2017 IEEE International Conference on Data Mining

iNEAT: Incomplete Network Alignment

Si Zhang, Hanghang Tong
Arizona State University
{szhan172, hanghang.tong}@asu.edu

Jie Tang
Tsinghua University
jietang@tsinghua.edu.cn

Jiejun Xu
HRL Laboratories
jxu@hrl.com

Wei Fan
Baidu Big Data Lab
fanwei03@baidu.com

***Abstract*—Network alignment and network completion are two fundamental cornerstones behind many high-impact graph mining applications. The state-of-the-arts have been addressing these tasks *in parallel*. In this paper, we argue that network alignment and completion are inherently complementary with each other, and hence propose to jointly address them so that the two tasks can benefit from each other. We formulate it from the optimization perspective, and propose an effective algorithm (iNEAT) to solve it. The proposed method offers two distinctive advantages. First (*Alignment accuracy*), our method benefits from higher-quality input networks while mitigates the effect of incorrectly inferred links introduced by the completion task itself. Second (*Alignment efficiency*), thanks to the low-rank structure of the complete networks and alignment matrix, the alignment can be significantly accelerated. The extensive experiments demonstrate the performance of our algorithm.**

1. Introduction

In the era of big data, networks are often *incomplete* (i.e., veracity) and *multi-sourced* (i.e., variety), e.g., transaction networks from multiple financial institutes [1]. As such, network alignment (i.e., to find node correspondence across multiple networks) and network completion (i.e., to infer the missing links) become two key tasks in many graph mining applications. Although the multi-sourced and incomplete characteristics often *co-exist* in many real networks, the state-of-the-arts have been largely addressing them *in parallel*. That is, most existing network alignment methods have implicitly assumed the topology of the input networks for alignment are perfectly known apriori [2], whereas the existing network completion methods admit either a single network (i.e., matrix completion [3]) or multiple aligned networks (e.g., tensor completion [4]). How can we align two networks when one or both of them have missing edges?

A natural choice could be *completion-then-alignment*. That is, we first run network completion task on each of the two input networks separately, and then align the two complete networks. However, this strategy has some fundamental limits. First (*Alignment accuracy*), the promise of the *completion-then-alignment* strategy lies in that by inferring the missing links, it would provide higher-quality input networks for the alignment task. However, the completion task itself might introduce noisy links, which might compromise, or even outweight the benefits of the correctly inferred missing links for the alignment task. Second (*Alignment efficiency*), the network alignment alone is already computationally costly. Most of the existing methods (even with approximation, such as [5]) have a time/space complexity that is at least $O(n^2)$, where n is the number of nodes of the input networks, mainly due to the computation/storage of the sparse matrix-matrix multiplication between the input adjacency matrices and the alignment matrix. By inferring the missing links, network completion would make each input network denser. If we simply conduct the network alignment task on such densified networks, it might make the computation even more intensive.

To address these limitations, we hypothesize that network alignment and network completion are inherently complementary with each other due to the following reasons. First, *(H1) alignment helps completion*. If two nodes in two networks are aligned together, intuitively, they might share similar connectivity patterns. Therefore, the knowledge about the existence or absence of links in one network could help inferring the missing links in another network via alignment. Second, *(H2) completion helps alignment*. As mentioned above, network completion could potentially improve the alignment accuracy by providing higher-quality input networks. Moreover, network completion itself implicitly assumes a low-rank structure of input networks, which, if harnessed appropriately, will actually *accelerate* the alignment process as we will show in the paper.

Armed with these hypotheses, we propose to *jointly* address network alignment and network completion problems. We formulate it as an optimization problem with the following two key ideas. First, in order to leverage alignment for the completion task, we impose the low-rank structure on the underlying (true) network, which matches not only the observed links of the corresponding network, but also the *auxiliary observations* from the other network via the alignment matrix. Second, in order to leverage the network completion for the alignment, we recast the network alignment problem via the low-rank structures of *complete* networks, which not only improves the alignment accuracy, but also reduces the alignment to *linear*.

The rest of the paper is organized as follows. Section 2 defines the incomplete network alignment problem and provides the preliminaries. Section 3 presents the proposed optimization formulation and Section 4 gives its optimization algorithm. Section 5 presents some experimental results. Related work and conclusion are given in Section 6 and 7.

2. Problem Definition and Preliminaries

2374-8486/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDM.2017.160

2.1. Problem Definition

Table 1 summarizes the main symbols and notations used throughout the paper. We use bold uppercase letters for matrices (e.g., $\mathbf{A}$), bold lowercase letters for vectors (e.g., $\mathbf{s}$), and lowercase letters (e.g., α) for scalars. We use $\mathbf{A}(i,j)$ to denote the entry at the intersection of the i-th row and j-th column of the matrix $\mathbf{A}$. We denote the transpose of a matrix by a superscript T (e.g., $\mathbf{A}^T$ is the transpose of $\mathbf{A}$). The vectorization of a matrix (in the column order) is denoted by $\text{vec}(\cdot)$, and the result vector is denoted by the corresponding bold lowercase letter (e.g., $\mathbf{s} = \text{vec}(\mathbf{S})$). Equivalently, the transformation of a vector to its corresponding matrix is denoted by a de-vectorization operator $\text{mat}(\cdot)$ (e.g., $\mathbf{S} = \text{mat}(\mathbf{s})$).

TABLE 1: Symbols and Notations

Symbols	Definition
$\mathcal{G}_1$, $\mathcal{G}_2$	incomplete networks
$\mathbf{A}_1, \mathbf{A}_2$	two adjacency matrices of $\mathcal{G}_1$ and $\mathcal{G}_2$
n_1, n_2	# of nodes in $\mathbf{A}_1$ and $\mathbf{A}_2$
$\mathbf{S}$	an $n_2 \times n_1$ alignment matrix between $\mathbf{A}_2$ and $\mathbf{A}_1$
$P_\Omega(\cdot), P_{\bar{\Omega}}(\cdot)$	an operator to project only to observed (unobserved) entries
$\mathbf{U}_1, \mathbf{V}_1, \mathbf{U}_2, \mathbf{V}_2$	low rank factorizations of $\mathbf{A}_1$ and $\mathbf{A}_2$
$\mathbf{P}_{\Omega_1}, \mathbf{P}_{\Omega_2}$	projection matrix, all 1s at all observed entries
$\mathbf{1}_1, \mathbf{1}_2$	1s vectors of length n_1 and n_2 respectively
λ, γ, β	parameters
$\text{Tr}[\cdot]$	trace operator
$\text{diag}(\cdot)$	diagonal matrix of a vector
$\text{vec}(\cdot)$, $\text{mat}(\cdot)$	vectorization and de-vectorization operator
$\text{eig}(\cdot)$	eigenvalues of a matrix

