CONSTRAINED CLUSTERING BASED ON THE LINK STRUCTURE OF A DIRECTED GRAPH

Yinghui Yang, Graduate School of Management, University of California, Davis, CA, USA, yiyang@ucdavis.edu

Zijie Qi, Microsoft Corporation, Redmond, WA, USA, qizijie@gmail.com

Hongyan Liu, School of Economics and Management, Tsinghua University, Beijing, China liuhy@sem.tsinghua.edu.cn

Jun He, Department of Computer Science, Renmin University, Beijing, China, hejun@ruc.edu.cn

Abstract

In many segmentation applications, data objects are often clustered based purely on attribute-level similarities. This practice has neglected the useful information that resides in the link structure among data objects and the valuable expert domain knowledge about the desirable cluster assignment. Link structure can carry worthy information about the similarity between data objects (e.g. citation), and we should also incorporate the existing domain information on preferred outcome when segmenting data. In this paper, we investigate the segmentation problem combining these three sources of information, which has not been addressed in the existing literature. We propose a segmentation method for directed graphs that incorporates the attribute values, link structure and expert domain information (represented as constraints). The proposed method combines these three types of information to achieve good quality segmentation on data which can be represented as a directed graph. We conducted comprehensive experiments to evaluate various aspects of our approach and demonstrate the effectiveness of our method.

Keywords: Data Mining, Directed Graph, Graph Segmentation, Clustering

1 INTRODUCTION

In recent years, the amount of data businesses generate and record has grown drastically with the continuing development of powerful technologies. Consequently, the business of helping organizations to make sense of their proliferating data has expanded by leaps and bounds (The Economics Feb. 25, 2010). As a result, the adoption of data mining techniques in a wide variety of industries has picked up speed. Among the popular data mining techniques, clustering has been widely applied in various domains. Clustering is to divide data objects (e.g. customers, products, news articles, etc.) into groups such that objects inside the same group are similar and objects in different groups are unalike. Data objects are often described by attribute values. For example, customers are described by attributes such as gender, age, income, education level, etc. When determining whether two data points are similar, a distance function is often used to calculate the similarity between data points based on their attribute values. Examples of such distance functions include Euclidean distance and Manhattan distance (Deza and Deza 2009). Distance-based clustering techniques, such as k-means (Kanungo et al. 2002), can be easily applied on data objects with attribute values.

While it is convenient to cluster solely based on attribute values, omitting other available information could lead to suboptimal clustering results when information other than attribute values is available. In this paper, we take into consideration information on link structure as well as expert domain knowledge on desirable cluster assignment while performing segmentation on data objects. In many applications, data objects are connected with each other. For example, in a social network, each person has his or her own characteristics such as gender and age which can be captured by attribute values. At the same time, people are interconnected through the network. A link between two people could mean that they share the same interests or background, giving out further information about how similar they are. It is often difficult to quantify such information into attribute values since there are no predefined values and these are often relative values instead of absolute values (also these pair-wise similarities cannot be represented into attribute values). News articles can be represented by a vector of words (attributes), and similarity between articles can be derived based on the frequency of the words representing the articles. Aside from the attribute-level information, news articles can provide links to other related news articles which could be a much stronger indicator for similarity.

Aside from information about link structure, another valuable source of information which can significantly affect segmentation quality comes from the experts. Expert domain knowledge carries information about the preferred results (i.e. which data points should be groups together). For example, in the social network setting, experts might prefer to have two members of the social network grouped together because they are both very active users instead of users with similar demographics. This can be expressed as a constraint to the segmentation method so that we can guarantee these two users will be put into the same cluster. Domain expert knowledge can take many forms. In this paper, we specifically study two types, must-link constraints and cannot-link constraints. A must-link constraint specifies that a pair of data objects should belong to the same group. A cannot-link constraint states that two data objects should not belong to the same group. These types of constraints can guide the clustering algorithm to converge to a more desirable solution. For example, a newspaper wants to group demographically similar towns into the same editorial segment so that it can customize its content (Berry and Lindoff 2004). However, the newspaper is only interested in grouping similar towns that are close to each other into the same segment so that the

trucks delivering the localized editions could take sensible routes. Adding this preference information as clustering constraints can help the newspaper get the clusters they are looking for. Experts may want two specific towns be grouped together due to the feasibility of the truck route. This information can be fed into the clustering process as a must-link constraint. If two towns have very similar demographics but they are very far from each other, experts can create a cannot-link constraint to specify this requirement. This cannot-link constraint can be used to adjust the clusters to make sure these two towns are not in the same segment. In the social network setting, two users with very similar activities (e.g. link to many people) may be put into a must-link constraint, and two users with similar demographics but very different behavior (e.g. one has lots of connections, and the other has few) can be included in a cannot-link constraint.

