Mutual Community Detection across Multiple Partially Aligned Social Networks

Jiawei Zhang*, Songchang Jin†, Philip S. Yu‡
jwzhanggy@gmail.com jinsongchang87@gmail.com psyu@cs.uic.edu

Abstract

To enjoy more social network services, users nowadays are usually involved in multiple online social networks simultaneously. Networks that involve some common users are named as multiple "partially aligned networks". In this paper, we want to detect communities of multiple partially aligned networks simultaneously, which is formally defined as the "Mutual Community Detection" problem. To solve the mutual community detection problem, a novel community detection method, MCD (Mutual Community Detector), is proposed in this paper. MCD can detect social community structures of users in multiple partially aligned networks at the same time with full considerations of (1) characteristics of each network, and (2) information of the shared users across aligned networks. In addition, to handle large scale aligned networks, we extend method MCD and propose MCD-SCALE. MCD-SCALE applies a distributed multilevel k-way partitioning method to divide the networks into k partitions sequentially. Extensive experiments conducted on two real-world partially aligned heterogeneous social networks demonstrate that MCD and MCD-SCALE can solve the "Mutual Community Detection" problem very well.

Keywords: Mutual Community Detection; Multiple Aligned Social Networks; Big Data; Data Mining

1. Introduction

Nowadays, online social networks which can provide users with various services have become ubiquitous in our daily life. The services provided by social networks are very diverse, e.g., make new friends online, read and write comments on recent news, recommend products and locations, etc. Real-world social networks which can provide these services usually have heterogeneous information, involving various kinds of information entities (e.g., users, locations, posts) and complex connections (e.g., social links among users, purchase links between users and products). Meanwhile, among these services provided by social networks, community detection techniques play a very important role. For example, organizing online friends into different categories (e.g., "family members", "celebrities", and "classmates") in Facebook and group-level recommendations of products in e-commerce sites are all based on community structures of users detected from the networks.

Meanwhile, as proposed in Kong et al. (2013); Zhang et al. (2013, 2014a,b), to enjoy more social network services, users nowadays are usually involved in multiple online social networks simultaneously, e.g., Facebook, Twitter and Foursquare. Furthermore, some of these networks can share common information either due to the common network establishing purpose or because of similar network features Zhang & Yu (2015a). Across these networks, the common users are defined as the *anchor users*, while the remaining non-shared users are named as the *non-anchor users*. Connections between *anchor users'* accounts in different networks are defined as the *anchor links*. The networks partially aligned by *anchor links* are called *multiple partially aligned networks*.

In this paper, we want to detect the communities of each network across multiple *partially aligned social networks* simultaneously, which is formally defined as the *Mutual Community Detection* problem. The goal is to distill relevant informa-

*Florida State University, FL, USA
†National University of Defense Technology, Hunan, China
‡University of Illinois at Chicago, IL, USA. Institute for Data Science, Tsinghua University, China

tion from another social network to compliment knowledge directly derivable from each network to improve the clustering or community detection, while preserving the distinct characteristics of each individual network. The *mutual community detection* problem is very important for online social networks and can be the prerequisite for many concrete social network applications: (1) *network partition*: Detected communities can usually represent small-sized subgraphs of the network, and (2) *comprehensive understanding of user social behaviors*: Community structures of the shared users in multiple aligned networks can provide a complementary understanding of their social interactions in online social networks.

Besides its importance, the *mutual community detection* problem is a novel problem and different from existing clustering problems, including: (1) *consensus clustering*, Goder & Filkov (2008); Li et al. (2007); Nguyen & Caruana (2007); Lourenço et al. (2013); Lock & Dunson (2013) which aims at achieving a consensus result of several input clustering results about the same data; (2) *multi-view clustering*, Bickel & Scheffer (2004); Cai et al. (2013) whose target is to partition objects into clusters based on their different representations, e.g., clustering webpages with text information and hyperlinks; (3) *multi-relational clustering*, Yin et al. (2007); Bhattacharya & Getoor (2005) which focuses on clustering objects in one relation (called target relation) using information in multiple inter-linked relations; and (4) *co-regularized multi-domain graph clustering* Cheng et al. (2013), which relaxes the *one-to-one* constraints on node correspondence relationships between different views in multi-view clustering to "*uncertain*" mappings. In Cheng et al. (2013), prior knowledge about the weights of mappings is required and each view is actually a homogeneous network (more differences are summarized in Section 7). Unlike these existing clustering problems, the *mutual community detection* problem aims at detecting the communities for multiple networks involving both anchor and non-anchor users simultaneously and each network contains heterogeneous information about users' social activities.

In addition, in recent years, social network size drastically increases. A recent report from Business Insider[1] indicates that, among the largest social networks, Facebook has the largest user population at 1.16 billion monthly active users, which is followed by YouTube with 1 billion. China's social media network, Qzone, takes the third position with 712 million monthly active users. *Mutual community detection* of multiple partially aligned large-scale online social networks has never been studied before.

Despite its importance and novelty, the *mutual community detection* problem is very challenging to solve due to:

- *Closeness Measure*: Users in heterogeneous social networks can be connected with each other by various direct and indirect connections. A general closeness measure among users with such connection information is the prerequisite for addressing the *mutual community detection* problem.
- *Network Characteristics*: Social networks usually have their own characteristics, which can be reflected in the community structures formed by users. Preservation of each network's characteristics (i.e., some unique structures in each network's detected communities) is very important in the *mutual community detection* problem.
- *Mutual Community Detection*: Information in different networks can provide us with a more comprehensive understanding about the anchor users' social structures. For anchor users whose community structures are not clear based on in formation in one network, utilizing the heterogeneous information in aligned networks to refine and disambiguate the community structures about the anchor users. However, how to achieve such a goal is still an open problem.
- *Large Scale Networks*: Network size implies that it is difficult for stand-alone programs to apply traditional partitioning methods and it is a difficult task to parallelize the existing stand-alone network partitioning algorithms. For distributed algorithms, load balance should be taken into consideration and how to generate balanced partitions is another challenge.

To solve all these challenges, a novel cross-network community detection method, MCD (Mutual Community Detector), is proposed in this paper. MCD maps the complex relationships in the

[1] http://www.businessinsider.com

social network into a heterogeneous information network Sun et al. (2011) and introduces a novel meta-path based closeness measure, *HNMP-Sim*, to utilize both direct and indirect connections among users in closeness scores calculation. With full considerations of the network characteristics, MCD exploits the information in aligned networks to refine and disambiguate the community structures of the multiple networks concurrently. In addition, to deal with the large-scale aligned networks, we extend MCD and propose the distributed version MCD-SCALE in this paper.

This paper is organized as follows: In Section 2, we formulate the problem. Sections 3-4 introduce the *mutual community detection* methods MCD and MCD-SCALE. Section 5-6 show the experiment results. In Sections 7 and 8, we give the related works and conclude this paper.

2. Problem Formulation

The networks studied in this paper are Foursquare and Twitter. Users in both Foursquare and Twitter can follow other users, write tips/tweets, which can contain timestamps, text content and location check-ins. As a result, both Foursquare and Twitter can be modeled as heterogeneous information networks $G = (V, E)$, where $V = \mathcal{U} \cup \mathcal{P} \cup \mathcal{L} \cup \mathcal{T} \cup \mathcal{W}$ is the set of different types of nodes in the network and $\mathcal{U}$, $\mathcal{P}$, $\mathcal{L}$, $\mathcal{T}$, $\mathcal{W}$ are the node sets of users, posts, location check-ins, timestamps and words respectively, while $E = \mathcal{E}_s \cup \mathcal{E}_p \cup \mathcal{E}_l \cup \mathcal{E}_t \cup \mathcal{E}_w$ is set of directed links in the network and $\mathcal{E}_s$, $\mathcal{E}_p$, $\mathcal{E}_l$, $\mathcal{E}_t$ and $\mathcal{E}_w$ are the sets of social links among users, links between users and posts and those between posts and location-checkins, timestamps as well as words respectively. To illustrate the structure of the heterogeneous network studied in this paper, we also give an example in Figure 1. As shown in the figure, users in the network can be extensively connected with each other by different types of links (e.g., social links, co-location checkins connections).

The multiple aligned networks can be modeled as $\mathcal{G} = (G_{set}, A_{set})$, where $G_{set} = \{G^{(1)}, G^{(2)}, \ldots, G^{(n)}\}, |G_{set}| = n$ is the set of n heterogeneous information networks and $A_{set} = \{A^{(1,2)}, \ldots, A^{(1,n)}, A^{(2,3)}, \ldots, A^{((n-1),n)}\}$ is the set of undirected anchor links between different heterogeneous networks in G_{set}. In this paper, we will

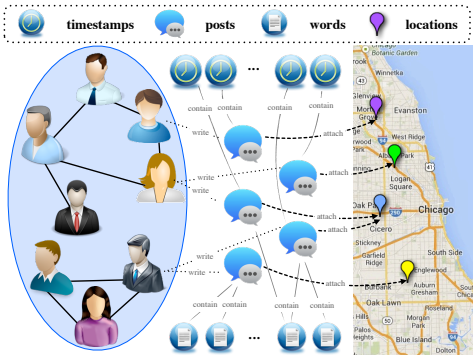


Figure 1: *Heterogeneous online social networks.*

follow the definitions about "anchor user", "non-anchor user", "anchor link", etc. proposed in Kong et al. (2013); Zhang et al. (2013, 2014a,b) and the constraint on anchor links is "*one-to-one*", i.e., each user can have one account in on network. The case that users have multiple accounts in online social networks can be resolved with method introduced in Tsikerdekis & Zeadally (2014), where these duplicated accounts can be aggregated in advance to form one unique vitural account in advance and the anchor links connecting these vitural accounts will be still "*one-to-one*". Different from Kong et al. (2013); Zhang et al. (2013), networks studied in this paper are all *partially aligned* Zhang et al. (2014a,b).

Mutual Community Detection Problem: For the given multiple aligned heterogeneous networks $\mathcal{G}$, the *Mutual Community Detection* problem aims to obtain the optimal communities $\{\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \cdots, \mathcal{C}^{(n)}\}$ for $\{G^{(1)}, G^{(2)}, \cdots, G^{(n)}\}$ simultaneously, where $\mathcal{C}^{(i)} = \{U_1^{(i)}, U_2^{(i)}, \ldots, U_{k^{(i)}}^{(i)}\}$ is a partition of the users set $\mathcal{U}^{(i)}$ in $G^{(i)}$, $k^{(i)} = \left|\mathcal{C}^{(i)}\right|$, $U_l^{(i)} \cap U_m^{(i)} = \emptyset, \forall\, l, m \in \{1, 2, \ldots, k^{(i)}\}$ and $\bigcup_{j=1}^{k^{(i)}} U_j^{(i)} = \mathcal{U}^{(i)}$. Users in each detected cluster are more densely connected with each other than with users in other clusters. In this paper, we focus on studying the hard (i.e., non-overlapping) clustering of users in online social networks.

3. Mutual Community Detection Framework for Small-Sized Aligned Networks

A co-regularization based multi-view clustering model was proposed in Cheng et al. (2013),

Table 1: *Summary of HNMPs.*

ID	Notation	Heterogeneous Network Meta Path	Semantics
1	U → U	User $\xrightarrow{follow}$ User	Follow
2	U → U → U	User $\xrightarrow{follow}$ User $\xrightarrow{follow}$ User	Follower of Follower
3	U → U ← U	User $\xrightarrow{follow}$ User $\xrightarrow{follow^{-1}}$ User	Common Out Neighbor
4	U ← U → U	User $\xrightarrow{follow^{-1}}$ User $\xrightarrow{follow}$ User	Common In Neighbor
5	U → P → W P ← U	User $\xrightarrow{write}$ Post $\xrightarrow{contain}$ Word	Posts Containing
		$\xrightarrow{contain^{-1}}$ Post $\xrightarrow{write^{-1}}$ User	Common Words
6	U → P → T ← P ← U	User $\xrightarrow{write}$ Post $\xrightarrow{contain}$ Time	Posts Containing
		$\xrightarrow{contain^{-1}}$ Post $\xrightarrow{write^{-1}}$ User	Common Timestamps
7	U → P → L ← P ← U	User $\xrightarrow{write}$ Post $\xrightarrow{attach}$ Location	Posts Attaching Common
		$\xrightarrow{attach^{-1}}$ Post $\xrightarrow{write^{-1}}$ User	Location Check-ins

which achieves the clustering results of nodes across multi-view by minimizing absolute clustering disagreement of all nodes (both shared and non-shared nodes). It cannot be applied to address the *Mutual Community Detection* problem, as in mutual community detection, we only exploit information across networks to refine the social community structures of anchor users only, while non-anchor users social community structures are not affected and can preserve their characteristics. To solve the *Mutual Community Detection* problem, a novel community detection method, MCD, will be proposed in this section. By mapping the social network relations into a heterogeneous information network, we use the concept of social meta path to define closeness measure among users in Section 3.1. Based on this similarity measure, we introduce the network characteristics preservation independent clustering method in Section 3.2 and normalized discrepancy based co-clustering method in Section 3.3. To preserve network characteristics and use information in other networks to refine community structures mutually at the same time, we study the mutual community detection problem in Section 3.4.