Many real-world networks are incomplete. That is, we only have knowledge about the existence (i.e., a value of 1) or absence (i.e., a value of 0) of certain entries (denoted by the set Ω) of its adjacency matrix. For the rest entries in the adjacency matrix, we do not know if the corresponding links exist or not. Figure 1 presents an illustrative example. As we can see in Figure 1(a), the nodes $(1, 2, 3, 4)$ in the first incomplete network have similar topology to the nodes $(6', 7', 8', 9')$, possibly leading to a wrong alignment result that these two sets of nodes are aligned within each other. However, the complete networks in Figure 1(b) (by filling all the red lines) are identical. Thus, the nodes $(1, 2, 3, 4)$ can be aligned to nodes $(1', 2', 3', 4')$ respectively, so can the rest of nodes. On the other hand, by completing two networks separately, noisy edges might be incorrectly added (e.g., edge $(4, 6)$) and the true network structure would fail to be recovered. The incorrectly recovered networks may further mislead the alignment results. Therefore, how to align incomplete networks while completing them is the key challenge this paper aims to address.

Problem 1. Incomplete Network Alignment.

Given: *(1) adjacency matrices* $\mathbf{A}_1$,$\mathbf{A}_2$ *of incomplete networks, (2) prior node similarity matrix* $\mathbf{H}$ *across networks.*

Output: *(1) the* $n_2 \times n_1$ *alignment matrix* $\mathbf{S}$*, where* $\mathbf{S}(x, a)$ *represents to what extent node-*a *in* $\mathcal{G}_1$ *is aligned with node-*x *in* $\mathcal{G}_2$*, and (2) complete adjacency matrices* $\mathbf{A}_1^*$*, and* $\mathbf{A}_2^*$.

2.2. Preliminaries

A - Network Alignment. Most existing network alignment algorithms (such as *IsoRank* [6] and *FINAL* [5]), explicitly or implicitly, are based on the *topology consistency* principle. Take *FINAL* as an example, the topology consistency

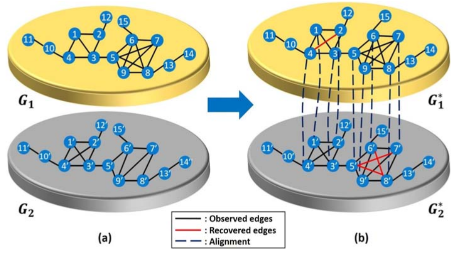


Figure 1: An illustrative example. Figure 1(a) shows the input incomplete networks and Figure 1(b) shows part of the alignment across two complete networks.

principle can be stated as follows[1]. Given two pairs of nodes, say (1) node-a in $\mathcal{G}_1$ and node-x in $\mathcal{G}_2$ and (2) node-b in $\mathcal{G}_1$ and node-y in $\mathcal{G}_2$, if nodes a and b are close neighbors and so are nodes x and y, it assumes the similarity between a and x, and that between their neighbors b and y to be consistent, i.e., small $[\hat{\mathbf{S}}(a,x) - \hat{\mathbf{S}}(b,y)]^2 \mathbf{A}_1(a,b)\mathbf{A}_2(x,y)$, where $\hat{\mathbf{S}}$ is the similarity matrix. Mathematically, this naturally leads to the following optimization problem.

$$\min_{\hat{\mathbf{s}}} \quad \alpha \hat{\mathbf{s}}^T (\mathbf{D} - \mathbf{A}_1 \otimes \mathbf{A}_2)\hat{\mathbf{s}} + (1-\alpha)\|\mathbf{D}\hat{\mathbf{s}} - \mathbf{h}\|_F^2 \tag{1}$$

where $\hat{\mathbf{s}}, \mathbf{h}$ are the vectorization of the similarity matrix $\hat{\mathbf{S}}$ and the prior node similarity matrix $\mathbf{H}$, $\mathbf{D} = \mathbf{D}_1 \otimes \mathbf{D}_2$ and $\mathbf{D}_1, \mathbf{D}_2$ are the degree matrix of $\mathbf{A}_1, \mathbf{A}_2$. Note that instead of using $\hat{\mathbf{S}}$ to infer the alignment as in [5], we use a scaled similarity matrix $\mathbf{S}$ as alignment matrix in our paper where $\mathbf{S}$ is the matrix form of $\mathbf{s} = \mathbf{D}\hat{\mathbf{s}}$ (i.e., $\mathbf{S} = \text{mat}(\mathbf{D}\hat{\mathbf{s}})$). In other words, the entries in the alignment matrix $\mathbf{S}$ measure to what extent two corresponding nodes are aligned together. The second regularization term is to avoid trivial solutions.

B - Network Completion. As mentioned earlier, incomplete networks might have many unobserved missing edges, which could significantly change the true network structure and hence mislead the topology-based network alignment. One straightforward way to address this issue is by using matrix completion. Most of the existing matrix completion methods are centered around minimizing the nuclear norm of the matrix. In [7], the authors show that the nuclear norm $\|\mathbf{A}_1\|_* = \min_{\mathbf{U}_1,\mathbf{V}_1} \frac{1}{2}(\|\mathbf{U}_1\|_F^2 + \|\mathbf{V}_1\|_F^2)$ where $\mathbf{A}_1 = \mathbf{U}_1\mathbf{V}_1^T$, which allows the factorization-based completion methods. Thus, we can recover the complete networks by minimizing

$$\begin{aligned} & J_1(\mathbf{U}_1, \mathbf{V}_1, \mathbf{U}_2, \mathbf{V}_2) \\ &= \frac{1}{2}\|P_{\Omega_1}(\mathbf{A}_1 - \mathbf{U}_1\mathbf{V}_1^T)\|_F^2 + \frac{\lambda}{2}(\|\mathbf{U}_1\|_F^2 + \|\mathbf{V}_1\|_F^2) \\ &+ \frac{1}{2}\|P_{\Omega_2}(\mathbf{A}_2 - \mathbf{U}_2\mathbf{V}_2^T)\|_F^2 + \frac{\lambda}{2}(\|\mathbf{U}_2\|_F^2 + \|\mathbf{V}_2\|_F^2) \end{aligned} \tag{2}$$

where the operator P_{Ω_1} projects the value to the observed set Ω_1 of $\mathbf{A}_1$, e.g., $P_{\Omega_1}((\mathbf{U}_1\mathbf{V}_1^T)(i,j)) = (\mathbf{U}_1\mathbf{V}_1^T)(i,j)$, $\forall(i,j) \in \Omega_1$, otherwise 0; and P_{Ω_2} is defined similarly.

