Front. Comput. Sci., 2024, 18(3): 183610
https://doi.org/10.1007/s11704-024-2604-8

LETTER

A biased edge enhancement method for truss-based community search

Yuqi LI[1], Tao MENG (✉)[1], Zhixiong HE[2], Haiyan LIU[3], Keqin LI[4]

1 School of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410082, China
2 School of Business, Central South University of Forestry and Technology, Changsha 410082, China
3 College of Information Engineering, Changsha Medical University, Changsha 410219, China
4 Department of Computer Science, State University of New York, New York 12561, USA

1 Introduction

A related study called community search, whose target is to find dense subgraphs containing the given node, has drawn a growing amount of attention recently [1]. To explore the higher-order structure of complex networks, truss-based community search methods [2] have been proposed. Nevertheless, the truss-based hypergraph constructed from the original graph is frequently fragmented and consists of numerous subgraphs and isolated nodes [3], which boils down to the fact that these methods often pay only attention to the truss connections but ignore the lower-order connectivity of the original graph.

To address the fragmentation issue faced by most of the existing truss-based community search methods, we propose a Biased edge Enhancement method for Truss-based Community Search (BETCS). Initially, we identify the rough subgraph containing the desired community based on the query node. Then, we measure the proximity of each node to it in the subgraph. Moreover, these nodes are divided into several levels according to their proximity values, and those at lower levels have higher proximity values. Afterwards, edges are enhanced by connecting nodes at a lower level to nodes at a level above them. Ultimately, we perform the edge-enhanced k-truss community search on the subgraph to obtain the desired community. Our contributions are summarized below.

- We propose a biased edge enhancement method to preserve and enhance the higher-order connectivity in hypergraphs with the fragmentation issue;
- We propose an edge-enhanced k-truss community search algorithm to address the truss-based hypergraph fragmentation issue.

2 Methodology

To accurately and efficiently determine the scope of the desired community, we identify an i-hop subgraph based on the query node, where nodes are no more than i-hop away from it. Initially, the i-hop subgraph is defined as:

Definition 1 (i-hop subgraph). Given a graph $G(V,E)$, an integer $i>0$, and a query node $q \in V$. We denote H as an i-hop subgraph of G if (1) H is connected; (2) the distance between each node in H and node q does not exceed i-hop.

After obtaining the i-hop subgraph, the next step is to perform the edge enhancement on it. The following is our strategy to enhance edges. Initially, we measure the proximity of each node to q in the i-hop subgraph. Then we divide these nodes into several levels according to their proximity values, and lower levels indicate higher proximity values. Ultimately, we connect nodes at a lower level to nodes at a level above them to achieve the purpose of enhancing edges. Next, we introduce measuring proximity, grading nodes, and implementing edge enhancement in turn.

Let $w(x)$ and $w(x,y)$ denote the degree of node x and edge weight between x and y, respectively. Then, the transition probability $p(x,y)$ from x to y is defined as

$$p(x,y) = w(x,y)/w(x). \tag{1}$$

Starting from a node x, the random walker iteratively moves to its neighbors with a probability proportional to the edge weight and returns to x with a constant probability c at each step. The steady-state probability that it ends up staying at the query node q is defined as the proximity of x to q. In summary, the random walk with restart (RWR) method is defined as

$$r(x) = \begin{cases} (1-c)\sum_{y\in N_x} p(y,x)r(y) + c, & \text{if } x = q, \\ (1-c)\sum_{y\in N_x} p(y,x)r(y), & \text{if } x \neq q, \end{cases} \tag{2}$$

where N_x denotes the neighbors of node x and constant c $(0<c<1)$ is defined as the restart probability.