1. HNMP-Sim

Many existing similarity measures, e.g., "Common Neighbor" Hasan & Zaki (2011), "Jaccard's Coefficient" Hasan & Zaki (2011), defined for homogeneous networks cannot capture all the connections among users in heterogeneous networks. To use both direct and indirect connections among users in calculating the similarity score among users in the heterogeneous information network, we introduce meta path based similarity measure HNMP-Sim in this section.

1.1 Meta Paths in Heterogeneous Networks

In heterogeneous networks, pairs of nodes can be connected by different paths, which are sequences of links in the network. Meta paths Sun et al. (2011, 2009) in heterogeneous networks, i.e., *heterogeneous network meta paths* (HNMPs), can capture both direct and indirect connections among nodes in a network. The length of a meta path is defined as the number of links that constitute it. Meta paths in networks can start and end with various node types. However, in this paper, we are mainly concerned about those starting and ending with users, which are formally defined as the *social HNMPs*. The notation, definition and semantics of 7 different *social HNMPs* used in this paper are listed in Table 1. To extract the social meta paths, prior domain knowledge about the network structure is required.

1.2 HNMP-based Similarity

These 7 different social HNMPs in Table 1 can cover lots of connections among users in networks. Some meta path based similarity measures have

been proposed so far, e.g., the *PathSim* proposed in Sun et al. (2011), which is defined for undirected networks and considers different meta paths to be of the same importance. To measure the social closeness among users in directed heterogeneous information networks, we extend *PathSim* to propose a new closeness measure as follows.

Definition 1 (HNMP-Sim): Let $\mathcal{P}_i(x \rightsquigarrow y)$ and $\mathcal{P}_i(x \rightsquigarrow \cdot)$ be the sets of path instances of HNMP # i going from x to y and those going from x to other nodes in the network. The HNMP-Sim (HNMP based Similarity) of node pair (x, y) is defined as

$$\text{HNMP-Sim}(x,y) = \sum_i \omega_i \left(\frac{|\mathcal{P}_i(x \rightsquigarrow y)| + |\mathcal{P}_i(y \rightsquigarrow x)|}{|\mathcal{P}_i(x \rightsquigarrow \cdot)| + |\mathcal{P}_i(y \rightsquigarrow \cdot)|} \right),$$

where ω_i is the weight of HNMP # i and $\sum_i \omega_i = 1$. In this paper, the weights of different HNMPs can be automatically adjusted by applying the technique proposed in Zhang & Yu (2015a).

Let $\mathbf{A}_i$ be the *adjacency matrix* corresponding to the HNMP # i among users in the network and $\mathbf{A}_i(m,n) = k$ iff there exist k different path instances of HNMP # i from user m to n in the network. Furthermore, the similarity score matrix among users of HNMP # i can be represented as $\mathbf{S}_i = (\mathbf{D}_i + \tilde{\mathbf{D}}_i)^{-1} \left(\mathbf{A}_i + \mathbf{A}_i^T\right)$, where $\mathbf{A}_i^T$ denotes the transpose of $\mathbf{A}_i$, diagonal matrices $\mathbf{D}_i$ and $\tilde{\mathbf{D}}_i$ have values $\mathbf{D}_i(l,l) = \sum_m \mathbf{A}_i(l,m)$ and $\tilde{\mathbf{D}}_i(l,l) = \sum_m (\mathbf{A}_i^T)(l,m)$ on their diagonals respectively. The HNMP-Sim matrix of the network which can capture all possible connections among users is represented as follows:

$$\mathbf{S} = \sum_i \omega_i \mathbf{S}_i = \sum_i \omega_i \left((\mathbf{D}_i + \tilde{\mathbf{D}}_i)^{-1} \left(\mathbf{A}_i + \mathbf{A}_i^T\right)\right).$$

2. Network Characteristic Preservation Clustering

Clustering each network independently can preserve each networks characteristics effectively as no information from external networks will interfere with the clustering results. Partitioning users of a certain network into several clusters will cut connections in the network and lead to some costs inevitably. Optimal clustering results can be achieved by minimizing the clustering costs.

For a given network G, let $\mathcal{C} = \{U_1, U_2, \ldots, U_k\}$ be the community structures detected from G. Term $\overline{U_i} = \mathcal{U} - U_i$ is defined to be the complement of set U_i in G. Various cost measure of partition $\mathcal{C}$ can be used, e.g., *cut* Wu & Leahy (1993) and *normalized cut* Shi & Malik (2000):

$$cut(\mathcal{C}) = \frac{1}{2}\sum_{i=1}^{k} S(U_i, \overline{U_i}) = \frac{1}{2}\sum_{i=1}^{k} \sum_{u \in U_i, v \in \overline{U_i}} S(u,v),$$

$$Ncut(\mathcal{C}) = \frac{1}{2}\sum_{i=1}^{k} \frac{S(U_i, \overline{U_i})}{S(U_i, \cdot)} = \sum_{i=1}^{k} \frac{cut(U_i, \overline{U}_i)}{S(U_i, \cdot)},$$

where $S(u,v)$ denotes the HNMP-Sim between u, v and $S(U_i, \cdot) = S(U_i, \mathcal{U}) = S(U_i, U_i) + S(U_i, \overline{U}_i)$.

For all users in $\mathcal{U}$, their clustering result can be represented in the *result confidence matrix* $\mathbf{H}$, where $\mathbf{H} = [\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_n}]^{\mathbf{T}}$, $n = |\mathcal{U}|$, $\mathbf{h}_i = (h_{i,1}, h_{i,2}, \ldots, h_{i,k})$ and $h_{i,j}$ denotes the confidence that $u_i \in \mathcal{U}$ is in cluster $U_j \in \mathcal{C}$. The optimal $\mathbf{H}$ that can minimize the normalized-cut cost can be obtained by solving the following objective function von Luxburg (2007):

$$\min_{\mathbf{H}} \ \text{Tr}(\mathbf{H}^T \mathbf{L} \mathbf{H}),$$
$$s.t. \ \mathbf{H}^T \mathbf{D} \mathbf{H} = \mathbf{I}.$$

where $\mathbf{L} = \mathbf{D} - \mathbf{S}$, diagonal matrix $\mathbf{D}$ has $\mathbf{D}(i,i) = \sum_j \mathbf{S}(i,j)$ on its diagonal, and $\mathbf{I}$ is an identity matrix.

Algorithm 1 Curvilinear Search Method ($\mathcal{CSM}$)

Input: $\mathbf{X}_k$ C_k, Q_k and function $\mathcal{F}$
parameters **ffl** $= \{\rho, \eta, \delta, \tau, \tau_m, \tau_M\}$
Output: $\mathbf{X}_{k+1}$, C_{k+1}, Q_{k+1}

$\mathbf{Y}(\tau) = \left(\mathbf{I} + \frac{\tau}{2}\mathbf{A}\right)^{-1} \left(\mathbf{I} - \frac{\tau}{2}\mathbf{A}\right) \mathbf{X}_k$
while $\mathcal{F}\left(\mathbf{Y}(\tau)\right) \geq \mathbf{C}_k + \rho\tau\mathcal{F}'\left((\mathbf{Y}(0))\right)$ **do**
  $\tau = \delta\tau$
  $\mathbf{Y}(\tau) = \left(\mathbf{I} + \frac{\tau}{2}\mathbf{A}\right)^{-1} \left(\mathbf{I} - \frac{\tau}{2}\mathbf{A}\right) \mathbf{X}_k$
end while
$\mathbf{X}_{k+1} = \mathbf{Y}_k(\tau)$
$Q_{k+1} = \eta Q_k + 1$
$C_{k+1} = \left(\eta Q_k C_k + \mathcal{F}(\mathbf{X}_{k+1})\right) / Q_{k+1}$
$\tau = \max\left(\min(\tau, \tau_M), \tau_m\right)$

3. Discrepancy based Clustering of Multiple Networks

Besides the shared information due to common network construction purposes and similar network features Zhang & Yu (2015a), anchor users

Algorithm 2 Mutual Community Detector (MCD)

Input: aligned network: $\mathcal{G} = \{\{G^{(1)}, G^{(2)}\}, \{A^{(1,2)}, A^{(2,1)}\}\}$;
number of clusters in $G^{(1)}$ and $G^{(2)}$: $k^{(1)}$ and $k^{(2)}$;
HNMP Sim matrices weight: **!**;
parameters: **ffl** $= \{\rho, \eta, \delta, \tau, \tau_m, \tau_M\}$;
function $\mathcal{F}$ and consensus term weight θ
Output: $\mathbf{H}^{(1)}$, $\mathbf{H}^{(2)}$

Calculate HNMP Sim matrices, $\mathbf{S}_i^{(1)}$ and $\mathbf{S}_i^{(2)}$
$\mathbf{S}^{(1)} = \sum_i \omega_i \mathbf{S}_i^{(1)}$, $\mathbf{S}^{(2)} = \sum_i \omega_i \mathbf{S}_i^{(2)}$
Initialize $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ with Kmeans clustering results on $\mathbf{S}^{(1)}$ and $\mathbf{S}^{(2)}$
Initialize $C_0^{(1)} = 0, Q_0^{(1)} = 1$ and $C_0^{(2)} = 0, Q_0^{(2)} = 1$
converge $=$ *False*
while *converge* $=$ *False* **do**
  /* update $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ with $\mathcal{CSM}$ */
  $\mathbf{X}_{k+1}^{(1)}, C_{k+1}^{(1)}, Q_{k+1}^{(1)} = \mathcal{CSM}(\mathbf{X}_k^{(1)}, C_k^{(1)}, Q_k^{(1)}, \mathcal{F}, \mathbf{ffl})$
  $\mathbf{X}_{k+1}^{(2)}, C_{k+1}^{(2)}, Q_{k+1}^{(2)} = \mathcal{CSM}(\mathbf{X}_k^{(2)}, C_k^{(2)}, Q_k^{(2)}, \mathcal{F}, \mathbf{ffl})$
  if $\mathbf{X}_{k+1}^{(1)}$ and $\mathbf{X}_{k+1}^{(2)}$ both converge **then**
    converge $=$ *True*
  end if
end while
$\mathbf{H}^{(1)} = \left((\mathbf{D}^{(1)})^{-\frac{1}{2}}\right)^T \mathbf{X}^{(1)}$, $\mathbf{H}^{(2)} = \left((\mathbf{D}^{(2)})^{-\frac{1}{2}}\right)^T \mathbf{X}^{(2)}$

can also have unique information (e.g., social structures) across aligned networks, which can provide us with a more comprehensive knowledge about the community structures formed by these users. Meanwhile, by maximizing the consensus (i.e., minimizing the "*discrepancy*") of the clustering results about the anchor users in multiple partially aligned networks, we refine the clustering results of the anchor users with information in other aligned networks mutually. We can represent the clustering results achieved in $G^{(1)}$ and $G^{(2)}$ as $\mathcal{C}^{(1)} = \{U_1^{(1)}, U_2^{(1)}, \cdots, U_{k^{(1)}}^{(1)}\}$ and $\mathcal{C}^{(2)} = \{U_1^{(2)}, U_2^{(2)}, \cdots, U_{k^{(2)}}^{(2)}\}$ respectively.

Let u_i and u_j be two anchor users in the network, whose accounts in $G^{(1)}$ and $G^{(2)}$ are $u_i^{(1)}$, $u_i^{(2)}$, $u_j^{(1)}$ and $u_j^{(2)}$ respectively. If users $u_i^{(1)}$ and $u_j^{(1)}$ are partitioned into the same cluster in $G^{(1)}$ but their corresponding accounts $u_i^{(2)}$ and $u_j^{(2)}$ are partitioned into different clusters in $G^{(2)}$, then it will lead to a *discrepancy* between the clustering results of $u_i^{(1)}$, $u_i^{(2)}$, $u_j^{(1)}$ and $u_j^{(2)}$ in aligned networks $G^{(1)}$ and $G^{(2)}$.

