**社区检测与排序优化算法**

**An optimized approach for community detection and ranking**

[**摘要(Abstract) 3**](#_Toc24178)

[**介绍(Introduction) 5**](#_Toc8312)

[**相关工作(Related work) 8**](#_Toc30215)

[**我们的方法(Our approach) 12**](#_Toc7523)

[初步(Preliminaries) 15](#_Toc32354)

[符号(Notations) 17](#_Toc10399)

[算法(Algorithm) 19](#_Toc2794)

[**分析(Analysis) 21**](#_Toc9974)

[时间复杂度分析(Time complexity analysis) 21](#_Toc6202)

[实验分析(Experimental analysis) 23](#_Toc2474)

[数据集(Data set) 23](#_Toc25940)

[**讨论和未来工作(Discussion and future work) 25**](#_Toc851)

[**结论(Conclusion) 29**](#_Toc5005)

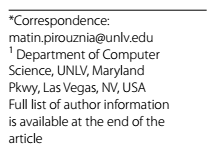
[研究成果总结(Summary of findings) 33](#_Toc27817)

[未来工作(Future work) 33](#_Toc10921)

[**参考文献(References) 34**](#_Toc8354)

IMG_256IMG_256 **RESEARCH Open Access**

IMG_256IMG_256



**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.

IMG_256**Keywords:** Betweenness, Breadth-first search, Community strength, Complex networks, Jaccard index, Modularity

© The Author(s) 2016. This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## 摘要(Abstract)

社交网络的社区结构和关系模式，以及对它们排序为我们提供了极大的网络知识。这些知识可用于目标市场营销或分类相似，但不同的节点。日益增长的各种社交网络需要分钟检测(detection of minute)和分散的社区(scattered communities)，这些在不同的研究领域，包括生物学，社会研究，物理学等是非常重要的问题。存在的社区检测算法，比如快速折叠或基于模块度，都不能发现[图异常](http://lab.toutiao.com/index.php/2016/12/28/daguimotuzhongdeyichangjiance.html)或是对于大的图非常慢且不实用。本次工作的主要贡献分两层：**（i）我们优化了吸引子算法，通过一个依赖于图复杂度的因素加快了算法的运行速度；即社交图越复杂，算法实现的结果越好;（ii）我们首次提出了社区排序算法。**前者通过合并循环，结合广度优先搜索算法进行边调整，填充缺失的缓存(to fill in the missing cache)，和保留一个等于图中边数的时间常数(a constant of time)实现。对于后者，首次尝试枚举一个给定的图中每个社区的影响力，并依据它们的标准化影响因子将它们排序。

**关键字：**中间状态(Betweenness)，广度优先搜索，社区强度(Community strength)，复杂网络，杰卡德系数，模块度。

**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.

## 介绍(Introduction)

社区本质上是在线社交网络成员之间的连接模式强度。这样的连接，比如Facebook上的友谊或跟随Instagram上的一个烹饪页面，会导致海量数据的生成，产生一个广泛的关系数组。当节点的统计特性被用于发现模式时，图论被用于描述社区节点间的关系。节点中心性定义社区边缘并影响邻接。质量函数的优化，即模块化，被表示为网络矩阵的特征向量。

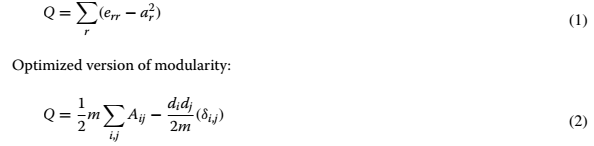
社区检测算法已经存在一段时间了，然而，随着今天网络大小的日益增长，如何在一个大的，拥有十亿个节点和边的图中发现小社区是一个主要的挑战。在线社交网络的图比如Facebook，Twitter等每天都在增长。这种增长包括大的和小的社区，例如那些少于100个成员的。在时间和处理开销方面，需要一种能够有效地发现大小社区的算法。此外，发现异常和社区离群值也证明有挑战性。一个关键的任务是在检测他们之后将对这些社区排序。比如，Facebook已经有超过20000个社区，并且在图中全体成员上计算有更多影响力的社区是很重要的。对一个社区而言更高的排名意味着积极和有影响力成员的概率更高。能够发现社区排名帮助我们分析网络图和关系，发现最合适的数据挖掘技术，预测信息流和理解公共情绪。

在这次研究中，检测和表征社区被讨论。优化吸引子算法(Optimized Attractor algorithm)，使用杰卡德距离(Jaccard distance)发现社区，和[吸引子算法](http://www.doc88.com/p-5367654660860.html)相似；然而对于更大或更小的图，该方法的性能不像吸引子算法那样变坏。此外，茅屋算法(a hovel algorithm)被介绍，它使用群落内比(the ratio of intra-community)和社区间的连接(inter-community links between links)为社区排序。不像大多数社区检测算法，优化吸引子算法基于一个临界值使社区大小规格化。优化吸引子的性能在已知的基准上被测量，胜过基于模块度的方法。

本文的剩余部分组织如下：“相关工作”部分对已存在的社区检测算法，从介算法(Betweenness algorithm)到最大模块度方法(modularity maximization approaches to)，进行了不可或缺的比较。“我们的方法”部分在讨论所需要的预备知识后介绍了提出的优化吸引子方法。“分析”部分描述了一个详尽的性能分析和优化吸引子的时间复杂度。对结果的讨论和未来预测在“讨论和未来工作”部分给出。“结论”部分概述结语。

**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:



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.

