THE NATIONAL UNIVERSITY
of SINGAPORE

School of Computing
Computing 1, 13 Computing Drive, Singapore 117417

TRB5/13

Force-directed Layout Community Detection

Yi Song and Stéphane Bressan

May 2013

T e c h n i c a l R e p o r t

Foreword

This technical report contains a research paper, development or tutorial article, which has been submitted for publication in a journal or for consideration by the commissioning organization. The report represents the ideas of its author, and should not be taken as the official views of the School or the University. Any discussion of the content of the report should be sent to the author, at the address shown on the cover.

OOI Beng Chin
Dean of School

Force-directed Layout Community Detection

Yi Song and Stéphane Bressan

School of Computing
National University of Singapore
{songyi, steph}@nus.edu.sg

Abstract. In this paper, we propose a graph-layout based method for detecting communities in networks. We first project the graph onto a Euclidean space using Fruchterman-Reingold algorithm, a force-based graph drawing algorithm. We then cluster the vertices according to their Euclidean distance. The idea is similar to that of dimension reduction. The graph drawing in two or more dimension provides a heuristic decision as whether vertices are connected by a short path based on their Euclidean distance. We study community detection for both disjoint and overlapping communities. For the case of disjoint communities, we use k-means clustering. For the case of overlapping communities, we use fuzzy-c means algorithm.
We evaluate the performance of our different algorithms for varying parameters and number of iterations. We compare the results to several state of the art community detection algorithms, each of which clusters the graph directly or indirectly according to geodesic distance. We show that, for non-trivially small graphs, our method is both effective and efficient. We measure effectiveness using modularity when the communities are not known in advance and precision when the communities are known in advance. We measure efficiency with running time. The running time of our algorithms can be controlled by the number of iterations of the Fruchterman-Reingold algorithm.

1 Introduction

Communities happen in complex networks from various fields and community detection can be applied to those fields such as sociology [16], biology [14], marketing [35] and computer science [12]. Communities exist when nodes in the network form a group in which they are better connected to each other than to the rest of the network. Identifying such groups or communities, e.g in online social networks, helps to disseminate or retrieve information in a easier and more efficient ways. In this paper, we propose methods for finding good communities.

We model social networks as simple graph $G(V, E)$ which is undirected, unweighted and without self-loop. V is a set of vertices. E is a set of edges. Our main idea is to obtain presentation in Euclidean space from graph presentation and then work on clustering based on Euclidean distances. This is different from what common graph clustering algorithms in that most of them cluster the

graph and detect communities directly or indirectly according to geodesic distance. We build our approach based on Fruchterman-Reingold's force-directed algorithm (*FR*) [18]. This graph layout approach transforms the connections among vertices to attractive forces and repulsive forces to pull vertices together or push them apart. In this way, the vertices with more connections are placed closer while the vertices without or with less connections are relatively further from each other in Euclidean space. This gives the hint that the vertices within one community are placed relatively closer since it's denser inside the community. In other words vertices within community have more connections to each other than connections to the vertices in other communities. It thus is a good opportunity to adopt the techniques in data clustering to look for the communities based on the graph layout. Furthermore, we extend FR from two dimension to one dimension and three dimensions, and even higher dimensions as well. We evaluate the significance of the number of dimensions on our method's effectiveness and efficiency. For disjoint community detection, the data clustering techniques we take advantage of are *k-means clustering (KM)*, while for overlapping community, we employ *Fuzzy C-mean clustering (FCM)* which can indicate the strength between each vertex and communities, and thus doesn't restrict each vertex to belong to one group only. *FCM* is a variant of *KM*. All these algorithms' complexities are not high and neither is *FR*'s. Our method building on these techniques is thus efficient for large social networks.

To evaluate the community, modularity is widely used. Modularity is defined based on this idea that edges between nodes in the same community are dense, and are sparse between different communities. As the size of the community is usually unknown, simply counting the edges inside or between communities is not adorable. To find communities with natural division, modularity is invented as the number of edges falling within groups minus the expected number in an equivalent graph with edges placed at random [32]. An equivalent graph here means that the graph has the same number of edges and the same degree distribution. In this paper, we apply modularity as the metric of community quality. For graphs with known community structures in advance, we measure the precision as well by comparing the memberships of communities that our approach discovers with those of the communities known in advance.

The rest of the paper is organized as follows. In Section 2 we introduce the related works on graph clustering and community detection. Section 3 briefly reviews the background and presents our approach to discovery community structures in networks. In Section 5 we describe the data sets that we use, and present and analyze the results of our experiments. Finally we conclude in Section 6.

2 Related Work

Community detection is sometimes referred to as graph clustering, though the two concepts are not exactly the same. Many works have been using graph clustering to detect communities in complex networks including social networks.

Graph clustering methods can be categorized into partition clustering, hierarchical clustering, divisive global clustering, and agglomerative global clustering

[38]. Large amount of specific methods are proposed such as Star clustering [2], Repeated Random Walks [29], Markov Clustering [40], *Richochet* clustering [42].

