Integrating Vertex-centric Clustering with Edge-centric Clustering for Meta Path Graph Analysis

Yang Zhou
Georgia Institute of Technology
Atlanta, GA 30332
yzhou@gatech.edu

Ling Liu
Georgia Institute of Technology
Atlanta, GA 30332
lingliu@cc.gatech.edu

David Buttler
Lawrence Livermore National Laboratory
Livermore, CA 94550
davidbuttler@llnl.gov

ABSTRACT

Meta paths are good mechanisms to improve the quality of graph analysis on heterogeneous information networks. This paper presents a meta path graph clustering framework, VEPathCluster, that combines meta path vertex-centric clustering with meta path edge-centric clustering for improving the clustering quality of heterogeneous networks. First, we propose an edge-centric path graph model to capture the meta-path dependencies between pairwise path edges. We model a heterogeneous network containing M types of meta paths as M vertex-centric path graphs and M edge-centric path graphs. Second, we propose a clustering-based multigraph model to capture the fine-grained clustering-based relationships between pairwise vertices and between pairwise path edges. We perform clustering analysis on both a unified vertex-centric path graph and each edge-centric path graph to generate vertex clustering and edge clusterings of the original heterogeneous network respectively. Third, a reinforcement algorithm is provided to tightly integrate vertex-centric clustering and edge-centric clustering by mutually enhancing each other. Finally, an iterative learning strategy is presented to dynamically refine both vertex-centric clustering and edge-centric clustering by continuously learning the contributions and adjusting the weights of different path graphs.

Categories and Subject Descriptors

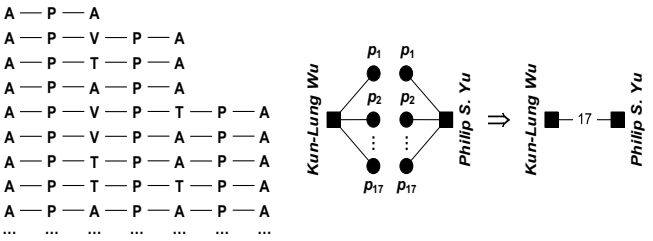
H.2.8 [**Database Applications**]: Data Mining

Keywords

Meta Path Graph Clustering; Vertex/Edge-centric Path Graph/Multigraph; Edge-centric Random Walk; Vertex/Edge-centric Clustering

1. INTRODUCTION

Heterogeneous information networks are graphs with heterogeneous types of entities and links. A meta path is a path connecting multiple types of entities through a sequence of heterogeneous meta links, representing different kinds of semantic relations among different types of entities. DBLP dataset has four types of entities: authors (A), publishing venues (V), papers (P) and paper terms (T). Figure 1 (a) gives nine example meta paths between authors in the DBLP dataset, each is composed of three types of meta links: A-P, V-P and T-P, representing different types of relationships between authors. More meta paths between authors can be generated through link combination and propagation. The meta path A-P-A captures the coauthor relationship, whereas the path A-P-V-P-A represents the relationship between a pair of authors through their papers published on the common venues. For each type of meta paths, we can construct a vertex-centric path graph to capture an individual type of relationships between authors. For example, Figure 1 (b) shows that we join one type of links (A-P) and its opposite form (P-A) to generate a vertex-centric A-P-A path graph, where vertices represent authors and edges denote the coauthor relationships between authors. For each pair of coauthors, say *Kun-Lung Wu* and *Philip S. Yu*, we can represent the A-P-A path by using parallel edges, each representing one of their coauthored papers ($p_1, \cdots, p_{17}$). By join composition, we obtain the total number of their coauthored papers (17). Clearly, mining heterogeneous information networks through multiple path graphs can provide new insights about how ideas and opinions on different subjects propagate differently among the same set of people.



(a) Meta Paths between Authors (b) Example Vertex-centric Path Graph

Figure 1: Example Meta Paths and Path Graphs from DBLP

Meta path-based social network analysis is gaining attention in recent years [1–6]. Existing efforts utilize a selection of meta paths between the same type of entities to improve the quality of similarity search, classification, clustering, link prediction and citation recommendation in heterogeneous networks. However, none of the existing methods have addressed all of the following challenges.

- **Vertex-centric clustering w.r.t. multiple path graphs.** As shown in Figure 1, different meta paths exhibit different semantic meanings about the same type of entities. Thus, the vertex clustering results based on different path graphs are typically not identical. It is critical to develop a unified clustering model that can efficiently integrate the clustering results from multiple path graphs and improve the overall clustering quality. Specifically, a dynamic weight assignment scheme should be employed to assign different weights to different path graphs to reflect their possibly different contributions towards the clustering convergence.
- **Fine-grained vertex assignment and clustering objective.** Meta-path graph analysis differentiates the semantics carried by different meta paths in a heterogeneous network. Consequently, it demands fine-grained vertex assignment and clustering objective to further improve the clustering quality. However, existing partitioning clustering approaches, such as K-Means and K-

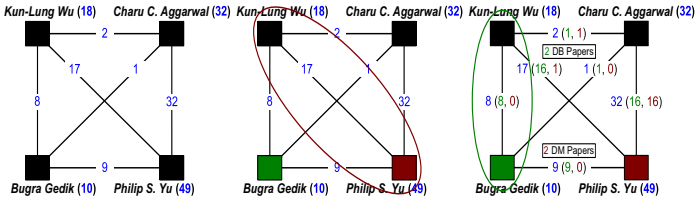
KDD'15, August 10-13, 2015, Sydney, NSW, Australia.
© 2015 ACM. ISBN 978-1-4503-3664-2/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2783258.2783328.

Medoids [7], usually assign each vertex to its closest center. We argue that this kind of vertex assignment may not always produce an accurate clustering result. Consider Figure 2 (a), by performing K-Means on the A-P-A path graph to assign *Kun-Lung Wu* to two centers of *Bugra Gedik* and *Philip S. Yu*, Figure 2 (b) shows a vertex assignment, i.e., by simply using the coarse path edge weight (the total number of coauthored papers) to measure vertex closeness, *Kun-Lung Wu* and *Philip S. Yu* are closer than *Kun-Lung Wu* and *Bugra Gedik*. However, in reality, *Kun-Lung Wu* and *Bugra Gedik* are known as database researchers with no or very few data mining papers but *Philip S. Yu* is a well-known expert on data mining with much more data mining papers than database publications, thus the vertex assignment in Figure 2 (c) is more accurate and better quality. This is because the similarity measures used in vertex assignment and clustering objective of existing methods are too coarse to reflect the above ground truth.

- **Edge-centric clustering w.r.t. multiple path graphs.** Conventional graph clustering models are usually based on the existence of vertex homophily. However, we argue that vertex homophily without edge clustering is insufficient for meta-path graph analysis on heterogeneous networks. Consider Figures 2 (b) and (c) again, there is only one of 17 coauthored papers between *Kun-Lung Wu* and *Philip S. Yu* published on DM conference (*KDD*) but all 8 coauthored papers between *Kun-Lung Wu* and *Bugra Gedik* are published on DB conferences, indicating that *Kun-Lung Wu*, *Bugra Gedik* and the path edge between them belong to cluster *DB* with very high probability. In comparison, it is highly probable that *Philip S. Yu* and the path edge between *Kun-Lung Wu* and *Philip S. Yu* belong to different clusters. Without considering edge clustering, the vertex homophily alone can lead to inaccurate vertex clustering.
- **Integrating vertex-centric clustering and edge-centric clustering.** Vertex clustering and edge clustering on heterogeneous networks may have individual clustering goals and due to the different semantic relationships implied by different meta paths. Relying on either of them alone may result in incomplete and possibly inaccurate clustering results. However, none of existing methods study how to effectively combine the above two techniques into a unified meta path graph clustering model.

To address the above challenges, we develop an efficient vertex/edge-centric meta path graph clustering approach, **VEPathCluster**, with four original contributions.

- We model a heterogeneous network containing multiple types of meta paths in terms of multiple vertex-centric path graphs and multiple edge-centric path graphs. Each meta path corresponds to one vertex-centric path graph and one edge-centric path graph.
- We propose a clustering-based multigraph model to capture the fine-grained clustering-based relationships between pairwise vertices and between pairwise path edges about given K clusters.
- We integrate multiple types of vertex-centric path graphs with different semantics into a unified vertex-centric path graph in terms of their contributions towards the clustering objective. We cluster both the unified vertex-centric path graph and each edge-centric path graph to generate vertex clustering and edge clusterings of the original heterogeneous network respectively.
- We design a reinforcement algorithm to tightly integrate vertex-centric clustering and edge-centric clustering by mutually enhancing each other: (1) good vertex-centric clustering promotes good edge-centric clustering and (2) good edge-centric clustering elevates good vertex-centric clustering. We devise an iterative learning method to dynamically refine both vertex-centric clustering and edge-centric clustering by continuously learning the contributions and adjusting the weights of different path graphs.



(a) A-P-A Vertex Graph (b) Clustering 1 (c) Clustering 2

Figure 2: Coarse Vertex Assignment/Clustering Objective

- Empirical evaluation over real datasets demonstrates the competitiveness of VEPathCluster against the state-of-the-art methods.

2. PROBLEM DEFINITION

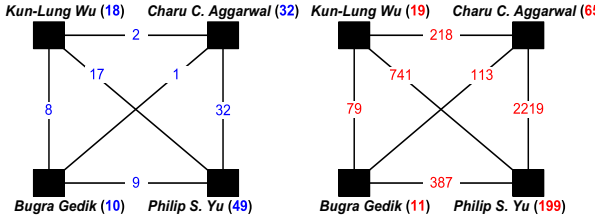
We define the problem of vertex/edge-centric meta path graph clustering in terms of the following four concepts.

A *heterogeneous information network* is denoted as $G = (V, E)$, where V is the set of heterogeneous entity vertices in G, consisting of s types of entity vertices, i.e., $V = \bigcup_{i=1}^{s} V_i$, each V_i $(1 \le i \le s)$ represents the i^{th} types of entity vertices. E is the set of heterogeneous meta links denoting the relationships between entity vertices in V. Due to heterogeneous entity vertices with s types, E can be divided into $s \times s$ subsets E_{ij} $(1 \le i, j \le s)$ such that $E = \bigcup_{i=1,j=1}^{s} E_{ij}$, where E_{ij} is the set of meta links connecting vertices of the i^{th} type (V_i) to vertices of the j^{th} type (V_j). E_{ji} is the opposite form of E_{ij}, specifying the set of meta links from V_j to V_i.

The m^{th} *meta path* of length l, denoted by $MP_m =< E_{a_0a_1}, E_{a_1a_2}, \cdots, E_{a_{l-1}a_l} >$, is a sequence of different types of meta links, with source vertex type V_{a_0} and destination vertex type V_{a_l} $(1 \le a_0, a_1, \cdots, a_l \le s)$, such that $< E_{a_0a_1}, E_{a_1a_2}, \cdots, E_{a_{l-1}a_l} >$ are l meta link types connected through join composition. For example, meta path A-P-A is of length 2 and comprises two meta link types: A-P and P-A.