3. Proposed Optimization Formulation

In this section, we present the proposed optimization formulation to solve Problem 1. First, we present how to formulate the alignment task and prove the low-rank structure of alignment matrix. Then we present how to leverage the alignment matrix to infer missing edges across networks.

1. In [5], the authors generalize the topology consistency to further accommodate node/edge attributes, which is outside the scope of this paper.

3.1. Network Completion Helps Alignment

We use the networks recovered by Eq (2) as the input networks whose adjacency matrices are of rank-r, i.e., $\mathbf{A}_1^* = \mathbf{U}_1\mathbf{V}_1^T$ and $\mathbf{A}_2^* = \mathbf{U}_2\mathbf{V}_2^T$. We adopt Eq. (1) to perform network alignment. Due to the potential asymmetry of $\mathbf{A}_1^*$ and $\mathbf{A}_2^*$, based on the *topology consistency* (i.e., small $[\hat{\mathbf{S}}(a,x) - \hat{\mathbf{S}}(b,y)]^2\mathbf{A}_1^*(a,b)\mathbf{A}_2^*(x,y)$ across two directed networks), the optimization problem is formulated as

$$\min_{\hat{\mathbf{s}}} \ \alpha\hat{\mathbf{s}}^T(\hat{\mathbf{D}} - \mathbf{A}_1^* \otimes \mathbf{A}_2^*)\hat{\mathbf{s}} + (1-\alpha)\|\hat{\mathbf{D}}\hat{\mathbf{s}} - \mathbf{h}\|_F^2 \quad (3)$$

where $\hat{\mathbf{D}} = \frac{\mathbf{D}_1\otimes\mathbf{D}_2+\hat{\mathbf{D}}_1\otimes\hat{\mathbf{D}}_2}{2}$, $\mathbf{D}_1 = \text{diag}(\mathbf{U}_1\mathbf{V}_1^T\mathbf{1}_1)$ and $\hat{\mathbf{D}}_1 = \text{diag}(\mathbf{1}_1^T\mathbf{U}_1\mathbf{V}_1^T)$ are the outdegree and indegree matrix of $\mathbf{A}_1^*$, respectively. $\mathbf{D}_2$ and $\hat{\mathbf{D}}_2$ are defined similarly.

However, directly solving the above problem requires at least $O(n^2)$ time complexity. To address this issue, we give the following lemma, which states the alignment matrix $\mathbf{S}$ under the *topology consistency* (i.e., Eq. (3)) intrinsically consists of a *low-rank* structure, thanks to the low-rank structure of two complete adjacency matrices.

Lemma 1. Low-Rank Structure of the Alignment Matrix S. *Let $\hat{\mathbf{s}}$ be the solution of Eq. (3) where $\mathbf{A}_1^* = \mathbf{U}_1\mathbf{V}_1^T$ and $\mathbf{A}_2^* = \mathbf{U}_2\mathbf{V}_2^T$ are complete rank-r_1 and rank-r_2 adjacency matrices. Let the alignment matrix $\mathbf{S}$ be the scaled similarity matrix $\mathbf{S} = mat(\hat{\mathbf{D}}\hat{\mathbf{s}})$ and $\mathbf{H}$ be the prior similarity matrix, then if $\alpha < 0.5$, the alignment matrix can be expressed as $\mathbf{S} = \alpha\mathbf{U}_2\mathbf{M}\mathbf{U}_1 + (1-\alpha)\mathbf{H}$ where $\mathbf{M}$ is an $r_2 \times r_1$ matrix and r_1, r_2 are the ranks of $\mathbf{A}_1^*$ and $\mathbf{A}_2^*$, respectively.*

Proof. The closed-form solution of Eq. (3) is computed by

$$\hat{\mathbf{s}} = (1-\alpha)\hat{\mathbf{D}}^{-1}\mathbf{h} + \alpha(1-\alpha)\hat{\mathbf{D}}^{-1}\mathbf{U}\mathbf{\Lambda}^{-1}\mathbf{V}^T\hat{\mathbf{D}}^{-1}\mathbf{h} \quad (4)$$

where $\mathbf{U} = \mathbf{U}_1 \otimes \mathbf{U}_2$, $\mathbf{V} = \mathbf{V}_1 \otimes \mathbf{V}_2$, $\mathbf{\Lambda} = \mathbf{I} - \alpha\mathbf{V}^T\hat{\mathbf{D}}^{-1}\mathbf{U}$.

First, we rewrite $\mathbf{\Lambda}^{-1}$ as follows. Since for any two matrices $\mathbf{X}, \mathbf{Y}$, the eigenvalues of their product satisfies $\text{eig}(\mathbf{XY}) = \text{eig}(\mathbf{YX})$ [8], we obtain

$$\begin{aligned}|\text{eig}(\mathbf{V}^{\hat{T}}\mathbf{D}^{-1}\mathbf{U})| &= |\text{eig}(\mathbf{U}\mathbf{V}^T\hat{\mathbf{D}}^{-1})| \leq |\text{eig}(2\mathbf{U}\mathbf{V}^T(\mathbf{D}_1 \otimes \mathbf{D}_2)^{-1}) \quad | \\ &= 2|\text{eig}((\mathbf{U}_1\mathbf{V}_1^T\mathbf{D}_1^{-1}) \otimes (\mathbf{U}_2\mathbf{V}_2^T\mathbf{D}_2^{-1}))|\end{aligned}$$

Here, the term $\mathbf{U}_1\mathbf{V}_1^T\mathbf{D}_1^{-1}$ represents a weighted directed network whose adjacency matrix has eigenvalues within $(-1, 1)$, so as $\mathbf{U}_2\mathbf{V}_2^T\mathbf{D}_2^{-1}$. Thus, if $\alpha < 0.5$, based on the spectrum property of Kronecker product, we have

$$2\alpha|\text{eig}((\mathbf{U}_1\mathbf{V}_1^T\mathbf{D}_1^{-1}) \otimes (\mathbf{U}_2\mathbf{V}_2^T\mathbf{D}_2^{-1}))| < 1 \quad (5)$$

