Pirouz *et al. J Big Data (2016) 3:22*
DOI 10.1186/s40537-016-0058-z

Journal of Big Data

RESEARCH Open Access

An optimized approach for community detection and ranking

Matin Pirouz[1*], Justin Zhan[1] and Shahab Tayeb[2]

*Correspondence:
matin.pirouznia@unlv.edu
[1] Department of Computer Science, UNLV, Maryland Pkwy, Las Vegas, NV, USA
Full list of author information is available at the end of the article

Abstract

Community structures and relation patterns, and ranking them for social networks provide us with great knowledge about network. Such knowledge can be utilized for target marketing or grouping similar, yet distinct, nodes. The ever-growing variety of social networks necessitates detection of minute and scattered communities, which are important problems across different research fields including biology, social studies, physics, etc. Existing community detection algorithms such as fast and folding or modularity based are either incapable of finding graph anomalies or too slow and impractical for large graphs. The main contributions of this work are twofold: (i) we optimize the Attractor algorithm, speeding it up by a factor depending on complexity of the graph; i.e. the more complex a social graph is, the better result the algorithm will achieve, and (ii) we propose a community ranker algorithm for the first time. The former is achieved by amalgamating loops and incorporating breadth-first search (BFS) algorithm for edge alignments and to fill in the missing cache, preserving a constant of time equal to the number of edges in the graph. For the latter, we make the first attempt to enumerate how influential each community is in a given graph, ranking them based on their normalized impact factor.

Keywords: Betweenness, Breadth-first search, Community strength, Complex networks, Jaccard index, Modularity

Introduction

Communities are essentially the strength of connection patterns among members of online social networks. Such connections as friendship over Facebook or following a cooking page on Instagram result in generation of voluminous data, causing a wide array of relationships. Graph theory is utilized to depict relations between community nodes while statistical properties of nodes are used to find the patterns. Node centrality defines community boundaries and impacts adjacencies [1]. Optimization of the quality function, i.e. modularity, is represented as eigenvectors of the network matrix [2].

Community detection algorithms have been around for some time now; however, with the growing size of todays networks, how to find small communities in big graphs, with billion of nodes and edges, is a major challenge. Graphs of Online Social Networks (OSN) such as Facebook, Twitter, etc. are growing every day. This growth introduces both large and small communities, for instance those with less than 100 members. There is a need for an algorithm capable of finding both large and small communities efficiently, in terms of time and processing overhead. Moreover, finding anomalies and

communities outliers also prove challenging [3, 4]. A key undertaking would be to rank these communities after detecting them. For example, Facebook has over 20,000 communities and it is important to calculate which community has more influence over the entire members in the graph. A higher rank for a community implies a higher probability of active and influential members. Being able to find communities' ranks helps us with (a) analyzing the network graph and relations, (b) finding the most suitable data mining techniques, (c) predicting the information flow, and (d) comprehending public sentiment.

In this study, detecting and characterizing communities are discussed. Optimized Attractor algorithm is introduced which finds communities using Jaccard distance, similar to Attractor algorithm; however, the performance of our approach does not deteriorate for much bigger or smaller graphs, unlike Attractor. Furthermore, a novel algorithm is introduced to rank the communities using the ratio of intra-community and inter-community links between links. Unlike most community detection algorithms, Optimized Attractor normalizes the size of communities based on a threshold. The performance of Optimized Attractor was measured on known benchmarks, where it outperforms modularity-based methods.

The rest of this paper is organized as following: "Related work" section reviews an indispensable comparison for existing community detection algorithms, be it from Betweenness algorithm to modularity maximization approaches to. "Our approach" section introduces the proposed method, Optimized Attractor, after discussing the required preliminaries. "Analysis" section depicts an exhaustive analysis of performance and time complexity of Optimized Attractor. Discussion of the results and future directions are given in "Discussion and future work" section. "Conclusion" section outlines the concluding remarks.