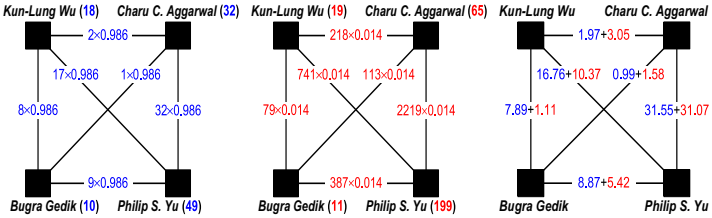
For each meta path in G, we construct a vertex-centric path graph to capture the meta-path based relationships between vertices. Formally, a *vertex-centric path graph* for MP_m is denoted as $VG_m = (V_{a_0}, V_{a_l}, E_m)$, where $V_{a_0} \in V$ is the set of source vertices and $V_{a_l} \in V$ is the set of destination vertices in MP_m, and $E_m \in E$ is the set of path edges between V_{a_0} and V_{a_l}. For the path edge set E_m, we compute its adjacency matrix $\mathbf{P}_m$ by multiplying adjacency matrix of each type of composite meta links $E_{a_0a_1}, E_{a_1a_2}, \cdots, E_{a_{l-1}a_l}$, denoted by $\mathbf{W}_{a_0a_1}, \mathbf{W}_{a_1a_2}, \cdots, \mathbf{W}_{a_{l-1}a_l}$ respectively. For Figure 1 (b), we use $\mathbf{W}_{AP}$ and $\mathbf{W}_{PA}$ to denote the adjacency matrices of two types of meta links A-P and P-A respectively. We calculate an adjacency matrix $\mathbf{P}_{AA} = \mathbf{W}_{AP} \times \mathbf{W}_{PA}$ to obtain the path edge between *Kun-Lung Wu* and *Philip S. Yu* with a value of 17. For presentation brevity, when the type of source vertices is the same as the type of destination vertices in VG_m, i.e., $V_{a_0} = V_{a_l} = V_c \in V$, we simplify $VG_m = (V_{a_0}, V_{a_l}, E_m)$ as $VG_m = (V_c, E_m)$, and path edges in E_m measure the pairwise closeness between vertices in V_c. We denote the size of V_c as $N_{V_c} = |V_c|$ and denote the size of E_m as $N_{E_m} = |E_m|$.

In VEPathCluster, for a specific clustering task, users can select a subset of entity vertices of a certain type as the *set of target vertices*, denoted by V_c, and a subset of M *target meta paths* MP_m. We construct M *vertex-centric path graphs* VG_m. The problem of **Vertex/Edge-centric meta Path graph Clustering** (VEPathCluster) is to simultaneously perform two clustering tasks: (1) assign all entity vertices in V_c to K soft clusters with an $N_{V_c} \times K$ clustering membership matrix $\mathbf{X}$ with each row summing to 1, and (2) cluster all path edges in each E_m $(1 \le m \le M)$ into K soft clusters with an $N_{E_m} \times K$ clustering membership matrix $\mathbf{Y}_m$ with each row summing to 1. The desired clustering result should achieve the two goals: (1) both path edges and their associated vertices should belong to the same clusters, and vertices within each cluster are close to each other in terms of path edges between them in the same cluster; and (2) vertices belonging to different clusters are relatively distant from each other in terms of clustered path edges between them.



(a) A-P-A Path Graph (b) A-P-V-P-A Path Graph

Figure 3: Vertex-centric Path Graph



(a) A-P-A Path Graph (b) A-P-V-P-A Path Graph (c) Unified Path Graph

Figure 4: Unified Vertex-centric Path Graph

Figure 3 gives an illustrative example of two vertex-centric path graphs about authors. For A-P-A path graph in Figure 3 (a), the number associated with an author vertex represents the number of coauthored papers by this author. Here, we only consider coauthored papers on three DB conferences: *SIGMOD*, *VLDB*, *ICDE* and three DM conferences: *KDD*, *ICDM*, *SDM*. For A-P-V-P-A meta path graph in Figure 3 (b), the number associated to an author, e.g., *Philip S. Yu* (199), represents the total number of papers published by this author on the above six venues. Similarly, the number on a path edge specifies the value of this path edge through link composition by multiplying adjacency matrices, e.g., $\mathbf{W}_{AP} \times \mathbf{W}_{PA}$ in Figure 3 (a), and $\mathbf{W}_{AP} \times \mathbf{W}_{PV} \times \mathbf{W}_{VP} \times \mathbf{W}_{PA}$) in Figure 3 (b).

3. THE VEPathCluster APPROACH

VEPathCluster improves the clustering quality by utilizing four novel mining strategies: (1) edge-centric random walk model; (2) clustering-based multigraph model; (3) integration of vertex-centric clustering and edge-centric clustering; and (4) dynamic weight learning. VEPathCluster iteratively performs the following three tasks to achieve high quality clustering: (1) fix edge clustering and weight assignment to update vertex clustering; (2) fix vertex clustering and weight assignment to update edge clustering; and (3) fix vertex clustering and edge clustering to update weight assignment.

3.1 Initialization

Given a heterogeneous network $G = (V, E)$, the set of target vertices $V_c \subset V$, and the M target meta paths, the number of clusters K, we first construct the M vertex-centric path graphs: $VG_1, \cdots, VG_M$. Then we initialize the weight assignment and produce the initial vertex clustering of V_c on K clusters.

Let $\omega_m^{(1)}$ $(1 \le m \le M)$ be the weight for the m^{th} vertex-centric path graph VG_m at the first iteration, and $\mathbf{P}_m$ be the adjacency matrix of VG_m. We use the initial weights $\omega_1^{(t)}, \cdots, \omega_M^{(t)}$ to integrate M vertex-centric path graphs into a unified vertex-centric path graph VG. The matrix form of VG, denoted by $\mathbf{P}^{(1)}$, is defined below.

$$\mathbf{P}^{(1)} = \omega_1^{(1)}\mathbf{P}_1 + \cdots + \omega_M^{(1)}\mathbf{P}_M \ s.t. \ \sum_{m=1}^{M} \omega_m^{(1)} = 1, \ \omega_1^{(1)}, \cdots, \omega_M^{(1)} \geqslant 0 \quad (1)$$

Random weight assignment often performs poorly and results in incorrect clustering results due to the sharp difference in edge values from path graph to path graph, e.g., the edge values in Figure 3 (a) are between 1 and 32 but the edge values in Figure 3 (b) are between 79 and 2219. We normalize edge values in each VG_m by assigning an initial weight for each VG_m in terms of its maximal edge value, i.e., $\omega_1^{(1)} = \frac{1/\max \mathbf{P}_1}{\sum_{m=1}^{M} 1/\max \mathbf{P}_m}, \ldots, \omega_M^{(1)} = \frac{1/\max \mathbf{P}_M}{\sum_{m=1}^{M} 1/\max \mathbf{P}_m}$, where $\max \mathbf{P}_m$ represents the maximal element in $\mathbf{P}_m$.

For two path graphs in Figure 3, we multiply the edge values by the initial weights $\frac{1/32}{1/32+1/2219} = 0.986$ and $\frac{1/2219}{1/32+1/2219} = 0.014$ to generate two path graphs in Figures 4 (a) and (b). Figure 4 (c) shows the combination of them with the above initial weights.

Next we employ a soft clustering method, Fuzzy C-Means (FCM) [8], on the unified vertex-centric path graph VG, to cluster each vertex to K clusters such that it has up to K membership probabilities. We use symbol $\mathbf{X}_k^{(1)}(i)$ to represent the membership probability of a vertex $v_i \in V_c$ $(1 \le i \le N_{V_c})$ belonging to cluster c_k $(1 \le k \le K)$ at the first iteration. Figure 6 (a) exhibits the FCM clustering result of author vertices in Figure 4 (c), where each green number and ochre number in the bracket denotes the membership probability of an author belonging to cluster *DB* or *DM* respectively.

3.2 Edge-centric Random Walk Model

Edge-centric random walk model is constructed by performing two tasks: (1) for each vertex-centric path graph, construct an edge-centric path graph and define its vertex values and edge values; and (2) define the transition probability on the edge-centric path graph.

Let VG_m be a vertex-centric path graph corresponding to the m^{th} meta path MP_m. We build an edge-centric path graph EG_m by converting the edges and vertices of VG_m to the vertices and edges of EG_m respectively. For example, we first transform the vertex-centric graph in Figure 5 (a) into a vertex/edge bipartite graph in Figure 5 (b) where rectangle vertices and circle vertices correspond to the vertices and the edges in Figure 5 (a). The circle vertex (W, Y) (17) in Figure 5 (b) corresponds to the edge between *Kun-Lung* ***Wu*** and *Philip* ***Yu*** with weight of 17 in Figure 5 (a).

Next we convert the bipartite graph in Figure 5 (b) to the edge-centric graph in Figure 5 (c) by shrinking each common rectangle vertex shared by any pair of circle vertices to an edge between these two circle vertices, and assign the edge value with the value of the common rectangle vertex in Figure 5 (b). For instance, a common rectangle vertex W (18) shared by two circle vertices (W, Y) (17) and (W, G) (8) in Figure 5 (b) is converted to the edge between (W, Y) (17) and (W, G) (8) in Figure 5 (c). In addition, to capture the fact that a circle vertex connects to two rectangle vertices in Figure 5 (b), we build a spin edge for each circle vertex in Figure 5 (c). The value of this spin edge is the sum of the values of two rectangle vertices linked to this circle vertex in the bipartite graph.

We define the transition probability on EG_m such that the edge-centric random walk model can be employed to measure the closeness between a pair of edge vertices in EG_m.

Definition 1. [Transition Probability on Edge-centric Path Graph] Let $VG_m = (V_c, E_m)$ be a *vertex-centric path graph* where V_c is the set of target vertices, E is the set of path edges between vertices in V_c and $EG_m = (E_m, E_m \times E_m)$ is a corresponding *edge-centric path graph*. The transition probability on EG_m is defined below.

$$\mathbf{T}_m(e_{mi}, e_{mj}) = \begin{cases} \dfrac{\mathbf{Q}_m(e_{mi}, e_{mj})}{\sum_{l=1}^{N_{E_m}} \mathbf{Q}_m(e_{ml}, e_{mj})}, (e_{mi}, e_{mj}) \in E_m \times E_m, \\ 0, \qquad\qquad otherwise. \end{cases}, \ 1 \le m \le M \quad (2)$$

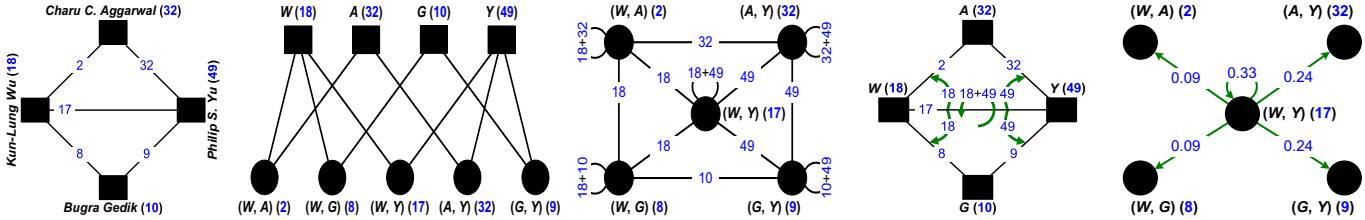
where $\mathbf{Q}_m$ is the adjacency matrix of EG_m and $\mathbf{T}_m(e_{mi}, e_{mj})$ represents the transition probability from vertex e_{mi} to vertex e_{mj} in EG_m.

