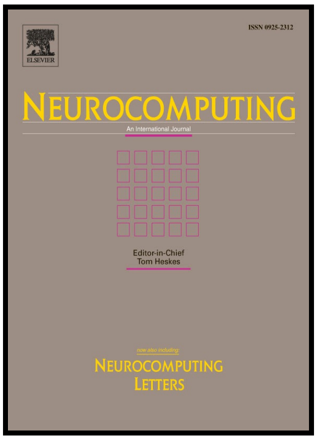


www.elsevier.com/locate/neucom

An Overlapping Community Detection Algorithm Based on Density Peaks

Xueying Bai, Peilin Yang, Xiaohu Shi

PII: S0925-2312(16)31400-X
DOI: http://dx.doi.org/10.1016/j.neucom.2016.11.019
Reference: NEUCOM17766

To appear in: *Neurocomputing*

Received date: 11 June 2016
Revised date: 28 August 2016
Accepted date: 7 November 2016

Cite this article as: Xueying Bai, Peilin Yang and Xiaohu Shi, An Overlappin Community Detection Algorithm Based on Density Peaks, *Neurocomputin* http://dx.doi.org/10.1016/j.neucom.2016.11.019

An Overlapping Community Detection Algorithm Based on Density Peaks

Xueying Bai[1], Peilin Yang[1], Xiaohu Shi[1,2*]

[1]College of Computer Science and Technology, Jilin University, 2699 Qianjin Street, Changchun 130012, PR China
[2]Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Changchun 130012, China

[*]Corresponding author, E-mail: shixiaohu@jlu.edu.cn

Abstract

Many real-world networks contain overlapping communities like protein-protein networks and social networks. Overlapping community detection plays an important role in studying hidden structure of those networks. In this paper, we propose a novel overlapping community detection algorithm based on density peaks (OCDDP). OCDDP utilizes a similarity based method to set distances among nodes, a three-step process to select cores of communities and membership vectors to represent belongings of nodes. Experiments on synthetic networks and social networks prove that OCDDP is an effective and stable overlapping community detection algorithm. Compared with the top existing methods, it tends to perform better on those “simple” structure networks rather than those infrequently “complicated” ones.

Keywords: overlapping community; density peak; community core; membership vector; social networks

1. Introduction：

Network is a universal and powerful tool to present the pattern of connection and interaction among parts of a system. Many systems in the real world are in the form of network, such as social network and neural network. Most of them are of high complexity, which are called ‘Complex Networks’. For those complex networks, community is one of the most common and important properties to reveal the hidden structures of a network [1]. Thus, community detection becomes one of the most crucial problems in this field. Its main purpose is to divide a network into several groups such that the nodes are densely connected within each group and sparsely connected between different groups [1].

There have been numerous researches focusing on community detection of complex networks, such as Spectral Partition [2], Kernighan-Lin algorithm [3], MFC algorithm [4] and GN algorithm [5]. The first two ones are the top-down dividing algorithms while the other two are bottom-up aggregation algorithms. The Spectral Partition utilizes the graph Laplacian [1] and its eigenvectors to divide a subset of nodes into two parts iteratively from the original whole set of network. The Kernighan-Lin algorithm starts with any division to divide nodes into two parts and then swaps the pair of vertices which reduces the cut size [3] by the largest amount or (if no pairs reduces it) increases the cut size by the smallest amount until no pairs can be swapped [1]. The MFC algorithm and GN algorithm detect communities both by removing links between communities. The MFC algorithm is based on the Max Flow-Min Cut theorem [6]. It recognizes connections among communities by calculating the cut set [4]. The GN algorithm is built around the idea of using centrality indices to find community boundaries [5]. Many other clustering methods have been also applied to community detection such as hierarchical clustering and optimization method [7].

However, in real world a node may belong to more than one community. Those nodes which belong to several communities are called overlapping nodes. And a community is called an overlapping community if it contains overlapping nodes. Overlapping communities often appear in real world networks. For instance, a protein network is commonly divided into communities according to protein functions. Because there are many proteins with more than one function, the protein network is overlapped. In the social network, a person may have multiple belongings, which means that the social network is also an overlapped network [1]. Overlapping communities interact with each other via overlapping nodes, which means ordinary algorithms for disjoint community detection are not accurate enough in such a situation. Because the number of communities a node belongs to is uncertain, algorithms like the Kernighan-Lin algorithm which assign a node into only one community are not appropriate enough. Removing links between communities is unable to detect overlapping communities as well.

