Community Search for multi-attributed weighted network

Mostafijur Rahman Akhond, Md Nasim Adnan

Abstract— Recently, community search over large networks has got significant interest. In applications such as analysis of social network, web network, protein interaction networks and collaboration networks, edges and nodes are likely to have attributes. Unfortunately, most previous community search algorithms ignore multiple attributes of nodes and edges. In this paper, we study the problem of multi-attributed weight driven community search, that is, given an undirected graph *G* where nodes and edges are associated with multiple attributes/weights, and an input query node *q*, desired community size k, find the community containing q, in which most of the members are cohesively connected with each other. For community construction we proposed a new distance calculation approach which is calculated using the attributes. Here distance indicates the degree of dissimilarities between any two pair of nodes. Experiments and analysis on large social network show the effectiveness of our proposed solution by considering both the node and edge attributes.

Index Terms— Community search, Edge weight, Multi-attribute, Node weight, Weighted graph, Network.

1 Introduction

REAL-world networks, such as online social media, web networks and biological networks enclose community structures. Uncover the underlying communities in network is a major research issue in network which has attracted significant attentions in recent years [1, 2, 15-19]. Another associated but quite different study is community search where the aim is to discover the most associated community which contains given query node [3-11]. The dissimilarity between these two researches is - community detection deals with identifying all the communities in a network using some criterions [1], while the *community search* is query dependent which aims to discover the community that contains given query vertex [3]. Community search is much faster technique than detection as it reduces the computational spaces from the whole graph to some specific portion.

Community has been defined widely as densely connected subgraph build over the topological properties of the graph. Additionally, some researches [5-8] considered the node attributes in community search which leads to more accuracy in semantics of community. Other research directions [9, 10] considered edge attributes and weights that give more values in the connection strength of two nodes. In practical scenarios both the edge attributes and node attributes are containing in the same network. For example, in the co-authorship network two authors having publication is the edge which can be associated with more attributes like number of common publications, average citation counts of mutual publications or impact factor of the common publications etc. On the other side the authors himself can be associated with multiple attributes like his total number of publication, citations, impacts or interested area etc. Figure 1 presents such a network where each of the nodes associated with two attribute value and edges are associated with single weight. To model such network, the concept of multi-attributed graph can be used wherein each vertex is represented by an n-dimensional vector and there may be multiple weighted edges between each pair of vertices. To perform community search in such networks we devise the weighted distance measure for multi-attributed graph. Distance is a well-established element in building community [3, 11]. The good community is formed with low distanced entities. There are many other goodness measures in community search study but the studies with aim to find defined sized community performs better with distance based goodness. Muhammad et al. [12] gave an initial thought of multi-attributed distance calculation in graphs. The study is very helpful for our problem scenario because it can compute a unified distance measurement among two adjacent nodes using the node and edge attributes. Further we modified their distance measure to suit into our problem. Their proposed model computes distance for non-adjacent nodes based on their node attributes only, while in community study topological structure is also essential so we consider using physical connection information for the non-adjacent nodes.

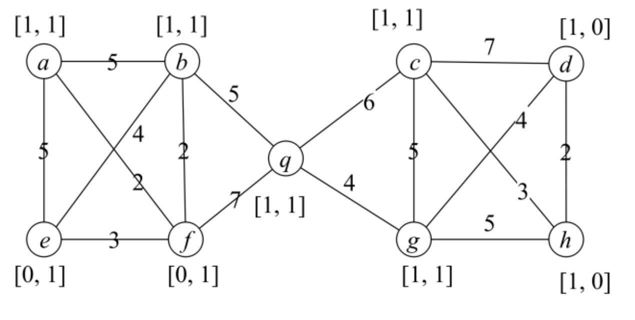

Fig. 1. Simple multi-attributed network

Existing works [5-8] on searching communities in attributed graph didn't consider edge attributes. And some are bounded to specify the attribute as query. All the existing methods are configured with k-core, k-truss, or quasi clique based models with distance calculation. We are not considering them as solution in our study to provide more emphasize to the query vertex itself, the distance based solutions are more focused to find community that closed to query, on the other side kore or truss based solutions search for denser region. Figure 1 shows an example multi-attributed graph. For the truss and core based solutions the result for query node *q* is {*a*, *b*, *e*, *f*, *q*} or {*c*, *d*, *g*, *h*, *q*} which is assigning away nodes {*a*, *e*} or {*d*, *h*} into *q*'s community. But distance based solution will find {*b*, *c*, *f*, *g*, *q*} as the solution.