Related work

Communities in the graphs are detected and defined based on similarity between their vertices; i.e. a community is a group of nodes with the highest degree of similarity between the vertices under that community in the graph. There are various methods to calculate the similarity values between nodes. Four of the most famous ones are Jaccard similarity, Cosine similarity, Pearson correlation, and Euclidean distance. After calculating the similarity between all nodes in the graph, all similar nodes will be placed in the same community and dissimilar edges between nodes will be cut out. However, after finding communities there is a need to find the ranking for these communities. Community detection is a helpful tool to analyze big and complex networks such as social media networks. Various methods have been used in order to find network communities. Back in 2002, Girvan and Newman [5] used Betweenness algorithm to find modularity change value. The way it worked was that every node was put in its own community initially. Then, the modularity value was calculated using Eq. 1. Next, the neighboring node with the highest Q value was combined with the designated community. Time complexity for the mentioned method is N^3. Newman [6] introduced a new method the next year which reduced the time complexity from the lowest value of M^2 for normal matrices to N^3 on sparse matrices to $((M+N)N)$ or N^2 on sparse matrices. Greedy algorithm [7] was introduced by Clauset and Newman the following year and improved the time

complexity of the latter method to N log^2 n. In 2005, External Optimization method was introduced by Duch and Arenas [8]. In this method, Duch and Arenas [8] have reached a better accuracy by using external optimization to optimize modularity, sacrificing the operational speed. Time complexity for this algorithm is $N^2 log^2 n$ which is slower than the Greedy method. Modularity formula:

$$Q = \sum_r (e_{rr} - a_r^2) \tag{1}$$

Optimized version of modularity:

$$Q = \frac{1}{2} m \sum_{i,j} A_{ij} - \frac{d_i d_j}{2m} (\delta_{i,j}) \tag{2}$$

On the other hand, Shao and Han developed a new algorithm called attractor based on distance dynamic [9]. Time complexity of Attractor algorithm is E which is counted as linear. Although it is faster than N-cut [10], Modularity, edge Betweenness [5], greedy and et, it is a bit slower than algorithms proposed by Louvan [12] and Infomap [13], and Metis [14]. The Louvan method [12] starts every node as a separate community; then in each iteration, it calculates the similarity between two communities and mixes them together.

The attractor method introduced by Shao et al. [9] uses the concept of distance instead of similarity. In this method, the similarity of nodes have been calculated by Eq. 3 as you can see the Jaccard similarity [16] has been used. Next the value found will get subtracted from one in order to find the distance between two nodes.

$$S(s,e) = 1 - \frac{\Upsilon(s) \cap \Upsilon(e)}{\Upsilon(s) \cup \Upsilon(e)} \tag{3}$$

Next step is to find the initial distance for all the nodes in the graph. There are three types of relation ship between the nodes in this algorithm. The first one is when two nodes are connecting directly. The distance between to direct relationship is calculated through Eq. 4

$$DC = -\left(\frac{1 - S(s,e)}{deg(s)} + \frac{1 - S(e,s)}{deg(e)} \right) \tag{4}$$

The second type is defined as the impact of mutual neighbors of two nodes on their direct link. In which is counted as a positive impact. It being counted positive, comes from the fact that having mutual interests usually makes a relationships stronger. The equation is shown in Eq. 5

$$\begin{aligned} MC = -\sum_{m \in MN} \Bigg(& \frac{(1 - S(m,s)).(1 - S(m,e))}{deg(s)} \\ & + \frac{(1 - S(m,e)).(1 - S(m,s))}{deg(e)} \Bigg) \end{aligned} \tag{5}$$

The last type is shown in Eq. 6 . The third type of impact is from exclusive neighbors. Having different or against interest can cause the relationships to become more distance. So the third type will usually be a positive impact on the link value between start and end node.