Then, we can use Taylor expansion on $\mathbf{\Lambda}^{-1}$ as

$$\mathbf{\Lambda}^{-1} = \sum_{k=0}^{\infty}(2\alpha)^k[\mathbf{V}^T(2\hat{\mathbf{D}})^{-1}\mathbf{U}]^k \quad (6)$$

Next, we rewrite $(2\hat{\mathbf{D}})^{-1}$ as follows. Denote $\bar{\mathbf{D}}_1 = \mathbf{D}_1 + \hat{\mathbf{D}}_1$ and $\bar{\mathbf{D}}_2 = \mathbf{D}_2 + \hat{\mathbf{D}}_2$, we have

$$\begin{aligned}(2\hat{\mathbf{D}})^{-1} &= (\mathbf{D}_1 \otimes \mathbf{D}_2 + \hat{\mathbf{D}}_1 \otimes \hat{\mathbf{D}}_2)^{-1} \\ &= [\mathbf{I} - (\bar{\mathbf{D}}_1^{-1} \otimes \bar{\mathbf{D}}_2^{-1})(\mathbf{D}_1 \otimes \hat{\mathbf{D}}_2 + \hat{\mathbf{D}}_1 \otimes \mathbf{D}_2)]^{-1}(\bar{\mathbf{D}}_1^{-1} \otimes \bar{\mathbf{D}}_2^{-1}) \\ &= \sum_{j=0}^{\infty}[(\bar{\mathbf{D}}_1^{-1}\mathbf{D}_1) \otimes (\bar{\mathbf{D}}_2^{-1}\hat{\mathbf{D}}_2) + (\bar{\mathbf{D}}_1^{-1}\hat{\mathbf{D}}_1) \otimes (\bar{\mathbf{D}}_2^{-1}\mathbf{D}_2)]^j \\ &= \sum_{j=0}^{\infty}\sum_{i=0}^{j}\binom{j}{i}[(\bar{\mathbf{D}}_1^{-1}\mathbf{D}_1)^i(\bar{\mathbf{D}}_1^{-1}\hat{\mathbf{D}}_1)^{j-i}] \otimes [(\bar{\mathbf{D}}_2^{-1}\hat{\mathbf{D}}_2)^i(\bar{\mathbf{D}}_2^{-1}\mathbf{D}_2)^{j-i}]\end{aligned}$$

Thus, Eq. (6) can be further derived as

$$\begin{aligned}\mathbf{\Lambda}^{-1} = \sum_{k=0}^{\infty}\sum_{j=0}^{\infty}\sum_{i=0}^{j}(2\alpha)^k\binom{j}{i}^k & [\mathbf{V}_1^T(\bar{\mathbf{D}}_1^{-1}\mathbf{D}_1)^i(\bar{\mathbf{D}}_1^{-1}\hat{\mathbf{D}}_1)^{j-i}\mathbf{U}_1]^k \\ & \otimes [\mathbf{V}_2^T(\bar{\mathbf{D}}_2^{-1}\hat{\mathbf{D}}_2)^i(\bar{\mathbf{D}}_2^{-1}\mathbf{D}_2)^{j-i}\mathbf{U}_2]^k \quad (7)\end{aligned}$$

Denote $\mathbf{s} = \hat{\mathbf{D}}\hat{\mathbf{s}}$ and $\hat{\mathbf{h}} = \hat{\mathbf{D}}^{-1}\mathbf{h}$. Due to $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})\text{vec}(\mathbf{B})$, by substituting Eq. (7) into Eq. (4), we obtain the alignment matrix $\mathbf{S} = \text{mat}(\hat{\mathbf{D}}\hat{\mathbf{s}})$ as

$$\mathbf{S} = \alpha\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T + (1-\alpha)\mathbf{H} \quad (8)$$

where $\mathbf{M}$ is an $r_2 \times r_1$ matrix and is computed by

$$\begin{aligned}\mathbf{M} = \sum_{k=0}^{\infty}\sum_{j=0}^{\infty}\sum_{i=0}^{j}2^k\alpha^k\binom{j}{i}^k[\mathbf{V}_2^T(\bar{\mathbf{D}}_2^{-1}\hat{\mathbf{D}}_2)^i(\bar{\mathbf{D}}_2^{-1}\mathbf{D}_2)^{j-i}\mathbf{U}_2]^k \quad (9) \\ \times (1-\alpha)\mathbf{V}_2^T\hat{\mathbf{H}}\mathbf{V}_1[\mathbf{U}_1^T(\bar{\mathbf{D}}_1^{-1}\mathbf{D}_1)^i(\bar{\mathbf{D}}_1^{-1}\hat{\mathbf{D}}_1)^{j-i}\mathbf{V}_1]^k \quad \square\end{aligned}$$

Remarks. In practice, the matrix $\mathbf{H}$ in Eq. (8) is either low-rank (e.g., a rank-one uniform matrix) or very sparse. This naturally leads to the following effective strategy. First, we temporarily treat the low-rank structure part as the alignment matrix to be solved in the optimization problems (i.e., $\mathbf{S} \approx \mathbf{U}_2\mathbf{M}\mathbf{U}_1$). We then can calibrate the result by averaging between the learned $\mathbf{S}$ and the prior knowledge $\mathbf{H}$, i.e., $\mathbf{S} \leftarrow (1-\alpha)\mathbf{H} + \alpha\mathbf{S}$. A direct benefit of this strategy is that we can reduce the overall complexity to be *linear*.

To take advantages of the low-rank structure of $\mathbf{S}$ under the above strategy, instead of minimizing the similarity matrix $\hat{\mathbf{S}}$ in Eq. (3), we alternatively optimize the topology consistency on the low-rank structure of alignment matrix $\mathbf{S} = \mathbf{U}_2\mathbf{M}\mathbf{U}_1$ without the second regularization term, i.e., minimizing $\mathbf{s}^T(\hat{\mathbf{D}} - \mathbf{A}_1^* \otimes \mathbf{A}_2^*)\mathbf{s}$. By $\text{vec}(\mathbf{A})^T\text{vec}(\mathbf{B}) = \text{Tr}(\mathbf{A}^T\mathbf{B})$ and $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})\text{vec}(\mathbf{B})$, network alignment across complete networks can be formulated as

$$\begin{aligned}& J_2(\mathbf{U}_1, \mathbf{V}_1, \mathbf{U}_2, \mathbf{V}_2, \mathbf{M}) \\ &= \frac{\gamma}{2}\mathbf{s}^T\text{vec}(\mathbf{D}_2\mathbf{S}\mathbf{D}_1 + \hat{\mathbf{D}}_2\mathbf{S}\hat{\mathbf{D}}_1) + \gamma\mathbf{s}^T\text{vec}(\mathbf{U}_2\mathbf{V}_2^T\mathbf{S}\mathbf{V}_1\mathbf{U}_1^T) \\ &= \frac{\gamma}{2}\text{Tr}(\mathbf{D}_2\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T\mathbf{D}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T + \hat{\mathbf{D}}_2\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T\hat{\mathbf{D}}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T) \\ &- \gamma\text{Tr}(\mathbf{U}_2\mathbf{V}_2^T\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T\mathbf{V}_1\mathbf{U}_1^T\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T) \quad (10)\end{aligned}$$

