Mach Learn (2016) 104:1–24
DOI 10.1007/s10994-016-5544-1

A novel probabilistic clustering model for heterogeneous networks

Zhi-Hong Deng[1] · **Xiaoran Xu**[2]

Received: 5 April 2015 / Accepted: 9 January 2016 / Published online: 5 February 2016

Abstract Heterogeneous networks, consisting of multi-type objects coupled with various relations, are ubiquitous in the real world. Most previous work on clustering heterogeneous networks either converts them into homogeneous networks or simplifies the modeling of the heterogeneity in terms of specific objects, structures or assumptions. However, few studies consider all relevant objects and relations, and trade-off between integrating relevant objects and reducing the noises caused by relations across objects. In this paper, we propose a general probabilistic graphical model for clustering heterogeneous networks. First, we present a novel graphical representation based on our basic assumptions: different relation types produce different weight distributions to specify intra-cluster probability between two objects, and clusters are formed around cluster cores. Then, we derive an efficient algorithm called PROCESS, standing for PRObabilistic Clustering modEl for heterogeneouS networkS. PROCESS employs a balance-controlled message passing algorithm and mathematical programming for inference and estimation. Experimental results show that our approach is effective and significantly outperforms the state-of-the-art algorithms on both synthetic and real data from heterogeneous networks.

Keywords Clustering · Heterogeneous networks · Probabilistic graphical model · Algorithm

Editor: Xiaoli Fern.

✉ Zhi-Hong Deng
zhdeng@cis.pku.edu.cn

[1] Key Laboratory of Machine Perception (Ministry of Education), School of Electronics Engineering and Computer Science, Peking University, Beijing, China

[2] Department of Computer Science, University of California at Los Angeles, Los Angeles, CA, USA

1 Introduction

Most real-world networks are heterogeneous, incorporating multi-type objects associated with multi-type relations. For instance, the usual bibliographic networks consist of papers, authors, publications (journals or conferences) and terms, which are associated with each other in terms of various relations such as paper-paper citation relations and author-paper authoring connections. Other typical heterogeneous networks include sensor networks, social networks, and transport networks. It is becoming more and more important and essential to analyze the heterogeneous objects and relations in such networks.

Clustering analysis is a practical and indispensable way to explore the structure of heterogeneous networks. The existing approaches for handling the heterogeneity of objects and relations in such networks intend to convert the data into a simpler format in the following ways. One focuses on how to embed objects in networks into an Euclidean space and transform each object to a multi-dimensional data point, so that the objects can be clustered by typical methods such as k-means. Such transformations are usually implemented based on the adjacent matrix representing the network via multi-dimensional vectors, ignoring explicit dependence across dimensions. Taking bibliographic networks as an example, in order to cluster papers, each row of the adjacent matrix of the "paper–term" relations may be converted into a vector to represent a paper, neglecting certain forms of "term–term" relations existing in the network. This method is called multi-dimensional clustering, including k-means (Hartigan and Wong 1979) and PLSA (Hofmann 1999). The other transforms various relations across heterogeneous objects into a homogeneous network with unique relations. For example, regarding papers as the clustering target, both the "paper–author–paper" authoring relations and the "paper–paper" citation relations are converted into "paper–paper" relations for clustering without differentiating the two types of relations. This approach is called graph partition, including the Kernighan–Lin algorithm (Kernighan and Lin 1970), Girvan–Newman algorithm (Girvan and Newman 2002), and spectral partitioning algorithm (Ng et al. 2001). In addition, algorithms such as NetClus (Sun et al. 2009b) and BibClus (Xu and Deng 2011) handle the heterogeneity based on or requiring specific data structures, lacking generality for clustering heterogeneous networks.

The motivation behind the above approaches is to reduce the heterogeneity of relations and simplify the presentation of objects, by a multi-dimensional vector or in a homogeneous network, so that most of existing approaches can be applied. In fact, due to the interaction between multi-type objects guided by multi-type relations, besides objects as clustering target, it is often necessary to explicitly consider the contributions by those non-target associated objects, and relations between target objects and non-target objects or even between non-target objects themselves. This involves the necessity of considering coupling relationships (Cao et al. 2012) between heterogeneous objects and between relations and objects.

In this paper, we present a novel probabilistic graphical model and derive an efficient algorithm, called PROCESS, for clustering heterogeneous networks. PROCESS performs inference and estimation iteratively under EM framework. The inference implements clustering reassignment for each object by using the balance-controlled message passing algorithm improved from the original one in Kschischang et al. (2001). The estimation updates model parameters for each cluster by transforming the original problem into several mathematical optimization problems, such as 0/1 integer programming (Nemhauser and Wolsey 1988). PROCESS considers both target and attribute objects, brings about a good trade-off between incorporating attribute objects and their possible noises to the clustering, and treats relations between targets and attributes differently and adaptively. Substantial experiments on both syn-

thetic and real datasets demonstrate the effectiveness of PROCESS and show that PROCESS outperforms the state-of-the-art algorithms for clustering heterogeneous networks.

The main contributions of our work are summarized as follows:

1. We handle the heterogeneity of objects in the network directly, without any simplification or reduction, through considering all associated types of objects and relations.
2. We study a mechanism to treat various relations in the network distinctively by learning relation weights automatically, which lead to a good trade-off between incorporating attribute objects and reducing noises.
3. We analyze the structure properties in a cluster, and propose the concept of cluster core, size of which can be adjusted dynamically to avoid an extremely unbalanced clustering.
4. We develop a novel inference approach based on message passing algorithm and it prove to be effective by extensive experiments.

The remainder of the paper is organized as follows. Section 2 introduces related work and discusses their limitations in clustering heterogeneous networks. The graphical model is proposed in Sect. 3. The PROCESS algorithm and its working mechanism are presented in Sect. 4. Experimental results and evaluation are presented in Sect. 5. Finally, Sect. 6 summarizes our work and points out promising directions for future research.

2 Related work

Clustering analysis of networks is a promising technique in studying complex networks. A typical network clustering method is graph clustering, also called node clustering, since a network is usually regarded as a simple graph consisting of nodes with edges. Node-clustering algorithms are generalizations of multi-dimensional clustering algorithms like k-means (Hartigan and Wong 1979). They use functions of the transformed multi-dimensional data points to define the distances and then minimize the inter-cluster similarity. A graphical version of k-medoids (Rattigan et al. 2007) has been proposed in this way. Mostly, graph clustering reduces to the problem of graph partition, aiming to partition the graph to minimize the weights of edges across the partitions. This method has been widely studied in different ways, such as the Kernighan–Lin algorithm (Kernighan and Lin 1970), the spectral clustering (Ng et al. 2001; Shiga et al. 2007), the min–max cut (Papadimitriou and Steiglitz 1998), the Girvan–Newman algorithm (Girvan and Newman 2002), and many other methods optimizing different predefined criteria (Aggarwal and Wang 2010). In addition, affinity propagation (Frey and Dueck 2007) is an exemplar-based clustering algorithm proposed recently, attracting much attention. It overcomes the sensitiveness of k-means to the initial points, and use the similarity between each pair of data points as input, leading to its flexibility to networked data. However, most of them focus on homogeneous networks and simplify the heterogeneous relations into homogeneous one for clustering.

Clustering heterogeneous networks is an emerging research topic, which is attracting more and more attention (Sun et al. 2009b; Xu and Deng 2011; Deng et al. 2011, 2013; Yu et al. 2014). Clustering heterogeneous networks is also called relational clustering of heterogeneous relational data (Philip 2010). There are two typical paradigms: deterministic approach and generative approach. The former mainly uses spectral analysis (Long et al. 2006) or modularity analysis (Tang et al. 2009b) based on collective matrix factorization or cross-dimension interaction, which leads to the computation of singular vectors or eigenvectors of certain graph affinity matrices. These methods preset the weights of different relations rather than learn adaptively, which may not reflect the data characteristics properly. The latter can

be traced from the well-known topic model, the Latent Dirichlet Allocation model (Blei et al. 2003). Such approaches usually have difficulty in making full use of all types of objects and relations. As a consequence, most of them just utilize partial network structure to conduct a generative process.

Probabilistic approaches are recently studied in network or graph clustering (Sun et al. 2009b, a, 2012b, a; Xu and Deng 2011; Xu et al. 2012; Zhou and Liu 2013; Perozzi et al. 2014). NetClus (Sun et al. 2009b), BibClus (Xu and Deng 2011) and PathSelClus (Sun et al. 2012a) are three typical probabilistic generative models for clustering general heterogeneous networks. NetClus and BibClus are constrained on specific data structures. NetClus needs a star network schema and BibClus requires a center linkage structure. PathSelClus need user to first provide a small set of object seeds for each cluster as guidance. Approaches proposed in Sun et al. (2009a, 2012b), Zhou and Liu (2013) handle the issue of clustering on special heterogeneous networks, such as bi-typed heterogeneous networks. The mixed membership relational clustering model (Long et al. 2007) is a more general generative model under a large number of exponential family distribution. However, it doesn't discriminate target objects from attribute objects, and will be unable to determine objects of which type should be clustered clearly, failing to reduce possible noises. There are other methods dealing with multi-type relations between homogeneous objects, called clustering with multiple graphs (Tang et al. 2009a), which have a limited application.

In summary, the dependency between heterogeneous objects and relations are not fully considered in existing methods. As far as we know, there is no work providing a systematic solution addressing the aforementioned challenges in clustering heterogeneous networks. In addition, the current methods tend to specify the contribution of each relation rather than automatically determining it by learning the data characteristics.