In this paper, we design a procedure that incorporates attribute information, link structure and constraints into the clustering process. In the first stage, we combine the attribute information and link structure into a directed graph whose edge weights carry both attribute similarity and link strength (i.e. link weight which measures how strong the link is). In the second stage, we transform the directed graph generated in the first stage to a compressed graph so that the must-link constraints expressed in domain information can be fully captured. Then, we perform graph partition on the compressed graph, and finally we transform the clusters on the compressed graph to the clusters on the original graph and adjust the clusters according to the cannot-link constraints. Using graph theory, we prove that the best partition that optimizes the clustering objective function on the compressed graph is equivalent to the best partition in the original graph with constraints incorporated.

Our main contribution is to provide a segmentation method on directed graphs with constraints. To our knowledge, this paper is the first to perform constrained clustering on directed graphs. The overall framework is also a great contribution. The entire framework integrates multiple steps, which include combining the attribute information and link structure into a directed graph, transforming the directed graph to a compressed graph to capture the constraints, partitioning the compressed graph, transforming the clusters on the compressed graph to the clusters on the original graph, and finally adjusting the clusters to satisfy the cannot-link constraints. This framework is a problem driven framework. The important problem we study is how to leverage all the information that is available (attribute values, link structure and domain information) to achieve good quality segmentation. This problem has not been studied before. The experiments conducted on multiple real-world data sets demonstrate that adding links and constraints can often achieve better quality clusters, and our method performs much better than the closest alternative (constraint clustering on undirected graphs). Furthermore, our method is not restricted to one specific type of clustering algorithm, and it works on many types of data.

The most related literature provides methods designed to combine link information with attribute values, especially in the document clustering domain (Bolelli et al. 2006). He et al. (2001) used the hyperlink structure as the dominant source of similarity between documents, and the link-based similarity measure is augmented by textual content similarity and citation similarity. Probabilistic models have also been developed to model cluster membership using both attribute information and link structure (Getoor et al. 2002, Cohn et al. 2003, Neville et al. 2003). Combining both attribute values and link information alone is not the main contribution of this paper. We focus more on the entire framework of clustering based on attribute values, link information and domain information. In the future, we plan to investigate more the aspect of combining link structure with attribute values and

compare our method with existing methods. Other related fields include graph segmentation (Long et al. 2008) and constrained clustering (Chapelle et al. 2006, Cohn et al. 2003). Due to space limit, we do not discuss further details.

The remainder of the paper is organized as follows. Section 2 defines the problem of directed graph segmentation with constraints, presents the entire framework and discusses details for each stage in the framework. Section 3 reports various experimental results to illustrate the effectiveness of our approach. Section 4 summarizes our study.

2 CONSTRAINED CLUSTERING ON DIRECTED GRAPHS

Many data sets have both attribute and link information such as web pages with hyperlinks and documents with citations. The intrinsic link structure forms a directed graph while the similarities between the objects derived from their attribute values can be integrated into the same directed graph. A clustering algorithm which makes use of both the attribute values and the link structure can potentially achieve better clustering quality compared to algorithms that do not consider both. Aside from the link structure and attributes, the domain information supplied by the experts can guide the algorithm to converge to a more desirable solution. For instance, if the experts specify that certain objects should or should not belong to the same cluster, we can cluster in a way that these constraints are fulfilled, generating results that align with domain information.

Given a directed graph, the goal of our work is to find a partition/clustering which divides the graph into k disjoint subgraphs such that these subgraphs are unrelated with weak links between them while the nodes inside the same subgraph are similar with strong links. When we measure the strength of the links between subgraphs and similarity within the same subgraph, we consider both the link information and the attribute information. Meanwhile, all the constraints are satisfied in the clustering results.