3.2. Network Alignment Helps Completion

In some applications, the information of a single network alone might be insufficient to correctly infer missing edges. Meanwhile, since the aligned nodes are likely to share similar connectivity patterns, the observed existing edges in one network could potentially help recover missing edges in the other network via alignment. To be specific, if node-a in $\mathcal{G}_1$ and node-x in $\mathcal{G}_2$ are aligned together, and the neighbor of x (say node-y) is aligned with node-b that is unobserved to connect with node-a, the completion based on the single network information might not succeed inferring this missing link. However, the facts that (1) a and x are aligned, (2) b and y are aligned, and (3) there is an edge between x and y might provide an *auxiliary confidence* about the edge existence between a and b. We can estimate such auxiliary confidence of the edge existence as

$$\mathbf{A}_1^*(a,b) \approx \sum_{x,y}^{n_2}\mathbf{S}(a,x)\mathbf{S}(b,y)\mathbf{A}_2(x,y) = (\mathbf{S}^T\mathbf{A}_2\mathbf{S})(a,b) \quad (11)$$

where $\mathbf{S} = \mathbf{U}_2\mathbf{M}\mathbf{U}_1^T$ is the alignment matrix learned from the topology consistency. In our experiments, we find that such auxiliary confidence is most powerful to estimate the existence/absence of an edge (a, b) when such an edge itself is not observed in $\mathcal{G}_1$ (i.e.,$(a, b) \in \bar{\Omega}_1$). Mathematically, this can be formulated as the following objective function.

$$
\begin{aligned}
&J_3(\mathbf{U}_1,\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2,\mathbf{M})\\
&=\frac{\beta}{2}\|P_{\bar{\Omega}_1}(\mathbf{U}_1\mathbf{V}_1^T-\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{A}_2\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T)\|_F^2\\
&+\frac{\beta}{2}\|P_{\bar{\Omega}_2}(\mathbf{U}_2\mathbf{V}_2^T-\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T\mathbf{A}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T)\|_F^2
\end{aligned}
\tag{12}
$$

where $\bar{\Omega}_1$ and $\bar{\Omega}_2$ are the unobserved set of $\mathbf{A}_1$ and $\mathbf{A}_2$.

3.3. Overall Objective Function

We impose the nonnegativity constraints on all the variables $\mathbf{U}_1,\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2,\mathbf{M}$ to guarantee the matrices $\mathbf{A}_1^*,\mathbf{A}_2^*,\mathbf{S}$ to be nonnegative. Combining Eq. (2), Eq. (10) and Eq. (12) together, the overall optimization problem is

$$
\begin{aligned}
\min_{\mathbf{U}_1,\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2,\mathbf{M}}\quad & J(\mathbf{U}_1,\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2,\mathbf{M})=J_1+J_2+J_3\\
\text{s.t}\quad & \mathbf{U}_1,\mathbf{U}_2,\mathbf{V}_1,\mathbf{V}_2,\mathbf{M}\geq\mathbf{0}
\end{aligned}
\tag{13}
$$

4. Proposed Optimization Algorithm

Since the overall objective function Eq. (13) is not jointly convex, we optimize it by block coordinate descent, i.e., alternatively minimizing w.r.t one variable group while fixing the others until convergence. Due to the space limitation, we only show the minimization procedures over $\mathbf{U}_1$. Other variables $\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2,\mathbf{M}$ can be solved similarly as $\mathbf{U}_1$. The derivative of Eq. (2) w.r.t $\mathbf{U}_1$ is computed by

$$\frac{\partial J_1}{\partial \mathbf{U}_1}=\mathbf{X}_1-\mathbf{Y}_1 \tag{14}$$

where
$$
\begin{aligned}
\mathbf{X}_1&=[\mathbf{P}_{\Omega_1}\odot(\mathbf{U}_1\mathbf{V}_1^T)]\mathbf{V}_1+\lambda\mathbf{U}_1\\
\mathbf{Y}_1&=(\mathbf{P}_{\Omega_1}\odot\mathbf{A}_1)\mathbf{V}_1
\end{aligned}
$$

and $\mathbf{P}_{\Omega_1}(i,j)=1$ for $(i,j)\in\Omega_1$, otherwise $\mathbf{P}_{\Omega_1}(i,j)=0$.

The derivative of Eq. (10) over $\mathbf{U}_1$ is computed by

$$\frac{\partial J_2}{\partial \mathbf{U}_1}=\mathbf{X}_2-\mathbf{Y}_2 \tag{15}$$

where

$$
\begin{aligned}
\mathbf{X}_2=&\frac{\gamma}{2}[(\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{D}_2\mathbf{U}_2\mathbf{M})\odot\mathbf{U}_1]\mathbf{1}_{r_1}\mathbf{1}_1^T\mathbf{V}_1\\
&+\frac{\gamma}{2}\mathbf{1}_1\mathbf{1}_{r_1}^T[(\mathbf{M}^T\mathbf{U}_2^T\hat{\mathbf{D}}_2\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T)\odot\mathbf{U}_1^T]\mathbf{V}_1\\
&+\gamma(\mathbf{D}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{D}_2\mathbf{U}_2\mathbf{M}+\hat{\mathbf{D}}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\hat{\mathbf{D}}_2\mathbf{U}_2\mathbf{M})\\
\mathbf{Y}_2=&\gamma\mathbf{V}_1\mathbf{U}_1^T\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{U}_2\mathbf{V}_2^T\mathbf{U}_2\mathbf{M}\\
&+\gamma\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{U}_2\mathbf{V}_2^T\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T\mathbf{V}_1\\
&+\gamma\mathbf{U}_1\mathbf{V}_1^T\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{V}_2\mathbf{U}_2^T\mathbf{U}_2\mathbf{M}
\end{aligned}
$$