Definition 2 (Discrepancy): The discrepancy between the clustering results of u_i and u_j across aligned networks $G^{(1)}$ and $G^{(2)}$ is defined as the difference of confidence scores of u_i and u_j being partitioned in the same cluster across aligned networks. Considering that in the clustering results, the confidence scores of $u_i^{(1)}$ and $u_j^{(1)}$ ($u_i^{(2)}$ and $u_j^{(2)}$) being partitioned into $k^{(1)}$ ($k^{(2)}$) clusters can be represented as vectors $\mathbf{h}_i^{(1)}$ and $\mathbf{h}_j^{(1)}$ ($\mathbf{h}_i^{(2)}$ and $\mathbf{h}_j^{(2)}$) respectively, while the confidences that u_i and u_j are in the same cluster in $G^{(1)}$ and $G^{(2)}$ can be denoted as $\mathbf{h}_i^{(1)}(\mathbf{h}_j^{(1)})^T$ and $\mathbf{h}_i^{(2)}(\mathbf{h}_j^{(2)})^T$. Formally, the discrepancy of the clustering results about u_i and u_j is defined to be $d_{ij}(\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) = \left(\mathbf{h}_i^{(1)}(\mathbf{h}_j^{(1)})^T - \mathbf{h}_i^{(2)}(\mathbf{h}_j^{(2)})^T\right)^2$ if u_i, u_j are both anchor users; and $d_{ij}(\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) = 0$ otherwise. Furthermore, the discrepancy of $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$ will be:

$$d(\mathcal{C}^{(1)}, \mathcal{C}^{(2)}) = \sum_i^{n^{(1)}} \sum_j^{n^{(2)}} d_{ij}(\mathcal{C}^{(1)}, \mathcal{C}^{(2)}),$$

where $n^{(1)} = |\mathcal{U}^{(1)}|$ and $n^{(2)} = |\mathcal{U}^{(2)}|$. In the definition, non-anchor users are not involved in the discrepancy calculation, which is totally different from the clustering disagreement function (all the nodes are included) introduced in Cheng et al. (2013)

However, considering that $d(\mathcal{C}^{(1)}, \mathcal{C}^{(2)})$ is highly dependent on the number of anchor users and anchor links between $G^{(1)}$ and $G^{(2)}$, minimizing $d(\mathcal{C}^{(1)}, \mathcal{C}^{(2)})$ can favor highly consented clustering results when the anchor users are abundant but have no significant effects when the anchor users are very rare. To solve this problem, we propose to minimize the *normalized discrepancy* instead, which significantly differs from the absolute clustering disagreement cost used in Cheng et al. (2013).

Definition 3 (Normalized Discrepancy) The normalized discrepancy measure computes the differences of clustering results in two aligned networks as a fraction of the discrepancy with regard to the number of anchor users across partially aligned

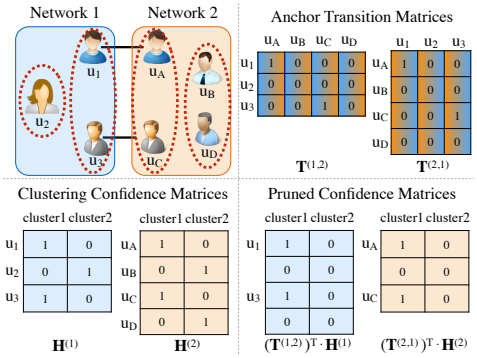


Figure 2: *An example to illustrate the clustering discrepancy.*

networks:

$$Nd(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = \frac{d(\mathcal{C}^{(1)},\mathcal{C}^{(2)})}{\left(\left|A^{(1,2)}\right|\right)\left(\left|A^{(1,2)}\right|-1\right)}.$$

Optimal consensus clustering results of $G^{(1)}$ and $G^{(2)}$ will be $\hat{\mathcal{C}^{(1)}},\hat{\mathcal{C}^{(2)}}$:

$$\hat{\mathcal{C}}^{(1)},\hat{\mathcal{C}}^{(2)} = \arg\min_{\mathcal{C}^{(1)},\mathcal{C}^{(2)}} Nd(\mathcal{C}^{(1)},\mathcal{C}^{(2)}).$$

Similarly, the normalized-discrepancy objective function can also be represented with the *clustering results confidence matrices* $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ as well. Meanwhile, considering that the networks studied in this paper are partially aligned, matrices $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ contain the results of both anchor users and non-anchor users, while non-anchor users should not be involved in the discrepancy calculation according to the definition of discrepancy. We propose to prune the results of the non-anchor users with the following *anchor transition matrix* first.

Definition 4 (Anchor Transition Matrix): Binary matrix $\mathbf{T}^{(1,2)}$ (or $\mathbf{T}^{(2,1)}$) is defined as the anchor transition matrix from networks $G^{(1)}$ to $G^{(2)}$ (or from $G^{(2)}$ to $G^{(1)}$), where $\mathbf{T}^{(1,2)} = (\mathbf{T}^{(2,1)})^T$, $\mathbf{T}^{(1,2)}(i,j) = 1$ if $(u_i^{(1)}, u_j^{(2)}) \in A^{(1,2)}$ and 0 otherwise. The row indexes of $\mathbf{T}^{(1,2)}$ (or $\mathbf{T}^{(2,1)}$) are of the same order as those of $\mathbf{H}^{(1)}$ (or $\mathbf{H}^{(2)}$). Considering that the constraint on anchor links is "*one-to-one*" in this paper, as a result, each row/column of $\mathbf{T}^{(1,2)}$ and $\mathbf{T}^{(2,1)}$ contains at most one entry filled with 1.

In Figure 2, we show an example about the clustering discrepancy of two partially aligned networks $G^{(1)}$ and $G^{(2)}$, users in which are grouped into two clusters $\{\{u_1,u_3\},\{u_2\}\}$ and $\{\{u_A,u_C\},\{u_B,u_D\}\}$ respectively. Users u_1, u_A and u_3, u_C are identified to be anchor users, based on which we can construct the "anchor transition matrices" $\mathbf{T}^{(1,2)}$ and $\mathbf{T}^{(2,1)}$ as shown in the upper right plot. Furthermore, based on the community structure, we can construct the "*clustering confidence matrices*" as shown in the lower left plot. To obtain the clustering results of anchor users only, the *anchor transition matrix* can be applied to prune the clustering results of non-anchor users from the *clustering confidence matrices*. By multiplying the *anchor transition matrices* $(\mathbf{T}^{(1,2)})^T$ and $(\mathbf{T}^{(2,1)})^T$ with *clustering confidence matrices* $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ respectively, we can obtain the "pruned confidence matrices" as show in the lower right plot of Figure 2. Entries corresponding anchor users u_1, u_3, u_A and u_C are preserved but those corresponding to non-anchor users are all pruned.

In this example, the clustering discrepancy of the partially aligned networks should be 0 according to the above discrepancy definition. Meanwhile, networks $G^{(1)}$ and $G^{(2)}$ are of different sizes and the pruned confidence matrices are of different dimensions, e.g., $(\mathbf{T}^{(1,2)})^T\mathbf{H}^{(1)} \in \mathbb{R}^{4\times 2}$ and $(\mathbf{T}^{(2,1)})^T\mathbf{H}^{(2)} \in \mathbb{R}^{3\times 2}$. To represent the discrepancy with the clustering confidence matrices, we need to further accommodate the dimensions of different pruned *clustering confidence matrices*. It can be achieved by multiplying one pruned *clustering confidence matrices* with the corresponding *anchor transition matrix* again, which will not prune entries but only adjust the matrix dimensions. Let $\bar{\mathbf{H}}^{(1)} = (\mathbf{T}^{(1,2)})^T\mathbf{H}^{(1)}$ and $\bar{\mathbf{H}}^{(2)} = (\mathbf{T}^{(1,2)})^T(\mathbf{T}^{(2,1)})^T\mathbf{H}^{(2)}$. In the example, we can represent the clustering discrepancy to be

$$\left\|\bar{\mathbf{H}}^{(1)}\left(\bar{\mathbf{H}}^{(1)}\right)^T - \bar{\mathbf{H}}^{(2)}\left(\bar{\mathbf{H}}^{(2)}\right)^T\right\|_F^2 = 0,$$

where matrix $\bar{\mathbf{H}}\bar{\mathbf{H}}^T$ indicates whether pairs of anchor users are in the same cluster or not.

Furthermore, the objective function of inferring clustering confidence matrices, which can minimize the normalized discrepancy can be repre-

sented as follows

$$\min_{\mathbf{H}^{(1)},\mathbf{H}^{(2)}} \frac{\left\|\bar{\mathbf{H}}^{(1)}\left(\bar{\mathbf{H}}^{(1)}\right)^T - \bar{\mathbf{H}}^{(2)}\left(\bar{\mathbf{H}}^{(2)}\right)^T\right\|_F^2}{\left\|\mathbf{T}^{(1,2)}\right\|_F^2\left(\left\|\mathbf{T}^{(1,2)}\right\|_F^2 - 1\right)},$$
$$s.t.\ (\mathbf{H}^{(1)})^T\mathbf{D}^{(1)}\mathbf{H}^{(1)} = \mathbf{I}, (\mathbf{H}^{(2)})^T\mathbf{D}^{(2)}\mathbf{H}^{(2)} = \mathbf{I}.$$

where $\mathbf{D}^{(1)}$, $\mathbf{D}^{(2)}$ are the corresponding diagonal matrices of HNMP-Sim matrices of networks $G^{(1)}$ and $G^{(2)}$ respectively.

4. Joint Mutual Community Detection of Multiple Networks

Normalized-Cut objective function favors clustering results that can preserve the characteristic of each network, however, normalized-discrepancy objective function favors consensus results which are mutually refined with information from other aligned networks. Taking both of these two issues into considerations, the optimal mutual community detection results $\hat{\mathcal{C}}^{(1)}$ and $\hat{\mathcal{C}}^{(2)}$ of aligned networks $G^{(1)}$ and $G^{(2)}$ can be achieved as follows:

$$\arg\min_{\mathcal{C}^{(1)},\mathcal{C}^{(2)}} \alpha \cdot Ncut(\mathcal{C}^{(1)}) + \beta \cdot Ncut(\mathcal{C}^{(2)}) + \theta \cdot Nd(\mathcal{C}^{(1)}, \mathcal{C}^{(2)})$$

where α, β and θ represents the weights of these terms and, for simplicity, α, β are both set as 1 in this paper.

By replacing $Ncut(\mathcal{C}^{(1)})$, $Ncut(\mathcal{C}^{(2)})$, $Nd(\mathcal{C}^{(1)}, \mathcal{C}^{(2)})$ with the objective equations derived above, we can rewrite the joint objective function as follows:

$$\min_{\mathbf{H}^{(1)},\mathbf{H}^{(2)}} \alpha\mathrm{Tr}((\mathbf{H}^{(1)})^T\mathbf{L}^{(1)}\mathbf{H}^{(1)}) + \beta\mathrm{Tr}((\mathbf{H}^{(2)})^T\mathbf{L}^{(2)}\mathbf{H}^{(2)}) + \theta\frac{\left\|\bar{\mathbf{H}}^{(1)}\left(\bar{\mathbf{H}}^{(1)}\right)^T - \bar{\mathbf{H}}^{(2)}\left(\bar{\mathbf{H}}^{(2)}\right)^T\right\|_F^2}{\left\|\mathbf{T}^{(1,2)}\right\|_F^2\left(\left\|\mathbf{T}^{(1,2)}\right\|_F^2 - 1\right)},$$
$$s.t.\ (\mathbf{H}^{(1)})^T\mathbf{D}^{(1)}\mathbf{H}^{(1)} = \mathbf{I}, (\mathbf{H}^{(2)})^T\mathbf{D}^{(2)}\mathbf{H}^{(2)} = \mathbf{I},$$

where $\mathbf{L}^{(1)} = \mathbf{D}^{(1)} - \mathbf{S}^{(1)}$, $\mathbf{L}^{(2)} = \mathbf{D}^{(2)} - \mathbf{S}^{(2)}$ and matrices $\mathbf{S}^{(1)}$, $\mathbf{S}^{(2)}$ and $\mathbf{D}^{(1)}$, $\mathbf{D}^{(2)}$ are the HNMP-Sim matrices and their corresponding diagonal matrices defined before.

The objective function is a complex optimization problem with orthogonality constraints, which can be very difficult to solve because the constraints are not only non-convex but also numerically expensive to preserve during iterations. Meanwhile, by substituting $\left(\mathbf{D}^{(1)}\right)^{\frac{1}{2}}\mathbf{H}^{(1)}$ and $\left(\mathbf{D}^{(2)}\right)^{\frac{1}{2}}\mathbf{H}^{(2)}$ with $\mathbf{X}^{(1)}$, $\mathbf{X}^{(2)}$, we can transform the objective function into a standard form of problems solvable with method proposed in Wen & Yin (2010):

$$\min_{\mathbf{X}^{(1)},\mathbf{X}^{(2)}} \alpha(\mathrm{Tr}((\mathbf{X}^{(1)})^T\tilde{\mathbf{L}}^{(1)}\mathbf{X}^{(1)}) + \beta\mathrm{Tr}((\mathbf{X}^{(2)})^T\tilde{\mathbf{L}}^{(2)}\mathbf{X}^{(2)}) + \theta\frac{\left\|\tilde{\mathbf{T}}^{(1)}\mathbf{X}^{(1)}\left(\tilde{\mathbf{T}}^{(1)}\mathbf{X}^{(1)}\right)^T - \tilde{\mathbf{T}}^{(2)}\mathbf{X}^{(2)}\left(\tilde{\mathbf{T}}^{(2)}\mathbf{X}^{(2)}\right)^T\right\|_F^2}{\left\|\mathbf{T}^{(1,2)}\right\|_F^2\left(\left\|\mathbf{T}^{(1,2)}\right\|_F^2 - 1\right)}),$$
$$s.t.\ (\mathbf{X}^{(1)})^T\mathbf{X}^{(1)} = \mathbf{I}, (\mathbf{X}^{(2)})^T\mathbf{X}^{(2)} = \mathbf{I}.$$

where $\tilde{\mathbf{L}}^{(1)} = ((\mathbf{D}^{(1)})^{-\frac{1}{2}})^T\mathbf{L}^{(1)}((\mathbf{D}^{(1)})^{-\frac{1}{2}})$, $\tilde{\mathbf{L}}^{(2)} = ((\mathbf{D}^{(2)})^{-\frac{1}{2}})^T\mathbf{L}^{(2)}((\mathbf{D}^{(2)})^{-\frac{1}{2}})$ and $\tilde{\mathbf{T}}^{(1)} = (\mathbf{T}^{(1,2)})^T(\mathbf{D}^{(1)})^{-\frac{1}{2}}$, $\tilde{\mathbf{T}}^{(2)} = (\mathbf{T}^{(1,2)})^T(\mathbf{T}^{(2,1)})^T(\mathbf{D}^{(2)})^{-\frac{1}{2}}$.