2.1 Partition Directed Graphs with Link Structure and Node Attributes

We consider a directed graph $G = (V, E_L)$ with a node set $V = \{v_i\}_{i=1}^{n}$ and a link set E_L. The edge set E is constructed to capture both the link information and the attribute information (Note, we use link to refer to the link structure between nodes in the original graph, and use edge to refer to the connection between nodes incorporating both the link information and the attribute information). We use E_L to represent the link set and E to represent the edge set. For a pair of nodes u and v, there are three weights we can measure. (a) Link weight $w_L(\,)$, which measures the strength of the link between these two nodes. Note that in a directed graph, $w_L(u,v) \neq w_L(v,u)$. If there is no link going from u to v, then $w_L(u,v) = 0$. (b) Similarity weight $w_S(\,)$, which measures the similarity between two nodes calculated based on the attribute values of these two nodes. $w_S(u,v) \geq 0$ and $w_S(u,v) = w_S(v,u)$. (c) Edge weight $w(\,)$, which is a measurement combining both the link weight and the similarity weight. The combination function we use to generate edge weights based on link weights and similarity weights is presented in Eq. (1).

$$w(u,v) = \alpha w_L(u,v) + \beta w_S(u,v) \qquad (1)$$

2.1.1. Determining $\boldsymbol{\alpha}$ *and* $\boldsymbol{\beta}$

In Eq. (1), α and β control the importance we assign to the link weight and the similarity weight. We use the following procedure to determine these two numbers.

Step 1. Assume that domain knowledge is applied on some small sample pairs of nodes, and each pair is assigned a label. Label 0 indicates that this pair of nodes should not belong to the same cluster, and label 1 indicates that the two nodes should belong to the same cluster.

Step 2. In order to determine α, we generate the following data structure illustrated in Table 1.

	Similarity value (continuous)	Label (0/1, whether the pair belong to the same cluster)
Pair 1	0.453	0
Pair 2	0.542	1
…	…	…
Pair 50	0.324	0

Table 1. Similarity values and Labels

Step 3. Calculate the correlation between the two columns of data in Table 1 according to the following formula. This formula is adopted because it is often used to calculate correlations between a continuous variable and a binary variable.

$$r_\alpha = \frac{\overline{X_p} - \overline{X_q}}{s_t} \cdot \sqrt{pq} \qquad (2)$$

p and q correspond to the proportion of 0s and 1s in the Label column.

$\overline{X_p}$ and $\overline{\mathrm{X_q}}$ are the average similarity value for the pairs with 0 or 1 Labels respectively.

$\mathrm{S_t}$ is the standard deviation for the continuous similarity values in the first data column.

Step 4. In order to determine β, we generate the following data structure illustrated in Table 2.

	Link exists? (0/1)	Label (0/1)
Pair 1	1	0
Pair 2	1	1
…	…	…
Pair 50	0	0

Table 2. Data Structure

Note that we did not use the link weights to indicate the strength of the link in the first data column, instead we use a binary variable to indicate whether a link exist between a pair of data points. This is because all the data sets we obtain only have 0/1 label to indicate whether a link exists. In the case where strength of the link is captured in the data, we can use Eq. (2) to replace Eq. (3) in Step 5.

Step 5. Calculate the correlation between the two columns of data in Table 2 according to the following formula. This formula is adopted because it is often used to calculate correlations between two binary variables.

$$r_\beta = \frac{ad-bc}{\sqrt{(a+b)(c+d)(a+c)(b+d)}}. \quad (3)$$

a, b, c, d are defined in Table 3. They represent the number of pairs with certain values.

	Label=0	Label=1
Link does not exist	a	b
Link exist	c	d

Table 3. Definition of a, b, c, d

Step 6. Sigmoid Transformation and Normalization

The range for both r_α and r_β is $(-\infty, +\infty)$, and we transform r_α and r_β to values between (0,1) by applying the sigmoid function $y = \frac{1}{1+e^x}$. We use r'_α and r'_β to denote the two values after the transformation. Then we normalize them, and obtain the final α and β. That is $\alpha = \frac{r'_\alpha}{r'_\alpha + r'_\beta}$ and $= \frac{r'_\beta}{r'_\alpha + r'_\beta}$. Since r_α and r_β went through the same transformation and normalization, their relative importance is preserved and the final α and β reflect the relative importance that similarities and links have on cluster assignments.

2.1.2. Objective Function for Clustering

After the transformation, the initial graph with sparse links becomes a fully connected graph, and any pair of nodes (u, v) will have two weights, $w(u, v)$ and $w(v, u)$ ($w(u, v) \geq 0$; $w(v, u) \geq 0$; $w(u, v) \neq w(v, u)$). It is possible that some edges have zero weight. In this case, the graph is not fully connected. The method we propose in this paper apply to both the fully connected graphs and the not fully connected graphs.