3 The proposed model

3.1 Motivation

Before introducing our model, we first explore the critical problem in clustering heterogeneous networks as shown in Fig. 1. Considering the network structure in Fig. 1a, it is a homogeneous network including single type of objects. Our task is to cluster these objects by using associated relations in the network. Obviously, the objects should be assigned to three clusters marked with dashed circles.

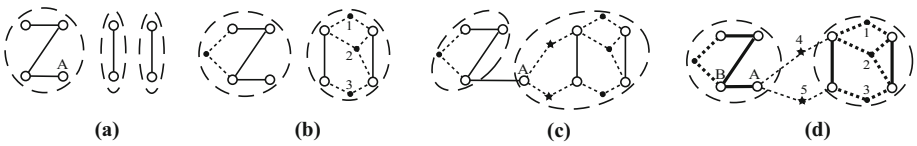


Fig. 1 **a** A homogenous network with target objects only. **b** An extended heterogenous network by adding four attribute objects. These objects provide supportive information for target objects clustering. **c**, **d** Another heterogenous network, which is built by adding two noisy attribute objects to the network in **b**. We use traditional graph partition algorithms to cluster the network in **c**. We treat two types of relations differently and use a heterogenous clustering method in the network of **d**. In these three networks, *big white circles* represent target objects, and *small black spots or stars* represent attribute objects. Among them, star-shape objects are noises, and spot-shape objects provide useful information

The above relations we take into account indicate direct interactions between objects. However, unconnected objects may be related as well, by possessing common attributes. For instance, papers without citation relation may own the same authors, terms, etc. To utilize these attributes, we introduce objects of more types to represent them, called *attribute objects*, and add them to the above homogeneous network. The original objects to be clustered are called *target objects*. It is worth mentioning that attribute objects provide extra information for clustering, but some of them may not belong to any cluster. Figure 1b shows a heterogeneous network by extending the homogeneous network in Fig. 1a, and introducing attribute objects represented by small black spots. Adding objects 1, 2 and 3 makes us confirm that the two right clusters in Fig. 1a should be merged into one. This indicates that bringing attribute objects into consideration supplements the insufficiency of the relation information in homogeneous networks.

However, can the addition of attribute objects guarantee a better clustering result? In Fig. 1c, another two attribute objects (star-shape) are brought into the network. Although the relations, including target-target relation (solid lines) and target-attribute relation (dashed lines), have been enriched, the network may be easily clustered in an inappropriate way as shown in Fig. 1c by using a graph partition algorithm. It treats all relations equally, and assigns object A to the right cluster instead of the left one. This error is caused by the lack of considering different roles played by different relations in the partition. It does not cater for the fact that relations between target objects carry more discriminative information than target-attribute ones in clustering, because the latter is only an auxiliary to the clustering of target objects.

As shown in Fig. 1d, to obtain an ideal clustering result, we consider that the weighting of relations between targets is higher than that of the target-attribute relations. Note that the relation weights are usually learned from the network structure rather than be pre-specified. Consequently A is assigned to the left cluster, because the relation between A and B is stronger than $A-4$ and $A-5$. We regard attribute objects like 4 and 5 as noises since they bring inter-cluster relations. On the contrary, attribute objects 1, 2 and 3 make us confirm that the four target objects on the right are in the same cluster.

The above example shows the following challenges in clustering heterogeneous networks:

- It is necessary to consider all associated objects of multiple types rather than just the target objects, and consider various relations rather than just relations between target objects in the clustering;
- Different relations, including target–target, target–attribute and attribute–attribute relations, play varying roles, which should be treated distinctively by learning relation weights adaptively and automatically in clustering;
- Attribute objects could contribute to the clustering of targets but might also bring noises leading to wrong clustering outcomes. This will lead to challenges in finding a proper trade-off between involving heterogeneous objects and reducing noises caused by relations across objects in different clusters.

In this paper, in order to develop a general and effective algorithm for clustering a heterogeneous network, we aim to take all associated types of objects and relations into account and treat them differently. Also, we study the structural properties in a cluster and characterize them well. Then, we employ two basic assumptions in our model:

- We regard relation weight as a random parameter to determine the probability of two connected objects belonging to the same cluster. To deal with multiple relation types discriminatively, hyper-parameters are introduced to allow relation weights of different types obey different probability distributions.

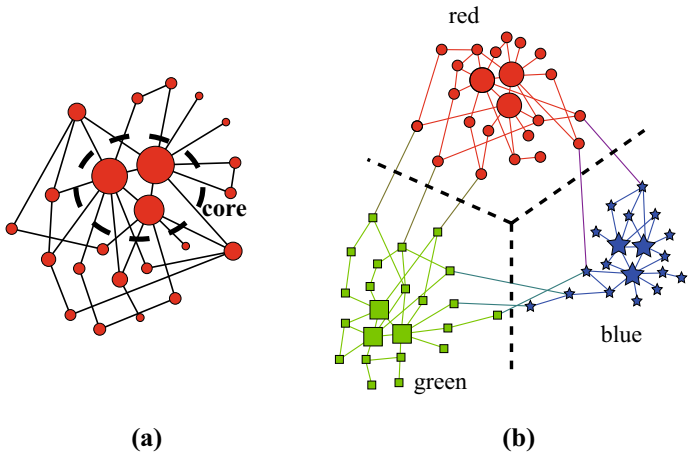


Fig. 2 **a** A single community which contains three core members encircled in the *dashed oval*, and the rest members linked around the core. **b** A clustered social network, in which one color represents one cluster. These core objects in each cluster, marked with larger *circles*, make each cluster tight and compact

– We define a *cluster core* consisting of a small number of objects in each cluster, since most of clusters possess a central structure. The rest objects form clusters centered on these cores.

For the first assumption, intuition tells us that relations are more likely to be created between intra-cluster objects than between inter-cluster objects, thus it is crucial to bring in more relations for clustering. However, some types of relations, linked objects mostly belonging to different clusters, may produce much noises rather than useful information, which may determine a small relation weight. In a word, by adjusting relation weights, we allow each relation type to play different roles in clustering. Here, under each relation type, the relation weights follow a distribution controlled by a specific hyper-parameter. These hyper-parameters are learned by the maximum likelihood estimation of objects forming several clusters giving the current network structure.

For the second assumption, we find that in a social network, a real community or cluster usually contains a central structure, composed of a minority of members or objects. Figure 2a shows that these members encircled in the dashed oval act as a pivot of the community, which we define as community cores or cluster cores. These members connect with many of the rest members. Figure 2b shows a clustered social network, in which each cluster possesses a cluster core, including three objects marked with larger circles. Then, objects can surround these cluster cores by linking to them directly or indirectly, and eventually form three clear clusters which are tight and compact.

Figure 2b shows that each cluster core is comprised of three objects, but in a more general case, the object number of each cluster core, i.e. cluster core size, is uncertain. How do we specify the size of each cluster core? How differently will it impact clustering results by assigning one or more objects to a cluster core? Given the network containing two clusters shown in Fig. 3a, suppose we assign one object, represented by the large circle, to each cluster core. Since the area around the green cluster core has a much higher relation density than the area around the red one, the green cluster inclines to an overwhelming coverage, leading to an extremely unbalanced clustering. However, by adding another object to the red cluster core, we obtain a relatively balanced clustering shown in Fig. 3b. Also, we observe that when

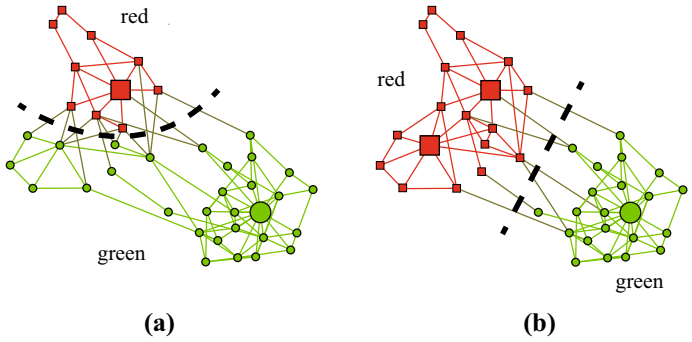


Fig. 3 **a** Two unbalanced clusters containing only one core object. **b** A relatively balanced clustering result by adding one core object to the *red* cluster. With two separated core objects, the *red* cluster looks less tight than the *green* one (Color figure online)

a cluster core has multiple separated objects, the corresponding cluster might become less tight like the red one. Through the above analysis, we know that, due to the different tightness of each cluster, the size and distribution of each cluster core should be adjusted dynamically to avoid an extremely unbalanced clustering, caused by a fixed core size.

Based on the above discussion, we build our model in the next subsection.

3.2 Model representation

In this section, we present a probabilistic graphical model for heterogeneous network clustering. Formally, we introduce some basic concepts and notations used throughout the paper.

Definition 1 Heterogeneous Network: A network $G = \langle V, E \rangle$ is a *heterogeneous network*, if $V = \bigcup_{s=1}^{S} V_s$ denotes a set of objects of S types, and $E = \bigcup_{t=1}^{T} E_t$ consists of T types of relations. $\mathfrak{S} = \{1, 2, \ldots, S\}$ represents all object types, and $\mathfrak{T} = \{1, 2, \ldots, T\}$ represents all relation types. For object types $s_1, s_2 \in \mathfrak{S}$, $t^{(s_1,s_2)} \in \mathfrak{T}$ indicates the corresponding relation type, if there are edges between V_{s_1} and V_{s_2}.

Definition 2 Target and Attribute Types: Given a heterogeneous network G, $s^* \in \mathfrak{S}$ is called *target type*, if objects in V_{s^*} are specified as the clustering target. The rest object types $s \in \mathfrak{S} \setminus \{s^*\}$ play an auxiliary role in clustering, called *attribute types*.