Wen et al. Wen & Yin (2010) propose a feasible method to solve the above optimization problems with a constraint-preserving update scheme. They propose to update one variable, e.g., $\mathbf{X}^{(1)}$, while fixing the other variable, e.g., $\mathbf{X}^{(2)}$, alternatively with the curvilinear search with Barzilai-Borwein step method until convergence. For example, when $\mathbf{X}^{(2)}$ is fixed, we can simplify the objective function into

$$\min_{\mathbf{X}} \mathcal{F}(\mathbf{X}), s.t.(\mathbf{X})^T\mathbf{X} = \mathbf{I},$$

where $\mathbf{X} = \mathbf{X}^{(1)}$ and $\mathcal{F}(\mathbf{X})$ is the objective function, which can be solved with the curvilinear search with Barzilai-Borwein step method proposed in Wen & Yin (2010) to update $\mathbf{X}$ until convergence and the variable $\mathbf{X}$ after the $(k+1)_{th}$ iteration will be

$$\mathbf{X}_{k+1} = \mathbf{Y}(\tau_k), \mathbf{Y}(\tau_k) = \left(\mathbf{I} + \frac{\tau_k}{2}\mathbf{A}\right)^{-1}\left(\mathbf{I} - \frac{\tau_k}{2}\mathbf{A}\right)\mathbf{X}_k,$$

$$\mathbf{A} = \frac{\partial\mathcal{F}(\mathbf{X}_k)}{\partial\mathbf{X}}\mathbf{X}_k^T - \mathbf{X}_k(\frac{\partial\mathcal{F}(\mathbf{X}_k)}{\partial\mathbf{X}})^T,$$

where let $\hat{\tau} = \left(\frac{\mathrm{Tr}((\mathbf{X}_k - \mathbf{X}_{k-1})^T(\mathbf{X}_k - \mathbf{X}_{k-1}))}{|\mathrm{Tr}((\mathbf{X}_k - \mathbf{X}_{k-1})^T(\nabla\mathcal{F}(\mathbf{X}_k) - \nabla\mathcal{F}(\mathbf{X}_{k-1})))|}\right)$, $\tau_k = \hat{\tau}\delta^h$, δ is the Barzilai-Borwein step size and h is the smallest integer to make τ_k satisfy

$$\mathcal{F}(\mathbf{Y}(\tau_k)) \leq C_k + \rho\tau_k\mathcal{F}'_\tau(\mathbf{Y}(0)).$$

Algorithm 3 Edge Weight based Matching ($\mathcal{EWM}$)

Input: Network G_h
Maximum weight of a node $maxVW = n/k$
Output: A coarser network G_{h+1}

map() Function:
for node i in current data bolck **do**
  if $match[i] == -1$ **then**
    $maxIdx = -1$
    $sortByEdgeWeight(NN(i))$
    for $v_j \in NN(i)$ **do**
      if $match[j] == -1$ and $VW(i) + VW(j) < maxVW$ **then**
        maxIdx = j
      end if
      $match[i] = maxIdx$
      $match[maxIdx] = i$
    end for
  end if
end for
reduce() Function:
new $newNodeID[n+1]$
new $newVW[n+1]$
set $idx = 1$
for $i \in \{1, 2, \cdots, n\}$ **do**
  if $i < match[i]$ **then**
    set $newNodeID[match[i]] = idx$
    set $newNodeID[i] = idx$
    set $newVW[i] = newVW[match[i]] = VW(i) + VW(match[i[)$
    $idx++$
  end if
end for

Terms C, Q are defined as $C_{k+1} = (\eta Q_k C_k + \mathcal{F}(\mathbf{X}_{k+1}))/Q_{k+1}$ and $Q_{k+1} = \eta Q_k + 1, Q_0 = 1$. More detailed derivatives of the curvilinear search method (i.e., Algorithm 1) with Barzilai-Borwein step is available in Wen & Yin (2010). Meanwhile, the pseudo-code of method MCD is available in Algorithm 2. Based on the achieved solutions $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$, we can get $\mathbf{H}^{(1)} = \left(\mathbf{D}^{(1)}\right)^{-\frac{1}{2}} \mathbf{X}^{(1)}$ and $\mathbf{H}^{(2)} = \left(\mathbf{D}^{(2)}\right)^{-\frac{1}{2}} \mathbf{X}^{(2)}$.

Algorithm 4 Synergistic Partitioning ($\mathcal{SP}$)

Input: Network G_h
Anchor Link Map $Map < anidx, pidx >$
Maximum weight of a node $maxVW = n/k$
Output: A coarser network G_{h+1}

Call Synergistic Partitioning-Map Function
Call Synergistic Partitioning-Reduce Function

4. Mutual Community Detection Framework for Large-Scale Aligned Networks

For large-sized networks, Data processing in MCD-SCALE can be roughly divided into two stages: datum generation stage and network alignment stage.

When got the anchor node set $A^{(1,2)}$, the framework will apply a distributed multilevel k-way partitioning method onto the datum network to generate k balanced partitions. During this process, the anchor nodes are ignored and all the nodes are given the same treatment. We call this process datum generation stage. When finished, partition result of anchor nodes will be generated, we store them in a set-$Map\langle anidx, pidx\rangle$, where $anidx$ is anchor node ID and $pidx$ represents the partition ID the anchor node belongs to.

After the datum generation stage, synergistic networks will be partitioned into k partitions according to the $Map\langle anidx, pidx\rangle$ to make the synergistic networks to align to the datum network, and during this process *discrepancy* and *cut* are the objectives to be minimized. We call this process network alignment stage.

1. Distributed Multilevel k-way Partitioning

Algorithms guaranteed to find out near-optimal partitions in a single network have been studied for a long period. But most of the methods are standalone, and performance is limited by the server's capacity. Inspired by the multilevel $k-way$ partitioning (MKP) method proposed by Karypis and Kumar Karypis & Kumar (1998, 1996) and based on our previous work Aggarwal et al. (2009), we try to use MapReduce Dean & Ghemawat (2008) to speedup the MKP method. As the same with

other multilevel methods, MapReduce based MKP also includes three phases: coarsening, initial partitioning and un-coarsening.

Coarsening phase is a multilevel process and a sequence of smaller approximate networks $G_i = (V_i, E_i)$ are constructed from the original network $G_0 = (V, E)$, where $|V_i| < |V_{i-1}|, i \in \{1, 2, \cdots, n\}$. To construct coarser networks, node combination and edge collapsing should be performed. The task can be formally defined in terms of matching Bui & Jones (1993). A matching is a set of node pairs $M = List\langle i, j\rangle, i \neq j$ and $e_{i,j} \in E$, in which each node can only appear for no more than once. For a network G_i with a matching M_i, if $\langle j, k\rangle \in M_i$ then $v(j)$ and $v(k)$ will form a new node $v(q) \in V_{i+1}$. The weight of $v(q)$ equals to the sum of weight $v(j)$ and $v(k)$, besides, all the links connected to $v(j)$ or $v(k)$ in G_i will be connected to $v(q)$ in G_{i+1}. The total weight of nodes will remain unchanged during the coarsening phase but the total weight of edges and number of nodes will be reduced. Define $W(T)$ to be the sum of edge weight in T and $N(T)$ to be the number of components in T. It will be that:

$$W(E_{i+1}) = W(E_i) - W(M_i),$$
$$N(V_{i+1}) = N(V_i) - N(M_i).$$

Analysis in Karypis & Kumar (1995a) shows that for the same coarser network, smaller edge-weight corresponds to smaller edge-cut. With the help of MapReduce framework, we use a local search method to implement an edge-weight based matching (EWM) scheme to collect larger edge weight during the coarsening phase. For the convenience of MapReduce, we design a new network representation format: each line contains essential information about a node and all its neighbors (NN), such as node ID, vertex weight (VW), edge weight (W), et al. The whole network data are distributed in distributed file system, such as HDFS Shvachko et al. (2010), and each data block only contains part of node set and corresponding connection information. Function $map()$ takes a data block as input and search locally to find node pairs to match according to the edge weight, $reduce()$ is in charge of node combination, renaming and sorting. With the new node IDs and matching, a simple MapReduce job will be able to update the edge information and write the coarser network back onto HDFS. The complexity of EWM is $O(|E|)$ in each iteration and pseudo code about EWM is shown in Algorithm 3.

Algorithm 5 Synergistic Partitioning-Map

Input: Network G_h
Anchor Link Map $Map < anidx, pidx >$
Maximum weight of a node $maxVW = n/k$
Output: A coarser network G_{h+1}
map() Function:
for node i in current data bolck **do**
  if $match[i] == -1$ **then**
    set $flag = false$
    $sortByEdgeWeight(NN(i))$
    if $v_i \in Map < anidx, pidx >$ **then**
      for $v_j \in NN(i)$ & $match[j] == -1$ **do**
        if $v_j \in Map < anidx, pidx >$ & $Map.get(v_i) == Map.get(v_j)$ & $VW(i) + VW(j) < maxVW$ **then**
          $match[i] = j, match[j] = i$
          $flag = true$, break
        end if
      end for
      if $flag == false$, no suitable anchor node **then**
        for $v_j \in NN(i)$ & $match[j] == -1$ & $VW(v_i) + VW(v_j) < maxVW$ **do**
          $indirectNeighbor = NN(v_j)$
          $sortByEdgeWeight(NN(i))$
          for $v_k \in indirectNeighbor$ **do**
            if $v_k \in Map < anidx, pidx >$ & $Map.get(v_i) == Map.get(v_k)$ **then**
              $match[i] = j, match[j] = i$
              $flag = true$, break
            end if
          end for
          if $flag == true$ **then**
            break
          end if
        end for
      end if
    else
      $sortByEdgeWeight(NN(i))$
      for $v_j \in NN(v_i)$ & $v_j \notin Map < anidx, pidx >$ & $VW(i) + VW(j) < maxVW$ & $match[j] == -1$ **do**
        $match[i] = j$, $match[j] = i$, break
      end for
    end if
  end if
end for

After several iterations, a coarsest weighted network $G^{(1)}$ consisting of only hundreds of nodes will be generated. For the network size of $G^{(1)}$, stand-alone algorithms with high computing complexity will be acceptable for initial partitioning. Meanwhile, the weights of nodes and edges of coarser networks are set to reflect the weights of the finer network during the coarsening phase, so $G^{(1)}$ contains sufficient information to intelligently satisfy the balanced partition and the minimum edge-cut requirements. Plenty of traditional bisection methods are quite qualified for the task. In this paper, we adopt the KL method with an $O(|E|^3)$ computing complexity to divide $G^{(1)}$ into two partitions and then take recursive invocations

of KL method on the partitions to generate balanced k partitions.

Un-coarsening phase is inverse processing of coarsening phase. With the initial partitions and the matching of the coarsening phase, it is easy to run the un-coarsening process on the MapReduce cluster.

2. Distributed Synergistic Partitioning Process

In this section we talk about the synergistic partitioning process on the synergistic networks with the knowledge of partition results of anchor nodes from datum network. The synergistic partitioning is also a MKP process but quite different from general MKP methods.

In the coarsening phase, anchor nodes are endowed with higher priority than non-anchor nodes. When choosing nodes to pair, we assume that anchor nodes and non-anchor nodes have different tendencies. Let $G^{(2)}$ be the datum network and $G^{(1)} = G_1^{(1)}, G_2^{(1)}, \cdots, G_{(t-1)}^{(1)}$ be the synergistic network set.