Let l, N be the number of levels and data, $\bar{z}$ be the mean of all data. First, calculate the sum of squared deviations from the array mean ($SDAM$) as follows:

Received September 27, 2022; accepted January 15, 2024

E-mail: mengtao@hnu.edu.cn

$$SDAM = \frac{1}{N}\sum_{j=1}^{N}\left(z_j - \bar{z}\right)^2. \quad (3)$$

Second, calculate the sum of squared deviations about the class mean($SDCM$) as follows:

$$SDCM = \sum_{i=1}^{l}\frac{1}{N_i}\sum_{j=1}^{N_i}\left(z_{ij} - \bar{z}_i\right)^2. \quad (4)$$

Third, calculate the goodness of variance fit (GVF) as follows:

$$GVF = 1 - \frac{SDCM}{SDAM}. \quad (5)$$

From the above equations, $SDAM$ is a constant and GVF ranges from 0 (worst fit) to 1 (perfect fit).

After dividing nodes into several levels according to their proximity values to the query node in the previous step, we consider how to enhance edges in this step.

Let $L(i)$ be the ith level, i.e., $L(i) = L(1), ..., L(n)(i = 1, ..., n)$. Then the enhanced edge set between $L(i)$ and $L(j)$ is defined as

$$E(i, j) = \{(x, y) \mid \forall x \in L(i), \forall y \in L(j)\}, \quad (6)$$

where $(i, j) = (1, 2), (2, 3), ..., (n-1, n)$. Then, the entire enhanced edge set is defined as

$$E_{\text{en}} = \{E(i, j) \mid \forall (i, j) \in \{(1, 2), ..., (n-1, n)\}\}. \quad (7)$$

Therefore, based on the entire enhanced edge set E_{en}, the enhanced graph G_{en} can be defined as

$$G_{\text{en}} = (V, E \cup E_{\text{en}}). \quad (8)$$

After edge enhancement, we consider performing an edge-enhanced k-truss community search on the i-hop subgraph to obtain the desired community containing the query node. Next, we first define the edge-enhanced k-truss community search, and then explain it with the algorithm in detail.

Definition 2 (edge-enhanced k-truss community search). Given a graph $G(V, E)$, an integer $k \geqslant 2$, a query node $q \in V$, and an enhanced edge set E_{en}, search all k-truss communities containing q based on the enhanced graph $G_{\text{en}}(V, E \bigcup E_{\text{en}})$.

Based on the above definition, the pseudo-code of the BETCS algorithm is given in Online Resource 1.

3 Experiment and analysis

To test the performance of the BETCS algorithm, we selected five benchmark algorithms for comparison, which are k-core [4], k-truss [2], DMF_M [5], LCDNN [6] and EquiTree [7]. For these algorithms, we specify the parameters to the values recommended by the authors.

Initially, we evaluate the effectiveness of these six algorithms on five large-scale networks with ground-truth communities. Figure 1 shows that our BETCS performs best in terms of F-score and NMI. In particular, the F-score obtained by BETCS is 15% and 6% higher than that of k-truss and EquiTree on the DBLP network, 22% and 1% higher than that of k-truss and EquiTree on the Amazon network, 64% and 35% higher than that of k-truss and EquiTree on the Youtube network, 19% and 3% higher than that of k-truss and EquiTree on the Orkut network, and 14% and 5% higher than that of k-truss and EquiTree on the Livejournal network. From this point of view, it demonstrates that the fragmentation problem faced by k-truss community search in these five large networks is well addressed by BETCS. In addition, LCDNN has the closest results to BETCS on the Amazon network, and DMF_M has the closest results to BETCS on DBLP and Orkut networks, but their results are worse than those of BETCS on the other networks.

From Fig. 2, we can see that the fastest running algorithm for community search is EquiTree and the slowest is k-truss, and our BETCS algorithm is the fourth fastest. Specifically, BETCS runs almost 3 times, 5 times, 11 times, 14 times, 21 times, faster than k-truss on the DBLP network, Amazon network, Youtube network, Orkut network, and Livejournal network, respectively. Although DMF_M, LCDNN and EquiTree all run faster than BETCS, they have worse accuracy than BETCS. In addition, our BETCS runs in less than 100 seconds, which is well accepted. More comparative experiments are given in Online Resource 1.