Some methods are specifically proposed for disjoint community detection. Girvan and Newman [20][31], the pioneering works on community detection, propose a divisive method to identify community. The edges with highest betweenness are removed iteratively, which splits the graph into communities. The quality of the obtained communities are measured by *modularity*. Clauset [7] defines a local measurement of community structure called *locally modularity* and proposes an agglomerative algorithm to maximize the *local modularity* of the communities detected. Clauset et al. [8] propose a greedy hierarchical agglomerative algorithm which starts from each vertex being a community and then joins two communities at each iteration. The whole process can be represented by a dendrogram. The selection of two joining community is based on the idea of maximizing modularity increment. Yan and Gregory [44] propose to optimize current community detection algorithms by comparing vertices. Pairwise vertex similarities are measured beforehand, and existing community detection algorithms are applied on the graph with the vertex similarities as edge weights. To evaluate communities found, they use modularity and normalized mutual information measurement [24]. Rosvall and Bergstrom [36] use information theoretic approach to detect community in weighted and directed network.

For overlapping community detection, Du et al. [13] use maximal cliques for community detecting instead of star-shaped subgraphs. The problem of star clustering, as they point out, is that the vertices with high degree doesn't necessarily mean involving in a community. Therefore, they propose an algorithm called *ComTector*. It enumerates all maximal cliques, finds clustering kernel in each group of the overlapping maximal cliques, assigns the rest vertices to closest kernels, and then merges fractional communities. Works such as [33] [22] and [23] are based on Newman's modulairty. Baumes et al. [3][4] propose two heuristics to detect locally dense subgraphs as communities. Two different subgraphs with significant overlap can be both locally optimal and thus they are overlapping communities. The first heuristic tries to find disjoined clusters by deleting high-ranking vertices and then adds the deleted vertices to one or more clusters. The second one starts from randomly chosen seeds and then adds or deletes one vertex at a time till the density metric cannot be improved further. Goldberg et al. [21] propose an additional requirement which requires the community to be a connected sub-graph. They modify Baumes' *iterative scan* algorithm [3] to be able to examine the connectivity of the cluster found, and they manage the large number of overlapping communities by post processing the results and merging the most similar communities. Palla et al. [19] propose the clique percolation method(CMP) which finds all cliques of size k and communities are detected by finding connected union of k-cliques. It 's based on an assumption that the network must have a large quantity of similar cliques. Chen et al. [6] detect overlapping communities utilizing concept from game theory. Lancichinetti et al. [24] and Michele et al. [9] locate communities locally. Lancichinetti et al. [26], considering various networks features, present a method called Order Statisti-

cal Local Optimization Method. It can be applied to weighted, directed graphs besides simple graphs. The hierarchical structures can be detected. Zhang et al. [45] propose a method that combines spectral mapping, fuzzy clustering and the optimization of a quality function. Ahn et al. [1] define clusters as sets of edges. They propose to group edges with an agglomerative hierarchical clustering technique. Jierui and Boleslaw [43] propose Speaker-listener Label Propagation Algorithm for overlapping community detection in large-scale networks. Some other works solve the problem from others perspectives such as [46][37].

3 Algorithm

3.1 Background

In this paper, we adopt the fundamental clustering techniques and force-directed notions. We introduce the basis before we introduce our algorithm.

force-directed algorithms The idea of force-directed algorithms is to achieve a "aesthetically pleasing" graph layout by simulating the whole graph as a physical system. Edges in the graph are replaced with springs and apply forces to vertices. Vertices are pulled closer together or pushed further apart. The positions of the vertices are adjusted and this procedure continues until the the system comes to an equilibrium state. Especially, Fruchterman and Reingold's force-directed algorithm [18] aims at even vertex distribution as well. They define attractive force and repulsive force as

$$f_a(d) = d^2/k$$
$$f_r(d) = -k^2/d$$

where $k = C\sqrt{\frac{area}{number\_of\_vertices}}$, and d is the distance between every pair of vertices. $area$ is the windows size for display the graph. In addition to the iteratively replacements of the vertices, Fruchterman and Reingold also add the technique of temperature control to control the degree of replacement. This practice is a special case of simulated annealing.

***k*-clustering** K-means clustering [28] is one of the state-of-the-art clustering methods in data mining. It partitions objects to k clustering, assign each object the cluster with the nearest mean and adjust their membership till optimum is achieved. Given a set of objects $(x_1, x_2, ..., x_n)$, k-means clustering algorithm allocates objects to k clusters such that the within-cluster sum of squares is minimized. The within-cluster sum of squares is defined as:

$$\sum_{i=1}^{c} \sum_{k \in C_i} ||x_k - \mu_i||_2$$

where C_i is the i-th cluster which contains a set of objects, and μ_i is the mean for cluster i: $\mu_i = \frac{\sum_{k=1}^{N_i} x_k}{N_i}, x_k \in C_i, N_i = |C_i|$

As a soft version of k-means, Fuzzy C-means clustering (FCM) [5] assigns each object a fuzzy degree of belonging to each cluster. Instead of belonging to only one cluster, objects classified via this algorithm can belong to several clusters with different strengths. Similar to k-means algorithm, FCM classifies the nodes to clusters to minimized the objective function:

$$\sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik})^m||x_k-\mu_i||_C^2$$

where $\mu_i$is the center of the i-th cluster, and u_{ik} is a function defined as:

$$\frac{1}{\sum_{j=1}^{c}(d(c_k,x)/d(c_j,x))^{\frac{2}{m-1}}}$$

where $d()$ is the distance function and m is a fuzzifier.

