EUROPEAN ACADEMIC RESEARCH
Vol. IV, Issue 9/ December 2016

Impact Factor: 3.4546 (UIF)
DRJI Value: 5.9 (B+)

ISSN 2286-4822
www.euacademic.org

Review on Different Methods of Community Structure of a Complex Software Network

MUNAWAR HUSSAIN[1]
JIA DONG REN
HAFIZ SHEHZAD AHMED
AWAIS AKRAM
College of Information Science and Engineering
Yanshan University
Qinhuangdao, P.R. China

Abstract:

Complex network is an emerging method to explore different aspects of a complex software systems example community structure. Several complex network systems show natural divisions of network nodes. Every division, or community, is a densely connected subgroup. That's why in this paper we will review different methods of community detection in complex software networks. These techniques can be analysis and selected by their origin. This type of community structure helps comprehension and suitable for wide application in different software networks.

Key words: complex software network, community structure

INTRODUCTION:

Analysis of complex software system has made many significantly discoveries in the field of software system in recent years. With the increase of complexity of software systems, the

[1] Corresponding author: munawar@stumail.ysu.edu.cn

overall structure of the system is be- coming more and more complicated, making the structure become one of the most important factors that influences the quality of the final software products, and also making it difficult for software maintenance and version updating. Fast and accurate identification of influential spreaders in a network is essential to the acceleration of information diffusion, inhibition of gossip and spread of a virus [1]. Research communities have found a tool called complex network to deal with complex software system in which consider all functions as nodes and relationship between them as edge in a network. Many software systems have been demonstrated to exhibit the structure and to possess the properties of a complex network[2].The investigation on network theory has found many properties of complex network which got much attention in recent years, where different quantities of measuring their complexities have been defined and implemented, and several computer programs and algorithms have been developed for solving these quantities, with broad influence on the knowledge and understanding in different discipline.

At present, cascading failure [3] is the major method in the process of analyzing the influence of node failure in complex network. Zhou Tao et al.[4] indicated that allocating more capacity to high degree nodes can effectively improve network capability against cascading failures. Fang Xinli et al.[5] focused their research on directed complex network and proposed the minimum in-degree attack strategy and maximum out-degree attack strategy. Baharan Mirzasoleiman et al.[6] investigated the effects of cascading failure in accordance with three strategies, namely, betweenness centrality of the edges, degrees of the end nodes and their betweenness centralities. Ding Lin et al.[7] constructed a cascading model to study the betweenness-dependent node weighting method on weighting networks. Bao Z J et al.[8] compared cascading failures in

small-world and scale-free networks subject with vertex and edge attacks.

One of the important features of complex networks is the possibility of partitioning them into smaller sub-networks preserving complexity features. Using the language of social networks these sub-networks are named communities, meaning that there are strongly interconnected nodes inside a single community, while the connections with nodes of other communities are sparse or weak [12]. From this point of view software systems are excellent candidates for complex networks with an underlying community structure, since they are structured in a hierarchical manner, where small units cooperate with each other.

The above discussions on complex network provide a theoretical and technical support for analysis of influence of nodes community and failure of nodes in software network. With regard to nodes' studies in software network, lots of researches are aimed at identifying influential nodes or critical nodes [8, 9], yet there are few research results on the impact of node failure in software network. Lan et al.[10] built a software cascading failure model based on coupled map lattice(CML)[18], and found that the bigger the in-node-strength is, the larger the failure scale caused by malicious attack on that node is. Hou G et al.[11wer] took advantage of the CML model in complex networks to simulate and analyze the cascading effect on software systems. However, they only treated in-node-strength of node as asimple metric to choose nodes to be attacked. And the CML hasn't described how software program runs neither.

In this paper we try to fill this gap, presenting a case study of the community structure of a software network, with the aim of investigating if some techniques which have been proved useful for characterizing the community structure can be of help for characterizing the properties of the related software network.