And the derivative of Eq. (12) over $\mathbf{U}_1$ is

$$\frac{\partial J_3}{\partial \mathbf{U}_1}=\mathbf{X}_3-\mathbf{Y}_3 \tag{16}$$

where

$$
\begin{aligned}
\mathbf{X}_3=&2\beta[\mathbf{P}_{\bar{\Omega}_1}\odot(\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{A}_2\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T)]\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{A}_2\mathbf{U}_2\mathbf{M}\\
&+2\beta\mathbf{A}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T[\mathbf{P}_{\bar{\Omega}_2}\odot(\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T\mathbf{A}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T)]\mathbf{U}_2\mathbf{M}\\
&+\beta[\mathbf{P}_{\bar{\Omega}_1}\odot(\mathbf{U}_1\mathbf{V}_1^T)]\mathbf{V}_1\\
\mathbf{Y}_3=&\beta[\mathbf{P}_{\bar{\Omega}_1}\odot(\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{A}_2\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T)]\mathbf{V}_1\\
&+\beta[\mathbf{P}_{\bar{\Omega}_1}\odot(\mathbf{U}_1\mathbf{V}_1^T+\mathbf{V}_1\mathbf{U}_1^T)]\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T\mathbf{A}_2\mathbf{U}_2\mathbf{M}\\
&+\beta\mathbf{A}_1\mathbf{U}_1\mathbf{M}^T\mathbf{U}_2^T[\mathbf{P}_{\bar{\Omega}_2}\odot(\mathbf{U}_2\mathbf{V}_2^T+\mathbf{V}_2\mathbf{U}_2^T)]\mathbf{U}_2\mathbf{M}
\end{aligned}
$$

and matrix $\mathbf{P}_{\bar{\Omega}_2}(i,j)=1$ for any $(i,j)\notin\Omega_2$.

A fixed-point solution of $\frac{\partial J}{\partial \mathbf{U}_1}=\mathbf{0}$ under nonnegativity constraint of $\mathbf{U}_1$ leads to the multiplicative update rule

$$\mathbf{U}_1(u,v)\leftarrow\mathbf{U}_1(u,v)\sqrt[4]{\frac{\mathbf{Y}_1(u,v)+\mathbf{Y}_2(u,v)+\mathbf{Y}_3(u,v)}{\mathbf{X}_1(u,v)+\mathbf{X}_2(u,v)+\mathbf{X}_3(u,v)}} \tag{17}$$

Initialization. Since the optimization problem in Eq. (13) is not a joint convex problem, a good initialization could play an important role of obtaining a good solution. For $\mathbf{U}_1$ and $\mathbf{U}_2$, we initialize them by solving the symmetric nonnegative matrix factorization of $\mathbf{A}_1$ and $\mathbf{A}_2$, e.g., minimizing $\|\mathbf{A}_1-\mathbf{U}_1\mathbf{U}_1^T\|_F^2$ over $\mathbf{U}_1\geq\mathbf{0}$. Same as [9], we use the following multiplicative update to obtain the solution

$$\mathbf{U}_1\leftarrow\mathbf{U}_1\odot[\frac{1}{2}+\frac{1}{2}\frac{\mathbf{A}_1\mathbf{U}_1}{\mathbf{U}_1(\mathbf{U}_1^T\mathbf{U}_1)}] \tag{18}$$

Then we set $\mathbf{V}_1=\mathbf{U}_1$ due to the symmetry of $\mathbf{A}_1$ and initialize $\mathbf{U}_2,\mathbf{V}_2$ similarly. Given the initial $\mathbf{U}_1,\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2$, we can initialize $\mathbf{M}$ based on Eq. (9) as

$$\mathbf{M}=(1-\alpha)\sum_{k=0}^{K}\alpha^{k+1}(\mathbf{U}_2^T\mathbf{D}_2^{-1}\mathbf{U}_2)^k\mathbf{U}_2^T\mathbf{D}_2^{-1}\mathbf{H}\mathbf{D}_1^{-1}\mathbf{U}_1(\mathbf{U}_1^T\mathbf{D}_1^{-1}\mathbf{U}_1)^k \tag{19}$$

where K can be set to a large number, e.g., 500.

Overall, the algorithm is summarized in Algorithm 1. It alternatively updates each variable group (line 3-7) until it converges or $t_{\max}$ is reached. In each iteration, the time complexity is $O(nr^2+\min\{|\bar{\Omega}|,|\Omega|\}r)$. The algorithm outputs the complete networks $\mathbf{A}_1^*,\mathbf{A}_2^*$ and alignment matrix $\mathbf{S}$ by averaging between $\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T$ and $\mathbf{H}$ (line 10).

Algorithm 1 iNEAT: Incomplete Network Alignment.

```
Input: (1) adjacency matrices $\mathbf{A}_1,\ \mathbf{A}_2$ of the incomplete networks
    $\mathcal{G}_1,\ \mathcal{G}_2$, (2) prior alignment preference $\mathbf{H}$, (3) the rank $r_1, r_2$,
    (3) parameters $\alpha,\lambda,\gamma,\beta$. and (4) maximum iteration number
    $t_{\max}$.
Output: (1) the alignment matrix $\mathbf{S}$, and (2) the complete adja-
    cency matrices $\mathbf{A}_1^*,\mathbf{A}_2^*$.
1: Initialize $\mathbf{U}_1,\mathbf{V}_1,\mathbf{U}_2,\mathbf{V}_2$ as Eq. (18), $\mathbf{M}$ as Eq. (19), $t=1$;
2: while not converge and $t\leq t_{\max}$ do
3:     Update $\mathbf{U}_1$ by Eq. (17) until convergence;
4:     Update $\mathbf{V}_1$ until its convergence;
5:     Update $\mathbf{U}_2$ until its convergence;
6:     Update $\mathbf{V}_2$ until its convergence;
7:     Update $\mathbf{M}$ until convergence;
8:     Set $t\leftarrow t+1$;
9: end while
10: $\mathbf{A}_1^*=\mathbf{U}_1\mathbf{V}_1^T$, $\mathbf{A}_2^*=\mathbf{U}_2\mathbf{V}_2^T$ and $\mathbf{S}=\alpha\mathbf{U}_2\mathbf{M}\mathbf{U}_1^T+(1-\alpha)\mathbf{H}$.
```

5. Experimental Results

5.1. Experimental Setup

Datasets. We evaluate the proposed algorithm on three types of real-world networks. The statistics of all the datasets are summarized in Table 2. Based on these datasets, we construct four pairs of incomplete networks by the following steps. For each dataset, we first generate a random permutation matrix and use it to construct the second permuted network. Then, in each of these two networks, we remove $0.1\%, 0.5\%, 1\%, 5\%, 10\%, 15\%, 20\%$ of the total number of edges uniformly at random to generate unobserved edges.