As a general version of k-means, Expectation-maximization algorithm (EM) [11] models clusters using statistic distributions. Starting with an initial guess for the parameters of the cluster probabilities, the mean and covariance of the objects in the cluster, EM iterates an expectation step (*E-step*) to compute the probability of the objects being drawn from the distributions of classes, and a maximization step (*M-step*) to re-compute the parameters until convergence. The reason we adopt k-means, rather than EM is that k-means is effective enough for this problem and k-means is more efficient. We will experimentally show this in Section 5.

3.2 Algorithm

We propose an algorithm that can systematically enumerating all possible number of clusters for the solution and find the one with which the clustering has the highest modularity. Therefore, theoretically the algorithm iterates by changing the value of k from 1 to $|V|$ which is the number of vertices in the network. We will show the changes of modularity with the change of k value. If the number of clusters is prior knowledge, we can set the number of iterations to be 1 to directly cluster the graph with the specific number of communities.

Our method starts from the FR algorithm which transforms the graph into the Euclidean space and places the vertices in the "right" place before clustering. The inputs for the algorithm are the edges of the graph only. Output is the coordinates of vertices in Euclidean Space. Then we sort the degrees of the vertices and initialize the centers of the clusters for the clustering by the vertices with highest degrees. The idea is that the vertices with high degree have higher chance of being the community centers. This is similar to the idea of star clustering, but different in that it's not determinant here. The centers may change during the clustering. We refine the clusters after the data clustering in Euclidean space. If there's any vertex that doesn't have any connection with other vertices in the same cluster, or it has less connections inside its cluster than outside its cluster, then it will be grouped to the cluster where it has the maximum number of connections. In other words, this vertex will be grouped to the cluster that has

most immediate neighbors. The refinement process may change the number of clusters, which is actually good for those who only roughly know the number of clusters. They can input the maximum number of clusters they believe and let our method find out the exact number of clusters in the network without trying all the value of k from 1 to $|V|$.

Algorithm 1: Force-directed Layout Community Detection Algorithm

Input: graph G with n vertices, the number of trials t, $t \leq n$.;
Result: Clusters C_i, $i \in (1, 2, ..., k')$

$v = Fruchterman\_Reingold(G)$, $v \in R^{n*2}$, v =[v_1;v_2;...;v_n];
Sort_degree(G);
$k \leftarrow 1$;
for *each* $k \leq t$ **do**
    $C'_i = K$-means(v);
    $C_i = Refinement(C'_i)$;
    Calculate modularity and record the maximum;
end
Return C_i,$i \in (1, 2, ..., k')$ with the maximum modularity;

Algorithm 2: Refinement

Input: Clusters C_i, $i \in (1, 2, ..., k)$;
Result: Clusters C'_i, $i \in (1, 2, ..., k')$;

for *i from 1* **to** k **do**
    for $v \in C_i$ **do**
        find the cluster C_j where v has the maximum number of immediate neighbors;
        if $i \neq j$ **then**
            Cluster v into C_j;
        end
    end
end
Return C'_i,$i \in (1, 2, ..., k')$;

We name the above algorithm *FR-KM* for the experiments. The other two versions of the algorithm are similar to *FR-KM* but depend on different clustering methods. Thus we name the one using expectation-maximization algorithm *FR-EM* and the one using fuzzy c-means algorithm *FR-FCM*. For *FR-FCM*, there's no refinement of the memberships for the vertices, since we intend to deal with overlapping community.

The modularity we use is the same as [31], defined as

$$modularity = \frac{1}{2m} \Sigma_{i,j \in V} (A_{ij} - \frac{k_i k_j}{2m}) \delta(c_i, c_j)$$

where $A_{ij} = 1$ if i and j are connected, otherwise $A_{ij} = 0$, and $\delta(c_i, c_j) = 1$ if i and j belong to the same cluster, otherwise $\delta(c_i, c_j) = 0$.

4 Experiment

We conduct experiments on both synthetic graphs and real world data including two benchmark graphs for community detection algorithm. The experiments ran on an Inter Core, 2 Quad CPU, 2.83GHz, 2GB machine running Windows 8 OS. The algorithms are implemented in C.