Thus, several types of algorithms for overlapping community detection have been developed in recent years, including clique percolation, line partition, local expansion, fuzzy detection and dynamical algorithms [8]. CPM [9] and CPMw [10] are typical algorithms of clique percolation. CPM first finds all *k*-cliques in the network. Then each pair of *k*-cliques that shares (*k*-1) public nodes is combined into a community. CPMw is an extension of CPM on weighted networks raised by Farkas in 2007, which introduces a threshold of *k*-cliques. Only *k*-cliques whose intensities are larger than this threshold are considered during the combination. Both algorithms are suitable for networks with dense connected parts [8].

Unlike clique percolation algorithms which are based on combination of nodes, algorithms of line partition are based on partition of links instead of nodes [8]. Because a link connects two nodes, if two links which both connect to a same node are divided into different clusters, this node is assigned into these two clusters as an overlapping node. The algorithm proposed in [11] calculates the similarity between each pair of edges, and then uses hierarchical clustering with similarity to decide belongings of edges. Another algorithm raised by Evans [12] creates a weighted graph (which is called line graph) whose nodes represents links and then applies disjoint community detection algorithms on it [8]. Some algorithms like that proposed in [13] are also based on the line graph. In 2015, a flexible multiscale approach was introduced to overlapping community detection [18]. It describes nodes using local sets of edges, and then agglomerates these sets to carve out communities by minimizing three scales at individual nodes, individual communities, and the network level.

Local expansion is another type of overlapping community detection algorithm which is based on nodes. The main idea of local expansion is to choose a seed community and then expand from it. Most algorithms of this kind need a local optimize function to quantify the connection of a group of nodes [8]. The classic algorithms include LFM [14] and OSLOM [15]. LFM first chooses a seed node randomly. Then other nodes are assigned to this community until the local optimize function reaches the local maxima. This process repeats among nodes which are not included in a community. With different expansion method, OSLOM is based on the local optimization of a fitness function expressing the statistical significance of clusters with respect to random fluctuations, which is estimated with tools of Extreme and Order Statistics [15].

Another method to represent belonging of overlapping nodes is based on belonging factors [20]. Fuzzy detection algorithms utilize membership vector to represent the belonging of each node. Main algorithms of fuzzy detection include SPAEM [16] and some algorithms based on NMF [8].However, this kind of algorithms has a drawback that it usually requires the number of communities before the detection, which restricts its application to some extent.

The dynamical algorithms are another widely used overlapping detection algorithms. Main algorithm of this kind is the label propagation algorithm like COPRA and SLPA. In COPRA [20], vertices have labels which contain information about more than one community. These labels then propagate between neighboring vertices so that members of a community reach a consensus on their community membership. The algorithm SLPA [17] is regarded to be the best overlapping community detection algorithm in [8]. It spread label among nodes during iterations and restore previous label information for each node, which is different from COPRA. The process of its label propagation is called a speaker-listener process, in which the distribution of label follows specific speaking or listening rules. In 2015, a new propagation algorithm to detect overlapping clusters was raised by Gaiteri et al. [22] This so-called SpeakEasy algorithm utilizes a bottom-up approach using neighboring information to clustering and a top-down approach using the information of the whole network. It updates nodes' labels based on their neighbors' labels, and then subtracts the expected frequency of these labels in each iteration. This algorithm is proved to have good results on overlapping community detection [22]

In 2014, Rodriguez et. al. have proposed a density peak clustering method in *Science* [19]. For the method is an effective and powerful tool, it is accorded great and prompt attention and has been widely applied to many clustering problems. Based on density-peak method, Wang et. al. proposed an overlapping detection algorithm and received better performance. However, this algorithm targets at only online social networks and could not be applied to other kinds of networks directly. In this paper, we propose a general density peak based overlapping community detection algorithm from another perspective. Firstly, a novel distance matrix representation based on similarity is designed. And then, the core (community centers) is selected using a more restricted method. At last, all other nodes are allocated to different communities according to their membership vectors. To evaluate our proposed method, it is applied to synthetic and real networks. Numerical results show the effectiveness of the method.

2. Clustering based on Density Peaks

Density peak clustering method was proposed by Rodriguez et. al. in *Science* [19]. This method is based on the assumption that cluster centers are with relatively high local density and that they are at a relatively large distance from any points with a higher local density. For the readers' easily understanding, a brief introduction of the algorithm is given below.