In the rest of the paper, we will only use the edge weight $w(u, v)$, and we will simply use "weight" to refer to the edge weight. Let W denote the affinity matrix where the entry at position (i, j) is $w(v_i, v_j)$. The degree of node u is the sum of the weights of the edges which start from u, $d_u = \sum_{i=1}^{n} w(u, v_i)$. Let the degree matrix D be a diagonal matrix where $D_{ii} = d_i = \sum_j W_{ij}$.

For a directed graph with non-negative weights $w(u, v) \geq 0$, we can normalize the weights so that sum of the weights from a node u to its out-neighbors sum up to 1. The normalized weight $p(u, v)$ on edge (u, v) is defined in Eq. (4). The normalized weights can be interpreted as transition probabilities of a random walk and $\sum_{i=1}^{n} p(u, v_i) = 1$. Let $P = D^{-1}W$ denote the transition matrix.

$$p(u, v) = \frac{w(u,v)}{d_u} \quad (4)$$

If the graph is strongly connected (which is the case in our problem) and aperiodic, there exists a unique row vector ϕ such that $\phi P = \phi$ and $\sum_{i=1}^{n} \phi(i) = 1$. This vector is called stationary distribution vector in random walk theory (Norris 1999). No matter which node the random walk starts from, the probability that the random walk converges to node v_i after infinite jumps is $\phi(i)$. We define a diagonal matrix Φ where $\Phi_{ii} = \phi(i)$.

In the graph partition problem on a directed graph, we aim to divide all the nodes into k disjoint sets $C_1, \ldots, C_k$, such that the random walk process seldom jumps between different clusters. Take $k = 2$ as

an example, the directed graph $G = (V, E)$ can be partitioned into two disjoint sets A and B ($A \cap B = \emptyset, A \cup B = V$). The transition probability $p(X_1 \in B | X_0 \in A)$ from A to B represents the probability of jumping from a node in A to a node in B (X_0 represents the node where the random walk is currently located and X_1 represents the node that the random walk jumps to next). We assume that the probability the random walk is at X_0 is specified in the stationary distribution (i.e. $p(X_0 = v_i) = \phi(i)$). The objective function of the clustering can be written in Eq. (5), which is also termed as normalized cut in graph theory (Shi and Malik 2000, Maila and Shi 2001).

$$Ncut(A, B) = p(X_1 \in B | X_0 \in A) + p(X_1 \in A | X_0 \in B) \quad (5)$$

By minimizing Eq. (5), we can minimize the probability of jumping between cluster A to cluster B. The spectral clustering algorithm (Ng et al. 2001) can be used to generate clusters A and B to minimize Eq. (5).

2.2 General Framework

The problem that we address in this paper is to find a good partition on a directed graph with link structure and node attributes such that the given constraints are satisfied. Figure 1 describes the main steps in our framework.

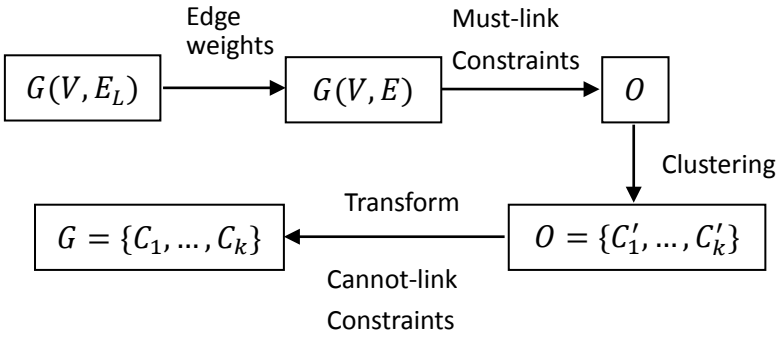


Figure 1: General Framework

In the first step, we transform $G = (V, E_L)$ to $G = (V, E)$ by gererating the edge weights based on both the link weights and similarity weights. In the second step, we encode the must-link constraint set $MCons$ into $G = (V, E)$ to form a compressed directed graph O. In the third step, we run the spectral clustering algorithm on O to partition the data on O. Then, we transform the clusters on O to the clusters on $G = (V, E)$. Finally, we adjust the clusters to satisfy the set of cannot-link constraints $CCons$. We will describe the details in the following sections.

2.3 Incorporating Must-link Constraints into a Directed Graph

Domain information about whether certain data objects should belong to the same cluster is expressed into a set of singular constraints which include must-link constraints ($MCons$) and cannot-link constraints ($CCons$). A constraint is between a pair of nodes (v_i, v_j). The type of constraints we consider in this paper is must-link constraint (Wagstaff and Cardie, 2000), $ML(v_i, v_j)$, which indicates that v_i and v_j must belong to the same cluster, and cannot-link constraint (Wagstaff and Cardie, 2000), $CL(v_i, v_j)$, which indicates that v_i and v_j must not belong to the same cluster.