Consider Figure 5, we compute the transition probabilities from (W, Y) to all five circle vertices: Given that $\sum_{l=1}^{N_{E_m}} \mathbf{Q}_m(e_{ml}, e_{mj}) = (18 + 49) + 18 + 18 + 49 + 49 = 201$, the transition probability from (W, Y) to (W, G) is $18/201 = 0.09$.

We express the above transition probability in a matrix form.

$$\mathbf{T}_m = \mathbf{Q}_m\mathbf{D}^{-1}, \ 1 \le m \le M \quad (3)$$

where $\mathbf{D}$ is a diagonal matrix $\mathbf{D} = \text{diag}(d_1, \cdots, d_{N_{E_m}})$ and $d_j = \sum_{l=1}^{N_{E_m}} \mathbf{Q}_m(e_{ml}, e_{mj})$ $(1 \le j \le N_{E_m})$.



(a) Vertex-centric Graph (b) Vertex/Edge Bipartite Graph (c) Edge-centric Graph (d) Transition between Edges (e) Transition Probability

Figure 5: Random Walk on Edges

3.3 Clustering-based Multigraph Model

The second novelty is to perform clustering analysis on vertex-centric multigraph and edge-centric multigraph to effectively combine vertex homophily with edge homophily. Recall Figure 2 (b), assigning *Kun-Lung Wu* to *Philip S. Yu* is due to the using of aggregated edge weight (i.e., the total number of coauthored papers) to measure the vertex closeness. We address this problem by introducing two clustering-based multigraph models, one for vertex-centric path graphs and another for edge-centric path graphs.

Given that a vertex-centric path graph $VG_m = (V_c, E_m)$, and the clustering result on the corresponding edge-centric path graph $EG_m = (E_m, E_m \times E_m)$ obtained at the previous iteration. A *vertex-centric path multigraph* i.e., $\mathbf{Y}_m^{(t-1)}$, denoted as $VMG_m = (V_c, F_m)$, is an edge augmented multigraph, where F_m is the set of edges satisfying the following condition: for each edge $(v_i, v_j) \in E_m$ in VG_m, we create a set of parallel edges between v_i and v_j in F_m. Each set of edges has up to K clustered edges and each of the parallel edges corresponds to a certain cluster c_k. The value of the parallel edge with label c_k between v_i and v_j in VMG_m at the t^{th} iteration, denoted by $\mathbf{P}_{mk}^{(t)}(v_i, v_j)$, are computed as follow.

$$\mathbf{P}_{mk}^{(t)}(v_i, v_j) = \mathbf{P}_m(v_i, v_j) \times \mathbf{Y}_{mk}^{(t-1)}((v_i, v_j)),\ 1 \le m \le M,\ 1 \le k \le K \quad (4)$$

where $\mathbf{P}_m(v_i, v_j)$ represents the value of the edge between v_i and v_j in VG_m. $\mathbf{Y}_{mk}^{(t-1)}$ denotes the k^{th} column vector of the edge clustering membership matrix $\mathbf{Y}_m^{(t-1)}$ and $\mathbf{Y}_{mk}^{(t-1)}((v_i, v_j))$ specifies the membership probability of vertex (v_i, v_j) belonging to cluster c_k in EG_m at the last iteration. $\mathbf{P}_{mk}^{(t)}$ is essentially a projection of $\mathbf{P}_m$ on c_k.

Similarly, let $\mathbf{X}^{(t)}$ $(t \ge 1)$ be the soft clustering result on the unified vertex-centric path multigraph VMG at the current iteration. For each edge-centric path graph $EG_m = (E_m, E_m \times E_m)$, we create an *edge-centric path multigraph* EMG_m: for each edge $(e_{mi}, e_{mj}) \in E_m \times E_m$, we create a set of up to K parallel edges. Each of parallel edges corresponds to cluster c_k. The edge values on EMG_m at the t^{th} iteration are defined as follow.

$$\mathbf{Q}_{mk}^{(t)}(e_{mi}, e_{mj}) = \begin{cases} \mathbf{Q}_m(e_{mi}, e_{mj}) \times \mathbf{X}_k^{(t)}(e_{mi} \wedge e_{mj}), & e_{mi} \ne e_{mj}, \\ \mathbf{R}_m(v_a) \times \mathbf{X}_k^{(t)}(v_a) + \mathbf{R}_m(v_b) \times \mathbf{X}_k^{(t)}(v_b), & e_{mi} = e_{mj}. \end{cases} \quad (5)$$
$$1 \le m \le M,\ 1 \le k \le K$$

where $\mathbf{Q}_m(e_{mi}, e_{mj})$ specifies the edge value between two vertices e_{mi} and e_{mj} in EG_m, $\mathbf{X}_k^{(t)}$ denotes the k^{th} column vector of the vertex clustering membership matrix $\mathbf{X}^{(t)}$ and $\mathbf{X}_k^{(t)}((e_{mi} \wedge e_{mj}))$ specifies the membership probability of common vertex of two edges e_{mi} and e_{mj} belonging to cluster c_k in the unified vertex-centric path graph VG at the t^{th} iteration. $\mathbf{Q}_{mk}^{(t)}$ is essentially a projection of $\mathbf{Q}_m$ on cluster c_k. When $e_{mi} = e_{mj}$, edge (e_{mi}, e_{mj}) is a spin edge associated to e_{mi} in EG_m. In this situation, e_{mi} and e_{mj} correspond to the same edge in VG_m, and e_{mi} and e_{mj} will have the same two endpoints (v_a and v_b) in VG_m, e.g., the spin edge $((W, Y), (W, Y))$ in Figure 5 (b) and the edge between *Kun-Lung Wu* and *Philip S. Yu* in Figure 5 (a). $\mathbf{R}_m(v_x)$ represents the value of endpoint v_x in VG_m, say 18 for *Kun-Lung Wu* in Figure 5 (a), and $\mathbf{X}_k^{(t)}(v_x)$ denotes the probability of v_x belonging to c_k in VG or VMG at the t^{th} iteration.

For ease of presentation, we omit all spin edges in Figure 6. Based on the A-P-A edge-centric path graph in Figure 6 (b) and its vertex soft clustering result in Figure 6 (a), we generate the A-P-A edge-centric path multigraph in Figure 6 (c). Using the probabilities of *Kun-Lung Wu* on clusters *DB* and *DM*: (0.96, 0.04) in Figure 6 (a) and the edge between (W, Y) and (W, A) in Figure 6 (b), we produce two parallel edges between (W, Y) and (W, A) in Figure 6 (c) as $18 \times 0.96 = 17.28$ and $18 \times 0.04 = 0.72$ respectively.

3.4 Edge-centric Clustering

We perform edge-centric soft clustering in two steps: (1) convert each edge-centric path graph EG_m to an edge-centric path multigraph EMG_m based on the vertex soft clustering $\mathbf{X}^{(1)}$ on the unified vertex-centric path graph VG or $\mathbf{X}^{(t)}$ $(t > 1)$ on the unified vertex-centric path multigraph VMG; and (2) compute the edge soft clustering $\mathbf{Y}_m^{(t)}$ on each edge-centric path multigraph EMG_m.

Different from traditional unsupervised graph clustering methods, at the first clustering iteration, we adopt a semi-supervised manner on each EG_m with the geometric mean of the probabilities of two endpoints belonging to cluster c_k as the initial membership probability of an edge on c_k. This is motivated by the observation that if the membership probabilities of two associated endpoints of an edge belonging to c_k are very large, then it is highly probable that this edge also has a large probability on c_k.

Formally, we convert each EG_m to an EMG_m by converting the adjacency matrix of EG_m to up to K independent adjacency matrices in terms of the cluster labels of the edges in EG_m, and then learns the cluster probabilities of edge vertices in EG_m on c_k based on the k^{th} adjacency matrix. Let (v_i, v_j) be an edge vertex in EG_m where v_i and v_j are the target vertices in the corresponding $VG_m = (V_c, E_m)$, and $\mathbf{X}_k^{(1)}(v_x)$ be the cluster membership probability of $v_x \in V_c$ belonging to cluster c_k at the first iteration. We define the initial edge clustering membership matrix $\mathbf{Y}_m^{(0)}$ for EMG_m below.

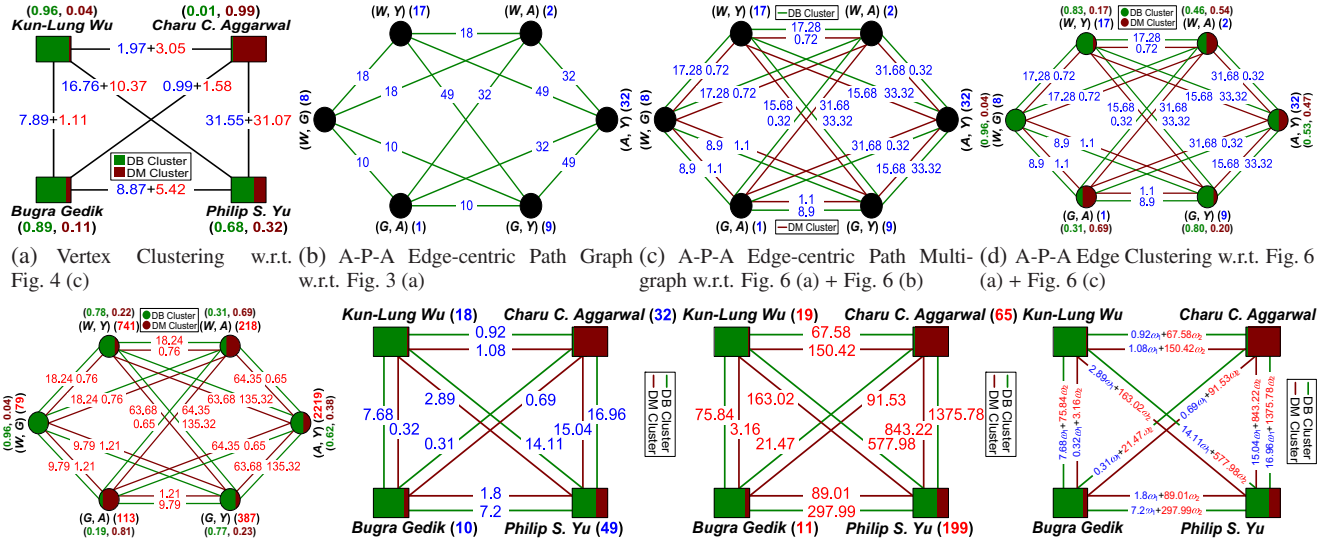
$$\mathbf{Y}_{mk}^{(0)}((v_i, v_j)) = \frac{\sqrt{\mathbf{X}_k^{(1)}(v_i) \times \mathbf{X}_k^{(1)}(v_j)}}{\sum_{l=1}^{K} \sqrt{\mathbf{X}_l^{(1)}(v_i) \times \mathbf{X}_l^{(1)}(v_j)}},\ 1 \le m \le M,\ 1 \le k \le K \quad (6)$$

where $\mathbf{Y}_{mk}^{(0)}$ is the k^{th} column vector of $\mathbf{Y}_m^{(0)}$, $\mathbf{Y}_{mk}^{(0)}((v_i, v_j))$ represents the initial membership probability of edge vertex (v_i, v_j) on c_k in EMG_m, and $\mathbf{X}_k^{(1)}(v_x)$ specifies the probability of v_x on c_k in VG.