$$XC = -\sum_{x \in XN(s)} \left(\frac{(1 - S(x,s)).(1 - S(x,s))}{deg(s)} - \sum_{x' \in XN(e)} \frac{(1 - S(x',e)).(1 - S(x',e))}{deg(e)} \right) \tag{6}$$

$$S(s,e,t+1) = S(s,e,t) + DC(t) + MC(t) + XC(t) \tag{7}$$

Our approach

Network structure evolves as time changes [17]. In this paper, G represents a graph. Every G contains set of nodes N and set of edges E. C is used to show set of communities. Every edge that connects two communities together is called as a between edge and shown bye e_b . n_s^e and n_t^e are two nodes in the sides of an edge, they are called starting and target node respectively. A counter is used to identify and counts the number of edges in which they have only one side in a community and the other side belongs to another community. And another counter is used to count number of edges that have nodes in different communities. Later counter is represented by (γ). The number of nodes in each community is shown by (η). Finally, rank of every community is recorded as R_c. The method used to calculate community rank is based on the number of nodes inside reach community and the total number of nodes in the graph. After finding the number of between edges for each community, the rank can be counted by dividing this number by all the nodes in the graph except for those already in the community. In this study, communities were found using an optimized version of Attractor algorithm. As it was explained in "Related work" section, Attractor algorithm works base on distance between nodes and their inclusive and exclusive neighbors. However, it was found that Attractor algorithm could work faster by removing the last loop of algorithm. In fact, the communities are found in the third loop by filtering the distance matrix. So, wherever the distance between two node is equal to one the edge should be cut from the graph. But this action could be applied with reversing the filtering function meaning taking the distances wherever it is equal to zero we could keep the edge in fact. Three major effort has been put in this study:

- First a breadth-first search (BFS) [18] is added to the algorithm before going thorough the initial distance calculations. It helps the program run almost ten percent faster (an algorithm for traversing or searching tree or graph data structures). It starts at the tree root (or some arbitrary node of a graph, sometimes referred to as a 'search key') and explores the neighbor nodes first, before moving to the next level neighbors [18].
- The original Attractor algorithm has been optimized by the method explained in "Algorithm" section. Using this optimization method, the program runs faster by a factor of almost one fifth. (shown in Table 1).

Table 1 Comparison between Attractor and Optimized Attractor

Data set	Attractor	Optimized Attractor	OA and BFS
Karate Club	0.012278	0.012091	0.011930
American Football	0.119706	0.101960	0.101330
Polbooks	0.149019	0.138020	0.137216
Friendship	446.488900	325.466700	305.337700
Amazon	379.636510	297.4342910	285.840700
Road	327.805610	317.141900015	303.112400

All the numbers are in milliseconds

- A new algorithm called "Community Ranker" is introduced. As the name suggest, CR (Community Ranker) method finds a value for each community that was found by Algorithm 1. How communities have been ranked is provided in "Algorithm" section by Algorithm 2.

Preliminaries

Social networks are graphs created by members of social media. These graphs are made from social media members and the relationships between them in forms of nodes and edges respectively. A friendship on the Facebook is an example of a two-sided (undirected) connection and following somebody's page Instagram or tweeter is an example of a one-sided (directed) connection and a connections in DBLP (Digital Bibliography & Library Project) network is an example of a weighed edge. In social networks, connections between people reveal a lot of information about the graph and communities existing in the graph. A community is a group of people with mutual interest. As same as real life in every social-media, there exist various communities. This communities are made from members of social media and their pages. Example of communities could be friendship groups or groups of people with same occupations or simply groups of people with same interests. In social graphs, relationship inside the community and relationships outside are respectively shown by internal edges and external edges. Number of internal-edges are dense within a community where number of external edges are low. In contrast between two separate communities there is no internal edge but a small number of external edges are existed. Among all the communities that exist in social media, there is a need to distinguish those who are active and more important in the graph. Figuring out the active communities can help with advertisings tasks, statistics and analyses needed for suggestions, searches and trend analytics. In addition, finding the ranking list of communities shows how important each community is. Communities with higher ranks have a better communication inside and outside the community. Local computations are good for big graphs. Community detection methods are divided into two subgroups. The first well known method is to cluster vertexes based on their similarity. And the second method is graph partitioning base on sparse cut.