REPRESENTATIONS OF DATA STRUCTURE

A graph data structure consists of a finite (and possibly mutable) set of vertices or nodes or points, together with a set of unordered pairs of these vertices for an undirected graph or a set of ordered pairs for a directed graph. These pairs are known as edges, arcs, or lines for an undirected graph and as arrows, directed edges, directed arcs, or directed lines for a directed graph. The vertices may be part of the graph structure, or may be external entities represented by integer indices or references.

Adjacency list

Vertices are stored as records or objects, and every vertex stores a list of adjacent vertices. This data structure allows the storage of additional data on the vertices. Additional data can be stored if edges are also stored as objects, in which case each vertex stores its incident edges and each edge stores its incident vertices.

Adjacency matrix

A two-dimensional matrix, in which the rows represent source vertices and columns represent destination vertices. Data on edges and vertices must be stored externally. Only the cost for one edge can be stored between each pair of vertices.

	A	B	C	D	E	F
A	0	1	0	0	1	1
B	1	0	1	0	0	1
C	0	1	0	1	1	0
D	0	0	1	0	0	1
E	1	0	1	0	0	0
F	1	1	0	1	0	0

Figure 1. Example of adjacency matrix

Adjacency lists

Representing a graph with adjacency lists combines adjacency matrices with edge lists. For each vertex i ii, store an array of the vertices adjacent to it. We typically have an array of |V| |V|verticalbar, V, vertical bar adjacency lists, one adjacency list per vertex. Here's an adjacency-list representation of the social network graph:

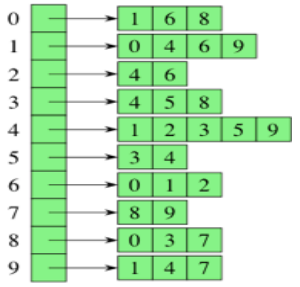

Figure 2. Example of adjacency list

Community structure complex software network with network theory

Community structure:

Modern software systems can be very large and can be made of tens of thousands, or even millions of lines of code. In the last decades, due to its simplicity, the use of Object Oriented (OO) programming paradigm largely increased and OO software systems are the majority in many applications. For any software system built according to the object oriented approach it is possible to easily define different kinds of networks, where the nodes represent specific software modules and connections among nodes represent relationships between software modules.

The investigation of complex networks received a large attention in the last decades, where many Quantities for measuring and characterizing their complexity have been defined and used, and many algorithms and software programs have been developed for computing these quantities, with large

impacts on the knowledge and understanding in very different fields and disciplines. One of the important features of complex networks is the possibility of partitioning them into smaller sub-networks preserving complexity features…

Minimum cut method

One of the oldest algorithms for dividing networks into parts is the minimum cut method (and variants such as ratio cut and normalized cut). This method sees use, for example, in load balancing for parallel computing in order to minimize communication between processor nodes.

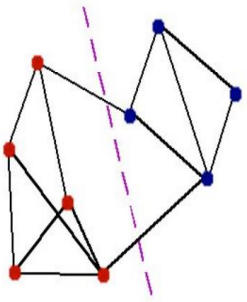

Figure 3. Example of Minimum cut method

A graph with n vertices can at the most have {\display style n(n-1)/2} n(n-1)/2 distinct minimum cuts.

In the minimum-cut method, the network is divided into a predetermined number of parts, usually of approximately the same size, chosen such that the number of edges between groups is minimized. The method works well in many of the applications for which it was originally intended but is less than ideal for finding community structure in general networks since it will find communities regardless of whether they are implicit in the structure, and it will find only a fixed number of them[13]

Hierarchical clustering

In this method one defines a similarity measure quantifying some (usually topological) type of similarity between node pairs. Commonly used measures include the cosine similarity, the Jaccard index, and the Hamming distance between rows of the adjacency matrix. Then one groups similar nodes into communities according to this measure. There are several common schemes for performing the grouping, the two simplest being single-linkage clustering, in which two groups are considered separate communities if and only if all pairs of nodes in different groups have similarity lower than a given threshold, and complete linkage clustering, in which all nodes within every group have similarity greater than a threshold. A novel approach in this direction is the use of various similarity or dissimilarity measures, combined through convex sums, [14] which has greatly improved the performance of this kind of methodology.

