Discovering Temporal Communities from Social Network Documents*

Ding Zhou[1] Isaac Councill[2] Hongyuan Zha[3] C. Lee Giles[2]

Computer Science and Engineering[1]
Information Science and Technology[2]
Pennsylvania State University,
University Park, PA

College of Computing[3],
Georgia Institute of Technology,
Atlanta, GA

Abstract

Discovering communities from documents involved in social discourse is an important topic in social network analysis, enabling greater understanding of the relationships among actors within a social network as well as topical trends in communication. This paper studies the discovery of communities from communication documents produced over time, including the discovery of temporal trends in community memberships. We first formulate static community discovery at a single time period as a tripartite graph partitioning problem. Then we propose to discover the temporal communities by threading the statically derived communities in different time periods using a new constrained partitioning algorithm, which partitions graphs based on topology as well as prior information regarding vertex membership. We evaluate the proposed approach on synthetic datasets and a real-world dataset prepared from the CiteSeer computer science research corpus. Quantitative evaluation on synthetic data demonstrates an high discovery precision and an improvement over the generalized normalized cut approach. Qualitative evaluation on CiteSeer data shows the effectiveness of the proposed approach in author community discovery and community summarization in research documents.

1 Introduction

Social network analysis (SNA) is an established field in sociology recently becoming popular for computer scientists [1, 10], which is motivated in part by the increasing amount of personal and social information available online. Community discovery is a classical problem in social network analysis, where the goal is to discover related groups of social actors such that they are intra-group close and inter-group loose [18]. The applications of community discovery have included viral marketing [6], collaborative filtering,and organizational structure analysis [17].

Well known graph-theoretic methods include spectral graph partitioning [14, 4], hierarchical community discovery [19], and clustering[1] based on random walks [11]. Spectral graph partitioning is a classical spectral method based on the Laplacian of the graph adjacency matrix [14, 4], with a characteristic focus on the design of cost functions for partitioning graphs. Hierarchical community discovery seeks to merge the vertices and edges based on the "closeness" between vertices measured by distances on graphs, such as the length of the shortest paths or the diffusion distance [19]. Finally, random walk-based clustering described in [11] applies random walks to the graphs iteratively such that the edge weight between two vertices is modified based on the probabilities that the random walk will circle back to one of the vertices through the other.

Despite the wide range of choices for partitioning homogeneous networks, research on discovering communities from heterogeneous social networks is rather limited [2]. Treating heterogeneous graphs the same as homogeneous ones leads to difficulty in normalization since different edge types may be incomparable [8]. However, observations of real-world networks often indicate diverse network structures, many of which can be modeled as heterogeneous networks of social actors and the other node types such as documents (e.g. emails, blogs, collaborative publications) or social events. In this paper, we are particularly interested in communication documents as these data sources represent the most widely available sources of information regarding social networks.

Discovering communities from documents is a recent trend. Popular approaches are either content-based or graph-theoretic. One popular content-based approach is

*Accepted at IEEE ICDM 2007

[1] In this paper, the term "clustering" and "community discovery" are used interchangably unless otherwise noted.

[2] Here we define a heterogeneous graph as a graph where there are many types of vertices and edges.

to mine information via probabilistic generative modeling, where the social actors or communities are considered as variables in the generation of document content [15, 22]. Alternatively, a graph-theoretic approach can consider the documents as an additional set of vertices connected to authors in a bipartite [21] or tripartite [8] graph structure. These methods, however, work with only a static snapshot of network data. The issues of document time and the temporal trends in communitiy development are generally overlooked.

This paper addresses the community discovery problem in a temporal heterogeneous social network consisting of authors, document content, and the venues in which the documents are published, all observed over time. We propose a new framework that addresses the two main challenges in this new problem: (a) handling of the heterogenous network and (b) incorporation of the temporal aspect of the data. For (a), we formulate community discovery in a heterogeneous social network (the social network is a network of authors, words, and publication venues) as a tripartite graph partitioning problem. A normalized cut (NCut) cost function is defined over the partitions. We show that partitioning a tripartite graph is a quadratically constrained quadratic programming (QCQP) problem. For (b), we introduce a new method for incorporating prior knowledge, such as prior community membership, into the current discovery process. The discovery of temporal communities is then performed by threading communities discovered at consecutive time periods using the output from the previous period as prior knowledge. At each time period, the constrained graph partitioning method is able to capture both the current graph topology and historical information regarding the vertex membership. This problem is efficiently solved using a proposed fractional orthogonal iteration algorithm (instead of pursuing the semidefinite program (SDP) as in [8], which is computationally intractable). We evaluate the proposed approach on synthetic datasets with various settings in order to explore the properties of the new algorithm. A great improvement in clustering precision is observed. In addition, we show the results of applying this method to a sample dataset obtained from CiteSeer (http://citeseer.ist.psu.edu).

The rest of this paper is organized as follows: Sec. 2 introduces related work; Sec. 3 defines the problem and the typical structure of heterogenous social networks that we are interested in; Sec. 4 and Sec. 5 propose a framework for partitioning temporal tripartite graphs; Sec. 6 gives the approximate solution to partitioning; Sec. 7 presents the experimental results and Sec. 8 concludes with comments on future work.

2 Related Work

Our work overlaps with two lines of research: (1) spectral graph partitioning and (2) social community discovery.

Spectral graph partitioning: Spectral graph partitioning is a classical spectral method for partitioning graphs [14, 4]. based on . Spectral methods have been applied in various domains including image segmentations [16] and text analysis [21, 5, 3, 8, 12]. The principal aim of spectral graph partitioning is to minimize the cost of cutting graphs as a function of the Laplacian of the graph adjacency matrix. The partitioning embeds a graph into a low-dimensional subspace subject to the minimal partitioning cost imposed by the graph adjacency matrix. After embedding the graph into the subspace, the clustering can be performed via an additional light-weight clustering algorithm (such as k-means) or by recursively searching for the binary cutting points [21] on the subspace axes. One traditional cost function uses the sum of weights on the edges between clusters [14]; however, this simple approach can bias towards unbalanced cutting points. Recent work proposes variants to the cost function, including ratio cut, normalized cut, and others (a survey can be found in [4]). The most popular cost function for partitioning graphs is the Normalized cut (NCut) [16]. The NCut cost function was originally applied to partitioning homogeneous or bipartite graphs [16, 21, 3]. Due to growing interest in analyzing correlated heterogeneous graphs, recent work generalizes NCut to the case of star-structured tri-partite graphs and a solution has been proposed based on semidefinite programming [8]. Another recent work introduces prior knowledge into the cost function so the partitioning will satisfy minimal violation of prior knowledge as well [12].