If there exists a must-link constraint $ML(v_i, v_j)$ between v_i and v_j, these two nodes will be merged into one group, which is termed as a connected component. After going through all the must-link

constraints in the constraint set MCons, we get a set of connected components which fully represent all the constraints, $\mathrm{M} = \{\mathrm{M_t}\}_{\mathrm{t=1}}^{\mathrm{p}}$ where p is the number of connected components. If a node $\mathrm{v_i}$ has no corresponding constraint, it will form its own connected component $\mathrm{M_t}$ where $\mathrm{M_t} = \{\mathrm{v_i}\}$ and $|\mathrm{M_t}| = 1$. For example, for a node set $\mathrm{V} = \{\mathrm{v_1}, \mathrm{v_2}, \mathrm{v_3}, \mathrm{v_4}\}$ and a must-link constraint set $\mathrm{MCons} = \{\mathrm{ML}(\mathrm{v_1}, \mathrm{v_2}), \mathrm{ML}(\mathrm{v_1}, \mathrm{v_3})\}$, there will be two connect components $\mathrm{M_1} = \{\mathrm{v_1}, \mathrm{v_2}, \mathrm{v_3}\}$ and $\mathrm{M_2} = \{\mathrm{v_4}\}$. We denote the mapping from the node set $\mathrm{V(G)}$ to the connected component set M as $\pi: \mathrm{V(G)} \rightarrow \mathrm{M}$.

The nodes within the same connected component are merged to form a node in the compressed directed graph O. The weight from a node $\tilde{\mathrm{v}}_\mathrm{a}$ to another node $\tilde{\mathrm{v}}_\mathrm{b}$ in O is defined as $\tilde{\mathrm{w}}(\tilde{\mathrm{v}}_\mathrm{a}, \tilde{\mathrm{v}}_\mathrm{b}) = \sum_{\mathrm{v_i} \in \mathrm{M_a}; \mathrm{v_j} \in \mathrm{M_b}} \mathrm{w}(\mathrm{v_i}, \mathrm{v_j})$, which is the sum of the weights from every node in $\mathrm{M_a}$ to every node in $\mathrm{M_b}$.

For the compressed directed graph O, its affinity matrix $\tilde{\mathrm{W}}_{\mathrm{p \times p}}$ can be written as $\tilde{\mathrm{W}} = \mathrm{Y^T W Y}$, where $\mathrm{Y_{n \times p}}$ is a subspace projection matrix defined in Eq.(6).

$$Y_{ij} = \begin{cases} 1 & if\ v_i \in M_j (1 \le j \le p), \\ 0 & otherwise. \end{cases} \tag{6}$$

2.4 Finding the Clusters

We apply spectral clustering on the compressed directed graph O to generate k clusters $\{\mathrm{C'_1}, \mathrm{C'_2}, \ldots, \mathrm{C'_k}\}$. Then, we transform the clusters $\{\mathrm{C'_1}, \mathrm{C'_2}, \ldots, \mathrm{C'_k}\}$ on O to clusters $\{\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}\}$ on G according to the mapping π. For example, if $\mathrm{C'_1}$ contains two connected components, then the nodes in these two connected components form the cluster $\mathrm{C_1}$ in the original space. Suppose the clustering in O is denoted by a $\mathrm{p} \times 1$ vector r' where $\mathrm{r'_i}$ indicates the cluster number for node $\mathrm{x_i}$, then we can retrieve the clustering r in the original space by using $\mathrm{r} = \mathrm{Yr'}$, where Y is defined in Eq. (6). Finally, we adjust the final clustering according to the cannot-link constraints. The procedure for the adjustments is detailed in Figure 2.

Input: $\{\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}\}$, a cannot-link constraint set *CCons*.
1. For each item $\mathrm{C_i}$ in $\{\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}\}$
2. If $\mathrm{C_i}$ does not violate any constraint in CCons:
3. Go to Step 1.
4. Else:
5. For every constraint $\mathrm{CL}(\mathrm{v_j}, \mathrm{v_k})$ violated:
6. Compute the average similaity between $\mathrm{v_j}$ (or $\mathrm{v_k}$) and other nodes in $\mathrm{C_i}$.
7. Take the one with the smaller similarity (say, $\mathrm{v_j}$) out of $\mathrm{C_i}$, and group
. it to another cluster which has the highest similarity if adding to
. the cluster does not violate any constraint in CCons.
8. If no cluster satisfies the no-violation condition, split that node ($\mathrm{v_j}$)
. to an individual cluster, and add this cluster to the set of clusters.
9. Output the clusters that contain enough number of nodes.