Based on the initial vertex clustering membership matrix $\mathbf{X}^{(1)}$ for VG or the vertex clustering membership matrix $\mathbf{X}^{(t)}$ $(t > 1)$ for VMG, we transform each EG_m into an edge-centric path multigraph EMG_m by Eq. (5). In the first clustering iteration, we update $\mathbf{Y}_m^{(1)}$ with $\mathbf{Y}_m^{(0)}$ based on $\mathbf{X}^{(1)}$ for VG through label propagation and update $\mathbf{Y}_m^{(t)}$ with $\mathbf{Y}_m^{(t-1)}$ in each subsequent iteration t $(t > 1)$.

Similar to Eq.(2), the transition probability on each EMG_m at the current iteration is defined by normalizing each kind of parallel edges with the same cluster labels in EMG_m as follow.

$$\mathbf{T}_{mk}^{(t)}(e_{mi}, e_{mj}) = \begin{cases} \dfrac{\mathbf{Q}_{mk}^{(t)}(e_{mi}, e_{mj})}{\sum_{l=1}^{N_{E_m}} \mathbf{Q}_{mk}^{(t)}(e_{ml}, e_{mj})}, & \mathbf{Q}_{mk}^{(t)}(e_{ml}, e_{mj}) \ne 0, \\ 0, & otherwise. \end{cases} \quad (7)$$
$$1 \le m \le M,\ 1 \le k \le K$$



(a) Vertex Clustering w.r.t. Fig. 4 (c)

(b) A-P-A Edge-centric Path Graph w.r.t. Fig. 3 (a)

(c) A-P-A Edge-centric Path Multigraph w.r.t. Fig. 6 (a) + Fig. 6 (b)

(d) A-P-A Edge Clustering w.r.t. Fig. 6 (a) + Fig. 6 (c)

(e) A-P-V-P-A Edge Clustering w.r.t. Fig. 3 (b) + Fig. 6 (a)

(f) A-P-A Vertex-centric Path Multigraph w.r.t. Fig. 3 (a) + Fig. 6 (d)

(g) A-P-V-P-A Vertex-centric Path Multigraph w.r.t. Fig. 3 (b) + Fig. 6 (e)

(h) Unified Vertex-centric Path Multigraph w.r.t. Fig. 6 (f) + Fig. 6 (g)

Figure 6: Iterative Vertex Clustering on Vertex-centric Path Multigraph and Edge Clustering on Edge-centric Path Multigraph

where $\mathbf{T}^{(t)}_{mk}(e_{mi}, e_{mj})$ denotes the transition probability with cluster label c_k on one of parallel edges between edge vertices e_{mi} and e_{mj} in EMG_m. The transition matrix on EMG_m is given below.

$$\mathbf{T}^{(t)}_{mk} = \mathbf{Q}^{(t)}_{mk}(\mathbf{D}^{-1}_{mk})^{(t)},\ 1 \le m \le M,\ 1 \le k \le K \tag{8}$$

where $(\mathbf{D}^{-1}_{mk})^{(t)}$ is a diagonal matrix $(\mathbf{D}^{-1}_{mk})^{(t)} = \mathrm{diag}(d_1, \cdots, d_{N_{E_m}})$, and $d_j = \sum_{l=1}^{N_{E_m}} \mathbf{Q}^{(t)}_{mk}(e_{ml}, e_{mj})$ $(1 \le j \le N_{E_m})$.

Thus, we produce K edge clustering kernels $\mathbf{T}^{(t)}_{mk}$, each corresponding to cluster c_k $(1 \le k \le K)$. The transition operation in each edge-centric path multigraph is divided into two steps: (1) choose those parallel edges with the objective cluster label by clustering objective; and (2) select an edge with the largest probability from the above edges to jump.

Let $\mathbf{Y}_m = [\mathbf{Y}_{m1}, \mathbf{Y}_{m2}, \cdots, \mathbf{Y}_{mK}] \in \mathbb{R}^{N_{E_m} \times K}$ be the edge clustering membership matrix for E_m in EMG_m $(1 \le m \le M)$. For each edge clustering membership vector $\mathbf{Y}_{mk}$ $(1 \le k \le K)$ based on cluster c_k, we use an individual clustering kernel $\mathbf{T}^{(t)}_{mk}$ to iteratively infer the membership probabilities of all edge vertices in E_m on c_k.

$$\begin{aligned} &Initilization:\ \mathbf{Y}_{mk} = \mathbf{Y}^{(t-1)}_{mk} \\ &Iteration:\ \mathbf{Y}_{mk} = \mathbf{T}^{(t)}_{mk}\mathbf{Y}_{mk} \end{aligned} \tag{9}$$

Based on the edge clustering membership matrix $\mathbf{Y}^{(t-1)}_{mk}$ at the last clustering round, VEPathCluster iteratively infers the membership probabilities of vertices in E_m until $\mathbf{Y}_{mk}$ converges. We then normalize each entry $\mathbf{Y}_{mk}(e_{mi})$ $(1 \le i \le N_{E_m})$ in $\mathbf{Y}_{mk}$ as follow.

$$\mathbf{Y}^{(t)}_{mk}(e_{mi}) = \frac{\mathbf{Y}_{mk}(e_{mi})}{\sum_{l=1}^{K} \mathbf{Y}_{ml}(e_{mi})} \tag{10}$$

where $e_{mi} \in E_m$ represents an edge vertex in EMG_m and $\mathbf{Y}^{(t)}_{mk}$ specifies the normalized edge clustering membership vector based on c_k. Thus, the edge clustering membership matrix is updated below.

$$\mathbf{Y}^{(t)}_m = \begin{bmatrix} \mathbf{Y}^{(t)}_{m1} & \mathbf{Y}^{(t)}_{m2} & \cdots & \mathbf{Y}^{(t)}_{mK} \end{bmatrix},\ 1 \le m \le M \tag{11}$$

For example, based on the vertex clustering in Figure 6 (a) and the edge-centric path multigraph in Figure 6 (c), we produce the A-P-A edge clustering in Figure 6 (d).

3.5 Vertex-centric Clustering

The vertex clustering on the unified vertex-centric path multigraph VMG follows the heuristic rule: if vertex $v_i \in V_c$ in each vertex-centric path graph VG_m has many neighbors with large probabilities on cluster c_k and the edges between v_i and these neighbors have large probabilities on c_k, then it is highly probable that v_i belongs to c_k with a larger probability. In each iteration, we use the edge clustering result on each edge-centric path graph EG_m at the previous iteration $(\mathbf{Y}^{(t-1)}_m)$ to perform the vertex clustering on VMG at the current iteration $(\mathbf{X}^{(t)})$ in three steps.

(1) Based on $\mathbf{Y}^{(t-1)}_m$ and Eq.(4), we first convert each VG_m to an vertex-centric path multigraph VMG_m by transforming the adjacency matrix of VG_m into K independent adjacency matrices in terms of the cluster labels of parallel edges. For example, based on the edge clustering result on the edge-centric path multigraph in Figure 6 (d) (or Figure 6 (e)), we convert the vertex-centric path graph in Figure 3 (a) (or Figure 3 (b)) to the vertex-centric path multigraph in Figure 6 (f) (or Figure 6 (g)).

(2) We combining M vertex-centric path multigraphs VMG_m into the unified vertex-centric path multigraph VMG based on each of K edge clusters with weighting factors $\omega^{(t)}_1, \cdots, \omega^{(t)}_N$. A dynamic weight tuning mechanism will be detailed in Section 3.6. Thus, we compute the value of the unified parallel edge between vertices v_i and v_j in VMG about cluster c_k at the t^{th} iteration as follow.

$$\begin{aligned} &\mathbf{P}^{(t)}_k(v_i, v_j) = \omega^{(t)}_1\mathbf{P}^{(t)}_{1k}(v_i, v_j) + \cdots + \omega^{(t)}_M\mathbf{P}^{(t)}_{Mk}(v_i, v_j),\ 1 \le k \le K \\ &s.t.\ \sum_{m=1}^{M} \omega^{(t)}_m = 1,\ \omega^{(t)}_1, \cdots, \omega^{(t)}_M \geqslant 0 \end{aligned} \tag{12}$$

where $\omega^{(t)}_m$ $(1 \le m \le M)$ represents the weight for the m^{th} vertex-centric path multigraph VMG_m at the t^{th} iteration, and $\mathbf{P}^{(t)}_{mk}(v_i, v_j)$ specifies the value of the parallel edge with label c_k between v_i and v_j in VMG_m. Note that $\mathbf{P}^{(t)}_k(v_i, v_j)$ keeps changing with $\omega^{(t)}_1, \cdots, \omega^{(t)}_M$ through dynamic weight learning during each iteration.

The matrix form of VMG is defined based on K kinds of clustered parallel edges.

$$\begin{aligned} &\mathbf{P}^{(t)}_1 = \omega^{(t)}_1\mathbf{P}^{(t)}_{11} + \omega^{(t)}_2\mathbf{P}^{(t)}_{21} + \cdots + \omega^{(t)}_M\mathbf{P}^{(t)}_{M1} \\ &\cdots \\ &\mathbf{P}^{(t)}_K = \omega^{(t)}_1\mathbf{P}^{(t)}_{1K} + \omega^{(t)}_2\mathbf{P}^{(t)}_{2K} + \cdots + \omega^{(t)}_M\mathbf{P}^{(t)}_{MK} \\ &s.t.\ \sum_{m=1}^{M} \omega^{(t)}_m = 1,\ \omega^{(t)}_1, \cdots, \omega^{(t)}_M \geqslant 0 \end{aligned} \tag{13}$$

Figure 6 (h) shows the unified vertex-centric path multigraph by combining the two vertex-centric path multigraphs in Figure 6 (f) and (g) with the weights of ω_1 and ω_2 respectively such that the clustered path edges with the same labels between the same pair of vertices from two vertex-centric path multigraphs are combined.