Document-based community discovery: Discovering communities in networks based on documents is an important topic of social network analysis, which focuses on analyzing the relationships between social actors in a network of inter-relations [18]. Traditional research has mostly focused on topological properties of social networks. However, real social networks are often embedded in particular social contexts defined by specific information carriers. For example, one of the most common information carriers in social networks is the *communication document*. Accordingly, a recent research trend proposes the content-based analysis of social networks where specific goals include community discovery [22], information flow detection [10], and tracking group evolution [1]. These works leverage text mining to interpret and understand the changes of topic dynamics in documents as well as the dynamics of social ties. Despite the increasing importance of mining communication documents, the analysis of temporal aspects of communication is in its early stage. Very often, temporal community discovery is performed by periodically clustering actors and examining the extracted temporal clusters [13]. There has been little work on discovering the communities of social actors and documents from temporally correlated text streams, that is, explicitly *threading* communities from different time periods.

3 Problem Statement

This paper considers social networks of researchers in the context of their collaborations on published work. The data in focus includes the co-occurrences of authors with documents, documents with words, and documents with venues. All data are associated with time stamps, which are the years of publication. The data is collapsed on documents yielding the (1) author-word co-occurrences and (2) word-venue co-occurrences, over a certain amount of time. Thus, within each time period there are at two correlated bipartite graphs, $G(V_X, V_Y, W_{XY})$ and $G(V_Y, V_Z, W_{YZ})$, where V_X is the author set, V_Y is the word set, V_Z is the venue set, W_{XY} is the bipartite edge weights between V_X and V_Y, and W_{YZ} is the edge weights for V_Y and V_Z. Here $G(V_X, V_Y, W_{XY})$ and $G(V_Y, V_Z, W_{YZ})$ share the vertex set V_Y. Name $G(V_X, V_Y, W_{XY})$ and $G(V_Y, V_Z, W_{YZ})$ as a *bipartite graph couple*, which can be seen as a generalized social network of authors, words, and documents. Two static communities in such a social network are illustrated in Fig. 2, where a *static community*, at a specific time, is defined on the snapshot below:

Definition 1 *A static community in a static social network is a composite of closely associated authors, words, and venues. Entities within the same community are closely related while entities in different communities are loosely associated if at all.*

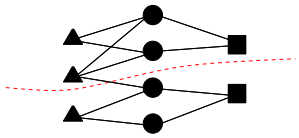


Figure 1. A static social network. triangles denote the authors, circles denote the words, and rectangles denote the venues. The graph between authors and words is inferred from the document authorship and the graph between words and venues is based on the publication records of documents. Two static communities are separated by the dashed line.

Over the entire time period, the underlying social network structure is dynamic. Accordingly, instead of observing a single static social network over the entire data set, a sequence of static social networks of various structures is generated, with consecutive snapshots showing significant overlap of entities. The definition of a temporal community thus embody the temporal aspects of the network:

Definition 2 *A temporal community in a dynamic social network is a threaded sequence of static communities at each time period. In a temporal community, the structure of a static community at a specific time depends on the previous* N *temporal networks, where* N *is a parameter that can be defined as the* order *of the temporal community.*

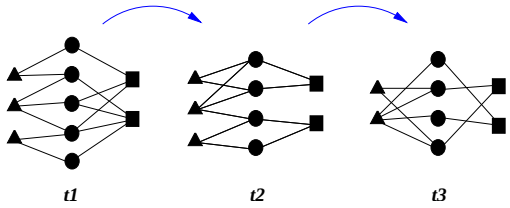


Figure 2. A dynamic social network. Three snapshots are included in the network with various numbers of authors (denoted by triangles), venues (denoted by rectangles), and words (denoted by circles).

A dynamic social network is illustrated in Fig. 2. Three snapshots are included, each having different network structures. It can be seen that each static social network is a bipartite graph couple.

The goal of this paper is to cluster authors, words and venues given their changing relationships over time. In the clustering results, one can easily see how a particular community evolves in its members and topical interest, expressed in terms of words. The temporal communities are discovered via threading the discovery of static communities at each time period. The desired number of communities k is assumed and given as a parameter.

4 Community Partitioning

We start from the discovery of static communities from a static social network. Suppose there are two bipartite graphs, $G_{XY} = G(V_X, V_Y, W_{XY})$ and $G_{YZ} = G(V_Y, V_Z, W_{YZ})$, where V_X is the author set, V_Y is the word set, and V_Z is the venue set; $W_{XY} \in \mathbb{R}^{+n_X \times n_Y}$ is a matrix where the elements represent the number of co-occurrences of an author and a word; and $W_{YZ} \in \mathbb{R}^{+n_Y \times n_Z}$ is a matrix whose elements are the number of co-occurrences of a word and a venue (n_X, n_Y, n_Z are the size of V_X, V_Y, V_Z). Note G_{XY} and G_{YZ} share V_Y.

Consider a community with two types of vertices from V_X and V_Y, say which is represented by two subsets S_i^X and S_j^Y. The weight of the community is:

$$W(S_i^X, S_j^Y) = \sum_{u \in S_i^X, v \in S_j^Y} w_{u,v}. \tag{1}$$

Given k as the desired number of communities, the cost function of *Normalized Cut (NC)* is defined as [21]:

$$J_2 = \sum_{i=1}^{k} \frac{W(S_i^X, \overline{S_i^Y}) + W(\overline{S_i^X}, S_i^Y)}{W(S_i^X, Y) + W(X, S_i^Y)} \tag{2}$$

where S_i^X, S_i^Y are the subsets of V_X and V_Y in community i; $\overline{S_i^X}$, $\overline{S_i^Y}$ are the subsets of V_X and V_Y not in community i. The sets $\{S_i^X\}_{i=1}^k$, $\{S_i^Y\}_{i=1}^k$ that minimize the cost J_2 belong to the discovered k communities.

Now define several indicator matrices. Let $X = [X_1, ..., X_k]$, where X_i is an indicator vector of whether the corresponding element belongs to community i, with 1 indicating so or 0 otherwise. Similarly, we have $Y = [Y_1, ..., Y_k]$ and $Z = [Z_1, ..., Z_k]$.