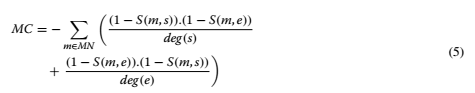
The attractor method introduced by Shao et al. [9] uses the concept of distance instead of similarity. In this method, the similarity of nodes have been calculated by Eq. 3 as you can see the Jaccard similarity [16] has been used. Next the value found will get subtracted from one in order to find the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

IMG_256

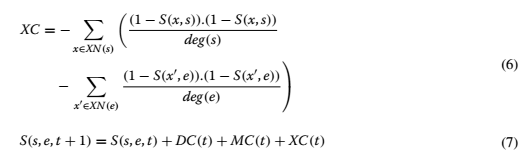
Next step is to find the initial distance for all the nodes in the graph. There are three types of relation ship between the nodes in this algorithm. The first one is when two nodes are connecting directly. The distance between to direct relationship is calculated through Eq. 4

IMG_256

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



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.



## 相关工作(Related work)

图中的社区是基于顶点之间的相似度被检测和定义的，即一个社区是一组有着最高相似度的节点。有各种各样的方法去计算节点间相似度的值,四个最有名的是：是杰卡德相似度(Jaccard similarity)，余弦相似度(Cosine similarity)，皮尔逊相关(Pearson correlation)和欧几里得度量(Euclidean distance)。在计算图中所有节点间的相似度之后，所有相似的节点将会被放到同一个社区，节点间不同的边将会被切割(dissimilar edges between nodes will be cut out)。然而，找到社区之后，仍需要找到这些社区间的排序。社区检测是一个有用的工具，可用于分析大的复杂的网络，比如社交媒体网络。为了发现网络社区各种各样的方法被使用。追溯到2002年，Girvan和Newman使用介算法(Betweenness algorithm)发现模块度改变值。它工作的方法是起初每个节点被放到它自己的社区。然后，模块度的值通过方程式被计算。紧接着，有着最高Q值的邻接节点被连接到指定的社区。被提到的这个方法的时间复杂度是N3。Newman在下一年介绍了一种新的方法，降低了时间复杂度，从对普通矩阵的最低值M2，到在稀疏矩阵上的N3，到稀疏矩阵上的((M+N)N)或者是N2(from the lowest value of M2 for normal matrices to N3 on sparse matrices to ((M+N)N) or N2 on sparse matrices)。贪心算法(Greedy algorithm)被Clauset和Newman在次年提出，并改进(improved)了后者方法的时间复杂度为Nlog2n。在2005年，外部优化方法(External Optimization)被Duch和Arenas介绍。在这个方法中，Duch和Arenas通过使用外部优化去优化模块度，牺牲操作速度达到了一个更好的准确度。这个算法的时间复杂度是N2log2n，比贪心算法更慢。模块度公式：

IMG_256

优化后的模块度公式：

IMG_256

另一方面，Shao和Han开发了一种基于距离动态(distance dynamic)叫做吸引子的新算法。吸引子算法的时间复杂度是被记为线性的E(which is counted as linear)。尽管它比标准化割(N-cut)，模块度，边介数(edge Betweenness)，贪心等快，它比Louvan提出的算法和Infomap和Metis有一点慢。Louvan方法开始每个节点是一个单独的社区；然后在每次迭代中，它计算两个社区间的相似度并将它们混合到一起。

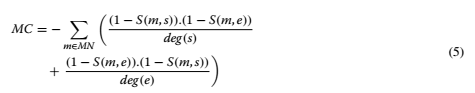
被Shao等人提出的吸引子算法使用距离的概念而不是相似度。在这个方法中，节点相似度通过公式被计算，正如你看到的，它使用了杰卡德相似度。为了找到两个节点间的距离，紧接着找到的值将从其中一个中减去(Next the value found will get subtracted from one in order to find the distance between two nodes)。

IMG_256

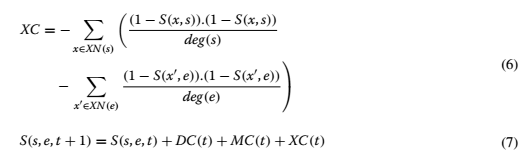
下一步是为图中所有节点找到初始距离。在这个算法中两个节点之间有三种关系类型。第一种是当两个节点直接连接。直接关系间的距离通过公式计算。

IMG_256

第二种类型被定义为两个相邻节点在他们的直接连接上的影响。其被视为积极影响。它被认为是积极地，来自于有共同兴趣的通常关系更强的事实(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.)。公式如下。



最后一种类型在方程式中展现。第三种影响方式来自独有邻居。具有不同的或相反的兴趣可能导致关系变得更远。因此第三种类型在开始和终止节点间的连接上通常有一个积极影响(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.)。



**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 *eb* . *nes* and *net* 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 *Rc*. 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

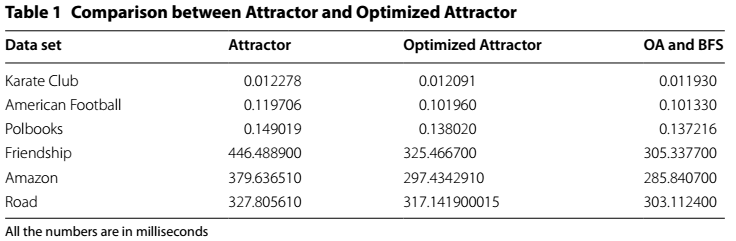
* A new algorithm called “Community Ranker” is introduced. As the name suggest, CR (Community Ranker) method finds a value for each community that was found by Algorithm 1. How communities have been ranked is provided in "Algorithm" section by Algorithm 2.