Notations

Adjacency matrix is a matrix of the size $n \times n$ where n is the number of nodes in the graph. It has ones whenever two nodes are connected and zero otherwise. Every

connection between two nodes is defined as an edge. Edges can be directed or undirected. Directed edges are those that connect two nodes in only one way. Undirected edges connect two nodes two-sided only with one edge. However, to show more features of nodes, for example the importance of each nod and edges connected to that node, there are weights defined for each edge. In this paper nodes are shown as n and the set of nodes is shown with N. Edges and set of edges are shown as l and L. Communities and the set of communities c and C. Start and end node are represented as s and R_c represents Rank of community c. And the number of nodes in each communities is represented by η. Also number of edges between two communities are show as γ. S is used to keep the similarity value. Mutual and exclusive neighbors are shown with m and x. And finally, neighboring Set is represented as Υ.

Algorithm

Algorithm 1 x Community Finder

Input: Graph G=(n, l), threshold δ
Output: C: {All communities in the graph}
Cal: Initial values
for $\forall l \in$ L **do**
    Cal: S_l using Eq. (1)
end for
for $\forall l \in$ L **do**
    if 0 **then** $<S_l>1$
        Cal: DC_l,CS_l,XC_l
        $TS = DC_l + MC_l + XC_l$
        $S_{t+1} = S_t + TS$
        if $S_{t+1} > 1$ **then** $S_{t+1} = 1$
        else $S_{t+1} = 0$ & remove the edge from G
        end if
    end if
end for

Algorithm 2 x Community Ranker

Input: C; {All communities in the graph} Graph G=(V, E), η
Output: R_i^c , C_w^i
Cal: Communities
for $\forall c \in$ G **do**
    for $\forall e \in$ C **do**
        if $n_s^e \& n_t^e \nexists C$ **then**
            Increment γ
        end if
        $R_c = Sin\frac{\gamma}{(N-\eta)}$
    end for
end for

Analysis

Time complexity analysis

The time complexity function of the existing Attractor algorithm is improved. Note that the improved version is working faster without sacrificing any accuracy. Overall, the latter linear time complexity is O($|E| + k|E|$) which is reduced by ($T|E|$) time. Even though Attractor and Optimized Attractor both have linear time complexities, the Optimized Attractor is faster by a factor of $O(T.|E|)$. As shown in Fig. 1, the optimized version of Attractor performs faster as the number of nodes and edges grow.

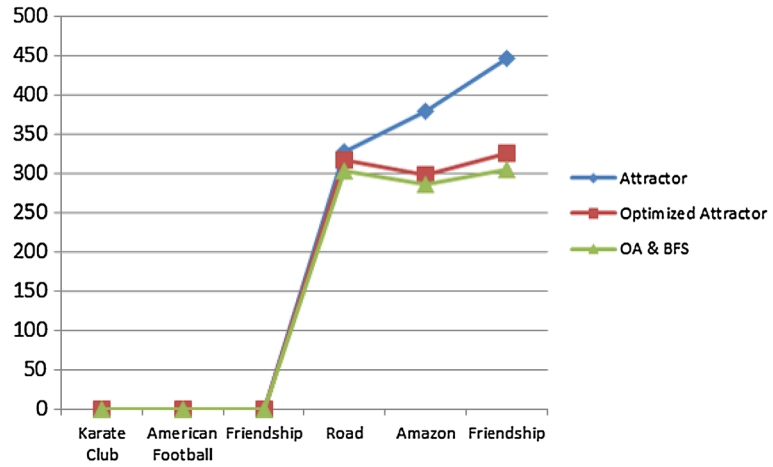

Fig. 1 Comparison chart for Attractor, Optimized Attractor without BFS , and OA with BFS

The reason for this phenomenon is that the process of finding communities was done using three loops in the original algorithm, which was performed over every graph edge. The Optimized Attractor reduces this to two loops; i.e. the algorithm is improve with a constant of $|E|$ for every datasets with edge number of $|E|$. Hence, Attractor time complicity which is $O(|E| + k|E| + T|E|)$ changed to $O(|E| + k|E|)$; therefore, it runs faster with constant factor of $O(|E|)$. Note that K is the average number of exclusive neighbors for two linked nodes and T is a constant number which is $3 \leq T \leq 50$. In OA, T is removed as all the calculations of third loop are compensated in second loop. The way, this has been possible is that instead of having third loop, all the filtering and flagging will be done in the second loop where similarities are being counted.