Define D_{XY} and D_{YZ} as diagonal matrices where the elements are the sums of rows in W_{XY} and W_{YZ}. Define D_{YX} and D_{ZY} as diagonal matrices where elements are the sums of columns in W_{XY} and W_{YX}. After some manipulations, we can rewrite Eq. 2 as:

$$J_2 = \sum_{i=1}^{k} \frac{X_i^T D_{XY} X_i + Y_i^T D_{YX} Y_i - 2X_i^T W_{XY} Y_i}{X_i^T D_{XY} X_i + Y_i^T D_{YX} Y_i} \quad (3)$$

$$= k - \sum_{i=1}^{k} \frac{2X_i^T W_{XY} Y_i}{X_i^T D_{XY} X_i + Y_i^T D_{YX} Y_i}. \quad (4)$$

The problem of searching for best solutions to the above minimization problem has been shown to be NP-hard. In order to obtain a solution efficiently, prior work relaxes the elements in X_i and Y_i to real values instead of the discrete set $\{0, 1\}$ [21]. Extending this work, we further scale X_i and Y_i to the denominator. In particular, assuming $X_i = D_{XY}^{-\frac{1}{2}} \hat{X}_i$ and $Y_i = D_{YX}^{-\frac{1}{2}} \hat{Y}_i$, we let $\hat{X}_i^T \hat{X}_i = \hat{Y}_i^T \hat{Y}_i = 1$. Thus, J_2 becomes:

$$J_2 = k - \sum_{i=1}^{k} \hat{X}_i^T D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}} \hat{Y}_i. \quad (5)$$

Here $D_{XY}^{-\frac{1}{2}} W_{XY} D_{YZ}^{-\frac{1}{2}}$ is in fact the normalized edge weight matrix. The minimization cost function J_2 is carried out over $\hat{X}_i$ and $\hat{Y}_i$ for $i = 1, ..., k$. Traditionally, the different minimizers are assumed to be orthogonal to each other [20], i.e. $\hat{X}^T \hat{X} = \mathbf{I}$ and $\hat{X}^T \hat{X} = \mathbf{I}$. We impose the same constraint on our solution.

Now let us generalize the cost function for a bipartite graph couple, where we have an additional set of vertices Z and the edge weights with Y in W_{YZ}. Similarly, define $\hat{X} = [\hat{X}_1, ..., \hat{X}_k]$, $\hat{Y} = [\hat{Y}_1, ..., \hat{Y}_k]$ and $\hat{Z} = [\hat{Z}_1, ..., \hat{Z}_k]$, where $\hat{X}^T \hat{X} = \hat{Y}^T \hat{Y} = \hat{Z}^T \hat{Z} = \mathbf{I}$. Let J_{XY} be the cost function of partitioning graph G_{XY} and J_{YZ} be the cost function for G_{YZ}. We introduce a parameter λ to balance the costs on both graphs. Based on Eq. 5, we define the new cost function J_3 on the bipartite graph couple as:

$$J_3 = \lambda J_{XY} + (1 - \lambda) J_{YZ}$$

$$= k - \lambda \sum_{i=1}^{k} \hat{X}_i^T D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}} \hat{Y}_i - (1 - \lambda) \sum_{i=1}^{k} \hat{Y}_i^T D_{YZ}^{-\frac{1}{2}} W_{YZ} D_{ZY}^{-\frac{1}{2}} \hat{Z}_i \quad (6)$$

where the second and third terms represent the cost functions on G_{XY} and G_{YZ}.

Thus, the minimization of cost function J_3 over $\hat{X}$, $\hat{Y}$, and $\hat{Z}$ becomes a maximization of the negative term in J_3:

$$\min_{\hat{X}, \hat{Y}, \hat{Z}} J_3$$

$$\equiv \max_{\hat{X}, \hat{Y}, \hat{Z}} \lambda \sum_{i=1}^{k} \hat{X}_i^T D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}} \hat{Y}_i - (1 - \lambda) \sum_{i=1}^{k} \hat{Y}_i^T D_{YZ}^{-\frac{1}{2}} W_{YZ} D_{ZY}^{-\frac{1}{2}} \hat{Z}_i \quad (7)$$

subject to

$$\hat{X} = [\hat{X}_1, ...\hat{X}_k], \ \ \hat{X}^T \hat{X} = \mathbf{I}; \quad (8)$$

$$\hat{Y} = [\hat{Y}_1, ..., \hat{Y}_k], \ \ \hat{Y}^T \hat{Y} = \mathbf{I}; \quad (9)$$

$$\hat{Z} = [\hat{Z}_1, ..., \hat{Z}_k], \ \ \hat{Z}^T \hat{Z} = \mathbf{I}; \quad (10)$$

where $\mathbf{I}$ is an identity matrix.

Now let us rewrite the problem in matrix form. Define $\widehat{W}_{XY} = D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}}$ and $\widehat{W}_{YZ} = D_{YZ}^{-\frac{1}{2}} W_{YZ} D_{ZY}^{-\frac{1}{2}}$. Define $U = [U_1, ..., U_k]$, where $U_i = [\hat{X}_i^T, \hat{Y}_i^T, \hat{Z}_i^T]^T$; Let there be a matrix M such that:

$$M = \begin{bmatrix} 0 & \lambda \widehat{W}_{XY} & 0 \\ \lambda \widehat{W}_{XY}^T & 0 & (1 - \lambda) \widehat{W}_{YZ} \\ 0 & (1 - \lambda) \widehat{W}_{YZ}^T & 0 \end{bmatrix}. \quad (11)$$

It is easy to verify that the cost function in Eq. 7 is $\frac{1}{2} U_i^T M U_i$. The problem thus becomes to minimize the trace of the matrix (The trace of a square matrix is defined as the sum of the diagonal elements):

$$\max_{U} \text{tr}(U^T M U) \quad (12)$$

subject to

$$U = [\hat{X}^T, \hat{Y}^T, \hat{Z}^T]^T \quad (13)$$

$$\hat{X}, \hat{Y}, \hat{Z} \text{ satisfy Eq. 8 - Eq. 10} \quad (14)$$

Here the optimization problem is a quadratically constrained quadratic programming problem [2]. Note that Eq. 8 - Eq. 10 is not equivalent to $U^T U = \mathbf{I}$. Constraints on U apply to its segments (i.e. $\hat{X}, \hat{Y}, \hat{Z}$) respectively.

5 Partitioning Temporal Graphs

The problem of community discovery has been formulated as a graph partitioning issue. Next we present a constrained graph partitioning method that threads community discovery across consecutive time periods.

5.1 Graphs with consistent vertices

We first focus on the case where graphs have consistent vertices. For each time period, we have M_t and U_t as described in Eq. 11 and Eq. 8 - Eq. 10, where $t = 1, ..., T$ are the time stamps and U^t contains the community membership of authors, words, and venues. Assume that the graphs have consistent vertices; thus, all U_t have the same dimensions. Now, let us define a cost function on the difference

between U^{t_i} and U^{t_j} for an arbitrary time stamp pair t_i, t_j, denoted $c(U^{t_i}, U^{t_j})$. The discovery of community structure at time t seeks to minimize the weighted sum of the distances between the current and previous community membership back to $t - \delta$:

$$\min_{U^t} \sum_{\pi=t-\delta}^{t-1} \alpha_\pi c(U^\pi, U^t) \tag{15}$$

where α_π is the weight on the distance to the community membership at π time periods ago. The weights on different historic periods are prescribed parameters. Hereafter, for simplicity, we concern ourselves only with the first-order dependency case where $\delta = 1$ and $\alpha_\pi = 1$.