4.1 Data Description

We use a batch of benchmark graphs [25] to evaluate the effectiveness of our method. These benchmark graphs are generated with known community structure including number of vertices, the average degree, maximum degree, minimum and maximum size of micro and macro community due to the hierarchical structure, the number of overlapping vertices and fraction of edges between vertices belonging to same/different communities. In our experiments, we generate graphs with 2000 vertices and different average degrees while the other parameter are the same. They have no overlapping communities.

The real-world benchmark graphs we use are Zachary's Karate Club data and American College Football data. They are widely used for evaluating community detection algorithms. We also test on the Email-URV data set, Wikipedia data set, and Facebook data set. They represent large online social network data.

Karate Club data is a social network of karate club members studied by the sociologist Wayne Zachary. The network has 34 members (vertices) and they separated into two different groups due to a controversy between one of the instructors and administrator of the club.

American College Football data is a network with 115 teams which are separated into 12 conferences. Each team represents a vertex and an edge exists between a pair of nodes if there is match between two teams. Community structural property exists in this network because there are more games happen among teams within the same conference rather than teams from different conferences.

Email-URV data, collected by Guimer et al. [10], contains user-to-user (address- to-address) links from the network of e-mail interchanges among faculty and graduate students at Rovira i Virgili University of Tarragona, Spain. It's available on Alex Arenas Website [15]. It has 1,133 vertices and 5,451 edges.

Wikipedia data, collected by Leskove et al. [27], contains user-to-user (who-vote- whom) links from Wikipedia network. It's available on SNAP website [39]. The graph is directed. It has 7,066 vertices and 100,736 edges. Each vertex represents a user. An edge is created from a user to a candidate if a user votes for Wikipedia admin candidates.

Facebook data, collected by Viswa-nath et al. [41], contains user-to-user links from Facebook New Orleans networks. It's available on *MPI* [30]. The graph has 90,269 vertices and 3,646,662 edges. Each vertex represents a user. An edge exists if two users are friends.

4.2 Analysis of non-overlapping community detection

We compare our method with the algorithm of Girvan and Newman(*GN*) [20][31], one of the state-of-the-art algorithms in community detection. Modularity is first proposed in this algorithm. We also compare our method with *Walktrap* algorithm [34] and *InfoMap* algorithm [36], which has been shown to perform quite well for community detection [17]. *Walktrap* is based on random walk while *InfoMap* is based on information theory.

Table 1 shows the performance of the algorithms. In this comparison, we use the normal two dimension *FR* algorithm with its iteration equal 400 for KarateClub and AmericanFootball data and 1000 for EmailURV data, and the number of trials is set to 30. For all three graphs, our method produces partitions with highest modularity among the four algorithms. Although *Walktrap* and *InfoMap* are faster than our method and *GN* is faster than our method for smaller graphs, the running time of our method is still tolerable. As the size of graph becomes larger, our method becomes faster. If the number of clusters is known in advance, then the number of trial is 1 instead of 30 that we set. If so, our method takes much less time. *GN* is much slower for larger graphs. For the other two real-world data sets, Wiki-Vote and Facebook, we are unable to make the comparison due to *GN*'s scalability, but we will show the running time of clustering these two graphs by our method.

Table 1. Performance Comparison between *FR-EM*,*FR-KM* and *GN*

	KarateClub		AmericanFootball		EmailURV	
	modularity	running time	modularity	running time	modularity	running time
GN	0.4013	0.016	0.5976	1.014	0.5323	3193.532
Walktrap	0.3944	0.0000001	0.6015	0.015	0.5250	0.92
InfoMap	0.402038	0.015	0.599176	0.047000	0.521420	5.912000
FR-KM	0.417406	0.020000	0.601731	2.179000	0.542659	15.388000

Figure 1 shows the performance comparison between multiple dimension *FR-KM* and *GN*. We extend the normal two dimension FR algorithm to one dimension and three, four, five dimensions. We set the number of trials 30. For karate club data, the number of trial is equal to its number of vertices 34. We run each *FR-KM* with the number of iterations of *FR* changing from 100 to 2000 with interval 100. As we can see, the larger the number of iterations of *FR-KM*, the longer time it takes. However, the number of iterations of *FR* doesn't have decisive influence on the modularity. This suggests that there is no need to increase the number of iterations to get higher modularity. In terms of dimension, we find that for small graphs, projecting them to one dimension or three dimension may get higher modularity sometimes, but for large graphs, the two dimension *FR-KM* performs best. It's faster and clusters graph with higher modularity. That is why we shall adopt the normal two dimension *FR* in our algorithm when it comes to large graphs. Comparing *FR-KM* with *GN*, *FR-KM* outperforms in both effectiveness and efficiency with large graphs.