(3) We compute the vertex clustering membership matrix $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_K] \in \mathbb{R}^{N_{V_c} \times K}$ for the target vertices V_c in VMG. We below define the transition probability on VMG in terms of each of K edge clusters.

$$\mathbf{S}_k^{(t)}(v_i, v_j) = \begin{cases} \dfrac{\mathbf{P}_k^{(t)}(v_i, v_j)}{\sum_{l=1}^{N_{V_c}} \mathbf{P}_k^{(t)}(v_l, v_j)}, \mathbf{P}_k^{(t)}(v_i, v_j) \neq 0, \\ 0, \qquad\qquad otherwise. \end{cases}, \ 1 \leq k \leq K \tag{14}$$

where $\mathbf{S}_k^{(t)}(v_i, v_j)$ denotes the transition probability with cluster label c_k on one of parallel edges between vertex v_i and vertex v_j in VMG.

The transition matrix on VMG is given as follow.

$$\mathbf{S}_k^{(t)} = \mathbf{P}_k^{(t)} (\mathbf{D}_k^{-1})^{(t)}, \ 1 \leq k \leq K \tag{15}$$

where $(\mathbf{D}_k^{-1})^{(t)}$ is a diagonal matrix $(\mathbf{D}_k^{-1})^{(t)} = \text{diag}(d_1, \cdots, d_{N_{V_c}})$, and $d_j = \sum_{l=1}^{N_{V_c}} \mathbf{P}_k^{(t)}(v_l, v_j)$ $(1 \leq j \leq N_{V_c})$.

Similar to edge-centric clustering, we produce K vertex clustering kernels $\mathbf{S}_k^{(t)}$, each corresponding to cluster c_k. The transition operation in the unified vertex-centric path multigraph VMG is divided into two steps: (1) choose those parallel edges with the objective cluster label; and (2) select an edge with the largest probability from the above edges to move.

For each vertex clustering membership vector $\mathbf{X}_k$ $(1 \leq k \leq K)$ based on c_k, we utilize an individual clustering kernel $\mathbf{S}_k^{(t)}$ to iteratively infer the membership probabilities of vertices in V_c on c_k.

$$\begin{aligned} &Initilization: \ \mathbf{X}_k = \mathbf{X}_k^{(t-1)} \\ &Iteration: \ \mathbf{X}_k = \mathbf{S}_k^{(t)} \mathbf{X}_k \end{aligned} \tag{16}$$

When the iterative vertex clustering converges, we further normalize each entry $\mathbf{X}_k(v_i)$ $(1 \leq i \leq N_{V_c})$ in $\mathbf{X}_k$ $(1 \leq k \leq K)$ below.

$$\mathbf{X}_k^{(t)}(v_i) = \frac{\mathbf{X}_k(v_i)}{\sum_{l=1}^{K} \mathbf{X}_l(v_i)} \tag{17}$$

where $v_i \in V_c$ denotes a target vertex in VMG and $\mathbf{X}_k^{(t)}$ represents the normalized vertex clustering membership vector based on c_k. Thus, the vertex clustering membership matrix is updated below.

$$\mathbf{X}^{(t)} = \begin{bmatrix} \mathbf{X}_1^{(t)} & \mathbf{X}_2^{(t)} & \cdots & \mathbf{X}_K^{(t)} \end{bmatrix} \tag{18}$$

$\mathbf{X}^{(t)}$ will be used to enter the next vertex clustering round.

3.6 Clustering with Weight Learning

The objective function of VEPathCluster is defined to maximize fuzzy intra-cluster similarity [22, 23] for both vertex clustering in the unified vertex-centric path multigraph VMG and edge clustering on each edge-centric path multigraph EMG_m.

Definition 2. [VEPathCluster Clustering Objective Function] Let VMG be a unified vertex-centric path multigraph, VMG_m $(m \in \{1, \cdots, M\})$ be M vertex-centric path multigraphs, EMG_m $(m \in \{1, \cdots, M\})$ be M edge-centric path multigraphs, $\omega_1, \cdots, \omega_M$ be the weighting factors for $VMG_1, \cdots, VMG_M$ and $EMG_1, \cdots, EMG_M$ defined in Eqs.(12) and (13) respectively, given K vertex soft clusters for VMG with a membership matrix $\mathbf{X}$ and K path edge soft clusters for each EMG_m with a membership matrix $\mathbf{Y}_m$, the goal of VEPathCluster is to maximize the following objective function.

$$\begin{aligned} O(\mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_M, \omega_1, \cdots, \omega_M) = &\sum_{i=1}^{N_{V_c}} \sum_{j=1}^{N_{V_c}} \sum_{k=1}^{K} \mathbf{X}_k(v_i) \mathbf{X}_k(v_j) \mathbf{P}_k(v_i, v_j) \\ &+ \sum_{m=1}^{M} \sum_{i=1}^{N_{E_m}} \sum_{j=1}^{N_{E_m}} \sum_{k=1}^{K} \mathbf{Y}_{mk}(e_{mi}) \mathbf{Y}_{mk}(e_{mj}) \mathbf{Q}_{mk}(e_{mi}, e_{mj}) \end{aligned}$$

$$\max_{\omega_1, \cdots, \omega_M} O(\mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_M, \omega_1, \cdots, \omega_M), \ s.t. \ \sum_{m=1}^{M} \omega_m = 1, \ \omega_1, \cdots, \omega_M \geqslant 0 \tag{19}$$

According to Eqs.(4)-(18), the objective function O is a fractional function of multi variables $\omega_1, \cdots, \omega_M$ with non-negative real coefficients. On the other hand, the numerator and the denominator of O are both polynomial functions of the above variables. Without loss of generality, we rewrite Eq.(19) as follow.

$$\max_{\omega_1, \cdots, \omega_M} O(\mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_M, \omega_1, \cdots, \omega_M) = \max_{\omega_1, \cdots, \omega_M} \frac{\sum_{i=1}^{p} a_i \prod_{j=1}^{M} (\omega_j)^{b_{ij}}}{\sum_{i=1}^{q} o_i \prod_{j=1}^{M} (\omega_j)^{r_{ij}}}$$

$$a_i, b_{ij}, o_i, r_{ij} \geq 0, b_{ij}, r_{ij} \in \mathbb{Z}, \ s.t. \ \sum_{m=1}^{M} \omega_m = 1, \ \omega_1, \cdots, \omega_M \geqslant 0 \tag{20}$$

where there are p polynomial terms in the numerator and q polynomial terms in the denominator, a_i and o_i are the coefficients of the i^{th} terms respectively, and b_{ij} and r_{ij} are the exponents of corresponding variables in the i^{th} terms respectively.

For ease of presentation, we revise the original objective as the following nonlinear fractional programming problem (NFPP).

Definition 3. [Nonlinear Fractional Programming Problem] Let $f(\omega_1, \cdots, \omega_M) = \sum_{i=1}^{p} a_i \prod_{j=1}^{M} (\omega_j)^{b_{ij}}$ and $g(\omega_1, \cdots, \omega_M) = \sum_{i=1}^{q} o_i \prod_{j=1}^{M} (\omega_j)^{r_{ij}}$, the clustering goal is revised as follow.

$$\max_{\omega_1, \cdots, \omega_M} \frac{f(\omega_1, \cdots, \omega_M)}{g(\omega_1, \cdots, \omega_M)}, \ s.t. \ \sum_{m=1}^{M} \omega_m = 1, \ \omega_1, \cdots, \omega_M \geqslant 0 \tag{21}$$

Our clustering objective is equivalent to maximize a quotient of two polynomial functions of multiple variables. It is very hard to perform function trend identification and estimation to determine the existence and uniqueness of solutions. Therefore, we want to transform this sophisticated NFPP into an easily solvable problem.

Definition 4. [Nonlinear Parametric Programming Problem] Let $f(\omega_1, \cdots, \omega_M) = \sum_{i=1}^{p} a_i \prod_{j=1}^{M} (\omega_j)^{b_{ij}}$ and $g(\omega_1, \cdots, \omega_M) = \sum_{i=1}^{q} o_i \prod_{j=1}^{M} (\omega_j)^{r_{ij}}$, the NPPP is defined as follow.

$$F(\gamma) = \max_{\omega_1, \cdots, \omega_M} f(\omega_1, \cdots, \omega_M) - \gamma g(\omega_1, \cdots, \omega_M), \ s.t. \ \sum_{m=1}^{M} \omega_m = 1, \ \omega_1, \cdots, \omega_M \geqslant 0 \tag{22}$$

Theorem 1. *The NFPP in Definition 3 is equivalent to the NPPP in Definition 4, i.e., $\gamma = \max_{\omega_1, \cdots, \omega_M} \frac{f(\omega_1, \cdots, \omega_M)}{g(\omega_1, \cdots, \omega_M)}$ if and only if $F(\gamma) = \max_{\omega_1, \cdots, \omega_M} f(\omega_1, \cdots, \omega_M) - \gamma g(\omega_1, \cdots, \omega_M) = 0$.*

Proof. If $(\overline{\omega}_1, \cdots, \overline{\omega}_M)$ is a feasible solution of $F(\gamma) = 0$, then $f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma g(\overline{\omega}_1, \cdots, \overline{\omega}_M) = 0$. Thus $f(\omega_1, \cdots, \omega_M) - \gamma g(\omega_1, \cdots, \omega_M) \leqslant f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma g(\overline{\omega}_1, \cdots, \overline{\omega}_M) = 0$. We have $\gamma = f(\overline{\omega}_1, \cdots, \overline{\omega}_M) / g(\overline{\omega}_1, \cdots, \overline{\omega}_M) \geqslant f(\omega_1, \cdots, \omega_M) / g(\omega_1, \cdots, \omega_M)$. Thus γ is a maximum value of NFPP and $(\overline{\omega}_1, \cdots, \overline{\omega}_M)$ is an optimal solution of NFPP.

Conversely, if $(\omega_1, \cdots, \overline{\omega}_M)$ solves NFPP, then we have $\gamma = f(\overline{\omega}_1, \cdots, \overline{\omega}_M) / g(\overline{\omega}_1, \cdots, \overline{\omega}_M) \geqslant f(\omega_1, \cdots, \omega_M) / g(\omega_1, \cdots, \omega_M)$. Thus $f(\omega_1, \cdots, \omega_M) - \gamma g(\omega_1, \cdots, \omega_M) \leqslant f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma g(\overline{\omega}_1, \cdots, \overline{\omega}_M) = 0$. We have $F(\gamma) = 0$ and the maximum is taken at $(\overline{\omega}_1, \cdots, \overline{\omega}_M)$.

Now the original NFPP has been successfully transformed into the straightforward NPPP. This transformation can efficiently speed up the clustering convergence due to the following properties.