In the second algorithm (Community Ranker), the time complexity is $O(|C| + |E|)$. The given time is coming from the number of communities and the number of existing edges in each community. The time complexity for CR is O(C), where C is the number of communities.

Experimental analysis

This experiment was performed on a computer with Intel Xeon(R) E5-1607 @ 3GHz processors and 16 GB RAM. A single core was used to run the algorithm. All the nodes and edges were loaded in the machines main memory before calculating the time spent for the Attractor or community Ranker algorithm. All the proposed algorithms were implemented in Python programming language, and the library used was *networkx* [18]. A *matplotlib* library was used for the Plots draw from the graphs.

Data set

To be able to illustrate the performance of Optimized Attractor algorithm, the most commonly used datasets for community detections have been chosen and used in this experiment. Data sets used are all real world datasets. We used the famous Karate Club dataset [19] which is used in almost all community detection related problems. The second commonly used dataset in this work is American football [20] with 115 nodes. The other datasets used are PolBooks–Krebs' Amazon books [21], Theory collaboration network [22], Brightkite [23], Pennsylvania road network [24], Amazon product

Table 2 The studied datasets

Data set	Number of nodes	Number of edges
Karate Club	34	78
American Football	115	613
Polbooks	105	441
Friendship	58,228	214,078
Amazon	334,863	925,872
Road	1,088,092	1,541,898

Table 3 Similarity analysis of the Optimized Attractor

Data set	Attractor	Modularity
Karate Club	1	0.916128
American Football	1	0.952536
Polbooks	1	0.854031

co-purchasing network, and ground-truth communities [25]. More details about datasets used in this study are given in Table 2.

Discussion and future work

The result of comparison between Optimized Attractor, Attractor and OA & BFS time performance is shown in Table 1. The optimization shows it's most effect on Amazon dataset. One can see the difference is almost 25% faster than the previous result. With these point being mentioned the only concerned remained is the accuracy of the proposed method, in which is addressed in Table 3. Not only the accuracy did not degrade but also it shows that Optimized Attractor has a better level of accuracy. For PolBooks dataset, the accuracy of Modularity method is 0.85 where the accuracy of Optimized Attractor is 1. The result of Optimized Attractor for datasets Karate and Football are shown in Figs. 2 and 3. Communities divisions are demonstrated with different colors.

As for Community Ranker algorithm, it's shown in Table 4 it takes less than a millisecond to rank a network with more than a million nodes. For the smaller datasets, the time consumed is almost close to zero. However this should be considered that all the communities have already been detected and loaded to the system. The time shown in the Table 4 shows only the execution time for ranking system.

Also the results for top four communities in Karate, Football and Polbooks datasets are shown in Fig. 4.

Conclusion

In this paper, we proposed a method achieving a better performance as shown in Table 5. As the networks specially social networks grow to more complex, the needs of dealing with the new sophisticated data grows. Community detection algorithm proposed in this paper improves the speed and performance of finding communities.