Rest of the paper is organized as follows – Section 2 discussed the related works, section 4 describes the proposed solution and related terminologies, section 4 presents the experiment and analysis and finally in section 5 conclude the study.

2 Related Work

2.1 Community Detection

Earlier studies on community focused to find out all the communities from the entire network. It is extensively studied and popularly known as community detection. Among them [1, 13-15] are the most popular works. These works aimed to find out communities based on different types of clustering, label propagation and matrix blocking. There also some works for community detection in attributed networks [16-18], they considered additional information with the vertexes along with the structural information. Most of them applied clustering techniques to identify the communities after applying their customized filters. In [18, 19] they considered the node attributes and edge strength in community detection

2.2 Community Search

Community search is finding the community only related to given query nodes. It removes the overheads of computing whole graph. Recently it has been extensively studied [3, 5-11]. Suzio et al. [3] was the first to introduce community search with aim to organize a party for given query, they used distance based community finding in their study. The study was focusing on removing the most remote node from graph until the graph come to desired size. They also proposed k-core based searching of community. It was an expensive solution as to examine all the nodes in graph. Later on the Cui at al. [11] proposed a better community search approach called local search of community, which is the opposite approach of the earlier one. Staring from the query and include the neighbor nodes one by with some k-core conditions. It was also having some issues with handling complex networks.

2.3 Attributed community search

Community search over large attributed graph is covered by [5-8], as the real networks are complex and like to have attributes on the vertexes so it got more intensions to the researchers. Fang et al. [6] proposed a method on large attributed graph, using the k-core cohesiveness. Another significant work is proposed by Shang et al. [8] but all of them lack on edge attributes. They impact of edge weight and attribute is also introduced [10].

3 Proposed Methodology

In this section, first we present the mathematical formulation of distance using the attribute and weight information. The distance for nodes connected with an edge is molded based on the weighted Euclidian norms [12] and the remaining distances are measured by weighted smallest path distance. After formulizing the distance, we define our expected community for multi-attributed weighted network.

4 Citations

IJSER style is to not citations in individual brackets, followed by a comma, e.g. "[1], [5]" (as opposed to the more common "[1, 5]" form.) Citation ranges should be formatted as follows: [1], [2], [3], [4] (as opposed to [1]-[4], which is not IJSER style). When citing a section in a book, please give the relevant page numbers [2]. In sentences, refer simply to the reference number, as in [3]. Do not use "Ref. [3]" or "reference [3]" At the beginning of a sentence use the author names instead of "Reference [3]," e.g., "Smith and Smith [3] show ... ." Please note that references will be formatted by IJSER production staff in the same order provided by the author.

3.1 Weighted Distance Measurement

We present the weighted distance measure for multi-attributed graph that is based on weighted Euclidean norm over the attributed network. First we evaluate the attributes of the vertex and then we amalgamate the effects of edges. Considering the *n* attributes of *a* vertex, we map it to an *n* dimensional real vector. Vertex *a* is expressed by $\boldsymbol{a} = (a_1, a_2, \dots, a_n)^T$. Similarly vertex *b* is $\boldsymbol{b} = (b_1, b_2, \dots, b_n)^T$. The multi-labelled edge between the node *a* and *b* is shown as, $e(a, b) = (e_1(a, b), e_2(a, b), \dots e_m(a, b))^T$. We used the weighted Euclidian norm [12] to calculate the distance between node *a* and *b*, $\Delta(a,b)$. The distance is measured with (1), where λ is the scalar value depending on the aggregate weighted attributes of edges. λ is calculated by (2). In (2) ω denotes the aggregated weight of edge (*a*, *b*) which is calculated using the equation (3). γ is a user-provided onset, giving the flexibility to tune the value of λ for calculating distance between a vertex-pair. In (3) α is a constant such that $\Sigma a = 1$. Note that I in (1) is an identity matrix of *n* size.