Comparison Methods.

- *Alignment.* To evaluate the alignment performance of our algorithm, we compare with the following existing network alignment algorithms, including (1) *NetAlign* [10], (2) *IsoRank* [6], (3) *FINAL-P+* [5]. Besides, to validate whether alignment and imputation are mutually beneficial from each other, we use the low-rank networks recovered solely by Eq. (2) as the input networks for *FINAL-P+*. We name this method as *FINAL-IMP*. We also show the alignment results by the degree similarity (*DegSim*), which is also used as the prior knowledge matrix $\mathbf{H}$ of iNEAT.
- *Completion.* To evaluate the completion performance, we compare with existing matrix completion methods which

TABLE 2: Statistics of Datasets.

Category	Network	# of Nodes	# of Edges
Collaboration	Gr-Qc	5,241	14,484
Infrastructure	Oregon	7,352	15,665
Social	Google+	23,628	39,194
Social	Youtube	1,134,890	2,987,624

are for a single network, including (1) the method based on Eq. (2) (*NMF-IMP*), (2) accelerated proximal gradient based nuclear norm minimization (*NNLS*) [11], (3) Riemannian trust-region based completion (*RTRMC*) [12].

5.2. Effectiveness Analysis

We first evaluate the alignment accuracy with different numbers of unobserved edges in the incomplete networks. We use a heuristic greedy matching algorithm as the post processing step on the alignment matrix to obtain the one-to-one mapping matrix between two input networks, then compute the alignment accuracy with respect to the ground-truth (i.e., the permutation matrix). The results are summarized in Figure 2. We have the following observations. First, we observe that iNEAT outperforms the baseline methods. To be specific, our method achieves an up to 30% alignment accuracy improvement, compared with the baseline methods that directly align across two incomplete networks (i.e., *NetAlign*, *IsoRank*, *FINAL-P+*). Second, the degree similarity (i.e., $\mathbf{H}$) alone gives a very poor performance on the alignment accuracy, whereas by averaging $\mathbf{H}$ and $\mathbf{U}_2\mathbf{M}\mathbf{U}_1$, the alignment matrix (i.e., results of iNEAT) provides a much better accuracy. This verifies the effectiveness of our strategy combining the low-rank structure of alignment matrix and prior knowledge $\mathbf{H}$. Third, the accuracy of iNEAT is higher than that of *FINAL-IMP*, which indicates that solving the alignment and completion tasks simultaneously indeed achieves a better performance than the *completion-then-alignment* strategy. Specifically, as Figure 2(a) and Figure 2(b) show, in some cases, the pure completion may introduce too much noise in the incomplete networks and lead to an even worse alignment result than that of other alignment baseline methods without performing network completion.

Second, to evaluate the effectiveness of iNEAT for network completion, we assume the missing edges are recovered if the corresponding entries of the completed adjacency matrix are larger than a certain threshold (e.g., set to be 0.3 in our paper). Then, we calculate the recovery rate over the total number of missing edges. The results are shown in Figure 3. As we can see, iNEAT has a higher recovery rate than other baseline methods, indicating that the completion performance is indeed improved by alignment.

5.3. Efficiency Analysis

Quality-Speed Trade-off. We evaluate the balance of our algorithm between the alignment accuracy and running time. Here, we show the trade-off results on the collaboration network with 10% unobserved edges in Figure 4. As we can see, the running time of our method is slightly higher than *IsoRank* and *FINAL-P+*, but it achieves a 15%-25% alignment accuracy improvement across the incomplete networks. Meanwhile, our method is much faster than *NetAlign*.

Scalability. We use the Youtube dataset to study the scalability of our method (i.e., running time vs. size of the network). As we can see from Figure 5, the running time is *linear* w.r.t the number of nodes in the networks.

6. Related Work

Network Alignment. Network alignment is a fundamental task in many applications, including bioinformatics [13], data mining [14], etc. Many network alignment algorithms are based on the *topology consistency*. One well-known method *IsoRank* computes the cross-network pairwise topology similarities by propagating the similarities of their neighboring pairs [6]. Koutra et al. propose *BigAlign* algorithm to align across the bipartite networks which assumes that one network is a noisy permutation of the other network [2]. Zhang et al. use the similar permutation assumptions of the networks and introduce the *transitivity* property to align multiple networks [15]. More recently, there are alignment algorithms to align the attributed networks. For instance, *COSNET* formulates the local consistency among the attributes of each node and the global topology consistency to find the alignment across networks [16]. Zhang et al. propose an attributed network alignment algorithm by adopting the topological, node and edge attribute consistency principles [5]. However, most, if not all, of the existing methods implicitly assume input networks are complete without missing edges.

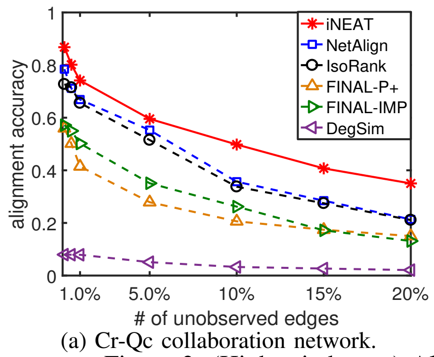
Network Completion. Kim et al. propose an Expectation Maximization (EM) based method that fits the network structure under the Kronecker graph model and re-estimates the parameters using a scalable Gibbs sampling approach [17]. Another work proposed by Masrour et al. leverages node similarity matrix [3]. In addition, inferring the missing edges can be considered as an adjacency matrix completion problem. One well-known matrix completion method is based on singular value thresholding (SVT) to minimize the nuclear norm [18]. To speed up, Toh and Yun propose an accelerated proximal gradient method to solve a nuclear norm regularized linear least squares problem [11]. In addition to matrix completion, tensor completion is very powerful in many areas [19]. Some work use tensor completion on multiple *aligned* networks [4]. Nonetheless, how input networks are aligned beforehand is not answered.