Firstly, the local density of each point is calculated according to

$$\rho_i = \sum_j \chi\left(d_{ij} - d_c\right) \tag{1}$$

$$\chi(x) = \begin{cases} 1 & if\;\; x < 0 \\ 0 & otherwise \end{cases} \tag{2}$$

where ρ_i is the local density of each point i, d_{ij} is the distance between point i and j, and d_c is the cutoff distance. And then, the minimum distance between any other point with higher density with itself could be found by

$$\delta_i = \begin{cases} \max\limits_j\left(d_{ij}\right) & if\;\; \rho_i = \max\limits_k\left(\rho_k\right) \\ \min\limits_{j:\rho_j > \rho_i}\left(d_{ij}\right) & otherwise \end{cases} . \tag{3}$$

Therefore, the decision graph is generated by taking ρ_i as x axis and δ_i as y axis. Those points with both relatively large ρ_i and δ_i are chosen as cores of clusters. And the other points should be assigned to the same cluster as its nearest neighbor of higher density.

However, this calculation of density Eqs. (1-2) might be unavoidably affected by large statistical errors for small data sets, an improvement is to adopt Gaussian kernel to set the local density [19, 21], which is defined by

$$\rho_i = \sum_j \exp\left(-\frac{d_{ij}^2}{d_c^2}\right) \tag{4}$$

3. Algorithm:

In this paper, a novel Overlapping Community Detection algorithm based on the Density Peak clustering (OCDDP) is proposed. For the community detection problem is different from clustering problem, there are some problems should be solved in the algorithm, for example: 1) how to set distances between each pair of nodes in the network; 2) how to set the criteria for cores of communities; 3) how to decide which communities a node belongs to. The main steps of OCDDP includes: 1) construction of distance matrix, 2) compute and regularization of ρ and δ, 3) selection of the community cores, 4) allocation of the other nodes. The flowchart of OCDDP is shown as Fig 1.

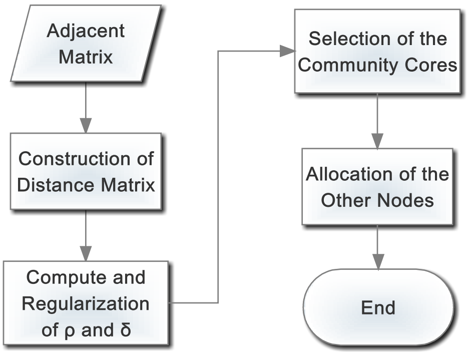

Adjacent Matrix
Construction of Distance Matrix
Compute and Regularization of ρ and δ
Selection of the Community Cores
Allocation of the Other Nodes
End

Fig 1. Flowchart of OCDDP

3.1. Construction of Distance Matrix

In density peak clustering method, the distance matrix is set as the input each entry of which represents the distance between one pair of points [19]. While in complex networks, the inputs are often set as adjacent matrixes. If the complex networks are unweighted, the elements of the adjacent matrixes are 0 or 1, otherwise they are nonnegative real values. That is to say, only those linking information could be obtained. So, the adjacent matrixes should be transferred to distance matrixes. One simple idea is to set the reciprocals of linkage values as the distances. However, for most of adjacent matrixes are sparse, it might cause too many infinite values in distance matrix which could not be distinguished even though they are greatly different. For example, assuming one pair of points could be linked with one intermediate node, and another pair of points needs many intermediate nodes to link together, but their distances both are infinite values. To solve this problem, the intermediate nodes of a pair of points are considered iteratively in the proposed algorithm. The number of routes by which one node reaches its counterpart in 2α steps ($\alpha \geq 1$) is used to represent the distance of the pair of nodes, which is called by absolute strength of linkage. However, only using absolute strength of linkage cannot show the connection accurately because the same number of routes of the nodes with different degrees does not contribute same for the connection. For example, in Fig.2, when α=1 the number of routes between node A and B is equal to that between node A and C. However, since all edges of C link A in two steps while two edges of B don't, the connection between A and B is not closer than that between A and C. So the relative strength of linkage based on node degree is also considered in the distance matrix construction algorithm which is described as following.