Using the equation (1) we measured the weighted multi-attribute distance between two nodes connected with an edge. But community can be larger in size than the number of neighbors of query node. To complete the community size requirement some scenarios might need to calculate the distance for neighbors of neighbors. To solve this issue, we searched the smallest distance for all the connected nodes from query vertex. The authors of [12] ignore edge contribution for non-directly connected nodes and find cohesiveness using the attribute information. In community search study structural cohesiveness is also important. It leads to consider the physical hop distance for the vertexes that don't have any edge with query vertex.

```
Algorithm 1 Distance Calculation
Input: V, E, q ,$\gamma$, $\alpha$
    {Description} $V$ contains the node list with attribute values, E contains the edge list with attribute values, q is the query nod. $\gamma$ is the tuning factor on edge, and $\alpha$ is the 1D array where $\alpha_i \geq 0$ and $\sum_{i=1}^{m} \alpha_i = 1$
Output: $dist_q$
    {Description} A java has map containing distance of all the nodes from q
 1: Hashmap $dist_q, dist_e$;
 2: loop For each e(a,b) in E
 3:     { // e is the connecting edge of node a and node b}
 4:     $\omega = \sum_{i=1}^{m} \alpha_i e_i(a, b)$
 5:     $\lambda = 1/(1 + \omega)^\gamma$
 6:     $\Delta = \sqrt{\lambda}*\sqrt{\sum_{i=1}^{n} N_a[i]^2 - N_b[i]^2}$
 7:     $dist_e.put(a\_b, \Delta)$
 8: end loop
 9: loop For each neighbor of x in N except q
10:     if $(q\_x \in dist_e)$ then
11:         $dist_q.put(x, dist_e.get(q\_x))$
12:     else
13:         $\Delta = find\_dist(q, x)$ { // find smallest distance between q and x }
14:         $dist_q.put(x, \Delta)$
15:     end if
16: end loop
17: return $dist_q$
```

Algorithm 1 shows the distance calculation of proposed system. It takes vertex list (V) with attributes, edges list (E) with attributes, query vertex q, the tuning parameter γ and linier combination of edge weights between a pair of nodes (α) as parameter. From line 3-7 we calculate the distance for existing edge list and line 9 to 17 it calculates the query distance for all the nodes in Graph. For the query distance calculation if the vertex has edge with query vertex then we adopt the distance already calculated in step 6. If the vertex doesn't have edge with query vertex we find the smallest distance between the vertex and query vertex (line 13). The *find_dist* is an efficient method to check the smallest distance between two vertexes considering their possible paths.

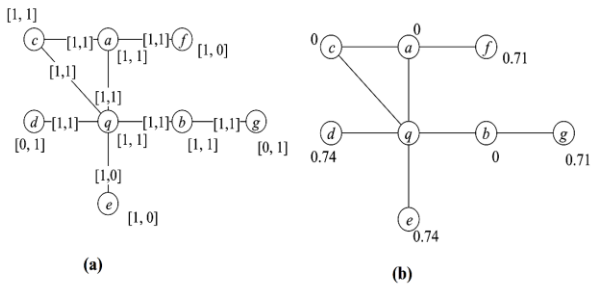

Fig. 2. (a) A multi-attributed graph, (b) Query distance for q, γ=1 and α=[0.5,0.5]

3.2 Multi-attributed Community

This section we formally introduce the definition of multi-attributed weighted community. The community we are searching from a multi-attributed network where both vertexes and edges can be represented as multidimensional vectors. Mathematically we define the graph as G (V, E), in which V contains the vertex list and each of the vertex is associated with *n* attribute values. Similarly, E represents the edge list with *m* attributes value for each of the edge. The problem of community search is to find a community, H for query vertex q from G where –

(1) H is a connected induced subgraph of G

(2) The query distance in H is minimized

(3) Size (H) ≤ k

Example: Figure 2 (a) represents a simple multi-attributed network, with both the nodes and edges are containing 2 attributes. Figure 1(b) represents the distance value calculated for each of the nodes considering the query vertex q. The subgraph H induced by {a, b, c, q} is the desired weighted multi-attributed community for k=4 as it has lowest possible query distance.