## 我们的方法(Our approach)

网络结构随着时间变化发展。在本文中，G代表一张图。每一个G包含节点集合N和边集合E。C被用来描述社区集合。连接两个社区的每一条边被叫做中间边(a between edge)并且由eb显示。nes和net是一条边的两个节点。它们分别被叫做源节点和目标节点。一个计数器被用来识别和统计只有一边在一个社区而另一边属于另一个社区的边的数目。另一个计数器被用来统计在不同的社区有节点的边的数目(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.)。后一个计数器用(γ)代表。每个社区中节点的数量用(η)表示。最后，每个社区的排名用Rc记录。计算社区排名的方法基于社区内(inside reach community)节点的数量和图中节点的总数量。在找到每个社区的中间边(between edge)的数量后，可以通过将该数除以图中除社区中已经存在的节点之外的所有节点来计算排名(the rank can be counted by dividing this number by all the nodes in the graph except for those already in the community.)。在本次研究中，使用吸引子算法的优化版本发现社区。正如在“相关工作”部分解释的那样，吸引子算法基于节点间的距离及其包容性(inclusive)和排他性(exclusive)邻居进行工作。然而，有人发现通过删除算法的最后一个循环，吸引子算法可以工作的更快。事实上，通过过滤距离矩阵(by filtering the distance matrix)社区可以在第三层循环找到。因此，无论哪两个节点之间的距离等于一，边应该从图中切出(wherever the distance between two node is equal to one the edge should be cut from the graph)。但是，这个动作可以应用于反转滤波函数(reversing the filtering function)，这意味着把距离等于零，我们实际上可以保持边(meaning taking the distances wherever it is equal to zero we could keep the edge in fact)。这项研究作了三大努力：

* 第一，在进行初始距离计算之前，广度优先搜索被添加到这个算法中。它帮助程序运行速度提高近10%（用于遍历和搜索树或图数据结构的算法）。它从树的根节点开始（或是图中某个任意节点，有时被称为‘搜索关键字’(sometimes referred to as a 'search key')），首先探索邻接节点，然后再移动到下一级邻居。
* 原始的吸引子算法通过在“算法”部分解释的方法被优化。使用这个优化方法，程序的运行速度快了将近五分之一(the program runs faster by a factor of almost one fifth)。（如表1）。

表1 吸引子和优化吸引子对比



所有的数字以毫秒为单位

* 一种称做“社区排名(Community Ranker)”的新算法被引入。顾名思义，CR(社区排名）方法为算法1找到的每个社区找到一个值。按“算法”部分的算法2提供社区的排名。

**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.

### 初步(Preliminaries)

社交网络是由社交媒体成员创建的图。这些图由社交媒体成员和他们之间的关系组成，分别以节点和边的形式表示。在Facdbook上的友谊是一个双向（无向）(a two-sided(undirected))连接的例子，跟随Instagram或tweeter上某人的页面是一个单向（有向）连接的例子，在[DBLP（Digital Bibliography & Library Project）](http://baike.baidu.com/link?url=RXNvr1yVRmuavW1Xuog1I-wMMJX6HeNqDSAmETIC1WEniPrNMiTBWf8LArDW5Vm6OAS6bOqNa2M62xTS9IQmvq)网络上的一个连接是一个加权边的例子。在社交网络上，人们之间的连接揭示了很多关于图和存在于图中的社区的信息。一个社区是一组有着共同兴趣的人。与每个社交媒体中的现实生活一样，存在各种各样的社区。这种社区由社交媒体成员和他们的页面组成。社区的例子可以是友谊团体或有着相同职业的一群人或只是有着共同爱好的一群人。在社交图中，社区内部的关系和社区外部的关系分别由内边(internal edges)和外边(external edges)表示。在一个社区内的内边数量较多，外边数量较少。相反，在两个不同的社区之间没有内边只有少量外边存在。在存在于社交媒体中的所有社区之间，有必要区分那些在图中活跃且更重要的社区。弄清活跃的社区可以帮助广告工作(advertisings tasks)，建议需要的统计和分析(statistics and analyses needed for suggestions)，搜索和趋势分析。另外，找到社区排名列表显示每个社区的重要性。有着更高排名的社区在社区内外有着更好的沟通。本地计算对大图有好处。社区检测方法分为两个子组。第一个众所周知的方法是基于它们的相似度来聚类顶点。第二个方法是基于稀疏切割(sparse cut)的图分割(graph partitioning)。

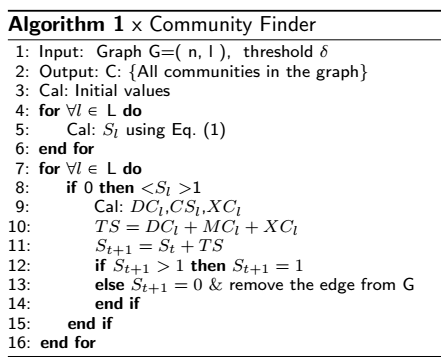
**Notations**

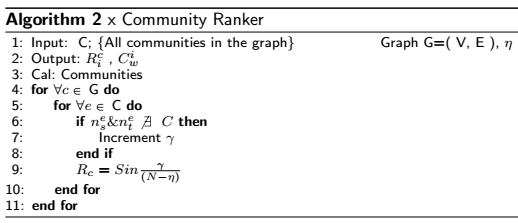
Adjacency matrix is a matrix of the size *n*×*n* where n is the number of nodes in the graph. It has ones whenever two nodes are connected and zero otherwise. Every connection between two nodes is defined as an edge. Edges can be directed or undirected. Directed edges are those that connect two nodes 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 *Rc* 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.

### 符号(Notations)

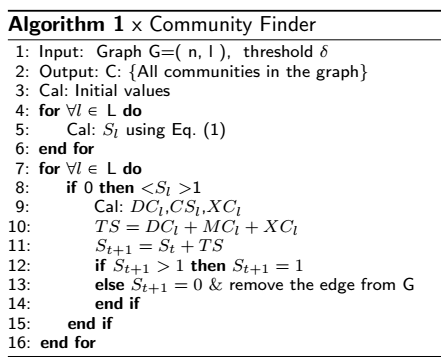
邻接矩阵是一个大小为n×n的矩阵，其中n表示图中节点的数量。当两个节点相连时值为1，否则为0(It has ones whenever two nodes are connected and zero otherwise)。两个节点间的每个连接被定义为一条边。边可以是有向的或无向的。有向边是仅以一种方式连接两个节点的边。无向边仅用一条边连接两个双向节点(Undirected edges connect two nodes two-sided only with one edge)。然而，为了显示节点的更多特性，比如每个节点和连接到该节点的边的重要性，为每条边定义权重。在本文中节点用n表示，节点集合用N表示。边和边集合分别用l和L表示。社区和社区集合分别用c和C表示。开始和终止节点用s代表，Rc代表社区c的排名。在每个社区中节点的数量用η代表。两个社区间边的数量用γ表示。S被用来保存相似度的值。相互(Mutual)和排它(exclusive)的邻居分别用m和x表示。最后，周边集(neighboring Set)用IMG_256表示。