The two definitions indicate that heterogeneous network clustering considers and merges the relation information across both homogeneous and heterogeneous objects. Due to the heterogeneity, with more useful information carried, abundant noises would be brought in as well. When we specify target objects, we should assign them into different clusters clearly against noises carried by attribute objects, which may blur cluster borders.

We use a generative model to characterize a heterogeneous network. First, we introduce some notations as follows:

Cluster core parameters

- $C = \{C_1, \ldots, C_K\}$: the set of cluster cores, where $C_k \subseteq V_{s^*}$, consisting of a few target objects, is also a subset of the kth cluster.
- $N = \{N_1, \ldots, N_K\}$: the sizes of cluster cores in C, where $N_k = |C_k|$ is usually a small number to guarantee a tight and compact cluster.

Relation weight parameters

- $W = \{W_{ij} \geq 0 | i, j \in V, i \neq j\}$: the set of relation weights, where each pair of objects i and j has a W_{ij}.
- $\beta = \{\beta_1, \ldots, \beta_T\}$: the set of hyper-parameters of the distributions on W, where β_t controls the relation weight distribution under relation type t. β can be learned to characterize the heterogeneity for different relation types.

Variables

- $X = \{X_{ij} \in \{0, 1\} | i, j \in V, i \neq j\}$: the set of observed Boolean relation states, reflecting whether a real relation exists, where for each pair of objects $i \in V_{s_1}$ and $j \in V_{s_2}$, $X_{ij} = 1$ if $(i, j) \in E_{t(s_1,s_2)}$, otherwise $X_{ij} = 0$.
- $Z = \{Z_i \in \{1, \ldots, K\} | i \in V\}$: the set of hidden cluster assignments, where Z_i stands for the cluster which object i belongs to.

Note that, we view X as observed variables, Z as hidden variables, C, W and β as estimated parameters, and N as the self-adapting parameter.

The generative process of generating a directed graphical model is as follows:

1. For each cluster k
   (a) Randomly sample core C_k from V_{s^*} with $|C_k| = N_k$
2. For each object i
   (a) If $i \in V_{s^*}$, then sample $Z_i \sim P(Z_i|C)$
   (b) Else sample Z_i from the uniform distribution on $\{1, \ldots, K\}$.
3. For each pair i and j with relation type t
   (a) Sample $W_{ij} \sim P(W_{ij}|\beta_t)$
   (b) Sample $X_{ij} \sim P(X_{ij}|Z_i, Z_j, W_{ij})$

Figure 4 shows the directed graphical model produced by the generative process. To conduct this process, we define the conditional probabilities mentioned above as:

– $P(Z_i|C)$: This probability indicates cluster assignment of a target object depending on cluster cores. For each Z_i, $i \in V_{s^*}$, $P(Z_i|C)$ is defined as:

$$P(Z_i = y|C) = \begin{cases} 1/K & i \notin \bigcup_k C_k \\ \varepsilon & i \in \bigcup_k C_k \wedge i \notin C_y \\ 1 - (K-1)\varepsilon & i \in C_y \end{cases}, \tag{1}$$

where $0 < \varepsilon \ll 1/K$ is a small quantity. The above formula indicates that the objects in cluster cores C should be assigned into one cluster with an extremely high probability, while the rest target objects have the uniform probability $1/K$ to belong to any cluster. These temporary assignments are derived only from cluster cores C without considering the network structure.

– $P(W_{ij}|\beta_t)$: This probability allows that relation weights of multiple types obey different distributions controlled by hyper-parameters, so that various relations can be treated differently. For each pair of objects i and j with their relation type t, since W_{ij} is non-negative, we need a unimodal distribution to characterize it, and the gamma distribution is a frequently used choice. We assume a gamma distribution for $P(W_{ij}|\beta_t)$ as follows:

$$P(W_{ij}|\beta_t) = \text{Gamma}(W_{ij}|\kappa, \beta_t\theta), \tag{2}$$

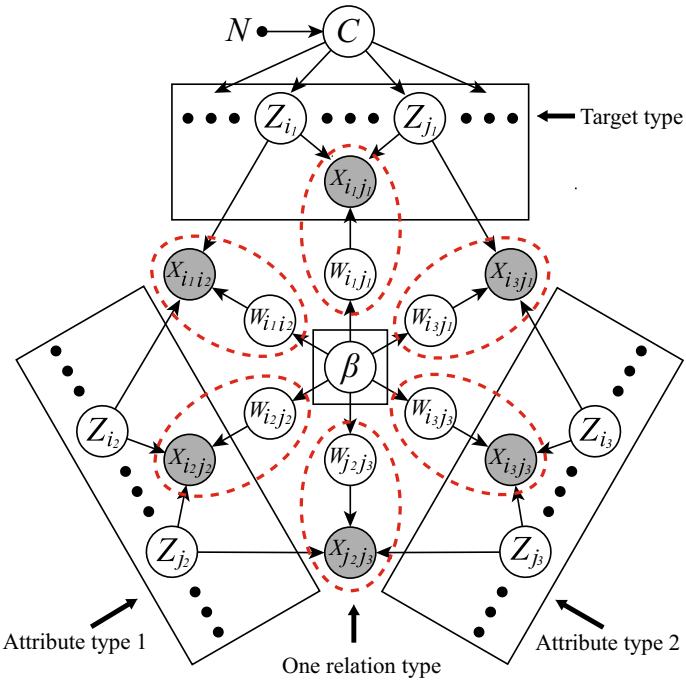


Fig. 4 The probabilistic graphical model for heterogeneous network clustering. The blank nodes in the *upper box* describe target objects, while the ones in the two *lower boxes* represent two types of attribute objects respectively. *Six red dashed ovals* represent six types of relations, including relations between homogeneous objects and between heterogeneous objects (Color figure online)

where κ, θ are fixed and $\beta_t > 0$. Here, we impose a harmonic average constraint as:

$$\frac{1}{T}\sum_t \frac{1}{\beta_t} = 1, \tag{3}$$

aiming to restrict the relative differences between β values to a fixed average level and also for the convenience of calculation.

– $P(X_{ij}|Z_i, Z_j, W_{ij})$: This probability implies the probability difference of creating a connection between two inter-cluster objects and between two intra-cluster objects. By using the Bayes' rule, we can derive $P(X_{ij}|Z_i, Z_j, w)$ from $P(Z_i, Z_j|X_{ij}, W_{ij})$ and $P(X_{ij})$ as:

$$P(X_{ij}|Z_i, Z_j, W_{ij}) \propto P(X_{ij})P(Z_i, Z_j|X_{ij}, W_{ij}). \tag{4}$$

For simplicity, we denote X_{ij} as x and W_{ij} as w for short. To derive $P(Z_i, Z_j|x, w)$, we rely on a basic intuition that objects i and j are more likely to share the same cluster membership if an edge with certain weight w exists between i and j, and define the probability as:

$$P(Z_i = u, Z_j = v|x, w) = \begin{cases} 1/K^2 & x = 0 \\ e^{wI(u=v)}/\gamma & x = 1 \end{cases}, \tag{5}$$

where $\gamma = K(e^w + K - 1)$ is a normalization constant. By substituting Eq. (5) into Eq. (4), we obtain the reduced formula as:

$$P(x|Z_i, Z_j, w) = \begin{cases} p_1 x + (1 - p_1)(1 - x) & Z_i = Z_j \\ p_2 x + (1 - p_2)(1 - x) & Z_i \neq Z_j \end{cases}, \tag{6}$$

where

$$\begin{aligned} p_1 &= K\eta/\big(K\eta + 1 + (K - 1)e^{-w}\big), \\ p_2 &= K\eta/\big(K\eta + K - 1 + e^{w}\big). \end{aligned} \tag{7}$$

Here, $0 < \eta = \frac{P(x=1)}{P(x=0)} \ll 1$, reflecting the relation density of the network, is usually an extremely small quantity, since most real-world networks are sparse.

In summary, we can obtain the fully joint probability as follows:

$$\begin{aligned} &P(Z, X, C, W; \beta, N) \\ &= \prod_{i \in V_s *} P(Z_i|C) \prod_{i<j} \Big(P(W_{ij}|\beta) P(X_{ij}|Z_i, Z_j, W_{ij}) \Big). \end{aligned} \tag{8}$$

4 PROCESS algorithm, its inference, and estimation

In this section, we take advantage of the typical EM framework to implement the clustering procedure through inference and estimation, and then develop an effective algorithm called PROCESS.

4.1 Clustering procedure and the EM framework

Generally, a clustering procedure contains two steps in each iteration:

1. reassign each object to a new cluster,
2. update model parameters for further clustering.

Such two steps are proceeded repeatedly until some criterion for convergence is achieved. Many existing clustering algorithms are derived from diversified ways for cluster reassignment and parameter update. As a simple example, k-means reassigns each object based on its distance to the cluster centers and updates the vectors of center points in each iteration. In this paper, the model parameters contains cluster core parameters as C, N and relation weight parameters as W, β, which are learned by using maximum likelihood estimation (MLE) under the EM framework.

The EM framework can be applied to learn the above model parameters, and consists of two iterative steps:

- *E-step* calculate the expected value of the log likelihood function by marginal inference for hidden variables, corresponding to the clustering step of cluster reassignment.
- *M-step* estimate the model parameters by maximizing the above expected value, corresponding to the clustering step of parameter update.