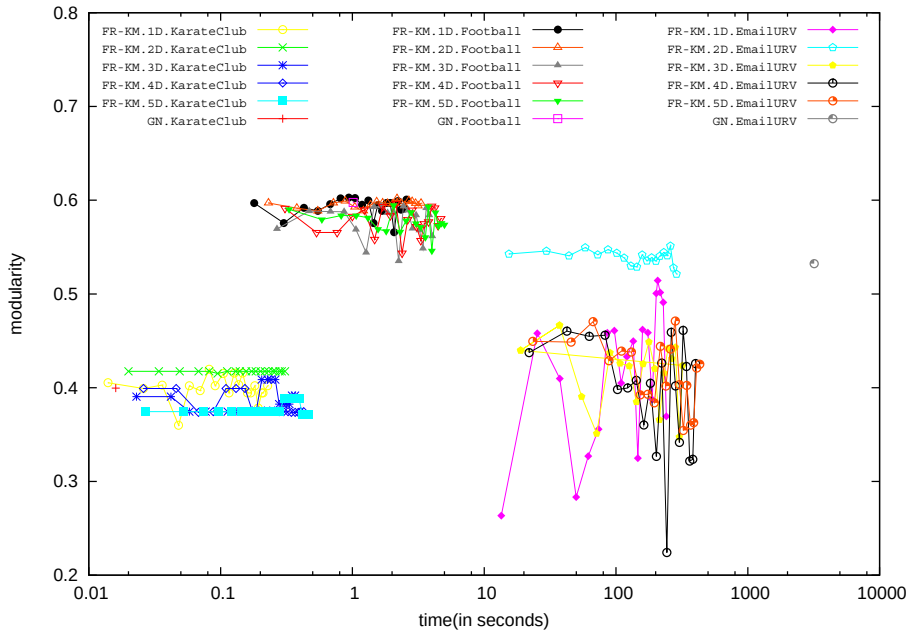
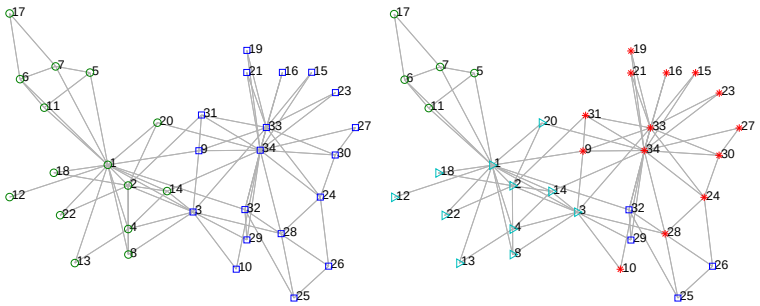


Fig. 1. Performance Comparison between multiple dimension *FR-KM* and *GN*

Figure 2(a) shows the communities when the Zachary's Karate Club network is divided into two clusters. The divergence of classification among many community detection algorithms often happens on the classification of vertices 3, 9, 10, 14 and 31 [24]. We believe our partition is reasonable because the vertices in controversy classified in this way have more inner group connections.

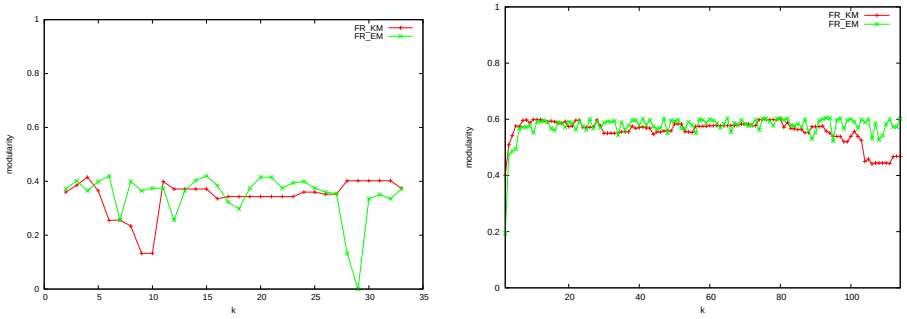
In Figure 2 (b) the graph is clustered into 4 groups. It is shown in Figure 3 (a) that in this case, the modularity is the highest. It indicates that some members in a group may better connect with some of the members than other members in the same group. Chen et al.'s algorithm [6] clusters this data into 6 groups. However, the small group phenomena is unverifiable from the existing information.



(a) k=2 (b) k=4

Fig. 2. Clustering of Zachary's Karate Club data

Figure 3 (b) shows the modularity for American Football data when the initial input number of clusters, k, is different. The final number of clusters may be different from the values of k on X-axis here. Our method changes the number of clusters during cluster refinements, which produces local optimum number of clusters. Therefore, we can see from the result that the trend of the line is horizontal in general. This suggests that even without knowing the number of clusters beforehand, we can find a local optimum around initial k value. This local maximum is probably the global optimum or close to the global optimum.



(a) Karate Club data (b) Football data

Fig. 3. Modularity for varying number of clusters

Figure 4 shows the 11 clusters found from American Football network data. This partition has the highest modularity among all the partitions with different number of clusters. Though in fact there are 12 teams in this network, which indicates 12 clusters, our method only misses one of them.