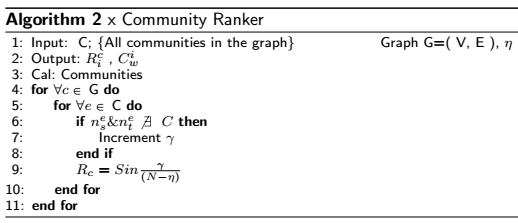
**Algorithm**





### 算法(Algorithm)

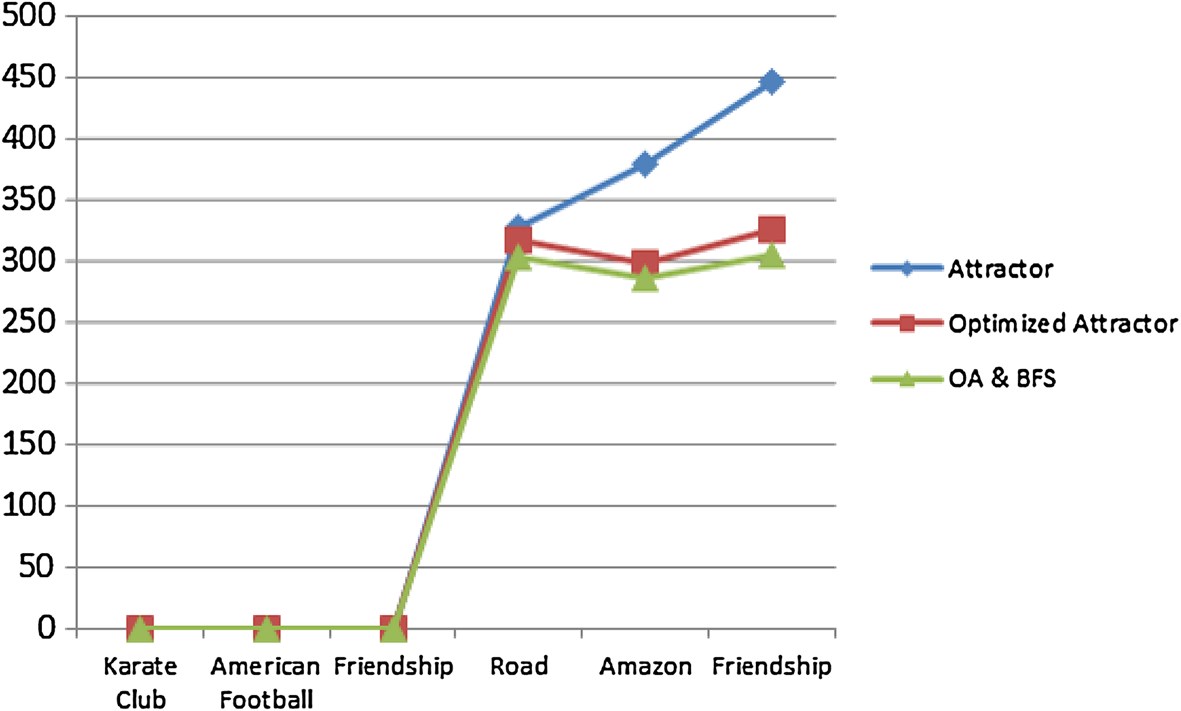




**Analysis**

**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

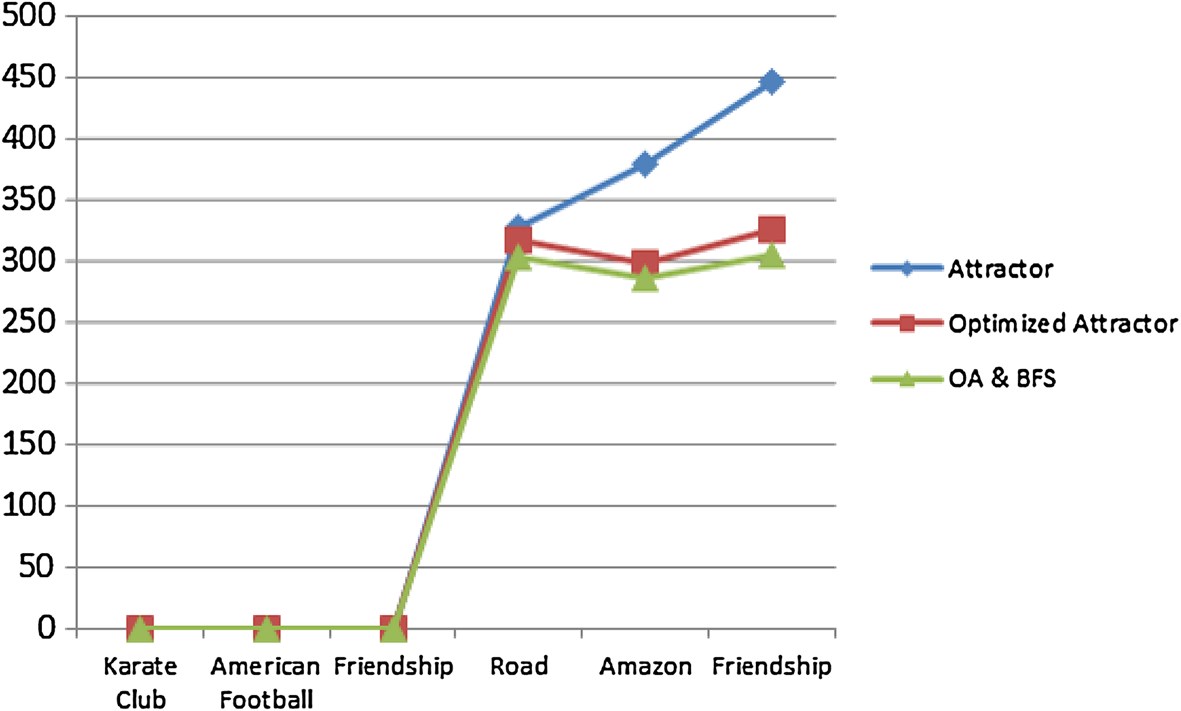
The reason for this phenomenon is that the process of finding communities was done using three loops in the original algorithm, which was performed over every graph edge. The Optimized Attractor reduces this to two loops; i.e. the algorithm is improve with a constant of |*E*| for every datasets with 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≤*T*≤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.