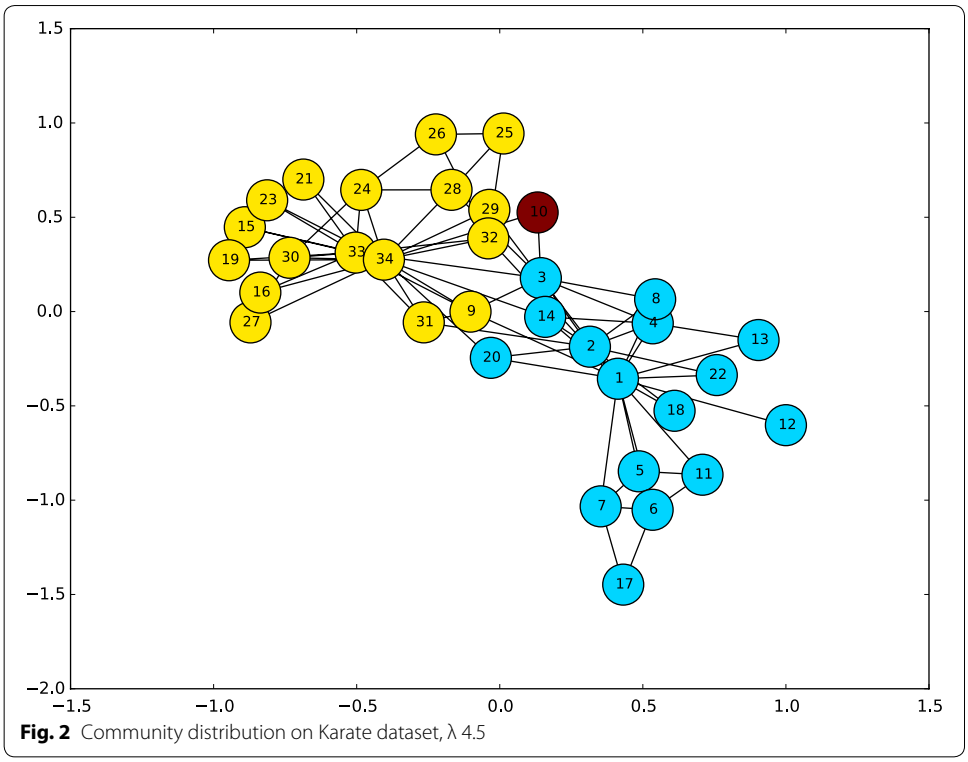

Fig. 2 Community distribution on Karate dataset, λ 4.5

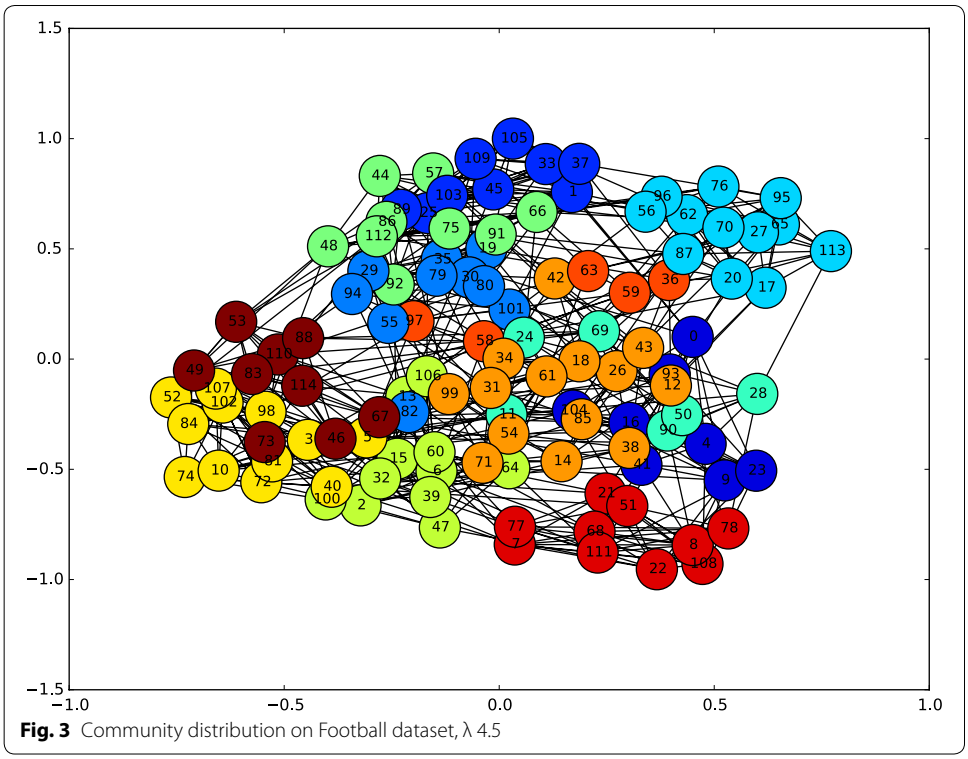

Fig. 3 Community distribution on Football dataset, λ 4.5

Table 4 Top four communities

Data set	Num of comm	Max nodes	Min nodes	Process time
Karate Club	3	17	1	0.001000
American Football	12	14	5	0.001000
Polbooks	4	44	1	0.001000
Friendship	15,280	15,280	1	0.132000
Amazon	33,931	1494	1	0.613000
Road	22,978	876,500	1	0.992000

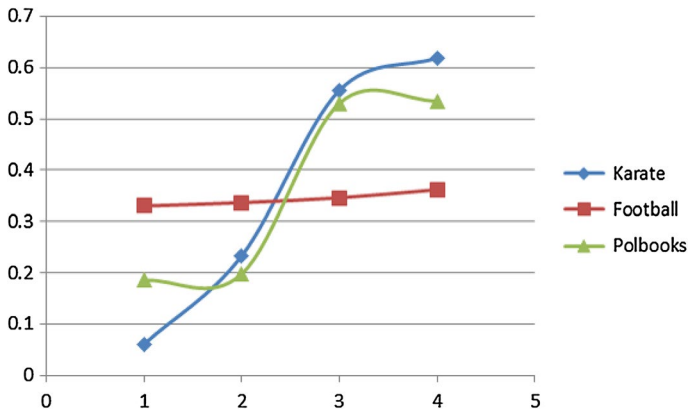

Fig. 4 Ranking of top four communities, Lambda 4.5

Table 5 Notations and symbols

Notation	Definition and description
G	Given graph
N, n	Set of nodes, node
L, *l*	Set of links, link
C, c	Set of community, community
η	Number of nodes in each C
R_c	Rank of community c
γ	Number of edges between two C
e_b	Designated edges between two C
n_s^e and n_t^e	Nodes on the side of an edge
S	Similarity value
s, e	Start and end node
m, x	Mutual, exclusive neighbor
Υ	Neighboring set

Summary of findings

We proposed a ranking system for communities in which rate communities based on their influence on the rest of network. As the interest in finding top communities and top nodes in each community grows, various methods have been developed to discover the ranking pattern. Our algorithm uses centrality concept to find top communities in each dataset. In this method, the number of intra community offspring of each node is counted and compared to the total number of nodes in the dataset and then a method