3.2 Finding the Community

This section elaborates the proposed solution to find the community for given multi-attributed network. We defined the distance in previous section. The distance is key factor to determine the cohesiveness between query vertex and other vertexes. In our algorithm we present a local approach to find the community. We start to explore the network for suitable community members from query node q and check the query distance of adjacent vertexes. The smaller the distance denotes the closeness of that pair of vertex. It leads us to add the vertexes into community H, based on the ascending order of their query distance. Algorithm 2 finds the expected community

```
Algorithm 2 Community Search
Input: $dist_q$, $k$, $q$
    {Description} $dist_q$ is the map containing vertex as key and the distance as value. $k$ is the desired size of community and $q$ is query vertex
Output: $H \subseteq G$
    {Description} H is the induced sub-graph of G containing query vertex.
sort $dist_q$ on value
$H \leftarrow q$
$size \leftarrow 1$
loop for each entry $(x, dist)$ in $dist_q$
    $H \leftarrow H \cup x$
    $size \leftarrow size + 1$
    if $size = k$ then
        $break$
    end if
end loop
return $H$
```

In algorithm 2 we first sort the query distance of the vertex. Then add the first *k-1* vertex into the solution. It ensures the closer k vertexes related with q are added into the community. In experimental section we show how it provides better performance from the existing models.

4 Result Analysis

In this study, the community search is conducted using a synthetic dataset considering a social network of 5000 users. We consider two attributes – online hours and number of posts in last 1 year. We further scaled the active hours and number of posts into scale of 10 using well-known rescaling technique (4). It contains 225000 edges with min 179 connections with each user. Two edge attributes are considered - number of mutual likes and number of lines in messages they communicated. This is also scaled using the (4). We consider α [0.03, 0.07] to emphasize more into the communication. γ is set to 1 for the evaluation. We tested the model for serval values of γ, which indicates the effectiveness of γ in distance measure over node-attributes and edge-attributes. We evaluate our model for query time on different *K*, Figure x shows the community searching time with various size of from 50 to 250.

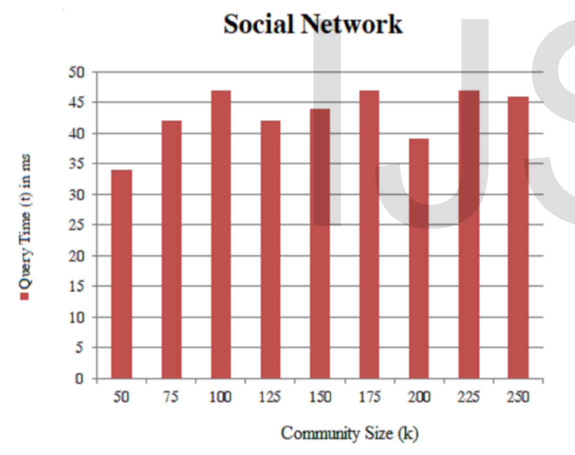

Fig. 3. Searching for community in social network.

$$v_{x[i]} = \left[\frac{a-a_{min}}{a_{max}-a_{min}} * 10\right] \qquad (4)$$

The specification of the computing machine is-
Windwos-10, 64 bit OS
Intel core(TM) i5-4460, 3.20 GHz CPU
8Gb ram

The Figure 3 shows community searching time for our generated social network. The running time varies from 34ms to 47ms, which is almost stable. The time includes Graph loading time also. Figure 4 and **Error! Reference source not found.**5 are showing the input graph with 500 nodes and resultant community.

Comparisons with existing solution:

The existing attributed community searching algorithms did not perform on both the edge and node attribute, they considered only node attributes in their measure while edge attributes play important roles in community relation. On the other hand, most of those solutions based on K-truss community structure which can find dense graph but the community members can be far away from the query node. Instead our community search emphasizes on distance with the query vertex which is counted on the attribute similarities from both vertex and edge perspectives.

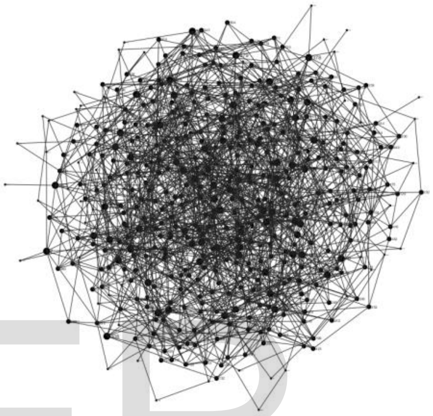

Fig. 4. Simplified Social Network with 500 Nodes.