Then, we denote the estimated parameters as $\Theta = (C, W, \beta)$. In E-step, the expected value of the log likelihood function with respect to the conditional distribution of Z given X under $\Theta^{(t)}$ can be described as:

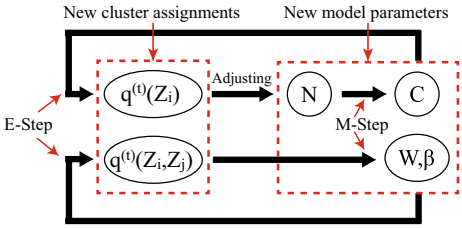


Fig. 5 The solving process of our model based on the EM framework

$$
\begin{aligned}
Q(\Theta|\Theta^{(t)}) &= \mathbb{E}_{Z|X;\Theta^{(t)}}[\log P(Z, X, C, W; \beta, N)] \\
&= \sum_{i \in V_s*} \sum_{Z_i} q^{(t)}(Z_i) \log P(Z_i|C) + \sum_{i<j} \Big(\log P(W_{ij}|\beta) + \\
&\quad \sum_{Z_i, Z_j} q^{(t)}(Z_i, Z_j) \log P(X_{ij}|Z_i, Z_j, W_{ij}) \Big),
\end{aligned} \tag{9}
$$

where $q^{(t)}(Z) := P(Z|X; \Theta^{(t)})$. In M-step, we estimate the parameters Θ as:

$$
\Theta^{(t+1)} = \arg\max_{\Theta} Q(\Theta|\Theta^{(t)}). \tag{10}
$$

Note that we have constraints for C and β defined as:

$$
|C_k| = N_k, \ k = 1, \ldots, K \quad \text{and} \quad \frac{1}{T} \sum_t \frac{1}{\beta_t} = 1. \tag{11}
$$

The working process of our model based on the EM framework can be illustrated in Fig. 5. Below, we will explain the marginal inference in E-step and the parameter estimation in M-step respectively.

4.2 Inference for $q^{(t)}(Z_i)$ and $q^{(t)}(Z_i, Z_j)$

In Eq. (9), we need to calculate the marginal probabilities $q^{(t)}(Z_i)$ and $q^{(t)}(Z_i, Z_j)$ from $q^{(t)}(Z) = P(Z|X; \Theta^{(t)})$ in each iteration. The probability $q^{(t)}(Z)$ involves the number of pairs of disconnected objects, which is much larger than that of the edges, and can be written as:

$$
\begin{aligned}
q^{(t)}(Z) = &\prod_{i \in V_s*} P(Z_i|C) \prod_{(i,j) \in E} P(X_{ij} = 1|Z_i, Z_j, W_{ij}) \\
&\prod_{(i,j) \notin E} P(X_{ij} = 0|Z_i, Z_j, W_{ij}).
\end{aligned} \tag{12}
$$

Since hidden variables Z in $q^{(t)}(Z)$ are fully coupled, it is intractable to calculate the marginal probabilities. We denote $P(X_{ij} = 0|Z_i, Z_j, W_{ij})$ as $p_{0|=}$ if $Z_i = Z_j$, otherwise $p_{0|\neq}$, written as:

$$
P(X_{ij} = 0|Z_i, Z_j, W_{ij}) = \begin{cases} \frac{1+(K-1)e^{-W_{ij}}}{K\eta+1+(K-1)e^{-W_{ij}}} = p_{0|=} & Z_i = Z_j \\ \frac{K-1+e^{W_{ij}}}{K\eta+K-1+e^{W_{ij}}} = p_{0|\neq} & Z_i \neq Z_j \end{cases}, \tag{13}
$$

where we have:

$$
\frac{1}{K\eta+1} \leq \frac{p_{0|=}}{p_{0|\neq}} = \frac{1}{K\eta+1} \Big(1 + \frac{K^2\eta(\eta+1)}{(K\eta+1)e^w + K - 1} \Big) \leq 1. \tag{14}
$$

Since $0 < \eta \ll 1$ reflects the network edge density, $\frac{p_{0|=}}{p_{0|\neq}}$ can be restricted to a very narrow range below but close to 1, and $\frac{p_{0|=}}{p_{0|\neq}} \approx 1$. Due to $p_{0|=}$ being slightly smaller than $p_{0|\neq}$, $\prod_{(i,j)\notin E} P(X_{ij} = 0|Z_i, Z_j, W_{ij})$, representing the joint distribution of a large amount of disconnected objects, tends to assign the objects separately to different clusters. Similarly, we denote $P(X_{ij} = 1|Z_i, Z_j, W_{ij})$ as $p_{1|=}$ if $Z_i = Z_j$, otherwise $p_{1|\neq}$, written as:

$$P(X_{ij} = 1|Z_i, Z_j, W_{ij}) = \begin{cases} \frac{K\eta}{K\eta+1+(K-1)e^{-W_{ij}}} = p_{1|=} & Z_i = Z_j \\ \frac{K\eta}{K\eta+K-1+e^{W_{ij}}} = p_{1|\neq} & Z_i \neq Z_j \end{cases}, \tag{15}$$

where we have:

$$1 \leq \frac{p_{1|=}}{p_{1|\neq}} = \frac{K\eta + K - 1 + e^{W_{ij}}}{K\eta + 1 + (K-1)e^{-W_{ij}}} \approx e^{W_{ij}} < +\infty. \tag{16}$$

Since $p_{1|=}$ is larger than $p_{1|\neq}$ and their ratio greatly depends on W_{ij}, $\prod_{(i,j)\in E} P(X_{ij} = 1|Z_i, Z_j, W_{ij})$, representing the joint distribution of a number of connected objects, tends to assign the objects to the same cluster.

Here, we use a trick to reduce the coupling among Z in $q^{(t)}(Z)$ by only considering pairs of connected objects, and factorize $q^{(t)}(Z)$ into many local factors in terms of vertices and edges as follows:

$$\begin{aligned} q^{(t)}(Z) &\propto \prod_{i\in V_{s*}} P(Z_i|C) \prod_{(i,j)\in E} P(X_{ij} = 1|Z_i, Z_j, W_{ij}) \\ &= \prod_{i\in V} \phi_i(Z_i) \prod_{(i,j)\in E} \psi_{ij}(Z_i, Z_j), \end{aligned} \tag{17}$$

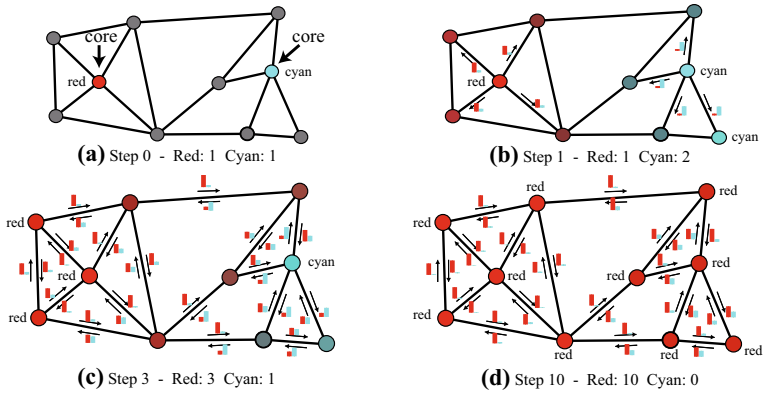
where $\phi_i(Z_i) \equiv 1$ if $i \notin V_{s*}$. Further, we build a factor graph and introduce an efficient method to calculate marginal distributions, called message passing (Kschischang et al. 2001), the time complexity of which increases by the edge number. Here, we adopt the sum-product algorithm, a method of the message passing family. The factor graph consists of variable nodes $\{Z_i|i \in V\}$ and factor nodes $\{\phi_i|i \in V\} \cup \{\psi_{ij}|(i, j) \in E\}$. We define the message from j to i across factor ψ_{ij} as a K-dimensional vector $m_{j\to i}(Z_i)$, which is calculated below:

$$m^{(r)}_{j\to i}(Z_i) \propto \sum_{Z_j} \left(\phi_j(Z_j)\psi_{ij}(Z_i, Z_j) \prod_{j'\in N(j)\setminus i} m^{(r-1)}_{j'\to j}(Z_j)\right), \tag{18}$$

where $\sum_{k=1}^{K} m^{(r)}_{j\to i}(Z_i = k) = 1$. After several iterations, it converges, and then $q^{(t)}(Z_i)$ and $q^{(t)}(Z_i, Z_j)$ can be calculated as follows:

$$\begin{aligned} q^{(t)}(Z_i) &\propto \phi_i(Z_i) \prod_{j\in N(j)} m_{j\to i}(Z_i) \\ q^{(t)}(Z_i, Z_j) &\propto \frac{q^{(t)}(Z_i)q^{(t)}(Z_j)\psi_{ij}(Z_i, Z_j)}{m_{j\to i}(Z_i)m_{i\to j}(Z_j)}. \end{aligned} \tag{19}$$

According to the factor graph theory, the message passing algorithm gives the exact marginal probabilities for all variable nodes in a cycle-free graph. However, it seems that the argument for the exactness will break down when cycles are present in graph. In fact, some equivalent algorithms have achieved excellent experimental results in error-correcting codes defined on Tanner graphs with cycles (Frey and MacKay 1997) and etc. Yedidia et al. (2003) showed that the fixed points of the algorithm correspond to Bethe free energy minima. And



(a) Step 0 - Red: 1 Cyan: 1 **(b)** Step 1 - Red: 1 Cyan: 2

(c) Step 3 - Red: 3 Cyan: 1 **(d)** Step 10 - Red: 10 Cyan: 0

Fig. 6 Assign objects to two clusters by the original message passing algorithm, causing an unbalanced clustering. *Red* and *cyan* (the complementary color of *red*) stands for two clusters. Each object has a probabilistic cluster assignment represented by a mix of *red* and *cyan*. Each *arrow* beside an edge represents a message flow, attached with a small histogram of cluster proportions in the message vector. **a** In the beginning, two objects are specified to cluster cores, but the rest are not assigned to any cluster since no messages are passed. **b** After one step, two core objects pass messages to their neighbors respectively. **c** After three steps, the objects assigned to the *red* cluster are more than to the *cyan* because the *red* parts of histograms in most messages dominate. **d** After ten steps, with messages fully propagated, all objects are assigned to the *red* cluster including the original core object of the *cyan* cluster (Color figure online)

McEliece et al. (1998) conjectured that the algorithm on graphs with cycles converges with a high probability to an approximate optimal solution.

