2015, pp. 303–318.