A key issue is the design of the cost function $c(\dot{U}, U)$. Here we let the cost function be the negative cosine distance between two subspaces. Suppose $\dot{X}$, $\dot{Y}$, and $\dot{Z}$ are the reference subspaces of X, Y, Z. We know that $\|\dot{X}\|^2 = \|\dot{Y}\|^2 = \|\dot{Z}\|^2 = 1$. Thus, the square of cosine distances between the desired subspace and the reference subspace are respectively $\|\dot{X}^T\hat{X}\|^2$, $\|\dot{Y}^T\hat{Y}\|^2$, and $\|\dot{Z}^T\hat{Z}\|^2$. In addition, we know that the cosine distances are within $[0, 1]$. We thus seek to maximize the cosine distances to minimize the cost imposed by the distance from the reference subspaces. In particular, define the cost function $c(\dot{U}, U)$:

$$-c(\dot{U}, U) = \alpha\|\dot{X}^T\hat{X}\|^2 + \beta\|\dot{Y}^T\hat{Y}\|^2 + \gamma\|\dot{Z}^T\hat{Z}\|^2 \tag{16}$$

$$= \alpha.\text{tr}(\hat{X}^T\dot{X}\dot{X}^T\hat{X}) + \beta.\text{tr}(\dot{Y}^T\dot{Y}\dot{Y}^T\hat{Y}) + \gamma.\text{tr}(\dot{Z}^T\dot{Z}\dot{Z}^T\hat{Z}) \tag{17}$$

$$= \text{tr}(U^T\dot{U}\dot{U}^TU), \tag{18}$$

where $\dot{U} = [\sqrt{\alpha}\dot{X}^T, \sqrt{\beta}\dot{Y}^T, \sqrt{\gamma}\dot{Z}^T]^T$, α, β and γ are the weight parameters of the membership differences in authors, words, and venues. Here, notice that $\dot{U}\dot{U}^T$ is essentially the covariance matrix between the vertices in the reference time period. Since we have assumed consistent vertices in the graphs across different time periods, we essentially minimize the the conflicts between the discovered U and the referenced covariance.

5.2 Graphs with evolving vertices

Now we generalize the previous section to graphs with evolving vertices. In practice, some vertices may disappear and other new ones may show up, thus the $\dot{U}$ obtained from previous period can disagree with the dimensionality of the U in the current time period. We introduce an additional step to adapt $\dot{U}$ to address this issue.

First, some vertices from previous time period may disappear. Since each vertex corresponds to a row in $\dot{U}$, we can delete these rows from $\dot{U}$, forming a matrix with same number of columns but a smaller number of rows, $\dot{U}'$. We call the first step shrink(). Thus we have:

$$\dot{U}' = \text{shrink}(\dot{U}) = [\dot{X}'^T, \dot{Y}'^T, \dot{Z}'^T]^T \tag{19}$$

where $\dot{U}'$ is the adapted subspace with disappeared vertices removed. $\dot{X}'$, $\dot{Y}'$, and $\dot{Z}'$ still correspond to the remaining $\dot{X}$, $\dot{Y}$, and $\dot{Z}$. Second, some new vertices may appear in the current time period. In this case, we have no prior knowledge about their membership. Therefore, we require zero co-variances of them with others, corresponding to zeros in the corresponding rows. Name this second step expand():

$$\dot{U}'' = \text{expand}(\dot{U}') = [\dot{X}'^T, 0, \dot{Y}'^T, 0, \dot{Z}'^T, 0]^T, \tag{20}$$

where $[\dot{X}'^T, 0]^T$, $[\dot{Y}'^T, 0]^T$, and $[\dot{Z}'^T, 0]^T$ respectively correspond to the newly observed X^t, Y^t, and Y^t; all $0's$ has the appropriate number of rows and k columns. We then arrive at the new reference covariance matrix $c(\dot{U}, U)$ as:

$$\dot{C} = \dot{U}''\dot{U}''^T, \tag{21}$$

which leads to the new cost function $c(\dot{U}, U)$ on U and reference $\dot{U}$ defined as: $-c(\dot{U}, U) == \text{tr}(U^T\dot{C}U)$, where $\dot{C}$ is given in Eq. 19 - Eq. 21.

Note the handling of new vertices here. Since the reference $\dot{U}''$ still has values in the rows corresponding to the old vertices, these previously observed vertices will be made consistent with the previous period. On the other hand, the new vertices will not be affected by such prior knowledge of the previous time period because of the zeros in the rest of $\dot{U}''$. To see this, note that the $\text{tr}(U^T\dot{C}U)$ has zero diagonals in the indices of those newly observed vertices regardless of the values of U in the corresponding rows.

Given the above, the combined community discovery problem at each time period is written as:

$$\begin{aligned} \min_U \widetilde{J} &= \min_U J_3 + c(\dot{U}, U) \\ &\equiv \max_U \text{tr}(U^TMU) + \text{tr}(U^T\dot{C}U) \\ &= \max_U \text{tr}(U^T(M + \dot{C})U) \end{aligned} \tag{22}$$

subject to

$$U = [\hat{X}^T, \hat{Y}^T, \hat{Z}^T]^T \tag{23}$$

$$\hat{X}, \hat{Y}, \hat{Z} \text{ satisfy Eq. 8 - Eq. 10} \tag{24}$$

$$M \text{ is given in Eq. 11} \tag{25}$$

$$\dot{U} = [\sqrt{\alpha}\dot{X}^T, \sqrt{\beta}\dot{Y}^T, \sqrt{\gamma}\dot{Z}^T]^T \tag{26}$$

$$\dot{C} \text{ is given by Eq. 19 - Eq. 21.} \tag{27}$$

where α, β and γ are the weight parameters for the membership differences in authors, words, and venues; $\dot{U}$ is the reference membership matrix. We arrive at a quadratically constrained quadratic programming problem.

6 Efficient Approximate Solutions

This section gives an efficient algorithm to solve the problem formulated in Eq. 22 - Eq. 27. It can be seen that Eq. 22 has a quadratic cost function of the matrix U. Here Eq. 22 can be rewritten as:

$$\max_{U} \sum_{i} U_i^T (M + \dot{C}) U_i) \tag{28}$$

where the U_i's are column vectors in U. We can see that this is a sum of a sequence of quadratic functions each corresponding to a subset of constraints in Eq. 23 - Eq. 27. Thus we have a sequence of quadratically constrained quadratic programming (QCQP) sub-problems. Note these QCQP problems are not isolated because their solution vectors U_i are required to be orthogonal.