The above reduction of $q^{(t)}(Z)$ yields an efficient algorithm for calculating marginal probabilities. However, for pairs of disconnected objects, although all $\frac{p_{0|=}}{p_{0|\neq}}$ are approximately equal to 1, the product of a large number of these pairs may produce a considerable effect. Ignoring this product may lose the effect of assigning disconnected objects to different clusters, leading to assigning most objects to one large cluster. In another word, it will cause an extremely unbalanced clustering by using the original message passing algorithm on such reduction illustrated in Fig. 6.

We therefore design a balance-controlled message passing algorithm by normalizing each dimension of all message vectors to restrict the differences among cluster sizes as:

$$\widetilde{m}^{(r)}_{j\to i}(Z_i = k) \propto \frac{1}{\pi_k^{\alpha}} m^{(r)}_{j\to i}(Z_i = k), \quad k = 1, \ldots, K \tag{20}$$

where $\pi_k = \sum m^{(r)}_{j\to i}(Z_i = k)$ and $\alpha \in [0, 1]$ is a control parameter to make the algorithm adaptive to data cluster balance. If $\alpha = 0$, there is no balancing and it becomes the same as the original message passing algorithm. When message passes, π_k corresponding to the largest cluster k goes far beyond other $\pi_{k'}$, meaning that most message vectors have a high proportion in the k^{th} dimension. If $\alpha = 1$, the dominant proportion in message vectors can be reduced by dividing the normalization constant π_k, leading to a strict balanced clustering. If $0 < \alpha < 1$, the balance is relaxed which causes a non-strict balanced clustering.

4.3 Estimation of $\Theta^{(t)}$

Since $Q(\Theta|\Theta^t)$ can be split into two parts: C and W, β, we estimate them respectively.

4.3.1 Calculating C

We define a logical matrix $M_{|V_s*|\times K}$ as: $m_{ik} = 1$ if $i \in C_k$ and otherwise $m_{ik} = 0$. The constraint on the sizes of C can be written as $\sum_i m_{ik} = N_k$. Since a target object cannot specified to more than one cluster core, we have $\sum_k m_{ik} \leq 1$. Also, the three cases about each target object i in Eq. (1) can be interpreted as:

	$i \notin \bigcup_k C_k$	$i \in \bigcup_k C_k - C_{Z_i}$	$i \in C_{Z_i}$
$\lambda_1 = 1 - \sum_{k=1}^{K} m_{ik}$	1	0	0
$\lambda_2 = \sum_{k=1}^{K} m_{ik} - m_{iZ_i}$	0	1	0
$\lambda_3 = m_{iZ_i}$	0	0	1

Therefore, $P(Z_i|C)$ can be written as:

$$P(Z_i|C) = (\frac{1}{K})^{\lambda_1} \varepsilon^{\lambda_2} \big(1 - (K-1)\varepsilon\big)^{\lambda_3}. \tag{21}$$

Then, the calculation of C can be converted into a 0/1 integer programming problem by maximizing the objective function:

$$\sum_{i \in V_s*} \sum_{k=1}^{K} q^{(t)}(Z_i = k) m_{ik}, \tag{22}$$

which is a 0/1 integer programming problem and can be solved by the following theorem.

Theorem 1 *Given a* 0/1 *integer programming problem described as:*

$$\begin{aligned} \max \ & f(X) = \sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij} x_{ij} \\ \text{s.t.} \ & \sum_{j=1}^{M} x_{ij} \leq 1, \quad i = 1, \ldots, N \\ & \sum_{i=1}^{N} x_{ij} = n_j, \quad j = 1, \ldots, M, \end{aligned} \tag{23}$$

where $x_{ij} \in 0, 1$, $N \geq M$. *The solution can be estimated approximately by*

$$\max f(X|_{x_{ij}=1}) - \max f(X|_{x_{ij}=0}) \approx r_g(i,j) + r_h(i,j) + a_{ij}, \tag{24}$$

so that

$$x_{ij} = \begin{cases} 1 & r_g(i,j) + r_h(i,j) + a_{ij} \geq 0 \\ 0 & r_g(i,j) + r_h(i,j) + a_{ij} < 0 \end{cases}, \tag{25}$$

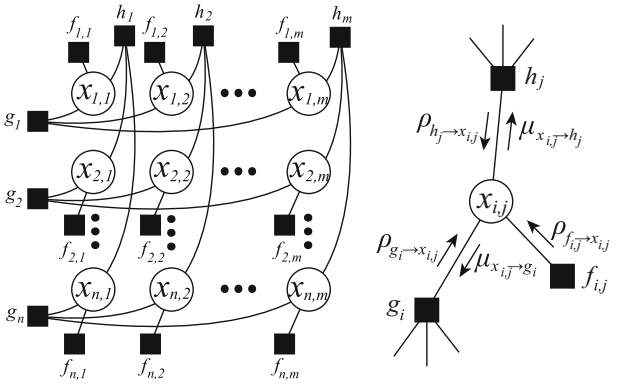


Fig. 7 The max-sum algorithm on a factor graph

where $r_g(i, j)$ *and* $r_h(i, j)$ *are the limits of the sequences* $\{r_g^{(0)}(i, j), \ldots\}$ *and* $\{r_h^{(0)}(i, j), \ldots\}$ *respectively, with the interactive recursive formulas:*

$$\begin{aligned} r_g^{(t)}(i, j) &= -\max\left\{0, \max_{j' \neq j}\left\{r_h^{(t-1)}(i, j') + a_{ij'}\right\}\right\} \\ r_h^{(t)}(i, j) &= -\text{largest}^{(n_j)}_{i' \neq i}\left\{r_g^{(t)}(i', j) + a_{i'j}\right\} \\ i &= 1, \ldots, N; \quad j = 1, \ldots, M, \end{aligned} \tag{26}$$

where $\text{largest}^{(n)}(S)$ *means the nth biggest number in set S.*

Proof We convert $f(X)$ into an equivalent objective function with no constraints:

$$\tilde{f}(X) = \sum_{i=1}^{N}\sum_{j=1}^{M} f_{ij}(x_{ij}) + \sum_{i=1}^{N} g_i(X_{i:}) + \sum_{j=1}^{M} h_j(X_{:j}) \tag{27}$$

where we denote $\{x_{i1}, \ldots, x_{iN}\}$ as $X_{i:}$, similarly for $X_{:j}$. Here, $f_{ij}(x_{ij}) = a_{ij}x_{ij}$, $g_i(X_{i:}) = 0$ if $\sum_{j=1}^{M} x_{ij} \leq 1$ and otherwise $g_i(X_{i:}) = -\infty$; and $h_j(X_{:j}) = 0$ if $\sum_{i=1}^{N} x_{ij} = n_j$ and otherwise $h_j(X_{:j}) = -\infty$. Thus, $\hat{X} = \arg\max \tilde{f}(X)$. In general graphs with loops, the message passing algorithm is an efficient approximate algorithm implemented in a factor graph. Here, we build a factor graph in Fig. 7 and use the max-sum algorithm. Five messages $\rho_{g_i \to x_{ij}}$, $\rho_{h_j \to x_{ij}}$, $\rho_{f_{ij} \to x_{ij}}$, $\mu_{x_{ij} \to g_i}$ and $\mu_{x_{ij} \to h_j}$ are defined in Fig. 7. These messages are updated according to below rules:

$$\begin{aligned} \rho_{g_i \to x_{ij}}^{(t)}(x_{ij}) &= \max_{X_{i:} \setminus x_{ij}}\left\{g_i(X_{i:}) + \sum_{j' \neq j} \mu_{x_{ij'} \to g_i}^{(t)}(x_{ij'})\right\} \\ &= \begin{cases} \sum_{j' \neq j} \mu_{x_{ij'} \to g_i}^{(t)}(0) & x_{ij} = 1 \\ \sum_{j' \neq j} \mu_{x_{ij'} \to g_i}^{(t)}(0) + \Delta_1 & x_{ij} = 0 \end{cases} \end{aligned}$$

$$\rho^{(t)}_{f_{ij}\to x_{ij}}(x_{ij}) = f_{ij}(x_{ij})$$
$$\mu^{(t)}_{x_{ij}\to h_j}(x_{ij}) = \rho^{(t)}_{g_i\to x_{ij}}(x_{ij}) + \rho^{(t)}_{f_{ij}\to x_{ij}}(x_{ij})$$
$$\rho^{(t)}_{h_j\to x_{ij}}(x_{ij}) = \max_{X_{:j}\setminus x_{ij}}\left\{h_j(X_{:j}) + \sum_{j'\neq j}\mu^{(t)}_{x_{i'j}\to h_j}(x_{i'j})\right\}$$
$$= \begin{cases} \sum\limits_{i'\neq i}\mu^{(t)}_{x_{i'j}\to h_j}(0) + \Delta_2(n_j - 1) & x_{ij} = 1 \\ \sum\limits_{i'\neq i}\mu^{(t)}_{x_{i'j}\to h_j}(0) + \Delta_2(n_j) & x_{ij} = 0 \end{cases}$$
$$\mu^{(t+1)}_{x_{ij}\to g_i}(x_{ij}) = \rho^{(t)}_{h_j\to x_{ij}}(x_{ij}) + \rho^{(t)}_{f_{ij}\to x_{ij}}(x_{ij})$$
$$\Delta_1 = \max\left\{0, \max_{j'\neq j}\left\{\mu^{(t)}_{x_{ij'}\to g_i}(1) - \mu^{(t)}_{x_{ij'}\to g_i}(0)\right\}\right\}$$
$$\Delta_2(n) = \operatorname*{topSum}^{(n)}_{i'\neq i}\left\{\mu^{(t)}_{x_{i'j}\to h_j}(1) - \mu^{(t)}_{x_{i'j}\to h_j}(0)\right\},$$