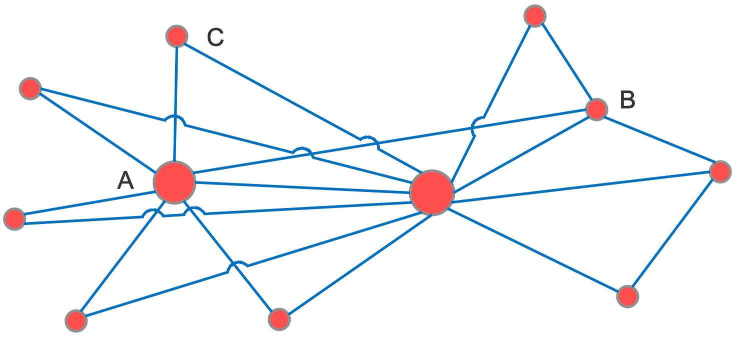


Fig 2. A sample network

For a network with n nodes, denote the adjacent matrix of the network as $A_{n\times n}$, the entry of which a_{ij} represents the linkage strength between node i and j. For each i and j, the absolute strength of linkage between them is denoted as sab_{ij}, which is defined as

$$\begin{cases} sab_{ij}^{(0)} = a_{ij} \\ sab_{ij}^{(\alpha)} = sab_{ij}^{(\alpha-1)} + \frac{1}{2}\sum_{k\neq i,j} sab_{ik}^{(\alpha-1)} sab_{kj}^{(\alpha-1)} \ (\alpha > 0) \end{cases} \tag{5}$$

Then the strength of linkage between node i and j is calculated by

$$str_{ij} = \frac{t\cdot sab_{ij}^{(\alpha)}}{mean\left(sab^{(\alpha)}\right)} + \frac{(1-t)\cdot n\cdot sab_{ij}^{(\alpha)}}{\sqrt{\sum_m sab_{im}^{(\alpha)} \sum_q sab_{qj}^{(\alpha)}}} \tag{6}$$

The first part of the right hand of Eq. (6) represents impact of absolute strength of linkage and the second part represents that of relative strength of linkage. Here t is the parameter representing weights of absolute and relative strength of linkage.

Finally, the distance between node i and j is calculated by

$$d_{ij} = \begin{cases} \frac{1}{str_{ij}+\varepsilon} & i\neq j \\ 0 & i = j \end{cases} \tag{7}$$

where ε is a small positive value which is to avoid being divided by 0.

3.2. Regularization of ρ and δ

Definitions of local density ρ_i and distance δ_i from the nearest node of higher densities in this algorithm are the same as those shown in section 2. The cutoff distance d_c is set to make the average number of neighbors around 1 to 2% of the total number of points in the data set [19]. In our algorithm, those pairs with distances equaling to 1/ε have no linkages, so they are not counted during the calculation of d_c. In addition, in this algorithm the nodes with δ_i =1/ε are considered as cores because they don't link to any other nodes with more densities.

Unlike the density peak algorithm in *Science* [19] which detects disjoint clusters, our algorithm is designed to detect also overlapping communities. However, for networks with overlapping communities, connections among nodes from two overlapping communities are generally closer than those from two disjoint communities. Assume that A is the core of a community and B is the nearest node with higher density than A. As shown in Fig 3, there are two possible relationships of A and B and communities they are in.

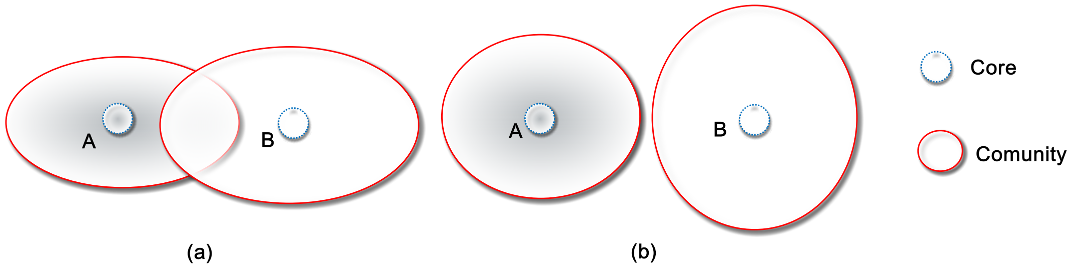


Fig 3. Two possible relationships of core A, its nearest node B with higher density and communities they are in: (a) Overlapping communities. (b) Disjoint communities.