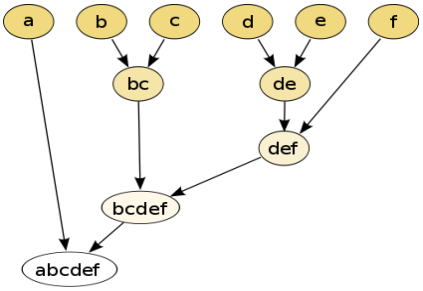

Figure 4. Example of Hierarchical Clustering

Girvan–Newman algorithm

Another commonly used algorithm for finding communities is the Girvan Newman algorithm.[16]This algorithm identifies edges in a network that lie between communities and then removes them, leaving behind just the communities themselves. The identification is performed by employing the graph-theoretic measure betweenness centrality, which assigns a number to each edge which is large if the edge lies "between" many pairs of nodes.

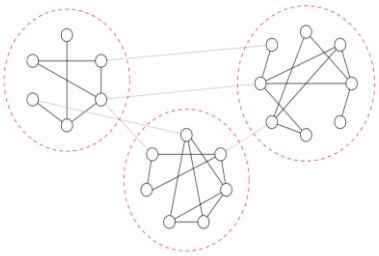

Figure 5. Example of Girvan-Newman algorithm

The Girvan–Newman algorithm returns results of reasonable quality and is popular because it has been implemented in a number of standard software packages. But it also runs slowly, taking time $O(m^2n)$ on a network of n vertices and m edges, making it impractical for networks of more than a few thousand nodes.[14]

Modularity maximization

In spite of its known drawbacks, one of the most widely used methods for community detection is modularity maximization. [15] Modularity is a benefit function that measures the quality of a particular division of a network into communities. The modularity maximization method detects communities by searching over possible divisions of a network for one or more that have particularly high modularity. Since exhaustive search over all possible divisions is usually intractable, practical algorithms are based on approximate optimization methods such as greedy algorithms, simulated annealing, or spectral optimization, with different approaches offering different balances between speed and accuracy.[17][18]A popular modularity maximization approach is the Louvain method, which iteratively optimizes local communities until global modularity can no longer be improved given perturbations to the current community state.[19] The currently best modularity maximization algorithm (winner of the10th DIMACS

Implementation Challenge) is an iterative ensemble algorithm.[20]

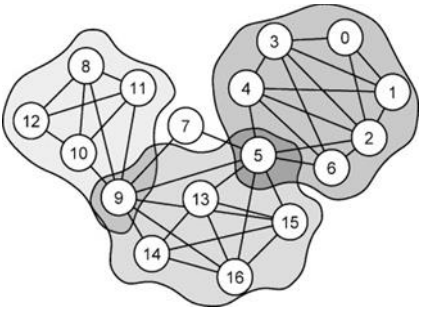

Figure 6. Example of Minimum cut method

The usefulness of modularity optimization is questionable, as it has been shown that modularity optimization often fails to detect clusters smaller than some scale, depending on the size of the network (resolution limit[21]); on the other hand the landscape of modularity values is characterized by a huge degeneracy of partitions with high modularity, close to the absolute maximum, which may be very different from each other.[22]

Statistical inference:

Methods based on statistical inference attempt to fit a generative model to the network data, which encodes the community structure. The overall advantage of this approach compared to the alternatives is its more principled nature, and the capacity to inherently address issues of statistical significance. Most methods in the literature are based on the stochastic [23] as well as variants including mixed membership,[24][25] degree-correction,[19] and hierarchical structures.[20] Model selection can be performed using principled approaches such as length and Bayesian model selection.[23] Currently many algorithms exist to perform efficient inference of stochastic block models, including belief propagation and agglomerative Monte Carlo.