4 Conclusion

Most truss-based community search methods are usually confronted with the fragmentation issue. We propose a Biased edge Enhancement method for Truss-based Community Search (BETCS) to address the issue. This paper mainly solves the fragmentation problem in truss community query

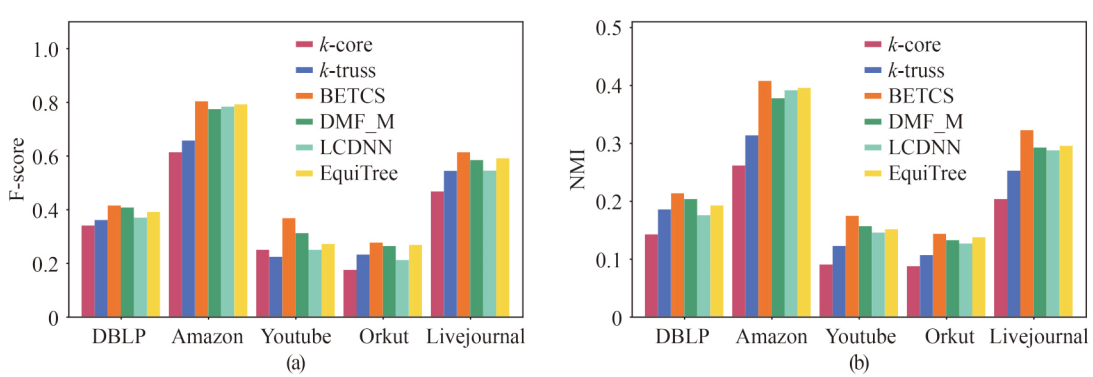

Fig. 1 Accuracy of six algorithms on large-scale networks in terms of average F-score and NMI. (a) F-score vs Networks; (b) NMI vs Networks

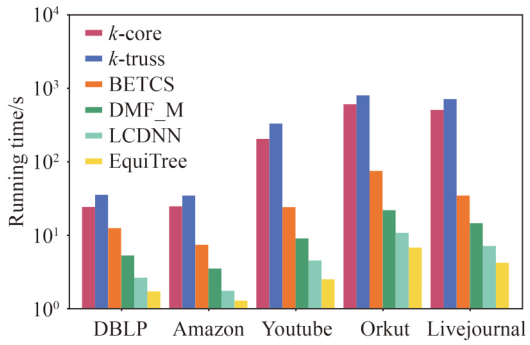

Fig. 2 Running time of six algorithms on large networks

through data enhancement. In future work, we will consider applying the methods in the text to directed graphs or dynamic graphs.

Acknowledgements This work was supported by the Research Foundation of Education Bureau of Hunan Province of China (Grant Nos. 20B625, 22B0275), and the Changsha Natural Science Foundation (Grant No. kq2202294).

Competing interests The authors declare that they have no competing interests or financial conflicts to disclose.

Electronic supplementary material Supplementary material is available in the online version of this article at journal. hep.com.cn and link.springer.com

References

1. Akbas E, Zhao P. Truss-based community search: a truss-equivalence based indexing approach. Proceedings of the VLDB Endowment, 2017, 10(11): 1298–1309
2. Huang X, Cheng H, Qin L, Tian W, Yu J X. Querying k-truss community in large and dynamic graphs. In: Proceedings of 2014 ACM SIGMOD International Conference on Management of Data. 2014, 1311–1322
3. Fang Y, Wang Z, Cheng R, Wang H, Hu J. Effective and efficient community search over large directed graphs. IEEE Transactions on Knowledge and Data Engineering, 2019, 31(11): 2093–2107
4. Cui W, Xiao Y, Wang H, Wang W. Local search of communities in large graphs. In: Proceedings of 2014 ACM SIGMOD International Conference on Management of Data. 2014, 991–1002
5. Luo W, Zhang D, Jiang H, Ni L, Hu Y. Local community detection with the dynamic membership function. IEEE Transactions on Fuzzy Systems, 2018, 26(5): 3136–3150
6. Luo W, Lu N, Ni L, Zhu W, Ding W. Local community detection by the nearest nodes with greater centrality. Information Sciences, 2020, 517: 377–392
7. Xu T, Lu Z, Zhu Y. Efficient triangle-connected truss community search in dynamic graphs. Proceedings of the VLDB Endowment, 2022, 16(3): 519–531