When communities of A and B overlap with each other (Fig 3(a)), distance δ of A is smaller than that when their communities are disjoint (Fig 3(b)). If there are many overlapping communities in a network, distance δ of most nodes may not be obviously large, including some cores of communities. To distinguish cores more accurately, we use a regularization to 'separate' two overlapping communities when calculating distance δ. The δ_i^* for node i is defined as

$$\delta_i^* = \exp\left(-\left(\frac{d_a}{\delta_i}\right)^2\right) \qquad (8)$$

The threshold d_a is selected from the list of δ. In this algorithm d_a locates around 80% of the ascending δ list in which all δs are smaller than 1/ε. A simple regularization is also need for local density ρ. The ρ_i^* for each node i is defined as

$$\rho_i^* = \frac{\rho_i}{\max_j \rho_j} \qquad (9)$$

This regularization extends distances which are around d_a. Some nodes with relatively small distance because of overlapping will stand out while others will still gather because of their similar distances. Fig 4(a) and Fig 4(b) show the distribution of $\delta_i^{'} = \frac{\delta_i}{\max_j \delta_j}$ and δ_i^*, Fig 4(c) and Fig 4(d) show their decision graphs.

3.3. Selection of the Cores

Fig 4(d) shows the decision graph after regularization. In the decision graph there are some nodes which we called marginal nodes like node A in Fig 4(d). For overlapping community detection, the core of a community is not always with an obviously large δ even after the regularization. In addition, if sizes of communities vary a lot in a network, the density of a community core may also be relatively small. In these aspects, cores of overlapping communities are divided into two groups: with both large ρ^* and δ^*, or with one large and another relatively small. For cores of the second group, we use the mutilation of ρ^* and δ^* to select them.