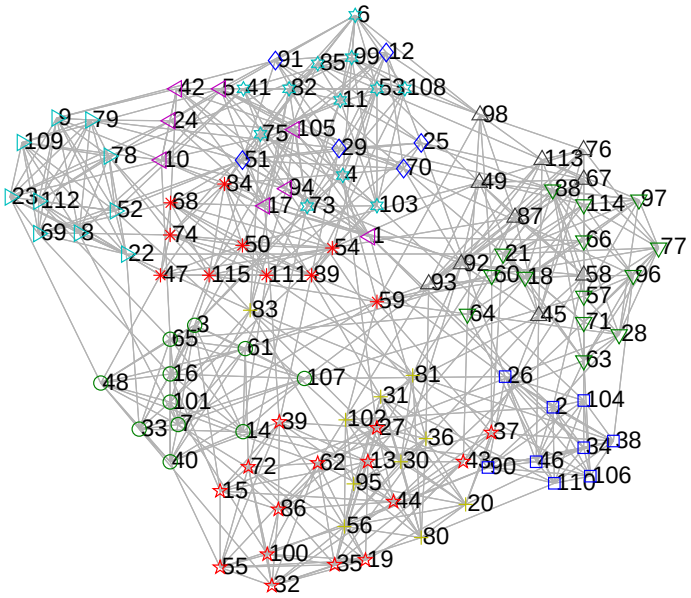
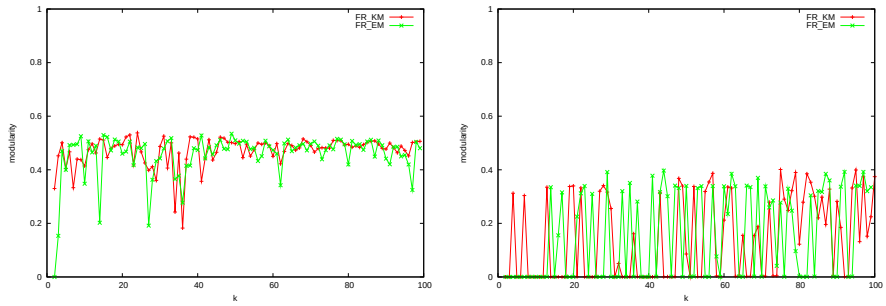


Fig. 4. American College Football is partitioned into 11 clusters

Clustering results of Zachary's karate club and American College Football illustrate the effectiveness of our method. Clustering results of the following three large social networks illustrate the efficiency of our method.



(a) Email-URV network (b) Wiki-Vote network

Fig. 5. Modularity for varying number of clusters for Email-URV network and for Wiki-Vote network

Figure 5(a) shows the modularity for varying initial number of clusters for Email-URV network. Figure 5(b) shows the modularity for varying number of clusters for Wiki-Vote network.

Figure 6 shows the computation time for varying number of clusters for Email-URV data, Wiki-Vote data and Facebook data. For each data set, the time for projecting the graph onto Euclidean space is the same, but the clustering time differs. For Facebook data, the computation time for graph layout is about 248.774 seconds, which is about 1/3 to 1/4 of the total time to cluster the graph once. *KM* computation time keeps the same in general as the initial number of clusters increases while *EM*'s computation time linearly increases as the initial number of clusters increases. Compared with *KM*, *EM* takes much more time. The trends are similar among results for the three data sets.

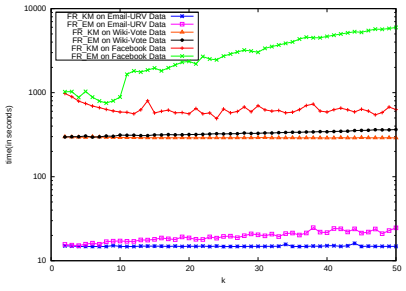


Fig. 6. Running time for varying number of clusters for Email-URV, Wiki-Vote, and Facebook data set

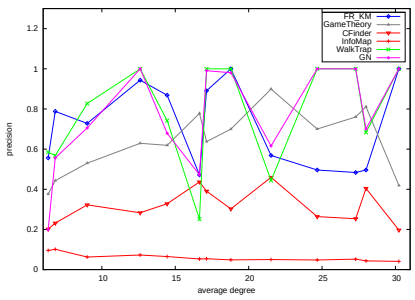


Fig. 7. Precision for varying average degree of synthetic graphs

We compare our method with *GN*, *InfoMap* and *WalkTrap* algorithm, and two other community detection algorithms, CFinder [19] and GameTheory algorithm [6]. Figure 7 shows the precision achieved by the algorithms on the gen-

erated graphs with different average degrees. Since the community structures are known, precision is obtained by counting the number of correctly clustered vertices. From the results we can see that our method outperforms the CFinder, *GN* and *InfoMap*, and produces results comparable with GameTheory algorithm and *WalkTrap*. The reason for CFinder having the low precision may be that not every vertex in the graph are clustered. The clusters consists of 3-cliques only in our experiment. The reason for *InfoMap* having the low precision may be that the number of community this method detects is large and most of the communities are of small size. Many communities are of size of two vertices only.