## 分析(Analysis)

### 时间复杂度分析(Time complexity analysis)

存在的吸引子算法的时间复杂度函数(The time complexity function)得到改进。要注意的是改进版本在不牺牲任何精度的情况下工作更快。总的来说，后者的线性时间复杂度(linear time complexity)是O(|E|+k|E|)，减少了(T|E|)的时间。即使吸引子和优化吸引子都有线性时间复杂度，优化吸引子通过一个O(T.|E|)的因子更快。正如在图1中所示，吸引子的优化版本随着节点和边数的数量增加执行速度更快。



**Fig. 1**

Comparison chart for Attractor, Optimized Attractor without BFS , and OA with BFS

这种现象的原因是使用原始算法中的三个循环完成了发现社区的过程，该循环是在每一个图边上执行的。优化吸引子将其减少到两个循环；即该算法对于每一个边数为|E|的数据集通过一个常数|E|进行改进。因此，吸引子时间并发(Attractor time complicity)由O(|E|+k|E|+T|E|)变为O(|E|+k|E|)；因此，有常数因子O(|E|)它运行的更快。注意K对于两个连接节点是排他邻居的平均数量(the average number of exclusive neighbors for two linked nodes)，T是一个常数且。在OA中，当第三层循环的所有计算在第二层循环中得到补偿，T被删除。这种方式，有可能取代拥有第三个循环，所有的过滤和标记都将在计算相似度的第二个循环中完成。

在第二个算法(社区排序)中，时间复杂度是O(|C|+|E|)。这个给出的时间来自于社区的数量和每个社区内存在的边的数量。CR的时间复杂度是O(C)，其中C是社区的数量。

**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 Opti** | **mized Attractor** | |  |
| **Data set** | | **Attractor** | **Modularity** |
| Karate Club | | 1 | 0.916128 |
| American Football | | 1 | 0.952536 |
| Polbooks | | 1 | 0.854031 |

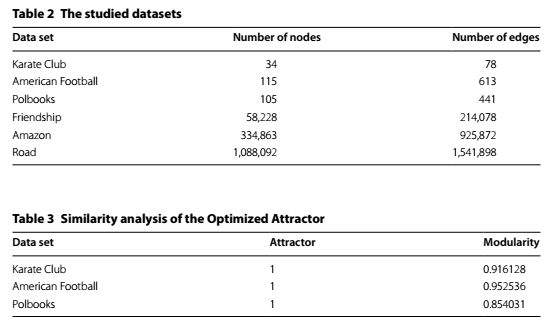
co-purchasing network, and ground-truth communities [25]. More details about datasets used in this study are given in Table 2.

### 实验分析(Experimental analysis)

该实验在具有Intel Xeon E5-1607 @ 3GHz处理器和16GB RAM的计算机上进行。使用单芯(a single core)运行算法。在计算吸引子或社区排序算法花费的时间之前，所有的节点和边被加载到主机内存中。所有提出的算法用Python程序语言实现，使用的库是networkx。为从图中抽出的Plots(for the Plots draw from the graphs)使用matplotlib库。

### 数据集(Data set)

为了能够说明优化吸引子算法的性能，在这个实验中，为社区检测选择并使用了最常用的数据集。使用的数据集都是现实世界中的数据集。我们使用了有名的空手道俱乐部(Karate Club)数据集，几乎所有的社区检测相关问题都使用了该数据集。本次研究使用的第二个常用数据集是拥有115个节点的美国足球(American football)数据集。使用的其它数据集是PolBooks-Krebs’ Amazon books,协作网络(Theory collaboration network)，Brightkite，Pennsylvania road network，Amazon product co-purchasing network，和ground-truth communities。关于用到的数据集的更多信息在表2中给出。