For each QCQP sub-problem alone, there exists a standard solution using semidefinite programming (SDP) [2]. For example, a related work [8] studied the binary clustering case and proposed an approximate solution using an interior-point method. However, we note that our optimizer here is a matrix ($U = [X^T, Y^T, Z^T]^T$) instead of a single vector. Thus, to apply SDP on each column vector and combine them together is overly complex. Nevertheless, one might construct a very high-dimensional vector by columns of U and still translate the problem into SDP, but difficulty still arises from the exploding dimensionality of the problem. Recall that $U \in \mathbb{R}^{(n_X+n_Y+n_Z)\times k}$, where n_X, n_Y, and n_Z are the numbers of authors, words, and venues. The translated SDP problem will have a $k(n_X + n_Y + n_Z)$-dimensional vector as the minimizer (with a $k(n_X + n_Y + n_Z) \times k(n_X + n_Y + n_Z)$ semidefinite matrix of constraints), which can easily surpass the capacity of most SDP solvers.

Instead, we propose an efficient algorithm that searches for approximate solutions. The new algorithm is based on algorithms for eigenvectors. First we are aware that the Eq. 22, without constraints, reaches the maximum when U contains the first k eigenvectors of the symmetric matrix $A = M - \dot{U}\dot{U}^T$. This is a standard result from matrix theory [9]. In addition, we have $\forall U \in \{U | U^T U = \mathbf{I}\}$, $U^T A U \leq \lambda_1 + ... + \lambda_k$, where $\lambda_1, ...\lambda_k$ are the first k largest eigenvalues of A. Second, we seek to preserve the constraints as much as possible while maximizing $\widetilde{J}$. We modify the *orthogonal iteration* method which is used to calculate the eigenvector space without constraints. The idea is to incorporate the constraints into the classical method. The new algorithm, *fractional orthogonal iteration*, is presented below:

Here eig(A, k) calculates the k-dimensional eigenvector space of A without constraints. This is the initial value for the subsequent orthogonal iteration. In the algorithm, step 9 - step 11 produce the normalized $\hat{X}$, $\hat{Y}$ and $\hat{Z}$ as specified in the constraints. Step 8 performs the power iteration as in the original *orthogonal iteration* method for calculating eigenvectors. Up to step 15, the algorithm has projected the original bipartite graph couple into an approximate k-dimensional eigenspace. The distribution of the points in the new space preserves the distribution of objects at the current time period, in addition to imposing the community membership from the last period. Then we run k-means to cluster the heterogeneous objects as current communities.

Algorithm 1 fractional orthogonal iteration

1: $\dot{U} = [\sqrt{\alpha}\dot{X}^T, \sqrt{\beta}\dot{Y}^T, \sqrt{\gamma}\dot{Z}^T]^T$;
2: $\dot{U}' \leftarrow$ shrink$(\dot{U})$ as in Eq. 19
3: $\dot{U}'' \leftarrow$ expend$(\dot{U}')$ as in Eq. 20
4: $\dot{C} \leftarrow \dot{U}''\dot{U}''^T$
5: $A = M + \dot{C}$
6: $[U, D] \leftarrow$ eig(A, k)
7: **for** $i = 1, 2, 3, ...$ **do**
8: $\begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix} \leftarrow A \times U$
9: $Q_X R_X \leftarrow \hat{X}$ // QR factorization
10: $Q_Y R_Y \leftarrow \hat{Y}$ // QR factorization
11: $Q_Z R_Z \leftarrow \hat{Z}$ // QR factorization
12: $U \leftarrow \begin{bmatrix} Q_X \\ Q_Y \\ Q_Z \end{bmatrix}$
13: **end for**
14: $U \leftarrow M \times U$
15: run k-means on U to obtain the desired partitioning, where each row in U denotes the original data object of the same index.

7 Experiments

A synthetic data generator was created to test the proposed method in various conditions, including different edge density-to-noise ratio, various proportions of $X/Y/Z$, different settings of λ, and different numbers of clusters (k). Two connected graphs G_{XY} and G_{YZ} are generated for the prescribed K and sizes of X, Y, and Z. All clusters contain the same number of entities with specified proportions of $X, Y, andZ$. The densities of all the clusters are the same, but the edge weights vary randomly. Random noise is added to the graph and density is determined by the given noise-signal ratio parameter (nsr). Setting $nsr = 1$ yields a random graph without cluster structures. Presumably, the community structures in the graph XY diminish as the noise-signal ratio (nsr) grows. Low nsr indicates that graph partitioning will be easier. Table **??** includes a complete list of parameters and their meanings.

abbr.	usages
fsi	fractional subspace iteration
par	partitioning static graphs using fsi
t-par	partitioning temporal graphs using fsi
k	number of clusters
$density$	the edge density of the graph clusters
nsr	noise-signal ratio, noise density / cluster density
x/z	the size of X / the size of Z
λ	the weight parameter in Eq. 11

7.1 Precision w.r.t. graph conditions

First, we focus on the clustering precision w.r.t. different densities and nsr for $k = 2$. As illustrated in Fig. 4 we present four values of nsr, indicating increasing difficulty for partitioning. In general, we observe that the precision decreases as nsr grows. In each subfigure, we can see that the clustering precision grows quickly as the graph clusters become denser. On graphs with less noise, the precision grows faster than on the highly noisy graphs. Comparatively, the proposed fsi algorithm outperforms the traditional *subspace iteration* algorithm (without consideration of constraints) for different nsr. We are able to see that the special scaling introduced in fsi improves the *subspace*

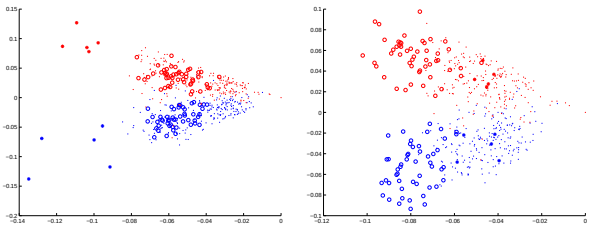
iteration. The fsi usually outperforms *subspace iteration* by a greater amount in the more difficult situations (large nsr). All precisions are measured using k-means with random initial medians. For each case, the k-means is repeated for 10 times and the averages are presented.