Figure 2: Procedure for Incorporating Cannot-link Constraints

2.5 The Complete Algorithm and an Illustrative Example

Figure 3 illustrates the complete algorithm.

Input: A data set V with attributes, the link set E_L, number of clusters - k, a must-link constraint set $MCons$, a cannot-link constraint set $CCons$
Output: Clusters $C_1,\ldots,C_k$.
• Derive a directed graph $G=\{V,E\}$ and its affinity matrix W in Eq. (1).
• Formulate the constraint matrix Y in Eq. (6)
• Embed the must-link constraint set $MCons$ into G to derive a compressed directed graph O, where the affinity matrix is $\widetilde{W}=Y^TWY$.
• Find the clustering in O using spectral clustering algorithm and use a vector r' to represent the cluster assignment.
• Derive the clustering in G by using $r=Yr'$.
• Adjust the clustering in G to satisfy the cannot-link constraints $CCons$.

Figure 3: Algorithm for Clustering Directed Graphs with Constraints

Our algorithm is designed to find k clusters on a directed graph which also satisfy the given constraints. First, we combine both attribute and link information to derive a directed graph $\mathrm{G}=\{\mathrm{V},\mathrm{E}\}$ (see Eq.(1)). Second, the given must-link constraint set MCons is described in a constraint matrix Y. The original graph G is then compressed into a graph O through matrix Y (see Section 2.3). We then find k clusters on O, which are represented by a vector r' (i.e. $\mathrm{r'_i}$ denotes which cluster $\tilde{\mathrm{v}}_\mathrm{i}$ belongs to). Then, the clusters in G are retrieved by using $\mathrm{r}=\mathrm{Yr'}$. Finally, the clusters are adjusted to satisfy the cannot-link constraints.

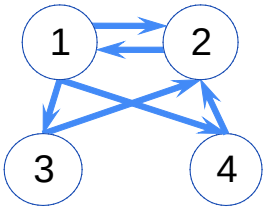
We use a simple example in Figure 4 to illustrate the steps in our algorithm (the last adjustment step is straightforward, thus not included in this example). Suppose the derived directed graph is $\mathrm{G}=\{\mathrm{V},\mathrm{E}\}$, $\mathrm{V}=\{1,2,3,4\}$, $\mathrm{E}=\{(1,3),(1,4),(3,2),(4,2),(1,2),(2,1)\}$ and k=2. We illustrate using a not fully connected graph which is more general than the fully connected graph. We assume that all the positive edge weights are 1s for simpler calculation. The affinity matrix W, the transition probability matrix P are as follows.

$$W=\begin{bmatrix}0&1&1&1\\1&0&0&0\\0&1&0&0\\0&1&0&0\end{bmatrix},\qquad P=\begin{bmatrix}0&0.33&0.33&0.33\\1&0&0&0\\0&1&0&0\\0&1&0&0\end{bmatrix}$$

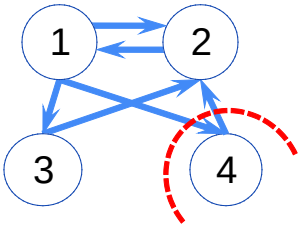
The clustering without must-link constraints generates two clusters which are illustrated in Figure 4(b). The two clusters are $\mathrm{A}=\{1,2,3\},\mathrm{B}=\{4\}$. However, if we prefer to have 3 and 4 in the same cluster, we can add the must-link constraint $\mathrm{ML}(3,4)$. After merging 3 and 4 into one node, we get the compressed directed graph $\mathrm{O}=\{\mathrm{V'},\mathrm{E'}\}$ in Figure 4(c) where $\mathrm{V'}=\{1,2,3'\}$ and $\mathrm{E'}=\{(1,2),(2,1),(1,3'),(3',2)\}$. The number on the edge indicates the new weight, and the weight for all other edges is 1. We formulated the constraint matrix Y, the affinity matrix $\widetilde{\mathrm{W}}$, the transition probability matrix $\widetilde{\mathrm{P}}$, as follows:

$$Y=\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\\0&0&1\end{bmatrix},\qquad \widetilde{W}=Y^TWY=\begin{bmatrix}0&1&2\\1&0&0\\0&2&0\end{bmatrix},\qquad \tilde{P}=\begin{bmatrix}0&0.33&0.67\\1&0&0\\0&1&0\end{bmatrix},$$