**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.

## 讨论和未来工作(Discussion and future work)

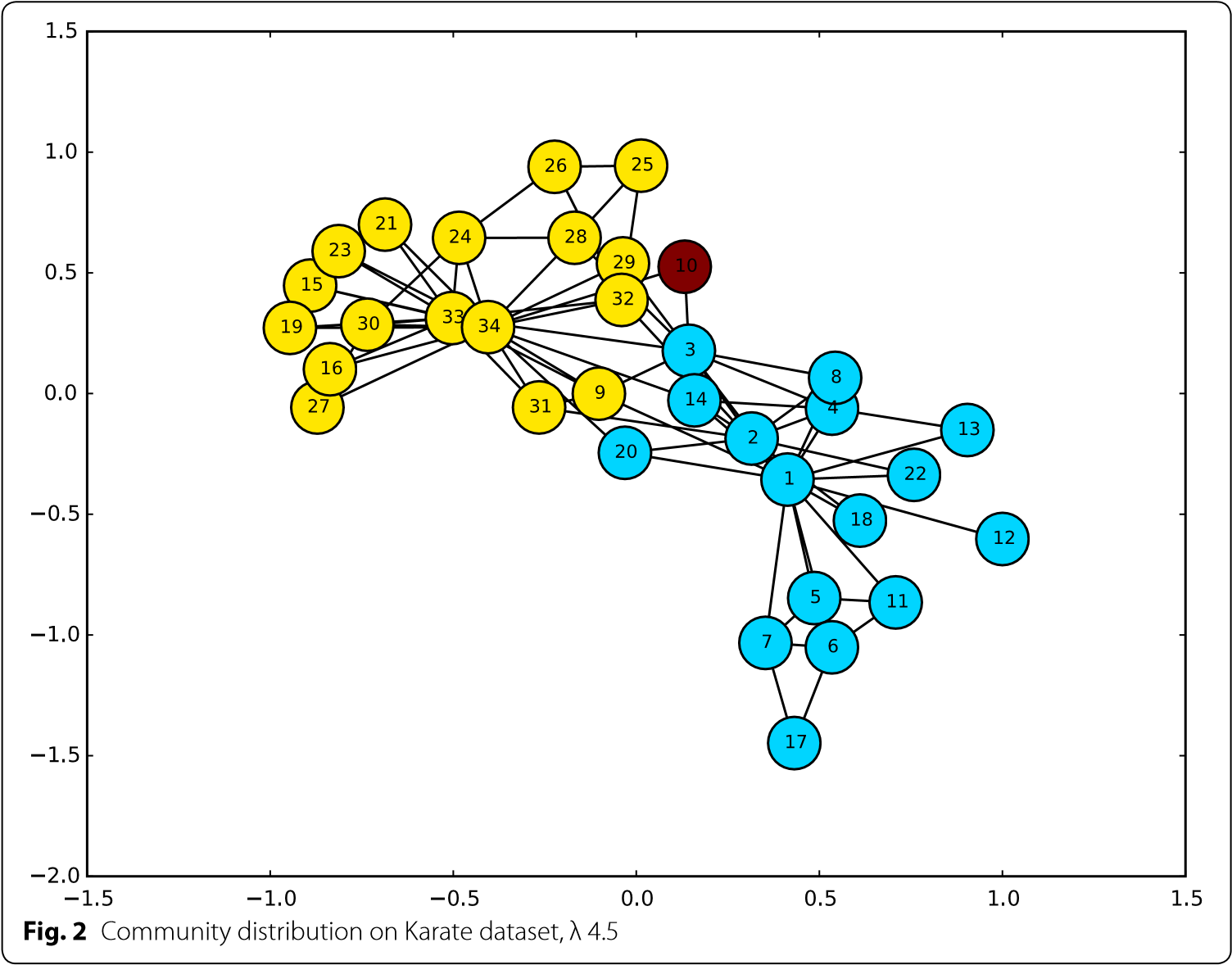
关于优化吸引子、吸引子和OA&BFS的时间性能的比较结果在表1中显示。优化版本表明它对Amazon数据集影响最大。人们可以看到比以前的结果快了将近25%的差异（one can see the difference is almost 25% faster than the previous result）。提到这一点，唯一关心的仍然是所提出的方法的准确性，见表3。不仅准确度没有降低，而且事实表明优化吸引子有一个更高的准确度。对于PolBooks数据集，模块化方法的准确度是0.85，优化吸引子的准确度是1。对于数据集Karate和Football优化吸引子的结果见图2和图3。社区分割用不同的颜色表示。

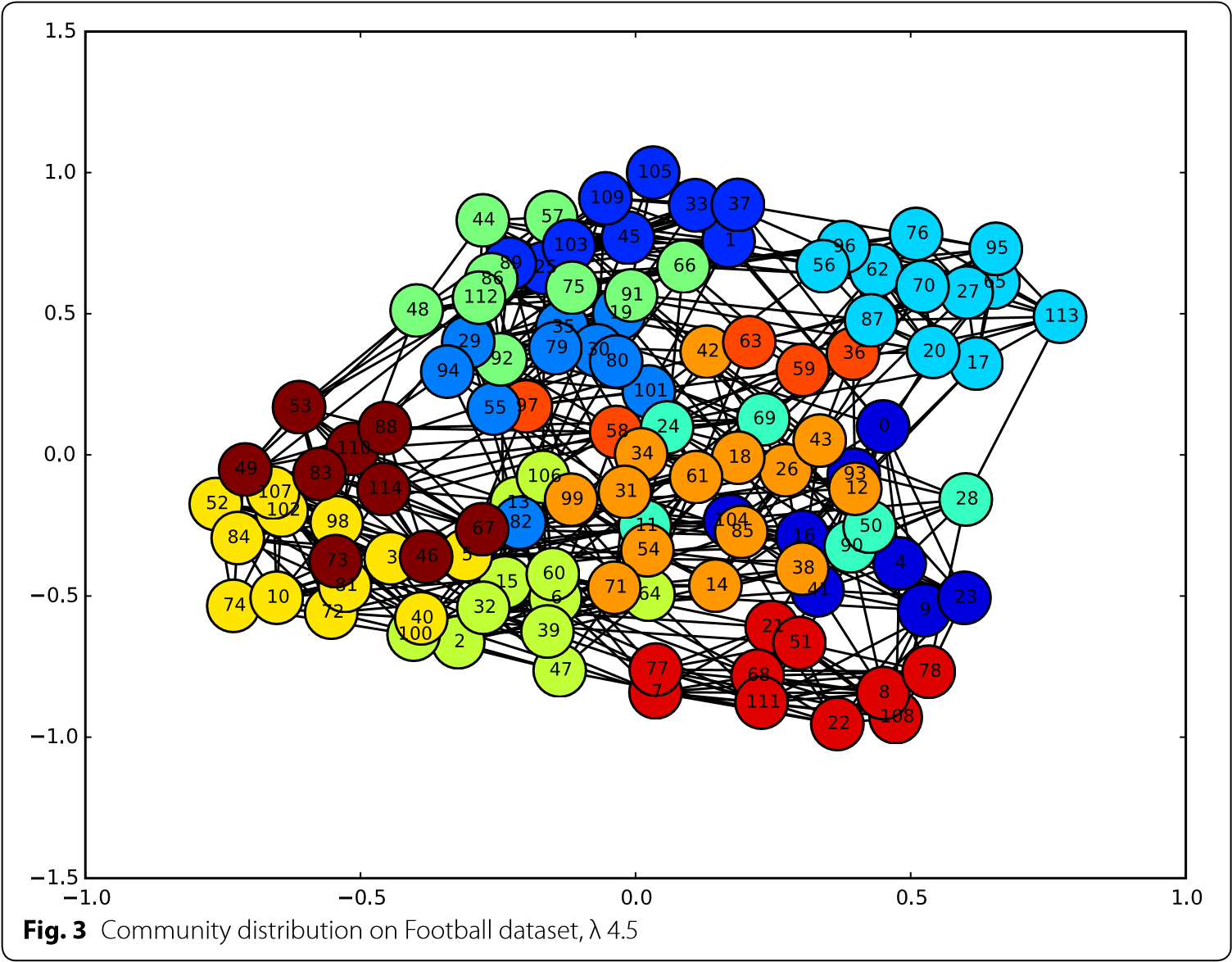
至于社区排序算法，如表4所示，对一个拥有超过一百万个节点的网络排序花费的时间少于1毫秒。对于更小的数据集，花费的时间几乎等于0。然而，应该考虑所有的社区已经被检测并加载到系统中的情况。表4所示的时间仅显示了排序系统的执行时间。