Second, we perform fsi on different settings of x/z ratios for a fixed setting of λ. In real world datasets, the sizes X and Z are usually not balanced. We compare fsi with *subspace iteration* for imbalanced data against fsi by varying the x/z ratio. Fig. 5 shows different settings of x/z for different densities. Recall that a large x/z indicates that the size of X is much greater than that of Z. Without loss of generality, we assume $x/z \geq 1$. We can see that for sparse graphs (small density) the fsi outperforms *subspace iteration* greatly (illustrated in the subfigure on the bottom). In simple cases (large density), the fsi generally outperforms *subspace iteration* for small x/z; however, fsi underperforms *subspace iteration* slightly for small x/z on dense graphs. Note that real-world graphs are usually very sparse; thus, fsi could be favored on many real-world datasets.

7.2 Precision w.r.t. parameter settings

Here we test different settings of parameters and their impact on community discovery precision. A set of experiments were run with different settings of λ in different x/z ratios. The results illustrated in Fig. 6 show that the favorable λ are different when x/z varies. When the X outnumbers Z by a large margin, a greater value in λ is favored; similarly, small λ performs better when there are few X entities compared with Z. This suggests that graphs with more edges deserve a larger weight in the cost evaluation.

In order to better visualize the effect of λ with different x/z, we present the subspace scatter plots for different λ. Note that here $|X|/|Y|/|Z| = 50 : 200 : 5$. The X outnumber Z, indicated by a great x/z ratio. In Fig. 3, we show precisions for $\lambda = 0.5, 0.8$. Here $k = 2$ so we have 2-D subspaces. In this case, a large λ better scales the edges in YZ and thus better embeds Z into the subspace.



(a) $\lambda = 0.5$ (b) $\lambda = 0.8$

Figure 3. Subspace plots for different λ when $|X|/|Y|/|Z|$ = 50:200:5. Different clusters are colored differently. Entities of different types have different markers (circles, dots, stars for X, Y, Z). Here $k = 2$.

Finally, we compare fsi with *subspace iteration* on different numbers of clusters, at different *subspace iteration*. We can see that, for large density, fsi still outperforms *subspace iteration* for large numbers of clusters. However, the *subspace iteration* seems to work better than fsi for the case of many clusters on sparse graphs. In practice, we can substitute fsi by recursively performing k-means using $k = 2$ for bi-partitioning the graph, similar to [21].

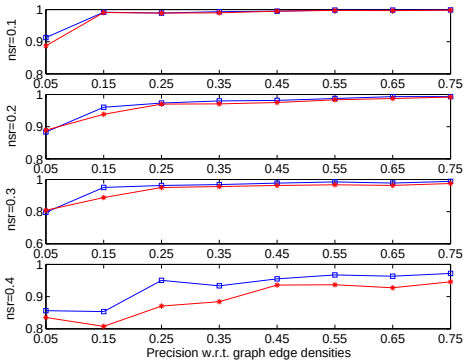


Figure 4. The clustering precision w.r.t. edge densities at different levels of noise-signal ratio (nsr). The line with square markers is the result for fsi. Here $k = 2$.

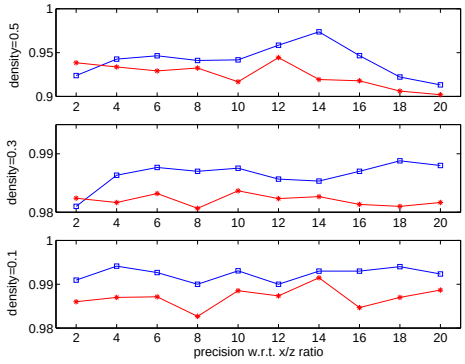


Figure 5. The precision w.r.t. different x/z ratio, at different edge density levels. Here $nsr = 0.1$, $k = 2$, $\lambda = 0.5$. The lines with square markers are fsi.

7.3 Higher precision using prior knowledge

The fsi algorithm uses the discovery results from the previous time period as prior knowledge for analyzing temporal graphs. This knowledge is then used as an additional constraint while discovering communities in the current time period. We simulate a 2-period temporal graph where communities in the first time period are clearly de-

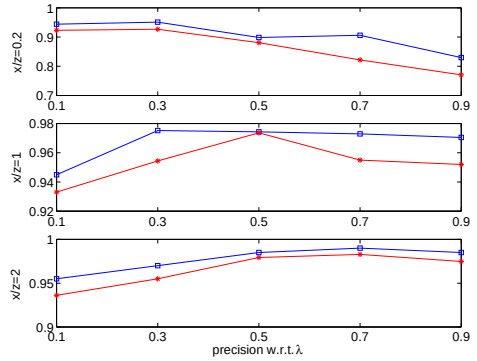


Figure 6. The precision w.r.t. λ, at different x/z ratio. Here $d = 0.3$, $nsr = 0.3$, $k = 2$. The line with square markers is the result for fsi.

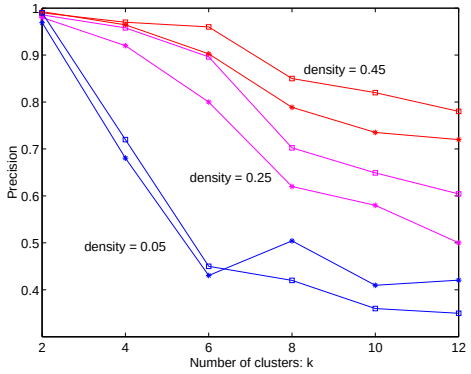


Figure 7. The precision w.r.t. k, at different densities: $density = 0.05$, $density = 0.25$, $density = 0.45$. The line with square markers is the result for fsi.

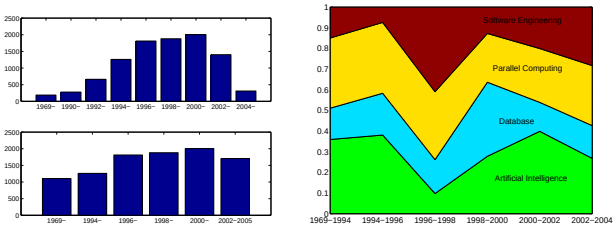
fined and then the community structure becomes vague in the second time period. The community membership from the first time period is used as the prior knowledge in the second time period.

methods	precisions	methods	precisions
par on g_1	0.9193	par on g_{12}	0.8212
par on g_2	0.2123	t-par on g_{12}	0.9169
average of the above	0.5658		

Table 1. Different methods on temoral graphs.

In Table 1, we illustrate the precisions of clustering on the snapshots from each time period and the average precision. It can be seen that the static partitioning precision is very high on g_1 (0.9193) and very low on g_2 (0.2123): the average of the two is about 0.5658. In addition, we perform clustering on the graph over the complete time periods, obtaining a precision of 0.8212. Then we perform the constrained partitioning t-par on the temporal graph, yielding the precision 0.9169. The precision is much higher than performing clustering periodically or on the complete graph.