Theorem 2. *$F(\gamma)$ is convex.*

Proof: Suppose that $(\overline{\omega}_1, \cdots, \overline{\omega}_M)$ is an optimum of $F((1-\lambda)\gamma_1 + \lambda\gamma_2)$ with $\gamma_1 \neq \gamma_2$ and $0 \leqslant \lambda \leqslant 1$. $F((1-\lambda)\gamma_1 + \lambda\gamma_2) = f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - ((1-\lambda)\gamma_1 + \lambda\gamma_2) g(\overline{\omega}_1, \cdots, \overline{\omega}_M) = \lambda(f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma_2 g(\overline{\omega}_1, \cdots, \overline{\omega}_M)) + (1-\lambda)(f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma_1 g(\overline{\omega}_1, \cdots, \overline{\omega}_M)) \leqslant \lambda \max_{\omega_1, \cdots, \omega_M} f(\omega_1, \cdots, \omega_M) - \gamma_2 g(\omega_1, \cdots, \omega_M) + (1-\lambda) \max_{\omega_1, \cdots, \omega_M} f(\omega_1, \cdots, \omega_M) - \gamma_1 g(\omega_1, \cdots, \omega_M) = \lambda F(\gamma_2) + (1-\lambda) F(\gamma_1)$. Thus, $F(\gamma)$ is convex.

Theorem 3. *$F(\gamma)$ is monotonically decreasing.*

Proof: Suppose that $\gamma_1 > \gamma_2$ and $(\overline{\omega}_1, \cdots, \overline{\omega}_M)$ is an optimal solution of $F(\gamma_1)$. Thus, $F(\gamma_1) = f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma_1 g(\overline{\omega}_1, \cdots, \overline{\omega}_M) < f(\overline{\omega}_1, \cdots, \overline{\omega}_M) - \gamma_2 g(\overline{\omega}_1, \cdots, \overline{\omega}_M) \leqslant \max_{\omega_1, \cdots, \omega_M} f(\omega_1, \cdots, \omega_M) - \gamma_2 g(\omega_1, \cdots, \omega_M) = F(\gamma_2)$.

Algorithm 1 Vertex/Edge-centric meta PATH graph Clustering

Input: M vertex-centric path graphs VG_m, M edge-centric path graphs EG_m, a clustering number K, and a parameter $\gamma^{(1)}$=0.
Output: vertex clustering membership matrix $\mathbf{X}$, M edge clustering membership matrices $\mathbf{Y}_1, \cdots, \mathbf{Y}_M$.

Initialize weights $\omega_1^{(1)}, \cdots, \omega_M^{(1)}$ in terms of the scales of edge values in each VG_m;
for t=1 **to** $F(\gamma^{(t)})$ converges to 0
  if t = 1
    Combine $\mathbf{P}_m$ of each VG_m into $\mathbf{P}^{(t)}$ of VG with Eq.(1);
    Invoke FCM to cluster vertices V_o in VG to generate $\mathbf{X}^{(t)}$ of VG;
  else
    Convert $\mathbf{P}_m$ of each VG_m into $\mathbf{P}_{mk}^{(t)}$ of each VMG_m with Eq.(4);
    Combine each VMG_m into VMG by computing all $\mathbf{P}_k^{(t)}$ in Eq.(13);
    Calculate $\mathbf{S}_k^{(t)}$ of VMG for each cluster c_k in Eqs.(14)-(15);
    Update $\mathbf{X}^{(t)}$ of VG with Eqs.(16)-(18);
  if t = 1
    Initialize $\mathbf{Y}_m^{(t-1)}$ of each EG_m with Eq.(6);
  Convert $\mathbf{Q}_m$ of each EG_m into $\mathbf{Q}_{mk}^{(t)}$ of each EMG_m with Eq.(5);
  Calculate $\mathbf{T}_{mk}^{(t)}$ of each EMG_m for each cluster c_k in Eqs.(7)-(8);
  Update $\mathbf{Y}_m^{(t)}$ of each EG_m with Eqs.(9)-(11);
  Compute $O(\mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_M, \omega_1, \cdots, \omega_M)$ in Eq.(19);
  Solve $F(\gamma^{(t)})$ in Eq.(22);
  Update $\omega_1^{(t+1)}, \cdots, \omega_M^{(t+1)}$;
  Refine $\gamma^{(t+1)}$=$f(\omega_1^{(t+1)}, \cdots, \omega_M^{(t+1)})/g(\omega_1^{(t+1)}, \cdots, \omega_M^{(t+1)})$;
Return $\mathbf{X}^{(t)}$ and $\mathbf{Y}_1^{(t)}, \cdots, \mathbf{Y}_M^{(t)}$.

THEOREM 4. *$F(\gamma) = 0$ has a unique solution.*

Proof: Based on the above-mentioned theorems, we know $F(\gamma)$ is continuous as well as decreasing. In addition, $lim_{\gamma \to +\infty} F(\gamma) = -\infty$ and $lim_{\gamma \to -\infty} F(\gamma) = +\infty$.

The procedure of solving this NPPP includes two parts: (1) find such a reasonable parameter γ ($F(\gamma) = 0$), making NPPP equivalent to NFPP; (2) given the parameter γ, solve a polynomial programming problem about the original variables $\omega_1, \cdots, \omega_M$. Our weight adjustment mechanism is an iterative procedure to find the solution of $F(\gamma) = 0$ and the corresponding weights after each clustering iteration. We first generate an initial matrix $\mathbf{P}^{(1)}$ with initial weights in terms of the scales of edge values in each vertex-centric path graph VG_m to produce an initial vertex clustering result through FCM [8] on the unified vertex-centric path graph VG. Based on the initial vertex clustering result, we construct an edge-centric path multigraph EMG_m for each edge-centric path graph EG_m. We then generate an initial edge clustering result on each EMG_m. According to the initial result of both vertex clustering and edge clusterings, we then calculate an initial $F(\gamma)$. Since $F(\gamma)$ is a monotonic decreasing function and $F(0) = \max_{\omega_1, \cdots, \omega_M} f(\omega_1, \cdots, \omega_M)$ is obviously non-negative, we start with an initial $\gamma = 0$ and solve the subproblem $F(0)$ by using existing fast polynomial programming model to update the weights $\omega_1, \cdots, \omega_M$. The parameter γ is gradually increased by $\gamma = f(\omega_1, \cdots, \omega_M)/g(\omega_1, \cdots, \omega_M)$ to help the algorithm enter the next round. The algorithm repeats the above-mentioned iterative procedure until $F(\gamma)$ converges to 0.

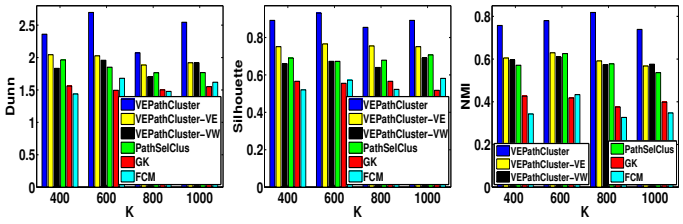
By assembling all the pieces in Section 3 together, we provide the pseudo code of our **VEPathCluster** algorithm in Algorithm 1.

4. EXPERIMENTAL EVALUATION

We have performed extensive experiments to evaluate the performance of **VEPathCluster** on three real graph datasets.

4.1 Experimental Datasets

The first real dataset is extracted from the DBLP Bibliography data [1], which contains 112,483 authors (A), 728,497 papers (P), 2,633 venues (V), and 45,968 terms (T). We choose three meta paths: A-P-A, A-P-V-P-A and A-P-T-P-A, to cluster authors and three kinds of path edges into soft clusters simultaneously.



(a) Dunn (b) Silhouette (c) NMI

Figure 7: Vertex Clustering Quality on DBLP

IMDb [2] is a searchable database of movies, TV and entertainment programs. We extract 48,975 actors (A), 31,188 movies (M), 4,774 directors (D), and 28 movie genres (G) from the original IMDb dataset. Three candidate meta paths: A-M-A, A-M-D-M-A and A-M-G-M-A, are used to assign each actor and three types of path edges to soft clusters.

The third real-world dataset is extracted from the Yelp's academic dataset [3], which includes 15,715 businesses (B), 470,212 reviews (R), 138,969 users (U), and 30,475 review terms (T). We select two meta paths: B-R-U-R-B and B-R-T-R-B, to generate the soft clusterings of businesses and two kinds of path edges.

4.2 Comparison Methods and Measures

We compare **VEPathCluster** with two representative soft clustering algorithms, **Fuzzy C-Means** (FCM) [8], **Gustafson-Kessel** (GK) [24], and one recently developed method **PathSelClus** [4]. For the first two clustering methods, we add the adjacency matrices of all vertex-centric path graphs together to get one single matrix. The first two methods perform vertex-centric soft clustering on a single graph and PathSelClus performs vertex-centric soft clustering on multiple graphs based on the assumption of vertex homophily.

We also evaluate three partial versions of VEPathCluster to show the strengths of edge clustering and weight learning respectively: (1)**VEPathCluster-VE** with only vertex clustering and edge clustering; (2) **VEPathCluster-VW** with only vertex clustering and weight update; and (3) **VEPathCluster-EW** with only edge clustering and weight update.

Evaluation Metrics We use three measures to evaluate the quality of vertex clustering by different methods. The fuzzy Dunn index [25,26] is defined as the ratio between the minimal fuzzy intra-cluster similarity and the maximal fuzzy inter-cluster similarity.

$$Dunn(\mathbf{X}) = \frac{\min_{1 \le k \le K} \left(\frac{1}{(\sum_{i=1}^{N_{V_c}} \mathbf{X}_k(v_i))(\sum_{j=i+1}^{N_{V_c}} \mathbf{X}_k(v_j))} \sum_{i=1}^{N_{V_c}} \sum_{j=i+1}^{N_{V_c}} \mathbf{X}_k(v_i)) \mathbf{X}_k(v_j)) \mathbf{P}(v_i, v_j) \right)}{\max_{1 \le k < l \le K} \left(\frac{1}{(\sum_{i=1}^{N_{V_c}} \mathbf{X}_k(v_i))(\sum_{j=1}^{N_{V_c}} \mathbf{X}_l(v_j))} \sum_{i=1}^{N_{V_c}} \sum_{j=1}^{N_{V_c}} \mathbf{X}_k(v_i)) \mathbf{X}_l(v_j)) \mathbf{P}(v_i, v_j) \right)} \quad (23)$$

where $\mathbf{X}$ is the vertex soft clustering membership matrix and $Dunn(\mathbf{X})$ is bounded in the range $[0, +\infty)$. A larger value of $Dunn(\mathbf{X})$ indicates a better clustering.

The following two metrics are often used to evaluate the hard clustering result, we thus map the soft clustering results by various methods into hard clustering results with the maximum probability of each vertex as its hard cluster labels.