4.3 Analysis of overlapping community detection

In Figure 8 we show that integrating soft clustering algorithm, *FCM*, in our algorithm, we find that the two communities are overlapping on four vertices (vertex 3, 9, 10, 31) that are marked both in green and blue. This is under the assumption that if the membership strength is larger than 0.8, then the vertex belongs to that cluster only. Figure 9 shows the membership strength of each vertex to each cluster. If we take 0.7 as a threshold for belonging strength, then the clusters will have more overlapping vertices. [24] points out that vertex 3, 9, 10, 14 and 31 are often misclassified by traditional algorithms. We believe that vertex 3, 9, 14 and 31 are shared between the two groups if overlapping is allowed.

4.4 Complexity

Our method's capability of working with large social network data contributes to low complexities of the basic techniques we utilize. *FR*'s complexity is $t*(\mathcal{O}|E|+\mathcal{O}(|V|^2))$, where $|E|$ is the number of edges, $|V|$ is the number of vertices and t is the number of iteration. Time complexity of k-means clustering is $\mathcal{O}(|V|*k*t)$, where $|V|$ is the number of vertices and t is number of iterations. Time complexity of refinement is $\mathcal{O}(|V|^2)$, since we check each vertex's immediate neighbors, the number of which is at most $|V|-1$. Therefore, for the whole algorithm, time complexity is $t*(\mathcal{O}|E|+\mathcal{O}(|V|^2))+\mathcal{O}(|V|*k*t)+\mathcal{O}(|V|^2)$. For large networks, $|V|$ and $|E|$ are the most significant elements that affect time complexity.

5 Conclusions

In this paper, we propose a graph-layout based community detection algorithm. We use Fruchterman-Reingold algorithm to project the graph onto a Euclidean space, and cluster the vertices according to their Euclidean distance. For non-overlapping community detection we use k-means clustering. For overlapping community detection we use fuzzy c-means clustering. Then we refer to the original graph information to refine the communities detected. We evaluate the effectiveness and efficiency on both real-world data and synthetic data. We measure effectiveness by modularity and precision. We measure efficiency by running

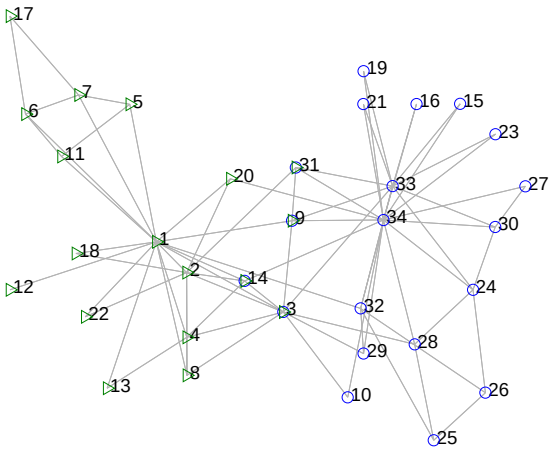


Fig. 8. Zachary's Karate Club data is partitioned into overlapping clusters

vertexID	C_1	C_2
1	0.996227	0.003773
2	0.940114	0.059886
3	0.598883	0.401117
4	0.882118	0.117882
5	0.929731	0.070269
6	0.823111	0.176889
7	0.863318	0.136682
8	0.809490	0.190510
9	0.473149	0.526851
10	0.344961	0.655039
11	0.865513	0.134487
12	0.914518	0.085482
13	0.877715	0.122285
14	0.771624	0.228376
15	0.045550	0.954450
16	0.045550	0.954450
17	0.751788	0.248212
18	0.931399	0.068601
19	0.223629	0.776371
20	0.797011	0.202989
21	0.045550	0.954450
22	0.974779	0.025221
23	0.045550	0.954450
24	0.023890	0.976110
25	0.061673	0.938327
26	0.045550	0.954450
27	0.045550	0.954450
28	0.157053	0.842947
29	0.293805	0.706195
30	0.045550	0.954450
31	0.427176	0.572824
32	0.151637	0.848363
33	0.022016	0.977984
34	0.029300	0.970700

Fig. 9. membership strength

time. For disjoint community detection, the results show that *FR-KM* is more effective on both small graphs and large graphs than *GN*, and is much more efficient than *GN* on large networks. *FR-KM* is also more effective than *Walktrap* and *InfoMap* algorithms in terms of modularity. Compared with *GN*, *CFinder*, *InfoMap*, *WalkTrap* and Gametheory algorithms on the synthetic graphs with known communities in advance, our method is more effective than *GN* and *CFinder* and has good performance comparable with *WalkTrap* Game Theory algorithm according to the results of precision. For overlapping detection, the result for Karate Club data shows that *FR-FCM* is reasonably effective.