where $\text{topSum}^{(n)}\{\cdot\}$ means the sum of the top-n values. Let $r^{(t)}_g(i,j) = \rho^{(t)}_{g_i\to x_{ij}}(1) - \rho^{(t)}_{g_i\to x_{ij}}(0)$ and $r^{(t)}_h(i,j) = \rho^{(t)}_{h_j\to x_{ij}}(1) - \rho^{(t)}_{h_j\to x_{ij}}(0)$, and we have

$$r^{(t)}_g(i,j) = -\max\left\{0, \max_{j'\neq j}\left\{r^{(t-1)}_h(i,j') + a_{ij'}\right\}\right\}$$
$$r^{(t)}_h(i,j) = -\operatorname*{largest}^{(n_j)}_{i'\neq i}\left\{r^{(t)}_g(i',j) + a_{i'j}\right\}.$$

Since $\max \tilde{f}(X|_{x_{ij}}) \approx \rho_{g_i\to x_{ij}}(x_{ij}) + \rho_{h_j\to x_{ij}}(x_{ij}) + \rho_{f_{ij}\to x_{ij}}(x_{ij})$, the conclusion is proved. □

The inspiration of the above proof is derived from the message passing way in affinity propagation (Frey and Dueck 2007).

4.3.2 Calculating W and β

First, we fix β to calculate W. Note that we only need to calculate w where $x = 1$. For each pair $(i,j) \in E$ with the relation type t, we can draw a function only with respect to w from $Q(\Theta|\Theta^{(t)})$:

$$f(w) \approx -\log(e^w + (K-1)) + \left(\rho - \frac{1}{\beta_t\theta}\right)w + (\kappa - 1)\log w, \tag{28}$$

where $\rho = \sum_{k=1}^{K} q^{(t)}(Z_i = k, Z_j = k)$. To maximize $f(w)$, we need to obtain the solution $\hat{w}$ such that $f'(\hat{w}) = 0$. Since $f''(\hat{w}) = -\frac{(K-1)e^{-w}}{(1+(K-1)e^{-w})^2} - \frac{\kappa-1}{w^2} < 0$, it has a unique global maximum, which can be solved by using the Newton–Raphson Method (Ypma 1995).

Then, we calculate β based on the updated W, by defining a function with respect to β:

$$f(\beta) = -\sum_t\left(\kappa|E_t|\log\beta_t + \frac{\sum_{(i,j)\in E_t} W_{ij}}{\theta\beta_t}\right), \tag{29}$$

Algorithm 1 PROCESS

Input: Heterogeneous network G, parameters κ, θ, α
Output: Cluster assignment Z
1: Initialize C_k by a random target object and set each $W_{ij} = (\kappa - 1)\theta$ and $\beta_t = 1$
2: **repeat**
3: Infer $q^{(t)}(Z_i)$ and $q^{(t)}(Z_i, Z_j)$ using the balanced message passing algorithm under α
4: Assign each object to a cluster by $q^{(t)}(Z_i)$
5: Decide whether adjust N_k to keep balance
6: Estimate new C_k under N_k using Theorem 1
7: Estimate new W, β by the Newton Method.
8: **until** Convergence

under the harmonic average constraint $\sum_t 1/\beta_t = T$. By using the Lagrange Multiplier λ,

$$\beta_t = \left(\frac{1}{\theta} \sum_{(i,j) \in E_t} W_{ij} + \lambda \right) / \kappa |E_t|, \tag{30}$$

where λ can be solved by calculating the root of the following equation:

$$\sum_t \left(\kappa |E_t| / \left(\frac{1}{\theta} \sum_{(i,j) \in E_t} W_{ij} + \lambda \right) \right) = T. \tag{31}$$

4.4 The PROCESS algorithm

Based on the above analysis, the entire process of PROCESS is shown in Algorithm 1, where the input parameters consist of κ, θ, α. Their roles and determination are discussed below:

- κ, θ: parameters in $P(W_{ij}|\beta_t) = \text{Gamma}(W_{ij}|\kappa, \beta_t\theta)$ to adjust the shape and scale of the distribution respectively.
- α: parameters in the balance-controlled message passing algorithm to balance cluster sizes to some degree.

The algorithm contains two main loops. The inner loop is to calculate the marginal probability by message passing, the time complexity of which is $\mathcal{O}(n_0|E|)$, where n_0 is the average iteration number of message passing. The outer loop is to update parameters and reassign the objects to clusters under the EM framework. The whole time complexity is about $\mathcal{O}(n_0|E| \cdot n_1)$, where n_1 is the average iteration number of EM. In fact, n_0 and n_1 are usually a small constant, so that the running time goes linearly with the number of edges in the heterogeneous Network.

Note that parameter $N = \{N_1, \ldots, N_K\}$,where $N_i = |C_i|$, controls the balance of cluster sizes. A balanced message passing algorithm is designed by adjusting α. However, there is no prior knowledge about how the data could be clustered before running the algorithm. When α gets slackened, it might cause extremely unbalanced clusters according to our experiments. Clustering adjustment on N is beneficial due to its self-adapting adjustment according to the situation happening during the clustering process. At the beginning, we set $|C_i| = 1 (1 \leq i \leq K)$. In each iteration, we set a bottom line to decide whether to accept a cluster result based on the available core sizes. For example, if $\max\{|C_i|(1 \leq i \leq K)\}/\min\{|C_i|(1 \leq i \leq K)\}$ becomes too large, we would increase the core size for the minimal cluster.

5 Experiments

5.1 Baselines

We evaluate the effectiveness of the PROCESS algorithm on both synthetic and real datasets by comparison with four state-of-the-art algorithms:

- The graphical k-medoids (k-medoids) (Rattigan et al. 2007): This method uses network structure indices to find the shortest path between two points and adapt the traditional k-medoids to networked data. It works for homogeneous networks.
- NetClus (Sun et al. 2009b): This method is able to handle heterogeneous networks and uses the star network schema, a sub-net picked from the whole network only consists of relations between target objects and attribute objects.
- BibClus (Xu and Deng 2011): This method also works for heterogeneous networks. Besides target-attribute relations, it can use relations among target objects based on center linkage structure, but ignoring relations between attribute objects.
- The spectral relational clustering (SRC) (Ng et al. 2001): This method is a general model for clustering heterogeneous networks based on the collective factorization on related matrices, without discriminating target and attribute objects.

Normalized mutual information (NMI) (Manning et al. 2008) and F-measure are used to measure how well each algorithm's clustering result matches the ground truth. Note that, we set $\kappa = 2$, $\theta = 2$, and $\alpha = 1$ in all experiments unless otherwise noted.

5.2 Synthetic datasets

Our synthetic data generator is comprised of a vertex generator $\text{GenV}(K, n_1, \ldots, n_S, \nu)$ and an edge generator $\text{GenE}\left(V, \langle s_1, s_2, p_{in}, p_{out}\rangle_{t=1}^{T}\right)$. For each cluster k, it contains objects of S types, denoted as $V^{(k)} = \bigcup_{s=1}^{S} V_s^{(k)}$ with $|V_s^{(k)}|$ being a normal random variable with expected value n_s and standard deviation n_s/ν. Given a set of vertices, a random clustered graph G is generated by inserting intra-cluster edges of specific type $t^{(s_1,s_2)}$ with probability p_{in} and inter-cluster edges of type $t^{(s_1,s_2)}$ with probability p_{out}. This data generation idea derives from Brandes et al. (2003). Here, we set three kinds of object types $\{1, 2, 3\}$ and designate the first type as the target one. Also, we consider four relations combining object types, i.e. $\{(1, 1), (1, 2), (1, 3), (2, 3)\}$. Under this setting, eight datasets are drawn with their parameters shown in Table 1. Note that (G1,G2), (G3,G4), (G5,G6) and (G7,G8) are comparison groups respectively, with one p_{out} varying to reflect the heterogeneity for clustering. This may not be the best choice but it is good enough to demonstrate the effectiveness of PROCESS.