另外，前四大社区(for top four communities)Karate，Football，和PolBooks数据集的结果如图4所示。

**Conclusion**

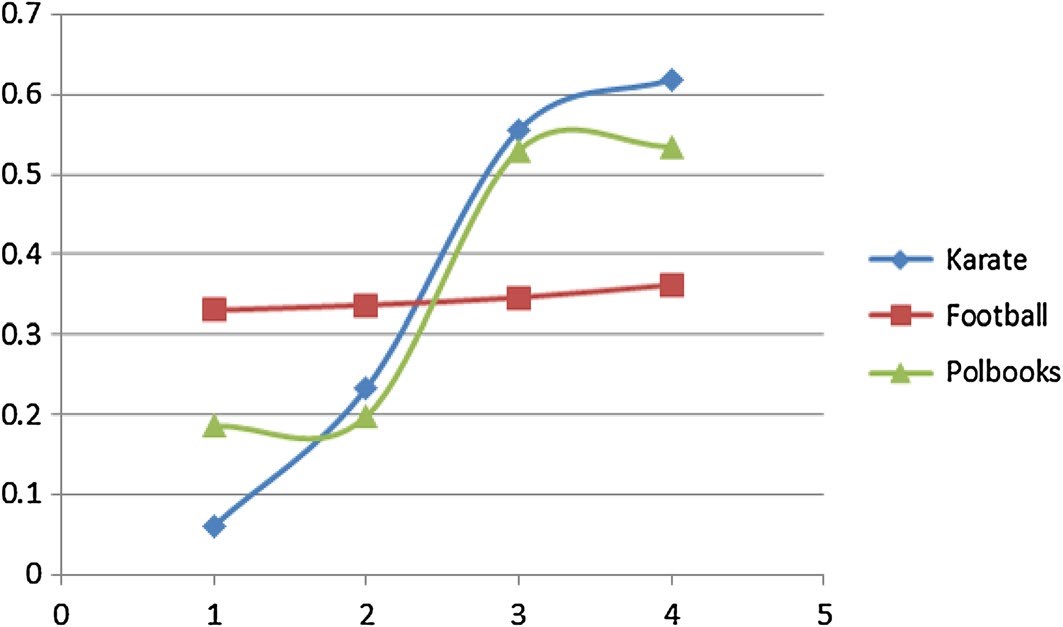
In this paper, we proposed a method achieving a better performance as shown in Table 5. As the networks specially social networks grow to more complex, the needs of dealing with the new sophisticated data grows. Community detection algorithm proposed in this paper improves the speed and performance of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





**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**

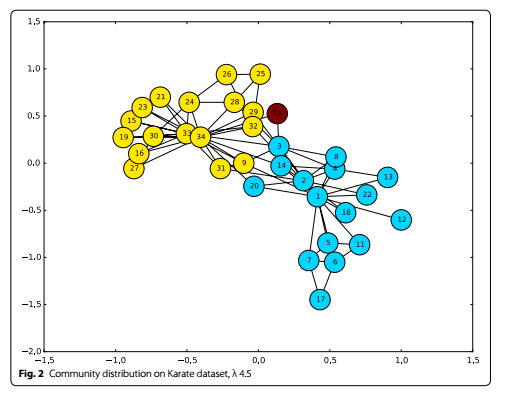
Ranking of top four communities, Lambda 4.5

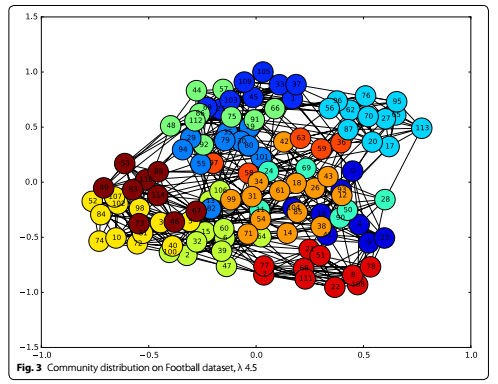
**Table 5 Notations and symbols**

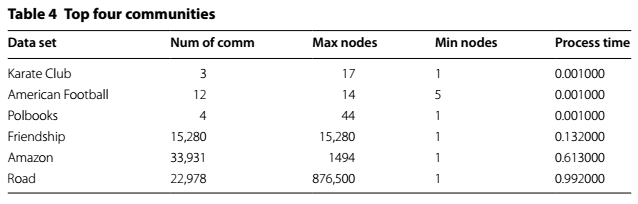
|  |  |
| --- | --- |
| **Notation** | **Definition and description** |
| G | Given graph |
| N, n | Set of nodes, node |
| L, *l* | Set of links, link |
| C, c | Set of community, community |
| η | Number of nodes in each C |
| *Rc* | Rank of community c |
| *eb* | Designated edges between two C |
|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