References

1. Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *NATURE*, 466:761, 2010.
2. J. A. Aslam, E. Pelekhov, and D. Rus. The star clustering algorithm for static and dynamic information organization. *J. Graph Algorithms Appl.*, 8:95–129, 2004.

3. J. Baumes, M. K. Goldberg, M. S. Krishnamoorthy, M. Magdon-Ismail, and N. Preston. Finding communities by clustering a graph into overlapping subgraphs. In *IADIS AC*, pages 97–104, 2005.
4. J. Baumes, M. K. Goldberg, and M. Magdon-Ismail. Efficient identification of overlapping communities. In *ISI*, pages 27–36, 2005.
5. J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.
6. W. Chen, Z. Liu, X. Sun, and Y. Wang. A game-theoretic framework to identify overlapping communities in social networks. *Data Min. Knowl. Discov.*, 21(2):224–240, Sept. 2010.
7. A. Clauset. Finding local community structure in networks. *PHYS.REV.E*, 72:026132, 2005.
8. A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *PHYS.REV.E*, 70:066111, 2004.
9. M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi. Demon: a local-first discovery method for overlapping communities. *CoRR*, 2012.
10. L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas. Self-similar community structure in a network of human interactions. *Physical Review E*, 68, 2003.
11. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.
12. Y. Dourisboure, F. Geraci, and M. Pellegrini. Extraction and classification of dense communities in the web. WWW '07, New York, NY, USA, 2007. ACM.
13. N. Du, B. Wu, X. Pei, B. Wang, and L. Xu. Community detection in large-scale social networks. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, WebKDD/SNA-KDD '07, pages 16–25. ACM, 2007.
14. M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. U.S.A.*, 95:14863–14868, 1998.
15. Email-URV. http://deim.urv.cat/ aarenas/data/welcome.htm.
16. B. Etling, J. Kelly, R. Faris, and P. John. Mapping the arabic blogosphere: Politics, culture, and dissent. *Berkman Center Research Publication*, (2006-06), 2009.
17. S. Fortunato and C. Castellano. Community structure in graphs. In *Encyclopedia of Complexity and Systems Science*, pages 1141–1163. 2009.
18. T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164, 1991.
19. I. F. Gergely Palla, Imre Derenyi and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, June 2005.
20. M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.
21. M. K. Goldberg, S. Kelley, M. Magdon-Ismail, K. Mertsalov, and A. Wallace. Finding overlapping communities in social networks. In *SocialCom/PASSAT*, pages 104–113, 2010.
22. S. Gregory. An algorithm to find overlapping community structure in networks. PKDD 2007, pages 91–102. Springer-Verlag, 2007.
23. S. Gregory. A fast algorithm to find overlapping communities in networks. ECML PKDD '08, pages 408–423. Springer-Verlag, 2008.

24. A. Lancichinetti, S. Fortunato, and J. Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11, 2009.
25. A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 78(4), 2008.
26. A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. *CoRR*, 2010.
27. J. Leskovec, D. Huttenlocher, and J. Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World Wide Web*. ACM, 2010.
28. S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28:129–137, 1982.
29. K. Macropol, T. Can, and A. K. Singh. Rrw: repeated random walks on genome-scale protein networks for local cluster discovery. *BMC Bioinformatics*, 2009.
30. MPI. http://socialnetworks.mpi-sws.org/.
31. M. Newman and M.Girvan. Finding and evaluating community structure in networks. *PHYS.REV.E*, 69:026113, 2004.
32. M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.
33. V. Nicosia, G. Mangioni.V.Carchiolo, and M. Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of statistical Mechanics: Theory And Experiment*, 2009.
34. P. Pons and M. Latapy. Computing communities in large networks using random walks. In *ISCIS*, 2005.
35. P. K. Reddy, M. Kitsuregawa, P. Sreekanth, and S. S. Rao. A graph based approach to extract a neighborhood customer community for collaborative filtering. DNIS '02, pages 188–200, London, UK, UK, 2002. Springer-Verlag.
36. M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences of the United States of America*, 105, 2008.
37. M. Sachan, D. Contractor, T. A. Faruquie, and L. V. Subramaniam. Using content and interactions for discovering communities in social networks. WWW '12. ACM, 2012.
38. S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.
39. SNAP. http://snap.stanford.edu/data.
40. S. van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, 2000.
41. B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *WOSN*, 2009.
42. D. T. Wijaya and S. Bressan. Ricochet: A family of unconstrained algorithms for graph clustering. In *DASFAA*, pages 153–167, 2009.
43. J. Xie and B. K. Szymanski. Towards linear time overlapping community detection in social networks. PAKDD'12. Springer-Verlag, 2012.
44. B. Yan and S. Gregory. Detecting communities in networks by merging cliques. *CoRR*, 2012.
45. S. Zhang, R. S. Wang, and X. S. Zhang. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications*, 374(1):483–490, 2007.
46. Y. Zhang and D.-Y. Yeung. Overlapping community detection via bounded non-negative matrix tri-factorization. KDD '12. ACM, 2012.