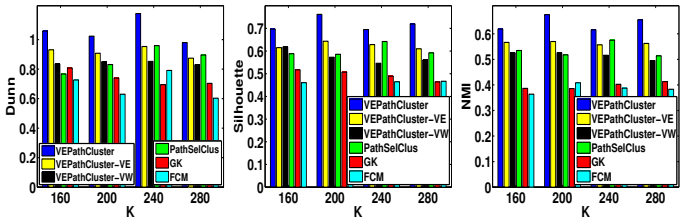
$$b(v_i) = \frac{1}{|VC_k| - 1} \sum_{v_j \in VC_k, j \ne i} \mathbf{P}(v_i, v_j), \quad a(v_i) = \max_{1 \le l \le K, l \ne k} \left(\frac{1}{|VC_l|} \sum_{v_j \in VC_l} \mathbf{P}(v_i, v_j) \right)$$

$$Silhouette(\{VC_k\}_{k=1}^K) = \frac{1}{K} \sum_{k=1}^{K} \left(\frac{1}{|VC_k|} \sum_{v_i \in VC_k} \frac{b(v_i) - a(v_i)}{\max\{a(v_i), b(v_i)\}} \right) \quad (24)$$

[1] http://dblp.uni-trier.de/xml/

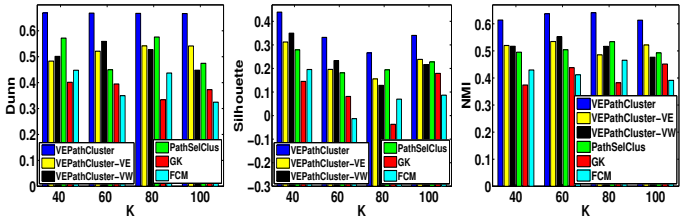
[2] http://www.imdb.com/interfaces

[3] http://www.yelp.com/academic_dataset



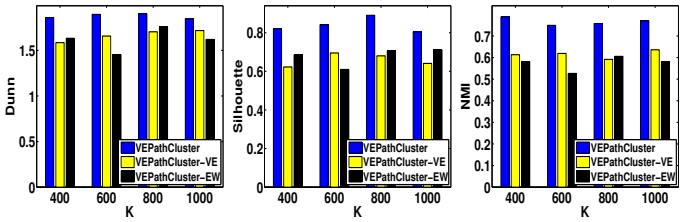
(a) Dunn (b) Silhouette (c) NMI

Figure 8: Vertex Clustering Quality on IMDb



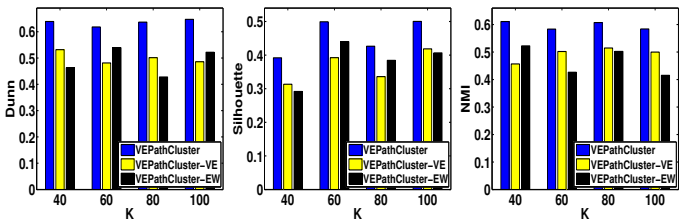
(a) Dunn (b) Silhouette (c) NMI

Figure 9: Vertex Clustering Quality on Yelp



(a) Dunn (b) Silhouette (c) NMI

Figure 10: Edge Clustering Quality on DBLP



(a) Dunn (b) Silhouette (c) NMI

Figure 11: Edge Clustering Quality on Yelp

where $\{VC_k\}_{k=1}^{K}$ represents the mapped hard clustering of the target vertices V_c, i.e., $V_c = \bigcup_{k=1}^{K} VC_k$ and $VC_k \bigcap VC_l = \phi$ for $\forall 1 \leq k, l \leq K, k \neq l$. $\mathbf{P}(v_i, v_j)$ is the edge value between two vertices v_i and v_j in the unified vertex-centric path graph VG. The silhouette coefficient [27] with the bound of [-1, 1] contrasts the average intra-cluster similarity with the average inter-cluster similarity. The larger the value, the better the quality.

Following the same strategy used in [4], we use $NMI(X, Y) = \frac{I(X;Y)}{\sqrt{H(X)H(Y)}}$ to compare the generated vertex clustering with the ground truth, where X and Y represent two cluster label vectors for the ground truth clustering and the calculated clustering by a clustering method respectively. $NMI(X, Y)$ is in the interval $[0, 1]$ and a larger NMI value indicates a better clustering.

Similarly, we use the same three measures to evaluate the quality of edge clustering by VEPathCluster. We report the average metric value for each measure based on M edge clustering results.

4.3 Vertex Clustering Quality

Figures 7-9 exhibit the vertex clustering quality on DBLP, IMDb and Yelp by varying the number of clusters. We divide six soft clustering methods into three categories: (1) FCM and GK perform the basic vertex clustering only based on the matrix of the unified vertex-centric path graph; (2) PathSelClus, VEPathCluster-VW and VEPathCluster-VE utilize partial optimization techniques to further improve the quality of vertex clustering; and (3) VEPathCluster makes use of both techniques of edge clustering and weight learning to achieve the promotion as much as possible.

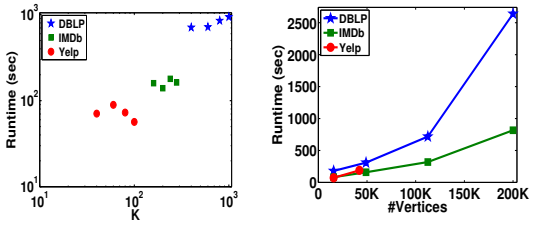
First, PathSelClus, VEPathCluster-VW and VEPathCluster-VE significantly outperform FCM and GK on all three evaluation measures. We know that the edges in different vertex-centric path graphs usually have values with different scales. As vertex-centric clustering methods, both PathSelClus and VEPathCluster-VW efficiency integrates the matrices of multiple vertex-centric path graphs through the iterative weight learning mechanism to learn the optimal weight assignment for these matrices. Thus, the measure scores obtained by them are often comparable to each other. On the other hand, VEPathCluster-VE integrates vertex clustering and edge clustering to mutually enhance each other. These results demonstrate that the importance of exploiting both edge clustering and weight learning for meta path graph clustering.

Second, it is observed that VEPathCluster-VE outperforms PathSelClus and VEPathCluster-VW on three graph datasets, even though the dynamic weight refinement is not used in VEPathCluster-VE while both PathSelClus and VEPathCluster-VW employed some iterative weight learning method to find the optimal weight assignment and improve the clustering quality. This is because both PathSelClus and VEPathCluster-VW are based solely on vertex homophily, without incorporating and integrating edge homophily into the clustering analysis. These results illustrate that employing edge clustering is more important than exploit weight learning in solve the meta path graph clustering problem.

Finally, among all six clustering methods, VEPathCluster achieves the best clustering performance for all three evaluation measures in most cases. Compared to other algorithms, VEPathCluster averagely achieves 18.7% Dunn increase, 14.1% Silhouette boost and 22.4% NMI improvement on DBLP, 10.6% Dunn growth, 10.4% Silhouette increase and 8.7% NMI boost on IMDb, and 17.7% Dunn increase, 23.9% Silhouette boost and 11.6% NMI improvement on Yelp, respectively. Concretely, there are three critical reasons for high accuracy of VEPathCluster: (1) the clustering-based multigraph model integrates both vertex-centric clustering and edge-centric clustering to accurately capture the cluster-specific relationships between vertices and between edges; (2) the edge-centric random walk model provides a natural way to capture the dependencies among path edges within each vertex-centric path graph; and (3) the iterative learning algorithm help the clustering model achieve a good balance among different types of vertex-centric path graphs and edge-centric path graphs.

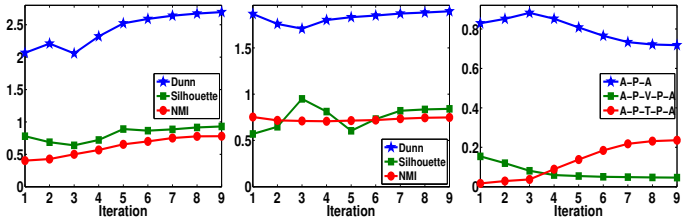
4.4 Edge Clustering Quality

Given that FCM, GK, PathSelClus and VEPathCluster-VW are vertex-centric soft clustering methods, we skip the experimental evaluation of edge clustering for these four approaches. Figures 10-11 present the edge clustering quality by three versions of VEPathCluster on two datasets with different K respectively. Similar trends are observed for the edge clustering quality comparison: VEPathCluster achieves the largest Dunn values (>0.62), the highest Silhouette around 0.39-0.89, and the largest NMI (>0.58), which are obviously better than other two methods. As K increases, the measure scores achieved by VEPathCluster remains relatively stable, while the measure scores of other two methods oscillate in a fairly large range. In addition, in terms of three evaluation measures, VEPathCluster-VE outperforms VEPathCluster-EW in some cases but VEPathCluster-EW performs better than VEPathCluster-VE in some cases. These results demonstrate that each of vertex clustering, edge clustering and weight learning plays an important role in meta path clustering. Thus, we should integrate three optimization techniques to further improve the clustering quality.



(a) Varying K (b) Varying #Vertices

Figure 12: Clustering Efficiency



(a) Vertex Clustering (b) Edge Clustering (c) Weight Assignment

Figure 13: Clustering Convergence

4.5 Clustering Efficiency

Figure 12 (a) presents the clustering time achieved by VEPathCluster on DBLP, Last.fm and IMDb with the same K setups in the experiments of clustering quality in Figures 7-11 respectively. Figure 12 (b) exhibits the scalability test of VEPathCluster by varying the number of target vertices on three datasets respectively. For DBLP and IMDb, we test four different setups of #Vertices, i.e., #Vertices = 15,715, 48,975, 112,483, 200,000 respectively. However, we only test #Vertices = 15,715, 42,153 for Yelp since the original Yelp dataset contains up to 42,153 businesses. We observe that VEPathCluster scales well with the size of graph for different graph datsets and shows good performance with varying K. A careful examination reveals that the bottleneck component of the overall time complexity for VEPathCluster is the execution time of iterative vertex clustering and edge clusterings, which mainly consist of a series of matrix-vector multiplications. Let K be the number of clusters, N_{V_c} be the number of target vertices in the unified vertex-centric path multigraph, N_{E_k} $(1 \le k \le K)$ be the number of parallel edges on the k^{th} cluster in the unified vertex-centric path multigraph, M be the number of edge-centric path multigraphs, N_{E_m} $(1 \le m \le M)$ be the number of vertices in the m^{th} edge-centric path multigraph, $N_{F_{mk}}$ $(1 \le k \le K)$ be the number of parallel edges on the k^{th} cluster in the m^{th} edge-centric path multigraph, t_i is the number of inner iterations, and t_o be the number of outer iterations in the clustering process. At the worst case, i.e., the original graph dataset is relatively dense, the complexity of performing vertex clustering on the unified vertex-centric path multigraph is equal to $O(t_o t_i K N_{V_c}^2)$ and the cost of performing edge clustering on each of M edge-centric path multigraphs is equal to $O(t_o \sum_{m=1}^{M} t_i K N_{E_m}^2)$. However, when the original graph dataset is very spare, the complexity of matrix-vector multiplication is approximately bounded by the size of edges. In this situation, the complexity of performing vertex clustering is reduced to $O(t_o t_i \sum_{k=1}^{K} N_{E_k})$ and the cost of performing edge clustering on all M edge-centric path multigraphs is decreased to $O(t_o \sum_{m=1}^{M} t_i \sum_{k=1}^{K} N_{F_{mk}})$.