7. Conclusion

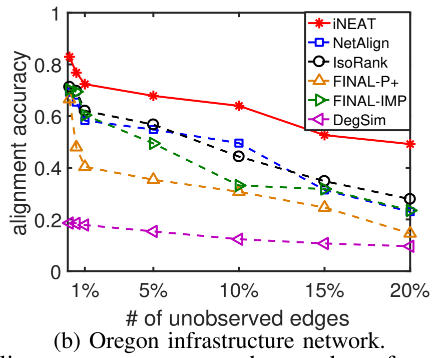
In the era of big data, the multi-sourced and incomplete characteristics often *co-exist* in many real networks. However, the state-of-the-arts have been largely addressing them *in parallel*. In this paper, we propose to jointly address network alignment and network completion so that the two tasks can benefit from each other. We formulate incomplete network alignment problem as an optimization problem and propose a multiplicative update algorithm (iNEAT). To our best knowledge, the proposed iNEAT algorithm is the first network alignment algorithm with a provable *linear* complexity. The empirical evaluations demonstrate both the effectiveness in network alignment and network completion, and the efficiency. Future work includes extending our algorithm to handle attributed networks.

8. Acknowledgement

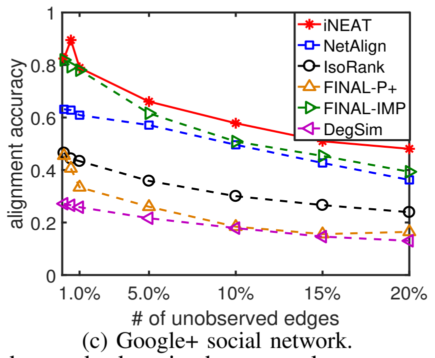
This material is supported by the National Science Foundation under Grant No. IIS-1651203, IIS-1715385, IIS-



(a) Cr-Qc collaboration network.

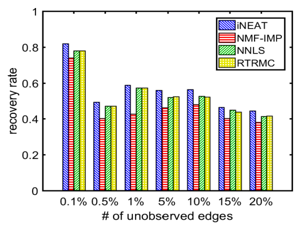


(b) Oregon infrastructure network.

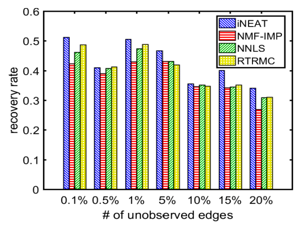


(c) Google+ social network.

Figure 2: (Higher is better.) Alignment accuracy vs. the number of unobserved edges in the networks.



(a) Cr-Qc collaboration network.



(b) Google+ social network.

Figure 3: Recovery rate vs. the number of unobserved edges.

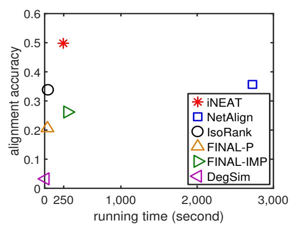


Figure 4: Balance between run time and accuracy.

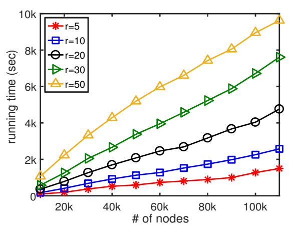


Figure 5: Run time vs. number of nodes.

1743040, and CNS-1629888, by DTRA under the grant number HDTRA1-16-0017, by Army Research Office under the contract number W911NF-16-1-0168, by National Institutes of Health under the grant number R01LM011986, National High-tech R&D Program (2015AA124102), National Natural Science Foundation of China (61561130160), a research fund supported by MSRA, the Royal Society-Newton Advanced Fellowship Award and by additional gifts from Huawei and Baidu. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] D. Zhou, S. Zhang, M. Y. Yildirim, S. Alcorn, H. Tong, H. Davulcu, and J. He, "A local algorithm for structure-preserving graph cut," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 655–664.

[2] D. Koutra, H. Tong, and D. Lubensky, "Big-align: Fast bipartite graph alignment," in *2013 IEEE 13th International Conference on Data Mining*. IEEE, 2013, pp. 389–398.

[3] F. Masrour, I. Barjesteh, R. Forsati, A.-H. Esfahanian, and H. Radha, "Network completion with node similarity: A matrix completion approach with provable guarantees," in *Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2015.

[4] Y. Liu, F. Shang, H. Cheng, J. Cheng, and H. Tong, "Factor matrix trace norm minimization for low-rank tensor completion," in *the 2014 SIAM International Conference on Data Mining*. SIAM, 2014.

[5] S. Zhang and H. Tong, "Final: Fast attributed network alignment," in *Proceedings of the 22th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM*, 2016.

[6] R. Singh, J. Xu, and B. Berger, "Global alignment of multiple protein interaction networks with application to functional orthology detection," *Proceedings of the National Academy of Sciences*, 2008.

[7] J. D. Rennie and N. Srebro, "Fast maximum margin matrix factorization for collaborative prediction," in *the 22nd international conference on Machine learning*. ACM, 2005, pp. 713–719.

[8] M. S. Pedersen, B. Baxter, B. Templeton, C. RishÃÿj, D. L. Theobald, E. Hoeghrasmussen, G. Casteel, J. B. Gao, K. Dedecius, and K. Strim, "The matrix cookbook," 2008.

[9] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM, 2005.

[10] M. Bayati, D. F. Gleich, A. Saberi, and Y. Wang, "Message-passing algorithms for sparse network alignment," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 7, no. 1, p. 3, 2013.

[11] K.-C. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems," *Pacific Journal of optimization*, vol. 6, no. 615-640, p. 15, 2010.

[12] N. Boumal and P.-a. Absil, "Rtrmc: A riemannian trust-region method for low-rank matrix completion," in *Advances in neural information processing systems*, 2011, pp. 406–414.

[13] C.-S. Liao, K. Lu, M. Baym, R. Singh, and B. Berger, "Isorankn: spectral methods for global alignment of multiple protein networks," *Bioinformatics*, vol. 25, no. 12, pp. i253–i258, 2009.

[14] B. Du, S. Zhang, N. Cao, and H. Tong, "First: Fast interactive attributed subgraph matching," in *the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017.

[15] J. Zhang and S. Y. Philip, "Multiple anonymized social networks alignment," in *Data Mining (ICDM)*. IEEE, 2015.

[16] Y. Zhang, J. Tang, Z. Yang, J. Pei, and P. S. Yu, "Cosnet: connecting heterogeneous social networks with local and global consistency," in *the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1485–1494.

[17] M. Kim and J. Leskovec, "The network completion problem: Inferring missing nodes and edges in networks," in *the 2011 SIAM International Conference on Data Mining*. SIAM, 2011, pp. 47–58.

[18] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[19] Y. Zhou and J. He, "Crowdsourcing via tensor augmentation and completion," in *25th International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.