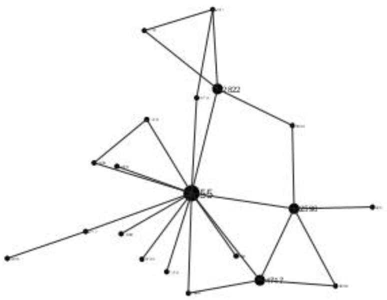

Fig. 5. Resultent Comuunity

5 Conclution And Future Work

This paper proposed a distance based community search model for multi-attributed large graph. We utilized the weighted Euclidian norm to calculate the distance between each pair of vertexes with multiple attributes. We showed the model can find the community where vertexes are very closely connected to each other as well as to the query node. The experimental results show the accuracy of model for several community sizes. In future we hope to extend the study with real social networks. Solution for distributed environment is also another future direction

REFERENCES

[1] Fortunato, Santo. "Community detection in graphs." Physics reports 486.3-5 (2010): 75-174.

[2] Xie, Jierui, Stephen Kelley, and Boleslaw K. Szymanski. "Overlapping community detection in networks: The state-of-the-art and comparative study." Acm computing surveys (csur) 45.4 (2013): 43.

[3] Sozio, Mauro, and Aristides Gionis. "The community-search problem and how to plan a successful cocktail party." Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2010.

[4] Cui, Wanyun, et al. "Online search of overlapping communities." Proceedings of the 2013 ACM SIGMOD international conference on Management of data. ACM, 2013.

[5] Li, Rong-Hua, et al. "Influential community search in large networks." Proceedings of the VLDB Endowment 8.5 (2015): 509-520.

[6] Fang, Yixiang, et al. "Effective community search for large attributed graphs." Proceedings of the VLDB Endowment 9.12 (2016): 1233-1244.

[7] Huang, Xin, and Laks VS Lakshmanan. "Attribute-driven community search." Proceedings of the VLDB Endowment 10.9 (2017): 949-960.

[8] Shang, Jingwen, et al. "An attribute-based community search method with graph refining." The Journal of Supercomputing (2017): 1-28.

[9] Huang, Xin, et al. "Approximate closest community search in networks." Proceedings of the VLDB Endowment 9.4 (2015): 276-287.

[10] Zheng, Z., Ye, F., Li, R. H., Ling, G., & Jin, T. (2017). Finding weighted k-truss communities in large networks. Information Sciences, 417, 344-360.

[11] Cui, Wanyun, et al. "Local search of communities in large graphs." Proceedings of the 2014 ACM SIGMOD international conference on Management of data. ACM, 2014.

[12] Abulaish, Muhammad. "A Novel Weighted Distance Measure for Multi-Attributed Graph." arXiv preprint arXiv:1801.07150 (2018).

[13] Newman, Mark EJ, and Michelle Girvan. "Finding and evaluating community structure in networks." Physical review E 69.2 (2004): 026113.

[14] Newman, Mark EJ. "Fast algorithm for detecting community structure in networks." Physical review E 69.6 (2004): 066133.

[15] Rosvall, Martin, and Carl T. Bergstrom. "Maps of random walks on complex networks reveal community structure." Proceedings of the National Academy of Sciences 105.4 (2008): 1118-1123.

[16] Liu, Yan, Alexandru Niculescu-Mizil, and Wojciech Gryc. "Topic-link LDA: joint models of topic and author community." proceedings of the 26th annual international conference on machine learning. ACM, 2009.

[17] Nallapati, Ramesh M., et al. "Joint latent topic models for text and citations." Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008.

[18] Ruan, Yiye, David Fuhry, and Srinivasan Parthasarathy. "Efficient community detection in large networks using content and links." Proceedings of the 22nd international conference on World Wide Web. ACM, 2013.

[19] Steinhaeuser, Karsten, and Nitesh V. Chawla. "Community detection in a large real-world social network." Social computing, behavioral modeling, and prediction. Springer, Boston, MA, 2008. 168-175.