4.6 Clustering Convergence

Figure 13 (a) and (b) exhibit the convergence trend of vertex clustering and edge clustering in terms of three evaluation measures on DBLP. Both the Dunn values and the NMI scores in two figures keep increasing or relatively stable and have convex curves when we iteratively perform the tasks of vertex clustering, edge clustering and weight update during the clustering process. On the other hand, the Silhouette values first fluctuate slightly within a range of [0.57, 0.95] and then converge very quickly. The entire clustering process converges in nine iterations for DBLP. Figure 13 (c) shows the tendency of weight update for three meta paths on DBLP. We keep the constraint of weights for three meta paths unchanged, i.e., $\sum_{m=1}^{M} \omega_m = 1$, during the clustering process. We observe that all three weights converge as the clustering process converges. An interesting phenomenon is that the weight for the A-P-A meta path first increases and then decreases with the iterations, the weight for the A-P-V-P-A meta path keeps decreasing and the weight curve for the A-P-T-P-A meta path has a converse trend. A reasonable explanation is that people who have many publications on the same conferences may have different research topics but people who have many papers with the same terms usually have the same research interests. On the other hand, for a pair of coauthors, their primary research areas are not always consistent in terms of the number of their coauthored papers, as illuminated in the example in Figure 2. Another interesting finding is that the weight for the A-P-A meta path is relatively large and other two weights are fairly small. This is because that the edges in different path graphs usually have values with different scales, as shown in Figure 3. In addition, the length of either of other two meta paths is larger than that of the A-P-A meta path, and there are many venues and terms in the DBLP dataset. To maintain a good balance among different meta paths, the algorithm needs to set larger weights for the path graphs with small-scale edges to maintain their contributions to clustering.

Author/Cluster	DB	DM	AI	IR
Ming-Syan Chen	0.258	0.588	0.021	0.134
W. Bruce Croft	0.058	0.006	0.026	0.909
Christos Faloutsos	0.346	0.539	0.012	0.102
Jiawei Han	0.373	0.459	0.057	0.111
H. V. Jagadish	0.904	0.048	0.014	0.034
Laks V. S. Lakshmanan	0.809	0.128	0.011	0.053
Hector Garcia-Molina	0.810	0.028	0.021	0.141
Eric P. Xing	0.009	0.123	0.830	0.038
Qiang Yang	0.012	0.265	0.512	0.210
Philip S. Yu	0.358	0.507	0.027	0.108
Chengqi Zhang	0.023	0.744	0.140	0.093

Table 1: Cluster Membership Probabilities of Authors Based on Three Meta Paths from DBLP

4.7 Case Study

We examine some details of the experiment results based on DBLP. Table 1 exhibits the set of authors and their cluster membership probabilities after nine iterations based on three meta paths: A-P-A, A-P-V-P-A and A-P-T-P-A. We only present most prolific DBLP experts in the area of database (DB), data mining (DM), artificial intelligence (AI) and information retrieval (IR). We observe that the predicted cluster memberships of authors are consistent with their actual research areas. For those researchers known to work in multiple research areas, the cluster membership distributions also correspond to their current research activities. For example, both *Jiawei Han* and *Philip S. Yu* are experts on data mining and database, though their *DM* probabilities are slightly higher since each of them and their circle of co-authors have more *DM* papers. Table 2 shows the set of path edges between the above authors in the A-P-A vertex-centric path graph and their cluster membership probabilities after nine clustering iterations. We have observed that most of author pairs associated to path edges usually have different primary research areas, e.g., the primary research areas of *W. Bruce Croft* and *Hector Garcia-Molina* are *IR* and *DB* respectively. In this situation, the cluster favorite of the path edges between the pairwise authors are often dominated by the primary research area of one associated author. For example, the path edge (*W. Bruce Croft*, *Hector Garcia-Molina*) has a main cluster favorite of *DB*. An interesting phenomenon is that although both *Ming-Syan Chen* and *Philip S. Yu* are experts on data mining, i.e., they both have more research publications in the area of data min-

Path Edge/Cluster	DB	DM	AI	IR
(Ming-Syan Chen, Philip S. Yu)	0.630	0.284	0.023	0.063
(W. Bruce Croft, Hector Garcia-Molina)	0.702	0.035	0.065	0.199
(Christos Faloutsos, H. V. Jagadish)	0.547	0.365	0.017	0.072
(Christos Faloutsos, Eric P. Xing)	0.238	0.713	0.015	0.034
(Jiawei Han, Laks V. S. Lakshmanan)	0.624	0.356	0.006	0.013
(Jiawei Han, Philip S. Yu)	0.518	0.424	0.013	0.045
(Qiang Yang, Philip S. Yu)	0.083	0.785	0.131	0.001
(Qiang Yang, Chengqi Zhang)	0.023	0.684	0.228	0.065

Table 2: Cluster Membership Probabilities of A-P-A Path Edges from DBLP

ing than in any other academic area such as database. However, the path edge (*Ming-Syan Chen*, *Philip S. Yu*) have a large probability on cluster *DB*. A careful examination reveals that most of coauthored publications between two experts are database specific.

5. RELATED WORK

Meta path-based social network analysis is gaining attention in recent years [1–6]. PathSim [1] presented a meta path-based similarity measure for heterogeneous graphs. [2] proposed a meta path-based ranking model to find entities with high similarity to a given query entity. HCC [3] is a meta-path based heterogeneous collective classification method. PathSelClus [4] utilizes user guidance as seeds in some of the clusters to automatically learn the best weights for each meta-path in the clustering. MLI [5] is a multi-network link prediction framework by extracting useful features from multiple meta paths.

Graph clustering has been extensively studied in recent years [9–21]. Shiga et al. [9] presented a clustering method which integrates numerical vectors with modularity into a spectral relaxation problem. SCAN [10] is a structural clustering algorithm to detect clusters, hubs and outliers in networks. MLR-MCL [11] is a multi-level graph clustering algorithm using flows to deliver significant improvements in both quality and speed. TopGC [14] is a fast algorithm to probabilistically search large, edge weighted, directed graphs for their best clusters in linear time. BAGC [16] constructs a Bayesian probabilistic model to capture both structural and attribute aspects of graph. GenClus [17] proposed a model-based method for clustering heterogeneous networks with different link types and different attribute types. CGC [18] is a multi-domain graph clustering model to utilize cross-domain relationship as co-regularizing penalty to guide the search of consensus clustering structure. FocusCO [20] solves the problem of finding focused clusters and outliers in large attributed graphs.

To the best of our knowledge, VEPathCluster is the first one to tightly integrate vertex-centric clustering and edge-centric clustering by mutually enhancing each other with combining different types of meta paths over heterogeneous information network.

6. CONCLUSIONS

We have presented a meta path graph clustering framework for mining heterogeneous information networks. First, we model a heterogeneous information network containing multiple types of meta paths as multiple vertex-centric path graphs and multiple edge-centric path graphs. Second, we cluster both vertex-centric path graph and edge-centric path graphs to generate vertex clustering and edge clusterings. Third, a reinforcement algorithm is provided to tightly integrate vertex clustering and edge clustering by mutually enhancing each other. Finally, an iterative learning strategy is proposed to dynamically refine both clustering results by continuously learning the degree of contributions of different path graphs.

Acknowledgement. The first two authors are partially supported by the NSF CISE under Grants IIS-0905493, CNS-1115375, IIP-1230740 and a grant from Intel ISTC on Cloud Computing. The third author performed this work under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

7. REFERENCES

[1] Y. Sun, J. Han, X. Yan, P. Yu, and T. Wu. PathSim: Meta path-based top-k similarity search in heterogeneous information networks. *PVLDB*, 4(11):992–1003, 2011.

[2] X. Yu, Y. Sun, B. Norick, T. Mao, and J. Han. User guided entity similarity search using meta-path selection in heterogeneous information networks. In *CIKM*, pages 2025–2029, 2012.

[3] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild. Meta path-based collective classification in heterogeneous information networks. In *CIKM*, pages 1567–1571, 2012.

[4] Y. Sun, B. Norick, J. Han, X. Yan, P. Yu, X. Yu. Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. In *KDD*, pages 1348–1356, 2012.

[5] J. Zhang, P. Yu, and Z. Zhou. Meta-path based multi-network collective link prediction. In *KDD*, pages 1286–1295, 2014.

[6] X. Ren, J. Liu, X. Yu, U. Khandelwal, Q. Gu, L. Wang, and J. Han. ClusCite: effective citation recommendation by information network-based clustering. In *KDD*, pages 821–830, 2014.

[7] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques, 3rd ed*. Morgan Kaufmann, 2011.

[8] J. C. Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Plenum Press, 1981.

[9] M. Shiga, I. Takigawa, H. Mamitsuka. A spectral clustering approach to optimally combining numericalvectors with a modular network. In *KDD*, pages 647–656, 2007.

[10] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger. Scan: a structural clustering algorithm for networks. In *KDD*, pages 824–833, 2007.

[11] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: Applications to community discovery. *KDD*, 2009.

[12] Y. Sun and Y. Yu and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD*, pages 797–806, 2009.

[13] Y. Zhou, H. Cheng, and J. X. Yu. Graph clustering based on structural/attribute similarities. *PVLDB*, 2(1):718–729, 2009.

[14] K. Macropol and A. Singh. Scalable discovery of best clusters on large graphs. *PVLDB*, 3(1):693–702, 2010.

[15] Y. Zhou, H. Cheng, and J. X. Yu. Clustering large attributed graphs: An efficient incremental approach. In *ICDM*, pages 689–698, 2010.

[16] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng. A model-based approach to attributed graph clustering. In *SIGMOD*, 505–516, 2012.

[17] Y. Sun, C. C. Aggarwal, and J. Han. Relation strength-aware clustering of heterogeneous information networks with incomplete attributes. *PVLDB*, 5(5):394–405, 2012.

[18] W. Cheng, X. Zhang, Z. Guo, Y. Wu, P. Sullivan, and W. Wang. Flexible and robust co-regularized multi-domain graph clustering. In *KDD*, pages 320–328, 2013.

[19] Y. Zhou and L. Liu. Social influence based clustering of heterogeneous information networks. In *KDD*, pages 338–346, 2013.

[20] B. Perozzi, L. Akoglu, P. Sanchez, and E. Muller. Focused clustering and outlier detection in large attributed graphs. In *KDD*, 2014.

[21] Y. Zhou and L. Liu. Activity-edge centric multi-label classification for mining heterogeneous information networks. In *KDD*, 2014.

[22] M. Roubens. Pattern classification problems and fuzzy sets. *Fuzzy Sets and Systems*, 1(4):239–253, 1978.

[23] M.-S. Yang. A survey of fuzzy clustering. *Mathematical and Computer Modelling*, 18(11):1–16, 1993.

[24] D. E. Gustafson and W. C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *CDC*, pages 761–766, 1979.

[25] J. C. Bezdek and N. R. Pal. Some new indexes of cluster validity. *TSMC*, 28(3):301–315, 1998.

[26] H. Hassar and A. Bensaid. Validation of fuzzy and crisp c-partitions. In *NAFIPS*, pages 342–346, 1999.

[27] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu. Understanding of Internal Clustering Validation Measures. In *ICDM*, pages 911–916, 2010.

[28] Y. Zhou, L. Liu, C.-S. Perng, A. Sailer, I. Silva-Lepe, and Z. Su. Ranking services by service network structure and service attributes. In *ICWS*, 2013.