A natural question is whether the community structure from previous time periods is *always* more reliable and informative than the current period. We would like to first point out that the reference community membership does not only encode the information from the immediate previous period but a combination of information from all previous periods. This is due to the recursive application of fsi on the snapshot sequence. Therefore, one might assume that the community discovered based on all historical data can be more reliable compared with the discovery on the current single snapshot. In practice, we can also allow manual manipulation of entity membership to be input as the prior knowledge for the *first time period*, in order to increase the validity of this assumption.



(a) Publication number (b) Community size

Figure 8. Amount of publications and community size over time. Two different grouping methods are shown, one by uniform grouping of years and the other by proportional grouping.

7.4 Real-world dataset and experiments

A real-world data set for further experimentation was generated by sampling documents from CiteSeer using combined document metadata from CiteSeer, the ACM Guide (http://portal.acm.org/guide.cfm), and the DBLP (http://www.informatik.uni-trier.de/ ley/db) for enhanced data accuracy and coverage. A set of venues was chosen from five fields in computer science (software engineering, data mining, artificial intelligence, databases, and distributed computing), such that data from each field included at least 2000 distinct author names and at least ten years of significant coverage. All documents contained in CiteSeer from each venue were obtained and the top 100 keyphrases were extracted from each document using the KEA keyphrase extraction algorithm [7]. The KEA algorithm was trained on the CSTR corpus provided with KEA containing 320 manually labeled abstracts from the computer science domain, and keyphrases were allowed to range from one to three words in length. Author names were

normalized such that only the initials of the first and middle names were kept along with the full last name. The correlated bipartite graphs were then generated for each year of data by linking authors with specific keyphrases and keyphrases with the venues in which they appeared. The final dataset contained 12,677 authors and 45,295 keyphrases from 30 distinct venues ranging over the years 1969 to 2004. The total number of documents used was 13,310.

Experiments on this data set began by empirically determining the appropriate number of clusters. While it is an open problem to determine the dimension of a subspace for embedding a graph, we used simple heuristics. We ran the proposed community discovery algorithm (fsi) with different k and chose the k corresponding to the smallest $\widetilde{J}$ (or the greatest $\gamma = \text{tr}(U^T(M + \dot{C})U)$) as in Eq. 22. We observed that the γ initially grows dramatically as k increases, but grows at a much lower rate as k becomes large. Thus we chose the smallest k that gave the near maximum γ. This gave us $k = 4$.

Then we ran the temporal community discovery (t-par) algorithm with $k = 4$ with various settings of λ. For screening the results, we judge the quality of discovery by examining the grouping of venues since their number is small. We observed that the quality is better for greater λ, supporting the results from synthetic datasets that suggest λ should be set proportionally to $|X|/|Z|$. Here we set $\lambda = 0.6$.

We observe that the resulting communities of authors, venues, and words are well grouped. Four communities are discovered for *artificial intelligence and machine learning*, *database and data mining*, *parallel and distributed computing*, and *software engineering*. We present two discovered communities and their authors in Table 2 and Table 3. In our experiments, we used the discovered venue set to manually produce community labels. The keyphrases (ranked by frequency) were considered as the summarization of a community.

Table 2 includes a subset of authors discovered in the *artificial intelligence and machine learning* community over six time periods. For presentation, we rank the authors by their number of papers within the corresponding periods. We can observe that the community memberships of authors are relatively stable but change over time. In the experiments, we observed that the top authors remained as the "core" members of the corresponding community and there were many more authors who had joined and left from the communities during these six time periods. The leftmost column shows the top venues. Similarly, authors from the *database and data mining* community are presented in Table 3.

We used the discovered clusters of words as the descrption for the corresponding communities. Summarizations of two communities are presented in Table 4 and Table 5. Words are ranked by their frequency of occurrence within the data. Those words that did not occur in the previous period are highlighted. Over the six time periods, we can see the emergence of new words, which presumably indicate the evolution of interests of the community.

Finally, we show the changes in communities' sizes over time in Fig. 8(b). The size of a community is measured by the number of distinct authors discovered within a particular time period. The sizes of the four communities are scaled to sum up to one.

8 Conclusion

This paper addresses an emerging problem of temporal community discovery from communication documents, by which one can observe the temporal trends in community membership over time. The problem is formulated as a tripartite graph partitioning problem with prior knowledge available of entity covariances. Temporal communities are discovered by threading the partitioning of graphs in different time periods, using a new constrained partitioning algorithm. Evaluation of the new algorithm is carried out on several synthetic datasets and a real-world dataset prepared from CiteSeer. Experiments on synthetic data reveal the properties of the new algorithm in various graph conditions. Experiments on CiteSeer data show the effectiveness of the proposed approach in author community discovery and community summarization. Future work will seek to track the community membership of individuals over time and investigate the applicability of the proposed methods to different domains such as viral marketing or recommendation services. Additionally, topical trends over time will be further investigated to track changing interests and events within communities.

References

[1] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54, New York, NY, USA, 2006. ACM Press.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, New York, NY, USA, 2003. ACM Press.

[4] C. Ding. A tutorial on spectral clustering. In *Proc. of the 25th International Conference on Machine Learning*, July 2004.

[5] C. Ding, X. He, H. Zha, M. Gu, and H. D. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM '01: Proceedings of International Conference on Data Mining*, pages 107–114, 2001.

[6] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66. ACM Press, 2001.

[7] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 668–673, 1999.

[8] B. Gao, T.-Y. Liu, X. Zheng, Q.-S. Cheng, and W.-Y. Ma. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 41–50, New York, NY, USA, 2005. ACM Press.