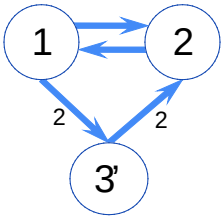
According to the spectral clustering algorithm, the directed graph is divided into two parts $\mathrm{A}=\{1,2\},\mathrm{B}=\{3'\}$ in Figure 4(d), and this partition can be denoted as $\mathrm{r'}=[1\ 1\ 2]$. Thus we derive the clustering $\mathrm{A}=\{1,2\},\mathrm{B}=\{3,4\}$ in the original directed graph G represented by $\mathrm{r}=\mathrm{Yr'}=[1\ 1\ 2\ 2]$, see Figure 4(e).



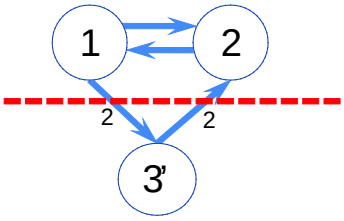
(a) The original directed graph *G*



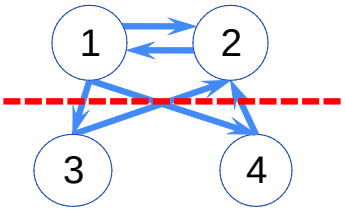
(b) Clusters without constraints



(c) Compressed graph *O* after adding *ML*(3,4)



(d) Clusters on graph *O*



(e) Clusters on graph *G*

Figure 4: An illustrative example

We leverage theories on coverings of directed graphs to prove that the best partition that optimizes the objective function on O is equivalent to the best partition in the original graph with must-link constraints incorporated. This will help us to justify the process we use to incorporate must-link constraints into G to generate the compressed graph O and cluster on O instead of clustering directly on G. Proof is available upon request.

3 EXPERIMENTS

We conduct comprehensive experiments to evaluate the effectiveness of our framework. We first compare with an existing constraint clustering method on undirected graphs. We also setup experiments to isolate the value of adding link information and constraints. We also conducted experiments to derive the following conclusions, but couldn't include the detailed results due to space limit. We show that our general framework works for different clustering methods with different distance metric and clustering objective functions. We apply our framework to a different type of data set to demonstrate that our framework is generalizable. We use synthetic data set to demonstrate the scalability of our approach.

To our knowledge, this paper is the first to perform constrained clustering on directed graphs. The closest alternative is to convert a directed graph to an undirected graph and perform constrained clustering on the undirected graph. We compare with a popular method for constrained clustering on undirected graphs (Yu and Shi, 2004). Yu and Shi (2004) provide a procedure of converting a directed graph to an undirected graph. We use six data sets, among which four are web page data consisting of a set of web pages from four universities: Cornell, Washington, Wisconsin, and Texas. These web pages are clustered into one of the following five clusters: course, faculty, student, project and staff. The other two data sets, Cora and Citeseer, consist of academic papers. The link structure captures the citation

network among the papers. The papers are clustered based on content. The results presented in Table 4 clearly demonstrate that our approach strongly dominates that of Yu and Shi (2004).

	Num Constraints	0	10	20	30	40	50
Cornell	only Sim	-0.2%	4.9%	24.6%	26.6%	19.5%	31.7%
	Sim + Link	19.0%	25.6%	43.2%	27.3%	36.5%	35.0%
Washington	only Sim	12.5%	11.9%	8.2%	10.9%	19.2%	19.1%
	Sim + Link	12.0%	12.4%	8.1%	16.5%	17.7%	20.1%
Wisconsin	only Sim	1.3%	12.0%	15.5%	12.7%	27.7%	19.4%
	Sim + Link	12.0%	21.6%	20.8%	28.5%	54.4%	40.2%
Texas	only Sim	2.7%	10.0%	2.2%	7.4%	9.7%	3.8%
	Sim + Link	19.8%	25.1%	14.6%	18.1%	25.5%	48.6%
	Num Constraints	0	100	200	300	400	500
Cora	only Sim	-5.0%	25.9%	37.1%	38.0%	37.5%	35.1%
	Sim + Link	-3.8%	4.8%	10.9%	50.6%	42.4%	37.3%
Citeseer	only Sim	9.0%	85.6%	115.9%	128.6%	135.1%	139.1%
	Sim + Link	5.2%	-12.1%	11.7%	78.6%	100.1%	90.1%

Table 4. Accuracy Improvement over Yu and Shi (2004)

Note: The numbers in the table represent the percentage improvement on accuracy our method has over Yu and Shi (2004).