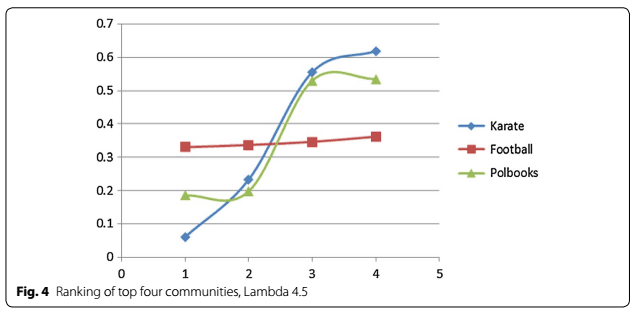
## 结论(Conclusion)

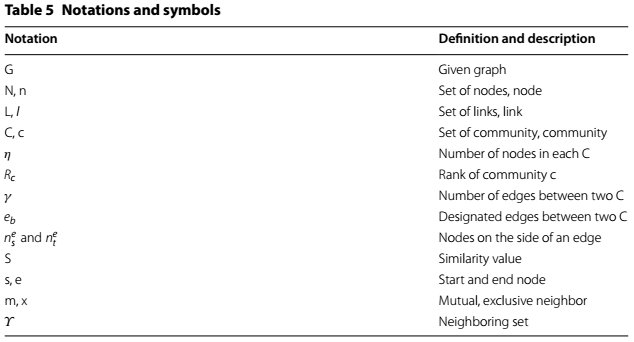
在本文中，我们提出了一种实现了更好性能的方法，如表5所示。随着网络，尤其是社交网络变得更复杂，处理新的复杂的数据的需求也随之增长。本文中提出的社区检测算法提高了发现社区的速度和性能。











**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

### 研究成果总结(Summary of findings)

我们提出了一个社区排序系统，其中速率社区基于对其他网络的影响(in which rate communities based on their influence on the rest of network)。随着每个社区寻找顶级社区(top communities)和顶级节点(top nodes)的兴趣的增长，各种各样的方法被发展去发现排序模式。我们的算法使用中心性(centrality)的概念去找到每个数据集中的顶级社区。在这个方法中，社区内每个节点的后代的数量(the number of intra community offspring of each node)被统计，并和数据集中的节点总数比较，然后应用归一化(normalization)的方法。结果，所提出的算法能够找到如何拓扑重要节点（对外部社区的更多后代表现出更大的影响力）(the proposed algorithm can find how topology important a node can be (more offspring to outside communities shows more influence))。

### 未来工作(Future work)

我们希望我们的结果将激励更多的对社区排序系统的研究。而且发现不仅基于节点关系，而且还涉及诸如评论，喜欢或跟随的节点之间的内容基础关系。发现社区的速度和性能在本次工作中得到提高，但是仍有必要去提高相关问题的准确度。另外，所提出的算法能够找到作为一个个体，但是有很多节点在不同的社区之间是共同的，并且不能被指定为特定社区的成员的社区。因此仍有必有发现共同的部分及重叠社区。

作者的贡献(Authors’ contributions)

MP，作为第一作者，进行了初步文献综述数据收集和实验，并起草了稿件。JZ和ST与MP合作开发算法，论文和框架。所有的作者阅读并批准最后的稿件。

作者细节(Author details)

美国内华达州拉斯维加斯马里兰州Pkwy计算机科学系，UNLV。 美国内华达州马里兰州皮克维电气与计算机工程系UNLV。

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

## 参考文献(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](http://dx.doi.org/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](http://dx.doi.org/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](http://dx.doi.org/10.1103/PhysRevE.70.066111).

1. Duch J, Arenas A. Community detection in complex networks using extremal optimization. Phys Rev E. 2005;72:027104.
2. 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.
3. Shi J, Malik J. Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell. 2000;22(8):888–905.
4. 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.
5. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. J Stat Mech Theory Exp. 2008;2008(10):P10008.
6. 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
7. Karypis George, Kumar Vipin. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput. 1998;20(1):359392.
8. Hubert L, Arabie P. Comparing partitions. J Classif. 1985;2(1):193–218.
9. 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](http://dx.doi.org/10.1109/TKDE.2015.2432815).
10. 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>.
11. Networkx. <https://networkx.github.io/>. Accessed 27 Sep 2016.
12. Zachary’s Karate Club. <https://networkdata.ics.uci.edu/data.php?id=105>[d](http://www-personal.umich.edu/%7emejn/netdata). Accessed 27 Sep 2016.[ata](http://www-personal.umich.edu/%7emejn/netdata). Accessed 27 Sep 2016.
13. 20. American College Foo[tball.](https://networkdata.ics.uci.edu/data.php?id=105) [http://www-personal.umich.edu/~mejn/net](http://www-personal.umich.edu/%7emejn/netdata)
14. PolBooks—Krebs’ Amazon books. <http://vlado.fmf.uni-lj.si/pub/networks/data/mix/mixed.htm>. Accessed 27 Sep 2016.
15. High energy physics—theory collaboration network. [https://snap.stanford.edu/data/ca-HepTh.html.](https://snap.stanford.edu/data/ca-HepTh.html) Accessed 27 Sep 2016.
16. Brightkite. <https://snap.stanford.edu/data/loc-brightkite.html>. Accessed 27 Sep 2016.
17. Pennsylvania road network. <https://snap.stanford.edu/data/roadNet-PA.html>. Accessed 27 Sep 2016.

Amazon product co-purchasing network and ground-truth communities.

1. Amazon product co-purchasing network and ground-truth communities.[https://snap.stanford.edu/data/comAmazon.html](https://snap.stanford.edu/data/com-Amazon.html). Access 27 Sep 2016.