For an anchor node $v(i)$, it would prefer to combine with another anchor node $v(j)$ which has the same partition ID in the datum network, i.e., $pidx(Gd, v(i)) = pidx(Gd, v(j))$ where $v(i) \in A$, $v(j) \in A$ and $i \neq j$. Second, if there is no appropriate anchor node, it would try to find a non-anchor node to pair. When planing to find a non-anchor node to pair, the anchor node, assuming to be $v(i)$, would like to find a correct direction, and it would prefer to match with the non-anchor node $v(j)$, which has lots of anchor nodes as neighbors with the same pidx with $v(i)$. When combined together, the new node will be given the same pidx as the anchor node. To improve the accuracy of synergistic partitioning among multiple social networks, an anchor node will never try to combine with another anchor node with different $pidx$.

For a non-anchor node, it would prefer to make a pair with an anchor node neighbor which belongs to the dominant partition in the non-anchor node's neighbors. Here, dominant partition in a node's neighbors means the number of anchor nodes with this partition ID is largest. Next, a non-anchor node would choose a general non-anchor node to pair. At last, a non-anchor node would not like to combine with an anchor node being part of the partitions which are in subordinate status. After combined together, the new node will be given the same $pidx$ as the anchor node. To ensure the balance among the partitions, about $\frac{1}{3}$ of the nodes in the coarsest network are unlabeled.

Algorithm 6 Synergistic Partitioning-Reduce

Input: Network G_h
Anchor Link Map $Map < anidx, pidx >$
Maximum weight of a node $maxVW = n/k$
Output: A coarser network G_{h+1}

```
reduce() Function:
new newNodeID[n + 1]
new newVW[n + 1]
set idx = 1
for i ∈ newNodeID[] do
    if i < match[i] then
        set newNodeID[match[i]] = idx
        set newNodeID[i] = idx
        set newVW[i] = newVW[match[i]] = VW(i) + VW(match[i[)
        idx + +
    end if
end for
new newPurity[idx + 1]
new newPidx[idx + 1]
for i ∈ [1, idx] do
    newPurity[i] = (purity[i]*VW(i)+purity[j]*VW(j)) / (VW(i)+VW(j))
    newPidx[i] = max{pidx[i], pidx[match[i]]}
end for
```

In addition to minimizing both the discrepancy and cut discussed at the beginning of Section 4, we also try to balance the size of partitions are the objectives in synergistic partitioning process. However, when put together, it is impossible to achieve them simultaneously. So, we try to make a compromise among them and develop a heuristic method to tackle the problems. First, according to the conclusion smaller edge-weight corresponds to smaller edge-cut and the pairing tendencies, we propose a modified EWM (MEWM) method to find a matching in the coarsening phase, of which the edge-weight is as large as possible. At the end of the coarsening phase, there is no impurity in any node, meaning that each node contains no more than one type of anchor nodes. Besides, a "*purity*" vector attribute and a $pidx$ attribute are added to each node to represent the percentage of each kind of anchor nodes swallowed up by it

and the *pidx* of the new node, respectively. Then, during the initial partitioning phase, we treat the anchor nodes as labeled nodes and use a modified label propagation algorithm to deal with the non-anchor nodes in the coarsest network. At the end of the initial partitioning phase, we will be able to generate balanced k partitions and to maximize the number of same kind of anchor nodes being divided into same partitions. Finally, we project the coarsest network back to the original network, which is the same as traditional MKP process. The pseudo code of coarsening phase in synergistic partitioning process is available in Algorithm 4.

5. Experiments on Small-Sized Aligned Networks

To demonstrate the effectiveness of MCD, we will conduct extensive experiments on two real-world partially aligned heterogeneous networks: Foursquare and Twitter, in this section.

1. Dataset Description

As mentioned in the Section 2, both Foursquare and Twitter used in this paper are heterogeneous social networks, whose statistical information is given in Table 2. These two networks were crawled with the methods proposed in Kong et al. (2013) during November, 2012. The number of anchor links obtained is 3,388. Some basic descriptions about datasets are as follows:

- **Foursquare**: Users together with their posts are crawled from Foursquare, whose number are 5,392 and 48,756 respectively. The number of social link among users is 76,972. All these posts written by these users and can attach locations checkins and, as a result, the numbers of write link and locate link are both 48,756. 38,921 different locations are crawled from Foursquare.
- **Twitter**: 5,223 users and all their tweets, whose number is 9,490,707, are crawled from Twitter and, on average, each user has about 1,817 tweets. Among these tweets, about 615,515 have location check-ins, which accounts for about 6.48% of all tweets. The number of locations crawled from Twitter is 297,182 and the number of social links among users is 164,920.

Table 2: *Properties of the Heterogeneous Social Networks*

		network	
	property	**Twitter**	**Foursquare**
# node	user	5,223	5,392
	tweet/tip	9,490,707	48,756
	location	297,182	38,921
# link	friend/follow	164,920	76,972
	write	9,490,707	48,756
	locate	615,515	48,756

For more information about the datasets and crawling methods, please refer to Kong et al. (2013); Zhang et al. (2013, 2014a,b).

2. Experiment Settings

2.1 Comparison Methods

The comparison methods used in the experiments can be divided into three categories,

Mutual Community Detection Methods

- MCD: MCD is the mutual community detection method proposed in this paper, which can detect the communities of multiple aligned networks with consideration of the connections and characteristics of different networks. Heterogeneous information in multiple aligned networks are applied in building MCD.

Multi-Network Clustering Methods

- SIclus: the clustering method proposed in Zhou & Liu (2013); Zhang & Yu (2015a) can calculate the similarity scores among users by propagating heterogeneous information across views/networks. In this paper, we extend the method proposed in Zhou & Liu (2013); Zhang & Yu (2015a) and propose SIclus to calculate the intimacy scores among users in multiple networks simultaneously, based on which, users can be grouped into different clusters with clustering models based on intimacy matrix factorization as introduced in Zhang & Yu (2015a). Heterogeneous information across networks is used to build SIclus.

Isolated Clustering Methods, which can detect communities in each isolated network:

Table 3: *Community Detection Results of Foursquare and Twitter Evaluated by Quality Metrics.*

			remaining anchor link rates σ									
network	metric	method	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Foursquare	ndbi	MCD	**0.927**	**0.924**	**0.95**	**0.969**	**0.966**	**0.961**	**0.958**	**0.954**	**0.971**	**0.958**
		SIclus	0.891	0.889	0.88	0.877	0.894	0.883	0.89	0.88	0.887	0.893
		Ncut	0.863	0.863	0.863	0.863	0.863	0.863	0.863	0.863	0.863	0.863
		Kmeans	0.835	0.835	0.835	0.835	0.835	0.835	0.835	0.835	0.835	0.835
	ent.	MCD	**1.551**	**1.607**	**1.379**	**1.382**	**1.396**	**1.382**	**1.283**	**1.552**	**1.308**	**1.497**
		SIclus	4.332	4.356	4.798	4.339	4.474	4.799	4.446	4.658	4.335	4.459
		Ncut	2.768	2.768	2.768	2.768	2.768	2.768	2.768	2.768	2.768	2.768
		Kmeans	2.369	2.369	2.369	2.369	2.369	2.369	2.369	2.369	2.369	2.369
	den.	MCD	**0.216**	**0.205**	**0.196**	0.163	**0.239**	**0.192**	**0.303**	**0.198**	0.170	**0.311**
		SIclus	0.116	0.121	0.13	0.095	0.143	0.11	0.13	0.12	0.143	0.103
		Ncut	0.154	0.154	0.154	0.154	0.154	0.154	0.154	0.154	0.154	0.154
		Kmeans	0.182	0.182	0.182	**0.182**	0.182	0.182	0.182	0.182	**0.182**	0.182
	sil.	MCD	**-0.137**	**-0.114**	**-0.148**	**-0.156**	**-0.117**	**-0.11**	**-0.035**	**-0.125**	**-0.148**	**-0.044**
		SIclus	-0.168	-0.198	-0.173	-0.189	-0.178	-0.181	-0.21	-0.195	-0.167	-0.18
		Ncut	-0.34	-0.34	-0.34	-0.34	-0.34	-0.34	-0.34	-0.34	-0.34	-0.34
		Kmeans	-0.297	-0.297	-0.297	-0.297	-0.297	-0.297	-0.297	-0.297	-0.297	-0.297
Twitter	ndbi	MCD	**0.962**	**0.969**	**0.955**	**0.969**	**0.97**	**0.958**	**0.952**	**0.96**	**0.946**	**0.953**
		SIclus	0.815	0.843	0.807	0.83	0.826	0.832	0.835	0.808	0.812	0.836
		Ncut	0.759	0.759	0.759	0.759	0.759	0.759	0.759	0.759	0.759	0.759
		Kmeans	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761	0.761
	ent.	MCD	**2.27**	**2.667**	**2.48**	**2.381**	**2.43**	**2.372**	**2.452**	**2.459**	**2.564**	**2.191**
		SIclus	4.780	5.114	5.066	4.961	4.904	4.866	5.121	4.629	4.872	5.000
		Ncut	3.099	3.099	3.099	3.099	3.099	3.099	3.099	3.099	3.099	3.099
		Kmeans	3.245	3.245	3.245	3.245	3.245	3.245	3.245	3.245	3.245	3.245
	den.	MCD	**0.14**	0.097	**0.142**	0.109	**0.15**	**0.158**	**0.126**	**0.149**	**0.147**	**0.164**
		SIclus	0.055	0.017	0.044	0.026	0.04	0.062	0.016	0.044	0.045	0.02
		Ncut	0.107	0.107	0.107	0.107	0.107	0.107	0.107	0.107	0.107	0.107
		Kmeans	0.119	**0.119**	0.119	**0.119**	0.119	0.119	0.119	0.119	0.119	0.119
	sil.	MCD	**-0.137**	**-0.179**	**-0.282**	**-0.175**	**-0.275**	**-0.273**	**-0.248**	**-0.269**	**-0.266**	**-0.286**
		SIclus	-0.356	-0.322	-0.311	-0.347	-0.346	-0.349	-0.323	-0.363	-0.345	-0.352
		Ncut	-0.424	-0.424	-0.424	-0.424	-0.424	-0.424	-0.424	-0.424	-0.424	-0.424
		Kmeans	-0.406	-0.406	-0.406	-0.406	-0.406	-0.406	-0.406	-0.406	-0.406	-0.406

- Ncut: Ncut is the clustering method based on normalized cut proposed in Shi & Malik (2000). Method Ncut can detect the communities in each social network merely based on the social connections in each network in the experiments.
- Kmeans: Kmeans is a traditional clustering method, which can be used to detect communities Qi et al. (2012) in social networks based on the social connections only in the experiments.

2.2 Evaluation Methods

The evaluation metrics applied in this paper can be divided into two categories: Quality Metrics and Consensus Metrics.

Quality Metrics: 4 widely and commonly used quality metrics are applied to measure the clustering result, e.g., $\mathcal{C} = \{U_i\}_{i=1}^{K}$, of each network.

- *normalized-dbi* Zhou & Liu (2013):

$$ndbi(\mathcal{C}) = \frac{1}{K}\sum_i \min_{j\neq i} \frac{d(c_i,c_j)+d(c_j,c_i)}{\sigma_i+\sigma_j+d(c_i,c_j)+d(c_j,c_i)},$$

where c_i is the centroid of community $U_i \in \mathcal{C}$, $d(c_i,c_j)$ denotes the distance between centroids c_i and c_j and σ_i represents the average distance between elements in U_i and centroid c_i. (Higher ndbi corresponds to better performance).

- *entropy* Zhou & Liu (2013):

$$H(\mathcal{C}) = -\sum_{i=1}^{K} P(i)\log P(i),$$

where $P(i) = \frac{|U_i|}{\sum_{i=1}^{K}|U_i|}$. (Lower entropy corresponds to better performance).