of normalization is applied. As a result, the proposed algorithm can find how topology important a node can be (more offspring to outside communities shows more influence).

Future work

We hope our results will motivate more studies on community ranking system. And having findings not only based on node relation but also on content base relations between nodes such as, comments, likes or follows. The speed and performance of finding communities is improved in this work but still there is a need to improve the accuracy related issues. Also the proposed algorithm can find community as an individual but there are a lot of nodes in which are common between different communities and can not be pointed as members of a specific communities. So there is a need to find the common part known as overlapping communities.

Authors' contributions

MP, as the first author, performed the primary literature review, data collection and experiments, and also drafted the manuscript. JZ and ST worked with MP to develop the algorithm, the paper, and the framework. All authors read and approved the final manuscript.

Author details

[1] Department of Computer Science, UNLV, Maryland Pkwy, Las Vegas, NV, USA. [2] Department of Electrical and Computer Engineering, UNLV, Maryland Pkwy, Las Vegas, NV, USA.

Acknowledgements

We gratefully acknowledge the United States Department of Defense (DoD Grant #W911 NF-13-1-0130), the National Science Foundation (NSF Grant #1560625), and Oak Ridge National Laboratory (ORNL Contract #4000144962) and United Hall Foundation (UHF #1592) for their support and finance for this project.