Table 2 shows the comparison results of NMI and F-measure (F. in the figure) scores. It shows that PROCESS performs extremely better than the baselines on all datasets. For (G1,G2), (G3,G4) and (G5,G6), p_{out} of the relation type $(1, 3)$ is raised in each group, and the rest of p_{out} keeps $0.1\,p_{in}$. Each time the raised p_{out} brings inter-cluster noises, the performance of the baselines drop obviously while PROCESS is affected slightly. In some datasets, PROCESS can cluster objects 100 % correctly. According to G1 and G3 in Table 3, all the algorithms have a better clustering result in a more dense network given the fixed cluster number K and ratio $|E_{in}| : |E_{out}|$. G3 and G5 have the similar density and $|E_{in}| : |E_{out}|$ but different cluster number K. Results show that the network with a larger K can be easily clustered, this may be because the edge number across one specific pair of clusters is reduced with K increasing. (G7,G8) has different expected object numbers. In general, clustering algorithms are more likely to be influenced by the object type with more

Table 1 Dataset description ($\nu = 5$, all p_{in} are the same)

	K	n_1, n_2, n_3	p_{in}	$p_{out}^{(1,1)}, p_{out}^{(1,2)}, p_{out}^{(1,3)}, p_{out}^{(2,3)}$
G1	5	50, 50, 50	0.1	0.01, 0.01, 0.01, 0.01
G2	5	50, 50, 50	0.1	0.01, 0.01, 0.05, 0.01
G3	5	50, 50, 50	0.05	0.005, 0.005, 0.005, 0.005
G4	5	50, 50, 50	0.05	0.005, 0.005, 0.01, 0.005
G5	10	50, 50, 50	0.1	0.005, 0.005, 0.005, 0.005
G6	10	50, 50, 50	0.1	0.005, 0.005, 0.001, 0.005
G7	5	50, 25, 75	0.05	0.005, 0.01, 0.005, 0.005
G8	5	50, 25, 75	0.05	0.005, 0.005, 0.01, 0.005

Table 2 Comparison results on eight datasets (PROCESS uses $\kappa = 2, \theta = 2, \alpha = 1$)

Dataset	k-medoids		NetClus		BibClus		SRC		PROCESS	
	NMI	F.	NMI	F.	NMI	F.	NMI	F.	NMI	F.
G1	0.5703	0.6483	0.8155	0.7955	0.8184	0.8011	0.9092	0.8729	1.0000	1.0000
G2	0.1470	0.3130	0.3808	0.4721	0.0000∗	0.3318∗	0.7059	0.7504	0.9599	0.9723
G3	0.3456	0.4633	0.6163	0.6765	0.5671	0.6685	0.7655	0.7943	0.9785	0.9860
G4	0.2801	0.4027	0.3733	0.4756	0.4340	0.4975	0.6630	0.7044	0.9464	0.9656
G5	0.5500	0.5002	0.8359	0.7807	0.8400	0.7256	0.9379	0.8635	1.0000	1.0000
G6	0.4246	0.4019	0.7760	0.7370	0.8306	0.6855	0.9269	0.8271	1.0000	1.0000
G7	0.1912	0.3571	0.3065	0.4228	0.0342∗	0.3359∗	0.4216	0.4274	0.8900	0.9252
G8	0.1694	0.3283	0.2726	0.4252	0.3074	0.4551	0.3626	0.4046	0.9197	0.9476

Table 3 Graph statistics on G1, G3, G5

	G1	G3	G5
Density	0.0220	0.0113	0.0110
Avg. degree	18.49	9.46	17.08
$\lvert E_{in}\rvert : \lvert E_{out}\rvert$	1:0.44	1:0.41	1:0.44

vertices, but PROCESS seems to have less dependency on that by adjusting relation weights adaptively. Admittedly, PROCESS benefits from the match between our model assumption and the idea of generating data, but the great advantage demonstrates its effectiveness.

For each algorithm, we test it for 10 iterations and use the averaged results after removing obvious anomalies to weaken the influence from initialization. PROCESS and SRC always have stable results, while NetClus and BibClus need the results from k-medoids as their initial inputs. To minimize the impact of the results from k-medoids, we run k-medoids 20 times and select the best run as the initial input for NetClus and BibClus. Among all algorithms, BibClus produces the most unstable result. The asterisks in Table 2 indicate that BibClus clusters almost all the objects into one large cluster, leading to a very low NMI, because BibClus does not consider the balance among clusters.

In order to explain how PROCESS adjusts relation weights adaptively in a heterogeneous network, we conduct another test to discover the change of parameter β when a relation

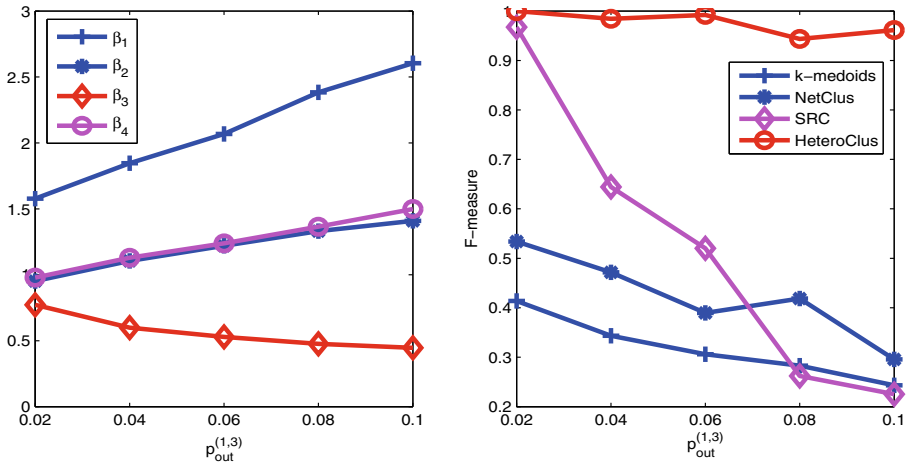


Fig. 8 The change of β with the inter-cluster probability $p_{out}^{(1,3)}$ varying. The fixed setting is $p_{in} = 0.1$, $p_{out}^{(1,1)} = p_{out}^{(1,2)} = p_{out}^{(2,3)} = 0.001$, $K = 5$ and $(n_1, n_2, n_3) = (50, 50, 50)$

type changes its contribution during clustering. In Fig. 8a, $p_{out}^{(1,3)}$ represents the inter-cluster probability between the target type and the attribute type 3. When $p_{out}^{(1,3)}$ ranges from 0.02 to 0.1 with other probability fixed, the corresponding parameter β_3 for the relation type $(1, 3)$ decreases, indicating its importance decreases. Note that all $\beta_t, t = 1, \ldots, 4$ hold a harmonic average constraint, so β_t except β_3 increases. Due to this adaptation, the F-measure of PROCESS stays in a high level as shown in Fig. 8b.

Also, the running time of PROCESS has been studied. Figure 9 shows the scalability testing of PROCESS on a scalable vertex set. In Fig. 9b, we construct three networks with different density levels, i.e. low, medium and high density. Like most real networks, the edge number increases linearly with the vertex number. Figure 9a shows in each network, the running time of PROCESS grows linearly with the vertex number as well, indicating that our algorithm has linear scalability. In the previous section, we have pointed out that the time complexity depends on the average iteration number and the edge number. Since messages can pass quickly in a dense network, it leads to a fast convergence, which explains why PROCESS has a shorter running time in the network with high density.

5.3 Real datasets

We crawl data from the ACM Digital Library (http://portal.acm.org) and build a bibliographic heterogeneous network consisting of papers, authors, conferences and terms. Then, we choose four closely relevant classes, data mining (DM), database system (DB), information retrieval (IR) and artificial intelligence (AI) and twenty representative conferences in these four areas to create a dataset, including 6193 papers, 10150 authors, 20 conferences and 11442 terms. The paper statistics in these four areas is shown in Table 4. Since the data source provides primary classification for each paper, we take them as the ground truth and view papers as target objects.

Table 5 shows the comparison results on the real datasets. PROCESS has a much higher quantity in both NMI and F-measure. From Table 4, we observe that the sizes of the four

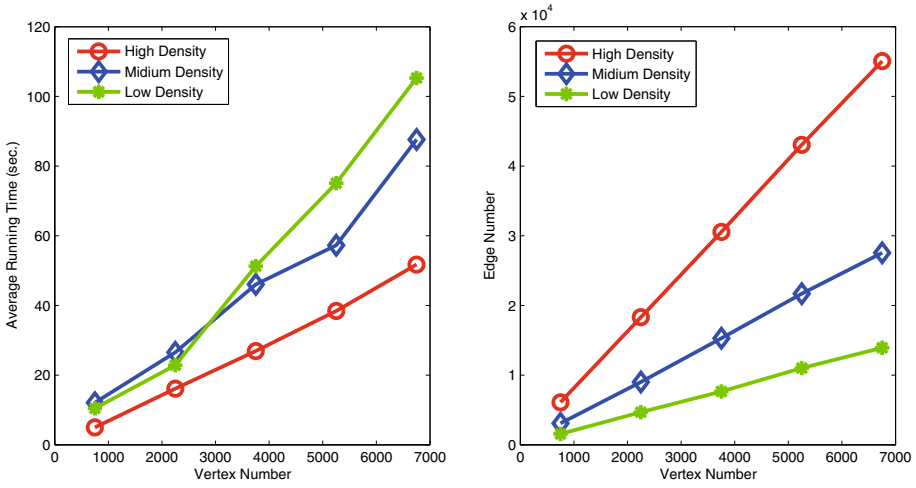


Fig. 9 The running time of PROCESS on a scalable vertex set. The average iteration numbers of EM in high, medium, low density networks are 3, 8, 15, and the average degrees are 4.15 ± 0.05, 8.16 ± 0.1, and 16.32 ± 0.1. The fixed setting is $K = 5$, $p_{in}/p_{out} = 10$