Table 4: *Community Detection Results of Foursquare and Twitter Evaluated by Consensus Metrics.*

		remaining anchor link rates σ									
measure	methods	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
rand	MCD	**0.095**	**0.099**	**0.107**	**0.138**	**0.116**	**0.121**	**0.132**	**0.106**	**0.089**	0.159
	SIclus	0.135	0.139	0.144	0.148	0.142	0.14	**0.132**	0.132	0.144	**0.141**
	Ncut	0.399	0.377	0.372	0.4	0.416	0.423	0.362	0.385	0.362	0.341
	Kmeans	0.436	0.387	0.4	0.358	0.403	0.363	0.408	0.365	0.35	0.363
vi	MCD	**3.309**	**4.052**	**4.058**	**3.902**	**4.038**	**4.348**	**3.973**	**3.944**	**4.078**	**2.911**
	SIclus	7.56	8.324	8.414	8.713	8.756	8.836	8.832	8.621	8.427	8.02
	Ncut	5.384	5.268	5.221	4.855	5.145	5.541	5.909	5.32	5.085	5.246
	Kmeans	5.427	5.117	5.355	5.326	5.679	5.944	5.452	5.567	5.513	4.686
nmi	MCD	0.152	**0.152**	**0.149**	**0.141**	**0.149**	**0.156**	**0.142**	**0.158**	**0.147**	0.146
	SIclus	**0.172**	0.097	0.081	0.06	0.056	0.069	0.078	0.093	0.105	**0.149**
	Ncut	0.075	0.074	0.111	0.108	0.109	0.099	0.05	0.036	0.042	0.106
	Kmeans	0.008	0.047	0.048	0.054	0.048	0.028	0.047	0.014	0.067	0.119
mi	MCD	0.756	**0.611**	**0.4**	0.258	**0.394**	**0.431**	**0.381**	**0.533**	**0.697**	0.689
	SIclus	**0.780**	0.446	0.367	**0.277**	0.258	0.325	0.374	0.44	0.489	**0.698**
	Ncut	0.188	0.181	0.261	0.232	0.252	0.243	0.138	0.092	0.111	0.31
	Kmeans	0.02	0.112	0.119	0.135	0.127	0.078	0.119	0.038	0.194	0.314

Table 5: *Community Detection Results of Foursquare and Twitter Evaluated by IQC Metrics.*

		remaining anchor link rates σ									
measure	methods	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
IQC^{ndbi}_{rand}	MCD	**-1.699**	**-1.695**	**-1.691**	**-1.662**	**-1.705**	**-1.676**	**-1.647**	**-1.703**	**-1.738**	**-1.594**
	SIclus	-1.459	-1.451	-1.44	-1.434	-1.444	-1.45	-1.465	-1.465	-1.442	-1.448
	Ncut	-0.824	-0.869	-0.878	-0.821	-0.789	-0.776	-0.899	-0.851	-0.897	-0.94
	Kmeans	-0.724	-0.821	-0.795	-0.88	-0.79	-0.87	-0.779	-0.865	-0.895	-0.869
$IQC^{ent.}_{vi}$	MCD	**10.439**	**12.379**	**11.975**	**11.566**	**11.902**	**12.45**	**11.681**	**11.897**	**12.028**	**9.509**
	SIclus	24.58	26.107	26.287	26.884	26.971	27.13	27.123	26.7	26.313	25.499
	Ncut	16.634	16.403	16.308	15.577	16.156	16.948	17.684	16.506	16.036	16.359
	Kmeans	16.468	15.847	16.325	16.267	16.972	17.503	16.519	16.748	16.641	14.986
$IQC^{dens.}_{nmi}$	MCD	**-0.659**	**-0.606**	**-0.636**	**-0.555**	**-0.686**	**-0.663**	**-0.713**	**-0.664**	**-0.611**	**-0.768**
	SIclus	-0.467	-0.317	-0.284	-0.243	-0.235	-0.261	-0.28	-0.309	-0.332	-0.421
	Ncut	-0.411	-0.409	-0.484	-0.477	-0.478	-0.458	-0.361	-0.333	-0.345	-0.473
	Kmeans	-0.317	-0.395	-0.397	-0.41	-0.398	-0.357	-0.396	-0.329	-0.436	-0.54
$IQC^{sil.}_{mi}$	MCD	**-1.239**	**-0.93**	**-0.371**	**-0.186**	**-0.396**	**-0.479**	**-0.479**	**-0.673**	**-0.979**	**-1.048**
	SIclus	-1.028	-0.361	-0.202	-0.022	0.016	-0.118	-0.216	-0.347	-0.446	-0.863
	Ncut	0.389	0.403	0.242	0.3	0.261	0.278	0.488	0.58	0.542	0.144
	Kmeans	0.664	0.479	0.465	0.433	0.45	0.546	0.466	0.628	0.316	0.074

- *density* Zhou & Liu (2013):

$$dens(\mathcal{C}) = \sum_{i=1}^{K} \frac{|E_i|}{|E|},$$

where E and E_i are the edge sets in the network and U_i. (Higher density corresponds to better performance).

- *silhouette* Liu et al. (2010):

$$sil(\mathcal{C}) = \frac{1}{K}\sum_{i=1}^{K}(\frac{1}{|U_i|}\sum_{u\in U_i}\frac{b(u)-a(u)}{\max\{a(u),b(u)\}}),$$

where $a(u) = \frac{1}{|U_i|-1}\sum_{v\in U_i, u\neq v} d(u,v)$ and $b(u) = \min_{j,j\neq i}\left(\frac{1}{|U_j|}\sum_{v\in U_j} d(u,v)\right)$. (Higher silhouette corresponds to better performance).

Consensus Metrics: Given the clustering results $\mathcal{C}^{(1)} = \{U_i^{(1)}\}_{i=1}^{K^{(1)}}$ and $\mathcal{C}^{(2)} = \{U_i^{(2)}\}_{i=1}^{K^{(2)}}$, the consensus metrics measuring the how similar or dissimilar the anchor users are clustered in $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$ include:

- *rand* Nguyen & Caruana (2007): $rand(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = \frac{N_{01}+N_{10}}{N_{00}+N_{01}+N_{10}+N_{11}}$, where $N_{11}(N_{00})$ is the numbers of pairwise anchor users who are clustered in the same (different) community(ies) in both $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$, $N_{01}(N_{10})$ is that of anchor users who are clustered in the same community (different communities) in $\mathcal{C}^{(1)}$ but in different communities (the same communities) in $\mathcal{C}^{(2)}$. (Lower

rand corresponds to better performance).

- *variation of information* Nguyen & Caruana (2007):

$$vi(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = H(\mathcal{C}^{(1)}) + H(\mathcal{C}^{(2)}) - 2mi(\mathcal{C}^{(1)},\mathcal{C}^{(2)}).$$

(Lower vi corresponds to better performance).

- *mutual information* Nguyen & Caruana (2007):

$$mi(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = \sum_{i=1}^{K^{(1)}}\sum_{j=1}^{K^{(2)}} P(i,j)\log\frac{P(i,j)}{P(i)P(j)},$$

where $P(i,j) = \frac{|U_i^{(1)} \cap_{\mathcal{A}} U_j^{(2)}|}{|\mathcal{A}|}$ and $|U_i^{(1)} \cap_{\mathcal{A}} U_j^{(2)}| = \left|\{u|u \in U_i^{(1)}, \exists v \in U_i^{(2)}, (u,v) \in \mathcal{A}\}\right|$ Kong et al. (2013). (Higher mi corresponds to better performance).

- *normalized mutual information* Nguyen & Caruana (2007):

$$nmi(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = \frac{mi(\mathcal{C}^{(1)},\mathcal{C}^{(2)})}{\sqrt{H(\mathcal{C}^{(1)})H(\mathcal{C}^{(2)})}}.$$

(Higher nmi corresponds to better performance).

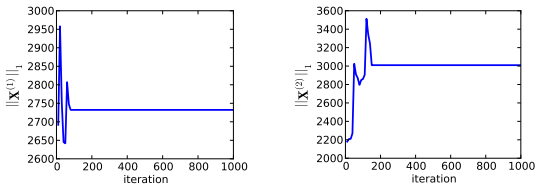
Definition 9 (Proportional and Inversely Proportional Metrics): Depending on relationship between the metric value and the clustering results, all the above metrics can be either *proportional* or *inversely proportional*. Metric M is proportional iff better clustering results corresponds to higher M value; M is inversely proportional iff better clustering result corresponds lower M value.

In the metrics introduced above, *normalized-dbi*, *density*, *silhouette*, *mutual information* and *normalized mutual information* are *proportional metrics*, *entropy*, *rand*, and *variation of information* are *inversely proportional metrics*.

To consider both the quality and consensus simultaneously, we introduce a new clustering metric, IQC metrics, in this paper, which is *inversely proportional*.

Definition 10 (IQC metrics): IQC is a linear combination of quality metrics Q and consensus metrics C.

$$IQC(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = I(Q)(\beta_1 Q(\mathcal{C}^{(1)}) + \beta_2 Q(\mathcal{C}^{(2)})) + I(C)(\beta_3 C(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) + \beta_4 C(\mathcal{C}^{(2)},\mathcal{C}^{(1)}))$$



(a) $\left\|\mathbf{X}^{(1)}\right\|_1$ (b) $\left\|\mathbf{X}^{(2)}\right\|_1$

Figure 3: $\left\|\mathbf{X}^{(1)}\right\|_1$ *and* $\left\|\mathbf{X}^{(2)}\right\|_1$ *in each iteration.*

where $\beta_1, \beta_2, \beta_3, \beta_4$ are weights of different terms, which are all set as 1 in this paper, and $I(Q), I(C) = -1$, if Q/C is proportional and 1, otherwise.

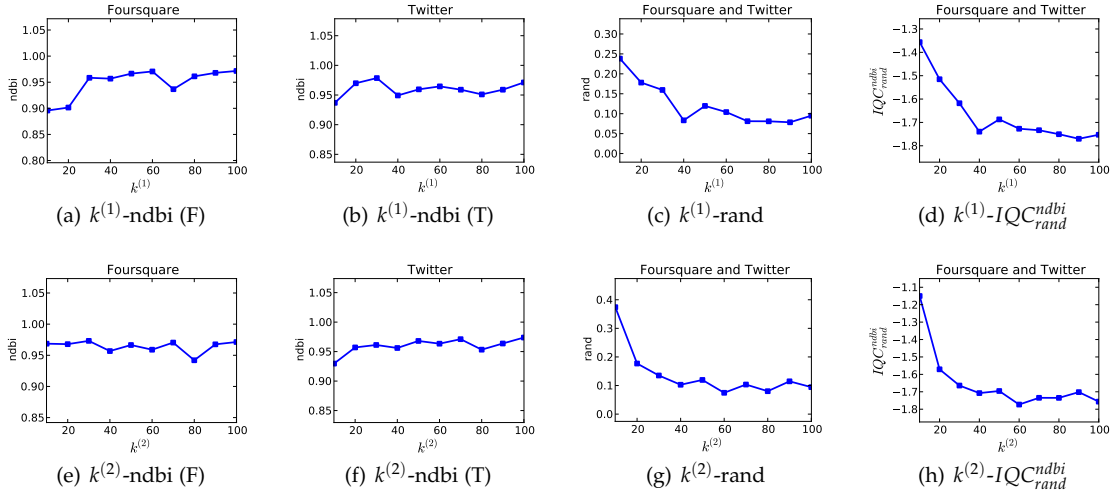
IQC Metrics used in this paper include:

- $IQC_{vi}^{ent}(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = H(\mathcal{C}^{(1)}) + H(\mathcal{C}^{(2)}) + 2vi(\mathcal{C}^{(1)},\mathcal{C}^{(2)})$
- $IQC_{mi}^{sil}(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = -sil(\mathcal{C}^{(1)}) - sil(\mathcal{C}^{(2)}) - 2mi(\mathcal{C}^{(1)},\mathcal{C}^{(2)})$
- $IQC_{rand}^{ndbi}(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = -ndbi(\mathcal{C}^{(1)}) - ndbi(\mathcal{C}^{(2)}) + 2rand(\mathcal{C}^{(1)},\mathcal{C}^{(2)})$
- $IQC_{nmi}^{dens}(\mathcal{C}^{(1)},\mathcal{C}^{(2)}) = -dens(\mathcal{C}^{(1)}) - dens(\mathcal{C}^{(2)}) - 2nmi(\mathcal{C}^{(1)},\mathcal{C}^{(2)})$

3. Experiment Results

The experiment results are available in Tables 3-4. To show the effects of the anchor links, we use the same networks but randomly sample a proportion of anchor links from the networks, whose number is controlled by $\sigma \in \{0.1, 0.2, \cdots, 1.0\}$, where $\sigma = 0.1$ means that 10% of all the anchor links are preserved and $\sigma = 1.0$ means that all the anchor links are preserved.

Table 3 displays the clustering results of different methods in Foursquare and Twitter respectively under the evaluation of ndbi, entropy, density and silhouette. As shown in these two tables, MCD can achieve the highest ndbi score in both Foursquare and Twitter for different sample rate of anchor links consistently. The entropy of the clustering results achieved by MCD is the lowest among all other comparison methods and is about 70% lower than SIclus, 40% lower than Ncut and Kmeans



(a) $k^{(1)}$-ndbi (F) (b) $k^{(1)}$-ndbi (T) (c) $k^{(1)}$-rand (d) $k^{(1)}$-IQC^{ndbi}_{rand}

(e) $k^{(2)}$-ndbi (F) (f) $k^{(2)}$-ndbi (T) (g) $k^{(2)}$-rand (h) $k^{(2)}$-IQC^{ndbi}_{rand}

Figure 4: *Analysis of parameters $k^{(1)}$ and $k^{(2)}$.*

in both Foursquare and Twitter. In each community detected by MCD, the social connections are denser than that of SIclus , Ncut and Kmeans. Similar results can be obtained under the evaluation of silhouette, the silhouette score achieved by MCD is the highest among all comparison methods. So, MCD can achieve better results than modified multi-view and isolated clustering methods under the evaluation of *quality metrics*.