Venues	1969-94	1994-96	1996-98	1998-2000	2000-02	2002-04
JMLR	M I Jordan	M I Jordan	W L Johnson	S Thrun	D Koller	A Blum
	L P Kaelbling	L P Kaelbling	N Friedman	C Boutilier	A W Moore	S Thrun
	J Y Halpern	Z Ghahramani	D Koller	T Sandholm	M I Jordan	S Zilberstein
PAMI	S P Singh	S P Singh	R E Schapire	D Koller	M L Littman	P Stone
	Z Ghahramani	M K Warmuth	Y Singer	N Friedman	S Thrun	J Langford
	M K Warmuth	T G Dietterich	R Dechter	Y Singer	D Schuurmans	T Eiter
ICML	T G Dietterich	T Dean	T J Sejnowski	A Mccallum	J Shawe-taylor	P Domingos
	T Dean	Y Bengio	H S Seung	L P Kaelbling	S P Singh	A K Jain
	Y Bengio	P Smets	D Poole	S P Singh	N Friedman	S Baker
AAAI/IAAI	P Smets	W Maass	M I Jordan	P R Cohen	N Cristianini	S Chawla
	W Maass	V Tresp	N Tishby	R Khardon	A Mccallum	R Dechter
	V Tresp	D Weinshall	R Greiner	M J Kearns	P Domingos	C Guestrin
UAI	D Weinshall	D Geiger	Y Mansour	K Nigam	Y Bengio	C Boutilier
	D Geiger	S Kambhampati	M K Warmuth	N Cristianini	D Freitag	M J Kearns
	D Poole	A Saffiotti	Y Freund	J Shawe-taylor	A Y Ng	T Lukasiewicz
IJCAI	R E Schapire	R E Schapire	D P Helmbold	C Baral	M K Warmuth	A Demiriz
	S Kambhampati	D S Nau	C Boutilier	A W Moore	G E Hinton	S P Singh
	C Baumlckstroumlm	H A Simon	M L Littman	D Fox	N Tishby	D Koller
JAIR	F Bacchus	F Bacchus	P Dayan	D Roth	A J Smola	D Schuurmans
	A Saffiotti	D Poole	A J Grove	M P Wellman	G Raumltsch	S Prabhakar

Table 2. Machine learning community during 1969-2004 in a CiteSeer sample.

Venues	1969-94	1994-96	1996-98	1998-2000	2000-02	2002-04
PODS	M Yannakakis	M Yannakakis	R Hull	A Mendelson	G Gottlob	S Abiteboul
	V Vianu	V Vianu	A Mendelzon	J Paredaens	V Vianu	L Popa
	A Gupta	J Y Halpern	Z M Zsoyoglu	C Papadimitriou	H Garcia-molina	T Milo
	Garciacute	Garciacute	H Garcia-molina	H Garcia-molina	J Widom	P G Kolaitis
SIGMOD	J Widom	J Widom	D Suciu	S Abiteboul	A Y Halevy	P S Yu
	J F Naughton	H Garcia-molina	A Silberschatz	D Florescu	C Faloutsos	F Neven
	H Garcia-molina	J F Naughton	A Y Levy	A Y Levy	D Suciu	C Beeri
	C Faloutsos	C Faloutsos	L Libkin	R Motwani	D Gunopulos	R Rastogi
VLDB	A Kemper	J Hammer	G Moerkotte	L V S Lakshmanan	S Lee	J Han
	K Ramamritham	A Biliris	S Seshadri	T Milo	J Han	D Srivastava
	G Moerkotte	K Ramamritham	S Abiteboul	S Cluet	W Fan	M N Garofalakis
	I S Mumick	A Kemper	J Widom	J Han	R Rastogi	J Widom
SIGMOD Record	A Biliris	C Baumlckstroumlm	R Agrawal	D Suciu	C S Jensen	A Y Halevy
	J Hammer	G Moerkotte	R Ramakrishnan	J S Vitter	H V Jagadish	C Li
	M Chen	I S Mumick	S Sudarshan	R Rastogi	D Kossmann	J Madhavan
	P S Yu	K Lin	K Ramamritham	G D Giacomo	D Srivastava	W Fan
ICDM	T Milo	S Berson	A Kemper	C S Jensen	K Chakrabarti	B Babcock
	D Suciu	D Suciu	D Florescu	D Srivastava	S Muthukrishnan	C Y Chan
	J Han	D Kossmann	P Atzeni	O Shehory	D S Weld	C Koch
	K Lin	C A Knoblock	M Benedikt	M Lenzerini	G D Giacomo	J Gehrke

Table 3. Database community during 1969-2004 in a CiteSeer sample.

years	**words**
1994-96	learning model training probability value image set action input points output variables goal point values search policy agent function selection examples error units distance knowledge classification representation recognition region test
1996-98	learning **state** model image value training probability **network** set values variables **class** error points input point action **vector** representation **sequence** agent search **distribution** recognition units **random** output classification case robot
1998-00	learning model state value training set image probability values action points policy error search point sequence **actions** noise function knowledge distribution classification robot parameters estimate **text** optimal estimation accuracy representation
2000-02	learning model training set error image probability **matrix** point sequence distribution **kernel** classification **random** features state estimation function representation **input** accuracy **strategy vector** text **prediction** parameters bound **approach selection**
2002-04	learning model set probability **policy points** training sequence image **variables** optimal **algorithm** function matrix **search** point error **distance erent** random bound classification max **robot** estimate representation case **expected** distribution vector

Table 4. Frequent words in the machine learning community during 1994-2004 in a CiteSeer sample.

years	**words**
1994-96	query data database queries object path event cost type user execution objects table class transaction local rules server client join name formula update rule attribute attributes view pages plan read
1996-98	query data queries database object cost **tree information** view user attributes pages objects rules join plan table update transaction type attribute **constraints** page access server disk **requests real-time label** client
1998-00	query data queries user information database pages rules constraints plan **path** attributes attribute view join formula table sources update objects **request** strategy **documents level instance items** rule **web** spatial **application**
2000-02	data query queries **points** information path **cost** xml database attributes **values** pages **tree** constraints table join plan **type** objects page **distance management example document** attribute update labeled items documents web
2002-04	data query **node** queries xml path values tree database attributes table document join name plan **service cache** objects **return selection** constraints type **patterns** label mapping attribute tuples index items **root**

Table 5. Most frequent words in the database community during 1994-2004 in a CiteSeer sample.

[9] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[10] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 491–501, 2004.

[11] D. Harel and Y. Koren. Clustering spatial data using random walks. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 281–286, 2001.

[12] X. Ji and W. Xu. Document clustering with prior knowledge. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 405–412, New York, NY, USA, 2006. ACM Press.

[13] J. Kubica, A. Moore, J. Schneider, and Y. Yang. Stochastic link and group detection. In *Proceedings of the 2002 AAAI Conference*, pages 798–804, 2002.

[14] A. Pothen, H. D. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[15] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *UAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494. UAI Press, 2004.

[16] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[17] J. R. Tyler, D. M. Wilkinson, and B. A. Huberman. Email as spectroscopy: automated discovery of community structure within organizations. *Communities and technologies*, pages 81–96, 2003.

[18] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[19] A. Y. Wu, M. Garland, and J. Han. Mining scale-free networks using geodesic clustering. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 719–724. ACM Press, 2004.

[20] H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. In *Neural Information Processing Systems*, volume 14, 2001.

[21] H. Zha, X. He, C. Ding, H. Simon, and M. Gu. Bipartite graph partitioning and data clustering. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 25–32, New York, NY, USA, 2001. ACM Press.

[22] D. Zhou, E. Manavoglu, J. Li, C. L. Giles, and H. Zha. Probabilistic models for discovering e-communities. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 173–182. ACM Press, 2006.