Table 4 Paper statistics in four areas

	DB	DM	IR	AI
#paper	1151	873	2807	1362

Table 5 Comparison results on the real data

	NMI	F-measure
k-medoids	0.1285	0.3989
NetClus	0.1717	0.4461
BibClus	0.2228	0.5204
SRC	0.2311	0.4850
PROCESS	0.4107 (0.5123)	0.5603(0.7173)

areas are not quite balanced, e.g. IR has three times of papers more than DM. In the previous section, we introduce a balancing parameter α to control the balance of message passing, and set $\alpha = 1$ for simplicity. However, it is not appropriate for this test. Here, we change α to 0.2 to relax the balancing control. That is why we have two evaluation values in Table 5 for PROCESS. The value outside the parenthesis corresponds to $\alpha = 1$ and the one inside the parenthesis is for $\alpha = 0.2$. It shows that after relaxing the balance, PROCESS performs much better. To illustrate it, we visualize the clustered sub-networks including only target objects, i.e. papers, by using the real data and prediction with different α in Fig. 10. We find that the prediction with $\alpha = 1$ holds more balance but the prediction with $\alpha = 0.2$ is much closer to the real one. Table 6 shows all 20 conferences in the clustered form. This result is obtained under the setting that papers are target objects. Also, we take the conferences as

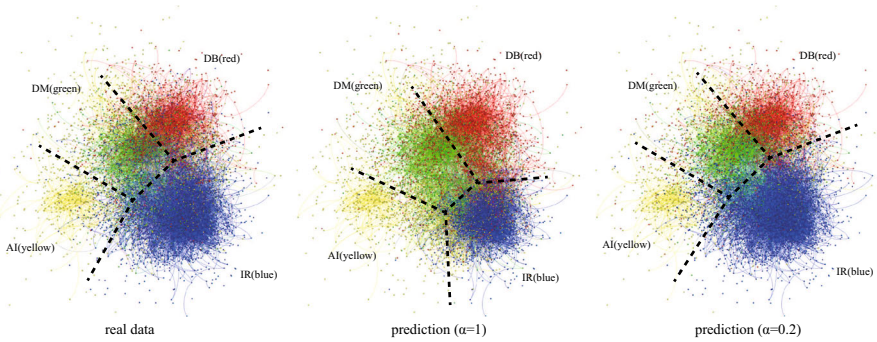


Fig. 10 The visualization of the real data and prediction with different balancing controls $\alpha = 1$ and $\alpha = 0.2$ (only show target objects). *Red* represents DB, *green* represents DM, *blue* represents IR and *yellow* represents AI. These dense networks consist of thousands of small colored points, which might make it difficult to tell cluster border clearly through color in the figure (Color figure online)

Table 6 Clustered conferences in four groups

	Conferences
DB	PODC PODS SAC SIGMOD CSC DOLAP
DM	KDD GIS
IR	MM RecSys SIGIR WI WIDM WWW CIKM
AI	ICAIL IEA/AEI COLT AGENTS GECCO

Table 7 Paper distribution on DB, DM, IR, AI in each conference

	DB	DM	IR	AI		DB	DM	IR	AI
PODC	14	0	2	3	PODS	149	9	16	1
SAC	89	55	143	100	SIGMOD	550	45	102	7
CSC	47	0	4	22	DOLAP	20	12	1	0
KDD	10	523	105	115	GIS	10	14	17	8
MM	18	6	266	16	RECSYS	0	2	33	7
SIGIR	24	2	1110	23	WI	2	32	123	37
WIDM	9	15	38	0	WWW	29	26	357	52
CIKM	173	91	468	37	ICAIL	2	0	5	203
IEA/AEI	3	3	1	209	COLT	0	0	0	123
AGENTS	1	0	12	155	GECCO	1	8	4	244

target objects and run the algorithm, but some conferences are difficult to be categorized into one single area, such as SAC and GIS shown in Table 7.

Based on the substantially experimental results on the synthetic and real datasets, we find that PROCESS wins a bigger margin to the existing state-of-the-art methods. This suggests that our assumptions in Subsection Motivation are reasonable and disclose the particular characteristics of heterogeneous networks. In addition, the experimental results also confirm that our method effectively models the assumptions and is more suitable for heterogeneous network clustering.

6 Conclusions

In this paper, we present a novel probabilistic model for clustering heterogeneous networks. This model is based on two assumptions: (1) different relation types produce different weight distributions to specify intra-cluster probability between two objects; and (2) clusters are formed around cluster cores. Under this model, we further propose PROCESS, an clustering algorithm that can differentiate target objects from attribute objects, incorporates relations between targets and attributes, and leverage between incorporating attribute objects and reducing noises. Substantial experiments show that PROCESS is effective and outperforms state-of-the-art algorithms for clustering heterogeneous networks.

In our future work, we will study the parallel/distributed algorithms of the proposed model to handle large-scale heterogeneous networks with plenty of attributes and a huge number of nodes. In addition, it is also an interesting work to extend our model to tackle other challenging learning tasks of heterogeneous networks such as ranking and classification.

Acknowledgments The authors would like to thank the editor and the anonymous reviewers for their helpful comments. This work is partially supported by Project 61170091 supported by National Natural Science Foundation of China and Project 2015AA015403 supported by the National High Technology Research and Development Program of China (863 Program).

References

Aggarwal, C. C., & Wang, H. (2010). *Managing and mining graph data*. Berlin: Springer.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, *3*, 993–1022.

Brandes, U., Gaertler, M., & Wagner, D. (2003). Experiments on graph clustering algorithms. In *Proceedings of the 11th annual European symposium on algorithms*, pp. 568–579.

Cao, L., Ou, Y., & Yu, P. (2012). Coupled behavior analysis with applications. *IEEE Transactions on Knowledge and Data Engineering*, *24*, 1378–1392.

Deng, H., Han, J., Ji, H., Li, H., Lu, Y., & Wang, H. (2013). Exploring and inferring user-user pseudo-friendship for sentiment analysis with heterogeneous networks. In *Proceedings of the 13th SIAM international conference on data mining*, pp. 378–386.

Deng, H., Han, J., Zhao, B., Yu, Y., & Lin, C. X. (2011). Probabilistic topic models with biased propagation on heterogeneous information networks. In *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1271–1279.

Frey, B. J., & MacKay, D. J. C. (1997). A revolution: Belief propagation in graphs with cycles. In *Proceedings of the 11th annual conference on neural information processing systems*, pp. 479–485.

Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *Science*, *315*, 972–976.

Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*, 7821.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *28*, 100–108.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval*, pp. 50–57.

Kernighan, B. W., & Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, *49*, 291–307.

Kschischang, F. R., Frey, B. J., & Loeliger, H. A. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, *47*, 498–519.

Long, B., Zhang, Z. M., & Yu, P. S. (2007). Spectral clustering for multi-type relational data. In *Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 470–479.

Long, B., Zhang, Z. M., Wu, X., & Yu, P. S. (2006). Spectral clustering for multi-type relational data. In *Proceedings of the 23rd international conference on machine learning*, pp. 585–592.

Manning, C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to information retrieval*. Cambridge: Cambridge University Press.

McEliece, R. J., MacKay, D. J. C., & Cheng, J. F. (1998). Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communications*, *16*, 140–152.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and combinatorial optimization*. New York: Wiley.

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Proceedings of the 15th annual conference on neural information processing systems*, pp. 849–856.

Papadimitriou, C. H., & Steiglitz, K. (1998). *Combinatorial optimization: Algorithms and complexity*. NY: Dover Publications.

Perozzi, B., Akoglu, L., Sanchez, P., & Muller, E. (2014). Focused clustering and outlier detection in large attributed graphs. In *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1346–1355.

Philip, S. Y. (2010). *Link mining: Models, algorithms, and applications*. Berlin: Springer.

Rattigan, M. J., Maier, M., & Jensen, D. (2007). Graph clustering with network structure indices. In *Proceedings of the 24th international conference on Machine learning*, pp. 783–790.

Shiga, M., Takigawa, I., & Mamitsuka, H. (2007). A spectral clustering approach to optimally combining numericalvectors with a modular network. In *Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 647–656.

Sun, Y., Han, J., Zhao, P., Yin, Z., Cheng, H., & Wu, T. (2009a). RankClus: Integrating clustering with ranking for heterogeneous information network analysis. In *Proceedings of the 12th international conference on extending database technology*, pp. 565–576.

Sun, Y., Norick, B., Han, J., Yan, X., Yu, P., & Yu, X. (2012a). Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1348–1356.

Sun, Y., Yu, Y., & Han, J. (2009b). Ranking-based clustering of heterogeneous information networks with star network schema. In *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 797–806.

Sun, Y., Aggarwal, C. C., & Han, J. (2012b). Relation strength-aware clustering of heterogeneous information networks with incomplete attributes. *Proceedings of the VLDB Endowment*, *5*(5), 394–405.

Tang, W., Lu, Z., & Dhillon, I. S. (2009a). Clustering with multiple graphs. In *Proceedings of the 9th IEEE international conference on data mining*, pp. 1016–1021.

Tang, L., Wang, X., & Liu, H. (2009b). Uncoverning groups via heterogeneous interaction analysis. In *Proceedings of the 9th IEEE international conference on data mining*, pp. 503–512.

Xu, X., & Deng, Z. H. (2011). BibClus: A clustering algorithm of bibliographic networks by message passing on center linkage structure. In *Proceedings of the 11th IEEE international conference on data mining*, pp. 864–873.

Xu, Z., Ke, Y., Wang, Y., Cheng, H., & Cheng, J. (2012). A model-based approach to attributed graph clustering. In *Proceedings of the ACM SIGMOD international conference on management of data*, pp. 505–516.

Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2003). Understanding belief propagation and its generalizations. In *Exploring artificial intelligence in the new millennium*, pp. 236–239.

Ypma, T. J. (1995). Historical development of the Newton–Raphson method. *SIAM Review*, *37*(4), 531–551.

Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick, B., & Han, J. (2014). Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on web search and data mining*, pp. 283–292.

Zhou, Y., & Liu, L. (2013). Social influence based clustering of heterogeneous information networks. In *Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 338–346.