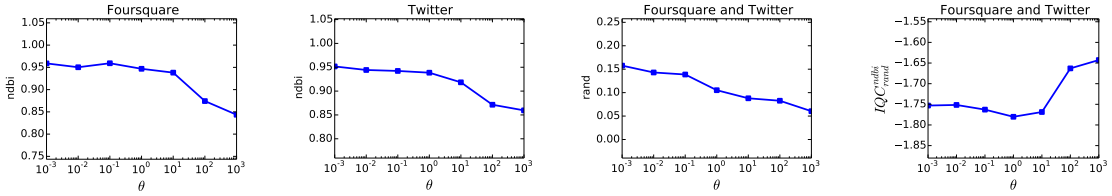
Table 4 shows the clustering results on the aligned networks under the evaluation of consensus metrics, which include rand, vi, nmi and mi. As shown in Table 4, MCD can perform the best among all the comparison methods under the evaluation of consensus metrics. For example, the rand score of MCD is the lowest among all other methods and when $\sigma = 0.5$, the rand score of MCD is 20% lower than SIclus, 72% lower than Ncut and Kmeans. Similar results can be obtained for other evaluation metrics, like when $\sigma = 0.5$, the vi score of MCD is about half of the the score of SIclus; the nmi and mi score of MCD is the triple of that ofKmeans. As a result, MCD can achieve better performance than both modified multi-view and isolated clustering methods under the evaluation of *consensus metrics*.

Table 5 is the clustering results of different methods evaluated by the IQC metrics. As shown in Table 5, the IQC^{ndbi}_{rand}, $IQC^{ent.}_{vi}$, $IQC^{dens.}_{nmi}$, $IQC^{sil.}_{mi}$ scores of MCD are all the lowest among all comparison methods. As mentioned above, lower IQC score corresponds to better clustering results, MCD can outperform all other baseline methods consistently under the evaluation of all IQC metrics. In sum, MCD can perform better than both modified multi-view and isolated clustering methods evaluated by IQC metrics.

According to the results shown in Tables 3-5, we observe that the performance of MCD doesn't varies much as σ changes. The possible reason can be that, in method MCD, normalized clustering discrepancy is applied to infer the clustering confidence matrices. As σ increases in the experiments, more anchor links are added between networks, part of whose effects will be neutralized by the normalization of clustering discrepancy and doesn't affect the performance of MCD much.

4. Convergence Analysis

MCD can compute the solution of the optimization function with Curvilinear Search method, which can update matrices $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ alternatively. This process will continue until convergence. To check whether this process can stop or not, in this part, we will analyze the convergence of $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. In Figure 3, we show the L^1 norm of matrices $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$, $\left\|\mathbf{X}^{(1)}\right\|_1$ and $\left\|\mathbf{X}^{(2)}\right\|_1$, in each iteration of the updating algorithm, where the L^p norm of matrix $\mathbf{X}$ is $\|\mathbf{X}\|_p = (\sum_i \sum_i {X_{ij}}^p)^{\frac{1}{p}}$. As



(a) θ-ndbi (F) (b) θ-ndbi (T) (c) θ-rand (d) IQC^{ndbi}_{rand} metric

Figure 5: *Analysis of parameter θ.*

shown in Figures 3, both $\left\|\mathbf{X}^{(1)}\right\|_1$ and $\left\|\mathbf{X}^{(2)}\right\|_1$ can converge in less than 200 iterations.

5. Parameter Analysis

In method MCD, we have three parameters: $k^{(1)}$, $k^{(2)}$ and θ, where $k^{(1)}$ and $k^{(2)}$ are the numbers of clusters in Foursquare and Twitter networks respectively, while θ is the weight of the normalized discrepancy term in the object function. In the pervious experiment, we set $k^{(1)} = 50$, $k^{(2)} = 50$ and $\theta = 1.0$. Here we will analyze the sensitivity of these parameters in details.

To analyze $k^{(1)}$, we fix $k^{(2)} = 50$ and $\theta = 1.0$ but assign $k^{(1)}$ with values in $\{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. The clustering results of MCD with different $k^{(1)}$ evaluated by *ndbi*, *rand* and IQC^{ndbi}_{rand} metrics are given in Figures 4(a)-4(d). As shown in the figures, the results achieved by MCD are very stable for $k^{(1)}$ with in range $[40, 100]$ under the evaluation of *ndbi* in both Foursquare and Twitter. Similar results can be obtained in Figures 4(c)-4(d), where the performance of MCD on aligned networks is not sensitive to the choice of $k^{(1)}$ for $k^{(1)}$ in range $[40, 100]$ under the evaluation of both *rand* and $IQC_{ndbi,rand}$. In a similar way, we can study the sensitivity of parameter $k^{(2)}$, the results about which are shown in Figures 4(e)-4(h).

An interesting phenomenon is that the predefined number of clusters in the Foursquare network can also affect MCD's performance in the Twitter network. As shown in Figure 4(b), the performance of MCD is the best in the Twitter network when $k^{(1)}$ is assigned with 30, as the *ndbi* score of MCD is the highest when $k^{(1)} = 30$. Figures 4(c)- 4(d) show the performance of MCD under the evaluation of *rand* and $IQC_{ndbi,rand}$. MCD performs the best when $k^{(1)} = 40$ under the evaluation of the *rand* metric and achieves the best performance when $k^{(1)} = 40$(or 90) evaluated by $IQC_{ndbi,rand}$.

To analyze the parameter θ, we set both $k^{(1)}$ and $k^{(2)}$ as 50 but assign θ with values in {0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0}. The results are shown in Figure 5, where when θ is small, e.g., 0.001, the *ndbi* scores achieved by MCD in both Foursquare and Twitter are high but the *rand* score is not good (*rand* is inversely proportional). On the other hand, large θ can lead to good *rand* score but bad *ndbi* scores in both Foursquare and Twitter. As a result, (1) large θ prefers consensus results, (2) small θ can preserve network characteristics and prefers high quality results. Meanwhile, considering the clustering quality and consensus simultaneously, MCD can achieve the best performance when $\theta = 1$, as the IQC^{ndbi}_{rand} is the lowest when $\theta = 1$ in Figure 5(d).

6. Experiments on Large-Sized Aligned Networks

MCD works very well for small-sized aligned networks but cannot be applied to networks involving billions of nodes and links. Meanwhile, MCD-SCALE is introduced in this paper mainly for large-sized aligned networks instead. Extensive experiments have been done on large-scale real-world aligned social networks in this section.

1. Dataset Description

Data sets used are parts of two real social networks: Foursquare and Twitter, which are crawled from May 7th to June 9th. In Foursquare, we crawl 1,064,011 users, among them about 413,182 have

Table 6: *Properties of the Heterogeneous Social Networks*

	network	
property	**Twitter**	**Foursquare**
user	5,223	5,392
friend/follow	164,920	76,972
radius	75	105

their accounts in Twitter as well. Numbers of followers and users of these 413,182 twitter users follow in Twitter are 1,030,855,018 and 187,295,465 respectively. To compare with the baseline method, we sample a smaller sized subnetwork of Twitter, which has about the same size with Foursquare, to perform experiments. The details about data sets can be found in Table 6.

2. Experiment Settings

The baseline method used to compare with the MCD-SCALE framework is an independent network partitioning method-Metis Karypis & Kumar (1995b), which is the state-of-the-art method for single network partitioning by now. With Metis, we first partition all the networks independently. Then for each partition $P_i^{(2)}, 1 \leq i \leq k$ in $G^{(2)}$, we search in each synergistic network $G_j^{(1)} \in G^{(1)}$, $1 \leq j \leq t-1$, the partition $P_j^{(1),l} \subseteq V^{(1)}$ containing the largest number of same kind of anchor users with $P_i^{(2)}$ will be chose to align with $P_i^{(2)}$.

To evaluate the accuracy and the balance performance of the framework MCD-SCALE, we compute the *nmi* of anchor nodes in the datum network and synergistic networks. To measure the edge-cut, we edge-cut ratio to represent the percentage of edges cut off by the partition boundaries.

In this part, all experiments are running on a Hadoop-1.1.1 cluster of Antivision Software Ltd., which consists of 20 PowerEdge R320 servers (Intel Xeon CPU E5-1410 @2.80GHz, memory 8GB) with 64-bit NeoKylin Linux OS connected by a Cisco 3750G-48TS-S switch.

3. Experiment Results

In this section, first, we simply test the performance of MCD-SCALE on datum network

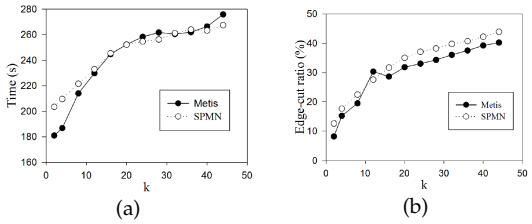


Figure 6: *Comparison of SPMN and Metis on Foursquare network which is chosen as datum network.*

Table 7: *Synergistic Network for Scalability Tests ($\times 10^6$)*

	D5	D6	D7	D8	D9
Node	1.5	1.8	2.1	2.4	2.7
Link	≈127.4	≈152.8	≈178.0	≈203.8	≈230.2

(Foursquare network) and the result is shown in Figure 6. From Figure 6(a), we can see that MCD-SCALE consumes more time than Metis when the partition number k is small. But when k increases, growth of running time slows down. When k climbs to 32, Metis becomes slower than MCD-SCALE. By and large, the increments in both methods are very small. The reason is as follows. The first phase consumes the most time in MKP methods. For MCD-SCALE, time consumption in the first and last phases have nothing to do with k and would be constants. Besides, they are running on MapReduce, so time consumption should be less than Metis. However, the coarsening phase depends two jobs (node pairing and edge renaming) and several iterations, and job initialization in MapReduce would take up great part. The initial partitioning phase takes recursive iterations to generate initial partitions and the number of iterations is linear to k which is about the same with Metis. For Metis, only the first phase doesn't concern partition number. If network size increases steadily, proportion of job initialization will decrease and the total time consumption in MCD-SCALE will be less than Metis. From Figure 6(b), we can see that the edge-cut size in MCD-SCALE is larger than in Metis, that is because Metis adds a refine process during the uncoarsening phase to further optimize the partitions.

Second, we verify the validity and practicability of MCD-SCALE on synergistic network (Twitter network). To increase the number of data sets and

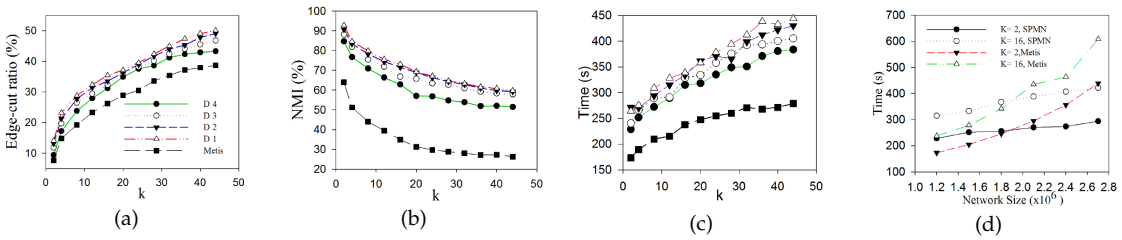


Figure 7: *Performance of SPMN on Synergistic Network. (a) Edge-cut test; (b) NMI test on the partitions of anchor nodes in datum network and synergistic network; (c) Running time test; (d) Scalability test on synergistic networks.*

guarantee the connectedness in all networks, we deliberately and randomly ignore parts of anchor nodes and construct another three data sets which are only different in the number of anchor nodes. D1 has 413, 182 anchor nodes, D2 has 328, 014 anchor nodes, D3 has 203, 491 anchor nodes and D4 has 109, 842 anchor nodes. To test the scalability performance of MCD-SCALE, we enlarge the synergistic network by adding randomly selected non-anchor node set based on D1, and all the non-anchor nodes are from our original Twitter data set which totally contains 1,030,855,018 nodes. The data sets are described in Table 6. With the partition results of anchor nodes in datum network, we conduct experiments on D1-D9 (information about D5-D9 is available in Table 7) and results are shown in Figure 7.

First, we compare MCD-SCALE with Metis. From Figure 7(a) and Figure 7(c), we can see that Metis cuts off less links and runs faster than MCD-SCALE during the network partitioning process. As described in previous section, Metis partitions the datum network and synergistic network independently, which means that it treats all the nodes as non-anchor nodes. MCD-SCALE has to consider the distribution of anchor nodes during the partitioning process. Moreover, Metis' extra refining process during the uncoarsening phase makes it more effective in controlling the edge-cut size and MCD-SCALE's job initialization account for large proportion of the total time consumption.

From the results of NMI and scalability tests shown in Figure 7(b) and Figure 7(d), we can see that MCD-SCALE is more effective than Metis. Higher NMI means that more anchor nodes are assigned to correct partitions and the mapping structure between partitions of different networks is more distinct. In distributed applications, partitions mapped together will be assigned to the same servers. In this situation, data locality will be improved and communication traffic among different servers will be reduced, both of which are very important for large scale data analysis. In traditional MKP methods, coarsening phase consumes the largest part of total time consumption. Metis is a stand-alone method and can not effectively accelerate the coarsening phase. However, MCD-SCALE, by employing the MapReduce computing model, is able to finish the process with relatively less time consumption. So, in terms of scalability, MCD-SCALE performs more excellent than Metis.