Differently from approaches which attempt to cluster the network given an ad-hoc quality function, this class of methods is based on generative models which not only serve as a description of the large-scale structure of the network, but also can be used to *generalize* the data, and predict the occurrence of missing or spurious links in the network.

Clique-based methods:

Cliques are sub graphs in which every node is connected to every other node in the clique. As nodes can not be more tightly connected than this, it is not surprising that there are many approaches to community detection in networks based on the detection of cliques in a graph and the analysis of how these overlap. Note that as a node can be a member of more than one clique, a node can be a member of more than one community in these methods giving an **overlapping community structure**. One approach is to find the **maximal cliques**, that is find the cliques which are not the sub graph of any other clique. The classic algorithm to find these is the Bron–Kerbosch algorithm. The overlap of these can be used to define communities in several ways. The simplest is to consider only maximal cliques bigger than a minimum size (number of nodes). The union of these cliques then defines a sub graph whose components (disconnected parts) then define communities.[26]Such approaches are often implemented in social such as UCInet.

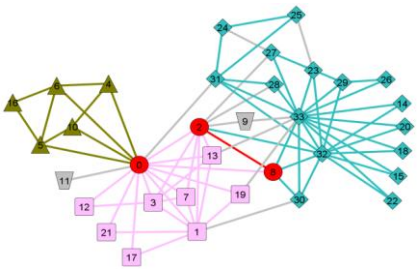

Figure 7.Example of Minimum cut method

The alternative approach to is to use cliques of fixed size, k. The overlap of these can be used to define a type of k-regular hyper graph or a structure which is a generalization of the line graph (the case when k=2) known as a **Clique graph**.[30] The clique graphs have vertices which represent the cliques in the original graph while the edges of the clique graph record the overlap of the clique in the original graph. Applying any of the previous community detection methods (which assign each node to a community) to the clique graph then assigns each clique to a community. This can then be used to determine community membership of nodes in the cliques. Again as a node may be in several cliques, it can be a member of several communities. For instance the clique percolation method defines communities as percolation clusters of k-cliques. To do this it finds all k-cliques in a network that is all the complete sub-graphs of k-nodes. It then defines two k-cliques to be adjacent if they share $k - 1$ nodes, that is this is used to define edges in a clique graph. A community is then defined to be the maximal union of k-cliques in which we can reach any k-clique from any other k-clique through series of k-clique adjacencies. That is communities are just the connected components in the clique graph. Since a node can belong to several different k-clique percolation clusters at the same time, the communities can overlap with each other.

CONCLUSION:

This paper discuss about the community structure of complex software system software in the different environment. Above mention several community detection methods are very suitable for find communities of different methods and classes in software network these techniques are very useful for new comer in this field. In Our future work will explore the classless of a software network with the help of complex network.

REFERENCES:

[1].Zhiying Zhao, Xiaofan Wang, Wei Zhang and Zhiliang Zhu. A Community-Based Approach to Identifying Influential Spreaders. Entropy 2015, 17, 2228-2252; doi:10.3390/e17042228
[2]:Giulio Concas,Cristina Monni, Matteoorru, Roberto Tonelli. A Study of the Community Structure of a Complex Software Network, Conference Paper·May2013. DO10.1109/WETSoM.2013.661933
[3]: P. Crucitti, V. Latora, M. Marchiori, A model for cascading failures in complex networks, Phys. Rev. E, vol.69, no.4, 2004, pp.045104.
[4]: P. I, B. H. Wang, H. Sun, P. Gao, T. Zhou, A limited resource model of failure-tolerant capability against cascading failure of complex network, EurPhysJB, vol.62, no.1, 2008, pp.101–104.
[5] X. Fang, Q. Yang, W. Yan, Modeling and analysis of cascading failure in directed complex networks, Safety Science, vol. 65, 2014, pp.1–9.
[6] B. Mirzasoleiman, M. Babaei, M. Jalili, MA. Safari, Cascaded failures in weighted networks, Phys. Rev. E, vol.84, no.4, 2011, pp.046114.
[7] L. Ding, S. Y. Zhang, Node weighting strategies on weighted networks against cascading failures. Journal of Computational Information Systems, vol.8, no.15, 2012, pp.6483-6490.
[8] B. Y. Wang, J. H. Lu¨, Software networks nodes impact analysis of complex software systems, Journal of Software, vol.24, no.12, 2013, pp.2814–2829(in Chinese). [16] X. Z. Zhang, G. L. Zhao, T. Y. Lu¨, Y. Yin, B. Zhang, Analysis on Key Nodes Behavior for Complex Software Network, Proceedings of the Third International Conference, ICICA 2012, Chengde, China, vol.7473, pp.59–66, 2012.
[9] Z. J. Bao, Y. J. Cao, L. J. Ding, G. Z. Wang, Comparison of cascading failures in small-world and scale-free networks