We also conducted comparisons to separate out the value and effects of the three information sources (attribute similarity, links, and constraints). Our experimental results show that the accuracy of the clusters generated based on data incorporating both similarity and link structure tends to be higher among the three across most of these six data sets. However, we need to acknowledge the fact that for some data sets (e.g. the Citeseer data), attribute similarity may contain more accurate information than links for clustering purpose. Adding links may dilute the input information. This is similar to the case of feature selection in the classification setting, which can increase accuracy by reducing the number of attributes. However, there is no readily available feature selection solution for the combination of attributes and links. This can be a direction of future research. However, the decision about whether to add information from links during the clustering process can be made based on an experimental procedure on a small portion of the data. Experiments with constraints show that adding constraints increased the accuracy. In majority of the cases, using similarity, link structure and constraints together perform the best.

7 CONCLUSION

In this paper, we designed a framework to effectively combine attribute values, link structure and domain information for better segmentation. We proposed a novel method to perform constrained clustering on directed graphs. The entire framework is problem-driven and can help solve the problem of better clustering when all the three sources of information (attribute, link, and domain) are available. The framework integrates multiple stages, which include combining the attribute information and link structure into a directed graph, transforming the directed graph to a compressed graph to capture the must-link constraints, partitioning the compressed graph, transforming the clusters on the compressed graph to the clusters on the original graph, and finally adjusting the clusters to satisfy the cannot-link constraints. The experiments conducted on real-world data sets demonstrate that adding links and constraints can often achieve better quality clusters, and our method performs much better than the closest alternative (constraint clustering on undirected graphs). Furthermore, our method is not restricted to one specific type of clustering algorithm, and it works on many types of data.

ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China under Grant No. 71272029 and 71490724, the 863 program under Grant No. 2014AA015204, and the Beijing Municipal Natural Science Foundation under No. 4152026.

REFERENCES

Berry, M., and Linoff, G. (2004). Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management, 2nd Edition. Wiley Computer Publishing.

Bolelli, L., Ertekin, S. and Giles, C. L. (2006). Clustering Scientific Literature Using Sparse Citation Graph Analysis. Lecture Notes in Computer Science, 4213, 30-41.

Chapelle, O., Schölkopf, B. and Zien, A. (2006). Semi-supervised learning. MIT Press, Cambridge.

Chung, F. (2005). Laplacians and the cheeger inequality for directed graphs. Annals of Combinatorics, 9(1), 1–19.

Cohn, D., Caruana, R. and McCallum, A. (2003). Semi-supervised clustering with user feedback. Technical Report TR2003-1892, Cornell University.

Deza, E., and Deza, M. M. (2009). Encyclopedia of Distances, Springer, 94.

The Economics. (Feb. 25, 2010). Data, Data Everywhere.

Getoor, L., Friedman, N., Koller, D., and Taskar, B. (2002). Learning probabilistic models of link structure. Journal of Machine Learning Research, 3, 679–707.

He, X., Zha, H., Ding, C., and Simon, H. D. (2002). Web document clustering using hyperlink structures. Computational Statistics and Data Analysis, 41, 19–45.

Kanungo, T., Mount, D. M., and Netanyahu, N. S. (2002). An Efficient k-Means Clustering Algorithm: Analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), 881 – 892.

Long, B., Zhang, M., Yu, P. S., and Xu, T. (2008). Clustering on Complex Graphs, In Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence. 659-664, Chicago.

Maila, M., and Shi, J. (2001). A random walks view of spectral segmentation. AI and STATISTICS.

Neville, J., Adler, M., and Jensen, D. (2003). Clustering relational data using attribute and link information. Proceedings of the Text Mining and Link Analysis Workshop, Eighteenth International Joint Conference on Artificial Intelligence, 9-15.

Ng, A., Jordan, M., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm, Advances in Neural Information Processing Systems, 14.

Shi, J., and Malik, J. (2000). Normalized Cuts and Image Segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8), 888-905.

Wagstaff, K., and Cardie, C. (2000). Clustering with instance-level constraints. In Proceedings of the Seventeenth International Conference on Machine Learning, 1103–1110.

Yu, S. X., and Shi, J. (2004). Segmentation Given Partial Grouping Constraints. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(2), 173-183.