Competing interests

The authors declare that they have no competing interests.

Received: 28 August 2016 Accepted: 3 November 2016
Published online: 10 November 2016

References

1. Latora V, Marchiori M. A measure of centrality based on network efficiency. New J Phys. 2007;9(6):188.
2. Newman MEJ. Modularity and community structure in networks. In: Proceedings of the National Academy of Sciences, vol 103.23. 2006. p. 8577–82.
3. Gupta M, Gao J, Aggarwal CC, Han J. Outlier detection for temporal data: a survey. IEEE Trans Knowl Data Eng. 2014;26(9):2250–67. doi:10.1109/TKDE.2013.184.
4. Wu S, Wang S. Information-theoretic outlier detection for large-scale categorical data. IEEE Trans Knowl Data Eng. 2013;25(3):589–602. doi:10.1109/TKDE.2011.261.
5. Girvan M, Newman ME. Community structure in social and biological networks. Proc Natl Acad Sci. 2002;99(12):7821–6.
6. Newman ME. The structure and function of complex networks. SIAM Rev. 2003;45(2):167–256.
7. Clauset A, Newman ME, Moore C. Finding community structure in very large networks. Phys Rev E. 2004;70:066111. doi:10.1103/PhysRevE.70.066111.
8. Duch J, Arenas A. Community detection in complex networks using extremal optimization. Phys Rev E. 2005;72:027104.
9. Shao J, Han Z, Yang Q, Zhou T. Community detection based on distance dynamics. In Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. New York: ACM; 2015. p. 10751084.
10. Shi J, Malik J. Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell. 2000;22(8):888–905.
11. Wang C, Tang W, Sun B, Fang J, Wang Y. Review on community detection algorithms in social networks. In: 2015 IEEE international conference on progress in informatics and computing (PIC). Piscataway: IEEE; 2015. p. 551-5.
12. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. J Stat Mech Theory Exp. 2008;2008(10):P10008.
13. Bohlin L, Edler D, Lancichinetti A, Rosvall M. Community detection and visualization of networks with the map equation framework. In: Measuring scholarly impact. Berlin: Springer International Publishing; 2014. p. 3–34
14. Karypis George, Kumar Vipin. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput. 1998;20(1):359392.
15. Hubert L, Arabie P. Comparing partitions. J Classif. 1985;2(1):193–218.
16. Du N, Jia X, Gao J, Gopalakrishnan V, Zhang A. Tracking temporal community strength in dynamic networks. IEEE Trans Knowl Data Eng. 2015;27(11):3125–37. doi:10.1109/TKDE.2015.2432815.

17. Leiserson CE, Schardl TB. A work-efficient parallel breadth-first search algorithm (or how to cope with the nondeterminism of reducers). In: Proceedings of the 22nd annual ACM symposium on Parallelism in algorithms and architectures (SPAA '10). New York: ACM; 2010. p. 303–14. doi:http://dx.doi.org/10.1145/1810479.1810534.
18. Networkx. https://networkx.github.io/. Accessed 27 Sep 2016.
19. Zachary's Karate Club. https://networkdata.ics.uci.edu/data.php?id=105. Accessed 27 Sep 2016.
20. American College Football. http://www-personal.umich.edu/~mejn/netdata. Accessed 27 Sep 2016.
21. PolBooks—Krebs' Amazon books. http://vlado.fmf.uni-lj.si/pub/networks/data/mix/mixed.htm. Accessed 27 Sep 2016.
22. High energy physics—theory collaboration network. https://snap.stanford.edu/data/ca-HepTh.html. Accessed 27 Sep 2016.
23. Brightkite. https://snap.stanford.edu/data/loc-brightkite.html. Accessed 27 Sep 2016.
24. Pennsylvania road network. https://snap.stanford.edu/data/roadNet-PA.html. Accessed 27 Sep 2016.
25. Amazon product co-purchasing network and ground-truth communities. https://snap.stanford.edu/data/com-Amazon.html. Accessed 27 Sep 2016.