Next, we focus on the performance of MCD-SCALE itself. In the Figure 7(a), we can see that for a certain synergistic network, the edge-cut size increases but the speed slows down as partition number grows. For the same partition number k, the more the anchor nodes are, the more the links would be cut off. Besides, for the same k, no significant difference in edge-cut ratio is observed among different synergistic networks. For NMI in Figure 7(b), the value decreases as k increases. When k increases, the probability of assigning an anchor node to incorrect partitions will be higher, then the NMI value will decrease. When k keeps increasing, speed of NMI slows, that is because the other partitions will share responsibility for the incorrect partition results.

For the running time test, compared with EWM scheme, taking the anchor nodes distribution into account in MEWM will add extra workload to coarsening phase and will consume more time to finish the coarsening task. Moreover, for a certain synergistic network, higher k means that more time is required for the LPA to generate initial

partitions. For the scalability test shown in Figure 7(d), we can see that magnitude of increase in time consumption is negligible as k increases. This is attributed to two reasons: 1) there are sufficient servers to process workload in the coarsening phase. 2) the size of the coarsest networks generated in the coarsening phase are similar, meaning time consumption in initial partitioning phase of those networks are close as well. Plus, the uncoarsening phase does not distinguish anchor nodes and non-anchor nodes.

7. Related Work

Clustering is a very broad research area, which include various types of clustering problems, e.g., consensus clustering Lourenço et al. (2013); Lock & Dunson (2013), multi-view clustering Bickel & Scheffer (2004); Cai et al. (2013), multi-relational clustering Yin et al. (2007), co-training based clustering Kumar & Daumé (2011), and dozens of papers have been published on these topics. Lourenco et al. Lourenço et al. (2013) propose a probabilistic consensus clustering method by using evidence accumulation. Lock et al. propose a bayesian consensus clustering method in Lock & Dunson (2013). Meanwhile, Bickel et al. Bickel & Scheffer (2004) propose to study the multi-view clustering problem, where the attributes of objects are split into two independent subsets. Cai et al. Cai et al. (2013) propose to apply multi-view K-Means clustering methods to big data. Yin et al. Yin et al. (2007) propose a user-guided multi-relational clustering method, CrossClus, to performs multi-relational clustering under user's guidance. Kumar et al. propose to address the multi-view clustering problem based on a co-training setting in Kumar & Daumé (2011).

A multi-view clustering paper which is correlated to the problem studied in this paper is Cheng et al. (2013), which relaxes the *one-to-one* constraint in traditional multi-view clustering problems to uncertain mappings. Weights of such mappings need to be decided by prior domain knowledge and each view is actually a homogeneous network. To regularize the clustering results, a cost function called *clustering disagreement* is introduced in Cheng et al. (2013), whose absolute value of all nodes in multiple views is involved in the optimization. Different from Cheng et al. (2013): (1) the constraint on anchor links in this paper is *one-to-one* and no domain knowledge is required, (2) each network involves different users and contains heterogeneous information, (3) we apply clustering discrepancy to constrain the community structures of anchor users only and non-anchor users are pruned before calculating discrepancy cost, and (4) the clustering discrepancy is normalized before being applied in mutual clustering objective function.

Clustering based community detection in online social networks is a hot research topic and many different techniques have been proposed to optimize certain measures of the results, e.g., modularity function Newman & Girvan (2004), and normalized cut Shi & Malik (2000). Malliaros et al. give a comprehensive survey of correlated techniques used to detect communities in networks in Malliaros & Vazirgiannis (2013) and a detailed tutorial on spectral clustering has been given by Luxburg in von Luxburg (2007). These works are mostly studied based on homogeneous social networks. However, in the real-world online social networks, abundant heterogeneous information generated by users' online social activities exist in online social networks. Sun et al. Sun et al. (2009) studies ranking-based clustering on heterogeneous networks, while Ji et al. Ji et al. (2011) studies ranking-based classification problems on heterogeneous networks. Coscia et al. Coscia et al. (2011) proposes a classification based method for community detection in complex networks and Mucha et al. study the community structures in multiplex networks in Mucha et al. (2010).

In recent years, researchers' attention has started to shift to study multiple heterogeneous social networks simultaneously. Kong et al. Kong et al. (2013) are the first to propose the concepts of *aligned networks* and *anchor links*. Across aligned social networks, different social network application problems have been studied, which include different cross-network link prediction/transfer Zhang et al. (2013, 2014a,b); Zhang & Yu (2015b), emerging network clustering Zhang & Yu (2015a) and large-scale network community detection Jin et al. (2014), inter-network information diffusion and influence maximization Zhan et al. (2015).

8. Conclusion

In this paper, we have studied the *mutual community detection* problem across multiple partially aligned heterogeneous online social networks. A novel clustering method, MCD and MCD-SCALE, has been proposed to solve the *mutual community detection* problem. We have proposed a new similarity measure, HNMP-Sim, based on social meta paths in the networks. MCD and MCD-SCALE can achieve very good clustering results in all aligned networks simultaneously with full considerations of network difference problem as well as the connections across networks. Extensive experiments conducted on two real-world partially aligned heterogeneous networks demonstrate that MCD and MCD-SCALE can perform very well in solving the *mutual community detection* problem.

9. Acknowledgement

This work is supported in part by NSF through grants CNS-1115234 and OISE-1129076, Google Research Award, and the Pinnacle Lab at Singapore Management University.

References

Aggarwal, C., Xie, Y., & Yu, P. (2009). Gconnect: A connectivity index for massive disk-resident graphs. *VLDB Endowment*.

Bhattacharya, I., & Getoor, L. (2005). Relational clustering for multi-type entity resolution. In *MRDM*.

Bickel, S., & Scheffer, T. (2004). Multi-view clustering. In *ICDM*.

Bui, T., & Jones, C. (1993). A heuristic for reducing fill-in in sparse matrix factorization. In *PPSC*.

Cai, X., Nie, F., & Huang, H. (2013). Multi-view k-means clustering on big data. In *IJCAI*.

Cheng, W., Zhang, X., Guo, Z., Wu, Y., Sullivan, P., & Wang, W. (2013). Flexible and robust co-regularized multi-domain graph clustering. In *KDD*.

Coscia, M., Giannotti, F., & Pedreschi, D. (2011). A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining*.

Dean, J., & Ghemawat, S. (2008). Mapreduce: Simplified data processing on large clusters. *Communications of the ACM*.

Goder, A., & Filkov, V. (2008). Consensus Clustering Algorithms: Comparison and Refinement. In *ALENEX*.

Hasan, M., & Zaki, M. J. (2011). A survey of link prediction in social networks. In C. C. Aggarwal (Ed.) *Social Network Data Analytics*.

Ji, M., Han, J., & Danilevsky, M. (2011). Ranking-based classification of heterogeneous information networks. In *KDD*.

Jin, S., Zhang, J., Yu, P., Yang, S., & Li, A. (2014). Synergistic partitioning in multiple large scale social networks. In *IEEE BigData*.

Karypis, G., & Kumar, V. (1995a). Analysis of multilevel graph partitioning. In *Supercomputing*.

Karypis, G., & Kumar, V. (1995b). Metis – unstructured graph partitioning and sparse matrix ordering system, version 2.0. Tech. rep.

Karypis, G., & Kumar, V. (1996). Parallel multilevel k-way partitioning scheme for irregular graphs. In *Supercomputing*.

Karypis, G., & Kumar, V. (1998). Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing*.

Kong, X., Zhang, J., & Yu, P. (2013). Inferring anchor links across multiple heterogeneous social networks. In *CIKM*.

Kumar, A., & Daumé, H. (2011). A co-training approach for multi-view spectral clustering. In *ICML*.

Li, T., Ding, C., & Jordan, M. I. (2007). Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In *ICDM*.

Liu, Y., Li, Z., Xiong, H., Gao, X., & Wu, J. (2010). Understanding of internal clustering validation measures. In *ICDM*.

Lock, E. F., & Dunson, D. B. (2013). Bayesian consensus clustering. *Bioinformatics*.

Lourenço, A., Bulò, S. R., Rebagliati, N., Fred, A. L. N., Figueiredo, M. A. T., & Pelillo, M. (2013). Probabilistic consensus clustering using evidence accumulation. *Machine Learning*.

Malliaros, F. D., & Vazirgiannis, M. (2013). Clustering and community detection in directed networks: A survey. *CoRR*.

Mucha, P., Richardson, T., Macon, K., Porter, M., & Onnela, J. (2010). Community structure in time-dependent, multiscale, and multiplex networks. *Science*.

Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*.

Nguyen, N., & Caruana, R. (2007). Consensus clusterings. In *ICDM*.

Qi, G., Aggarwal, C., & Huang, T. (2012). Community detection with edge content in social media networks. In *ICDE*.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *TPAMI*.

Shvachko, K., Kuang, H., Radia, S., & Chansler, R. (2010). The hadoop distributed file system. In *MSST*.

Sun, Y., Han, J., Yan, X., Yu, P., & Wu, T. (2011). Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *PVLDB*.

Sun, Y., Yu, Y., & Han, J. (2009). Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD*.

Tsikerdekis, M., & Zeadally, S. (2014). Multiple account identity deception detection in social media using nonverbal behavior. *Information Forensics and Security, IEEE Transactions on*.

von Luxburg, U. (2007). A tutorial on spectral clustering. *CoRR*.

Wen, Z., & Yin, W. (2010). A feasible method for optimization with orthogonality constraints. Tech. rep., Rice University.

Wu, Z., & Leahy, R. (1993). An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *TPAMI*.

Yin, X., Han, J., & Yu, P. (2007). Crossclus: user-guided multi-relational clustering. *Data Mining and Knowledge Discovery*.

Zhan, Q., Zhang, J., Wang, S., Yu, P., & Xie, J. (2015). Influence maximization across partially aligned heterogenous social networks. In *PAKDD*.

Zhang, J., Kong, X., & Yu, P. (2013). Predicting social links for new users across aligned heterogeneous social networks. In *ICDM*.

Zhang, J., Kong, X., & Yu, P. (2014a). Transferring heterogeneous links across location-based social networks. In *WSDM*.

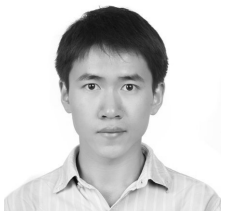
Zhang, J., & Yu, P. (2015a). Community detection for emerging networks. In *SDM*.

Zhang, J., & Yu, P. (2015b). Integrated anchor and social link predictions across partially aligned social networks. In *IJCAI*.

Zhang, J., Yu, P., & Zhou, Z. (2014b). Meta-path based multi-network collective link prediction. In *KDD*.

Zhou, Y., & Liu, L. (2013). Social influence based clustering of heterogeneous information networks. In *KDD*.

Authors:



Jiawei Zhang received the PhD degree in Computer Science from University of Illinois at Chicago, USA, in 2017. He received the bachelors degree in computer science from Nanjing University, China, in 2012. He has been working as an assistant professor in the Department of Computer Science in Florida State University since 2017. His main research areas are data mining and machine learning, especially multiple aligned social networks studies.

Songchang Jin received the B.S. degree in automation at Tsinghua University in 2008, and received the M.S. degree in computer science and technology at National University of Defense Technology (NUDT) in 2010. Now he is a Ph.D. candidate in Computer science and technology at NUDT. He visited the University of Illinois at Chicago in 2014. His research interests include big data analysis, social network analysis. He is an IEEE member.

Philip S. Yu received the BS degree in electrical engineering from National Taiwan University, the MS and PhD degrees in EE from Stanford University, and the MBA degree from New York University. He is a distinguished professor of computer science with the University of Illinois at Chicago and also holds the Wexler chair in information technology. Before joining UIC, he was with IBM, where he was the manager in Software Tools and Techniques Group, Watson Research Center. His research interest is on big data, including data mining, data stream, database, and privacy. He has published more than 970 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. He is the editor-in-chief of the ACM Transactions on Knowledge Discovery from Data. He is on the steering committee of the ACM Conference on Information and Knowledge Management and was a member of the steering committee of IEEE Data Engineering and IEEE Conference on Data Mining. He was the editor-in-chief of the IEEE Transactions on Knowledge and Data Engineering (2001-2004). He received the ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions on mining, fusion and anonymization of big data, the IEEE Computer Societys 2013 Technical Achievement Award for pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining and anonymization of big data, and the Research Contributions Award from IEEE International Conference on Data Mining (ICDM) in 2003 for his pioneering contributions to the field of data mining. He also received the ICDM 2013 10-year Highest-Impact Paper Award, and the EDBT Test of Time Award (2014). He has received several IBM honors including two IBM Outstanding Innovation Awards, an Outstanding Technical Achievement Award, two Research Division Awards, and the 94th plateau of Invention Achievement Awards. He was an IBM Master Inventor. He is a fellow of the ACM and IEEE.