subject to vertex and edge attacks, Phys.A,vol.388, no.20, 2009, pp.4491–4498.
[10] W. Lan, K. Zhou, J. Feng, Research on Software Cascading Failures, Multimedia Information Networking and Security (MINES), 2010 International Conference on. IEEE, pp.310–314, 2010.
[11] G. Hou, X. Wang, K. Zhou, Network Model Construction and Cascading Effect Analysis for Software Systems, Software Engineering (WCSE) 2012 Third World Congress on. IEEE, pp.9–12, 2012
[12] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. Phys. Rev. E, 69(2):026113, Feb. 2004
[13] M. E. J. Newman (2004). "Detecting community structure in networks". Eur. Phys. J. B. 38 (2): 321–330.
[14]Alvarez, Alejandro J.; Sanz-Rodríguez, Carlos E.; Cabrera, Juan Luis (2015-12-13). "Weighting dissimilarities to detect communities in networks". Phil. Trans. R. Soc. A. 373 (2056): 20150108. doi:10.1098/rsta.2015.0108. ISSN 1364-503X
[15]M. E. J. Newman (2004). "Fast algorithm for detecting community structure in networks". Phys. Rev. E. 69 (6): 066133. doi:10.1103/PhysRevE.69.066133
[16]M. Girvan; M. E. J. Newman (2002). "Community structure in social and biological networks". Proc. Natl. Acad. Sci. USA. 99 (12): 7821–7826.
[17] L. Danon; J. Duch; A. Díaz-Guilera; A. Arenas (2005). "Comparing community structure identification". J. Stat. Mech. 2005
[18]R. Guimera; L. A. N. Amaral (2004). "Functional cartography of complex metabolic networks". Nature. 433 (7028): 895–900.
[19]V.D. Blondel; J.-L. Guillaume; R. Lambiotte; E. Lefebvre (2008). "Fast unfolding of community hierarchies in large networks". J. Stat. Mech. 2008 (10): P10008.

[20] Michael Ovelgönne; Andreas Geyer-Schulz (2013). "An ensemble learning strategy for graph clustering". Graph Partitioning and Graph Clustering. Contemporary Mathematics. American Mathematical Society.
[21] S. Fortunato; M. Barthelemy (2007). "Resolution limit in community detection". Proceedings of the National Academy of Sciences of the United States of America.
[22] B. H. Good; Y.-A. deMontjoye; A. Clauset (2010). "The performance of modularity maximization in practical contexts". Phys. Rev. E. 81 (4): 046106.
[23]Holland, Paul W.; Kathryn BlackmondLaskey; Samuel Leinhardt (June 1983). "Stochastic blockmodels: First steps". Social Networks. 5 (2): 109–137.
[24] Airoldi, Edoardo M.; David M. Blei; Stephen E. Fienberg; Eric P. Xing (June 2008). "Mixed Membership Stochastic Blockmodels". J. Mach. Learn. Res. 9: 1981–2014.
[25]Ball, Brian; Brian Karrer; M. E. J. Newman (2011). "Efficient and principled method for detecting communities in networks". Physical Review E. 84 (3): 036103.
[26]M.G. Everett; S.P. Borgatti (1998). "Analyzing Clique Overlap Connections". Connections. 21: 49.