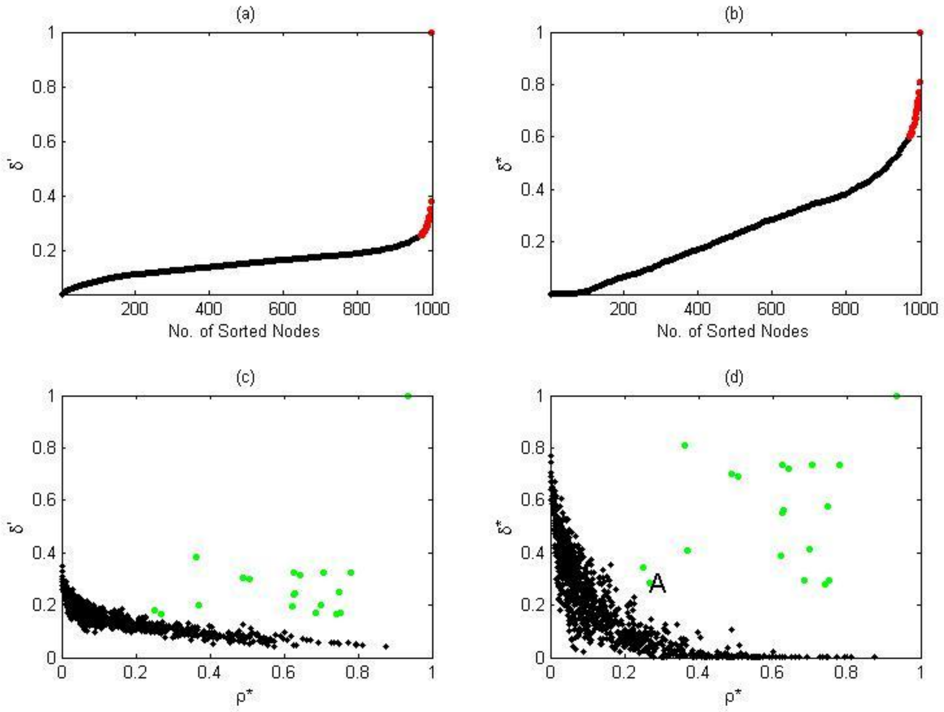


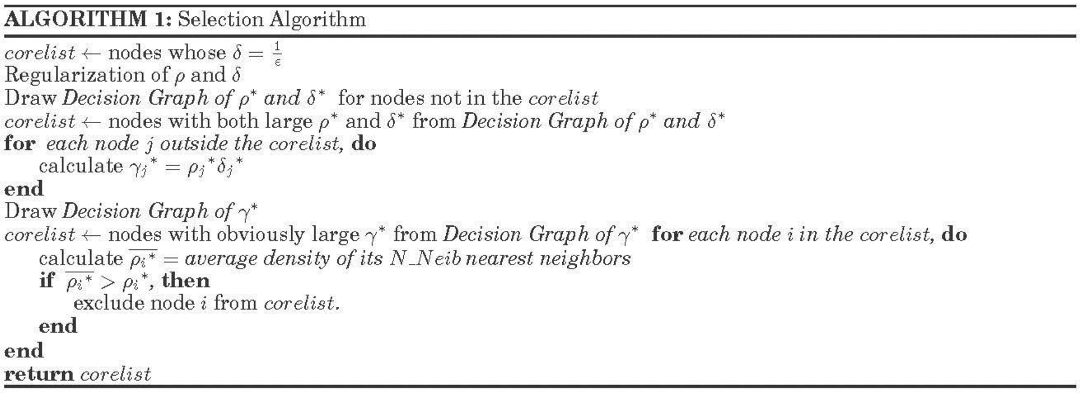
Fig 4. Distribution of δ before and after regularization, and their corresponding decision graphs: (a). Distribution of δ' on a network with 1000 nodes, community size is (20,100), μ=0.3, O_m=3, O_n=10%. (Details of the network are shown in section 4.1.) The first 30 nodes with large δ' are colored red. (b). Distribution of δ^* on the same network as that used in (a). The same 30 nodes as those in (a) are colored red. (c). Corresponding decision graph of δ' and ρ^*. Nodes with large δ' and $\rho^*$are colored green. (d). Corresponding decision graph of δ^* and ρ^*. The same nodes as those in (c) are colored green. To show the relationship of nodes clearly, the nodes with δ=1/ε are concealed.

Furthermore, we assume that the density of a core is not smaller than the average density of its *N_Neib* nearest neighbors. This assumption is proposed to reduce the impact of overlapping nodes during the core selection. An overlapping node usually locates at edges of two communities. It has connections with nodes of both communities so that its local density is sometimes relatively large. In addition, its distance from the nearest node with higher density is relatively large due to its sparse connections to one node. However, since many of its neighbors are closer to one community and have denser connections with nodes in the community, some of their local densities are larger than the overlapping node. So this restriction of cores' densities works well to avoid overlapping nodes being chosen as cores. In our paper, *N_Neib* is selected as 5.

So the selection of cores can be divided into three steps:
1) If δ_i =1/ε, the node is considered as a core. Then Choose nodes with both large ρ^* and δ^* from the decision graph into the core list;
2) Calculate $\gamma_j^*=\rho_j^*\delta_j^*$ for each other node, choose those with obviously large γ^* into

the core list (if the former decision graph can clearly separate cores and non-cores, don’t choose nodes in this step);
3) Exclude nodes with smaller densities than neighbors from the core list by comparing to their neighbors’ densities.

Nodes in the core list are considered to be cores of the overlapping communities. The pseudo code of the algorithm is shown in Fig. 5.



```
ALGORITHM 1: Selection Algorithm
corelist ← nodes whose $\delta = \frac{1}{\varepsilon}$
Regularization of $\rho$ and $\delta$
Draw Decision Graph of $\rho^*$ and $\delta^*$ for nodes not in the corelist
corelist ← nodes with both large $\rho^*$ and $\delta^*$ from Decision Graph of $\rho^*$ and $\delta^*$
for each node $j$ outside the corelist, do
    calculate $\gamma_j{}^* = \rho_j{}^* \delta_j{}^*$
end
Draw Decision Graph of $\gamma^*$
corelist ← nodes with obviously large $\gamma^*$ from Decision Graph of $\gamma^*$ for each node $i$ in the corelist, do
    calculate $\overline{\rho_i{}^*}$ = average density of its N_Neib nearest neighbors
    if $\overline{\rho_i{}^*} > \rho_i{}^*$, then
        exclude node $i$ from corelist.
    end
end
return corelist
```

Fig 5. Pseudo code of cores selection algorithm

3.4. Allocation of the Nodes to Communities

Assume that there are c ($c\geq 1$) communities, we use probability vector $p_i=\{p_{i,1}, p_{i,2},\ldots,p_{i,c}\}$to illustrate possibilities of node i belongs to different communities. The probability vector of node i is decided by its nearest *N_Neib* neighbors whose local densities are larger than ρ_i. Denote the core of community j as node n_j. Assume the core of a community cannot be an overlapping node. Then the probability vector of the core of community j is set as

$$p_{n_j,k} = \Delta_{n_j,k} \tag{10}$$

$$\Delta_{i,j} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{11}$$

For other nodes, we calculate their probability vectors in the descending order of local densities. The probability vector of node i is defined as

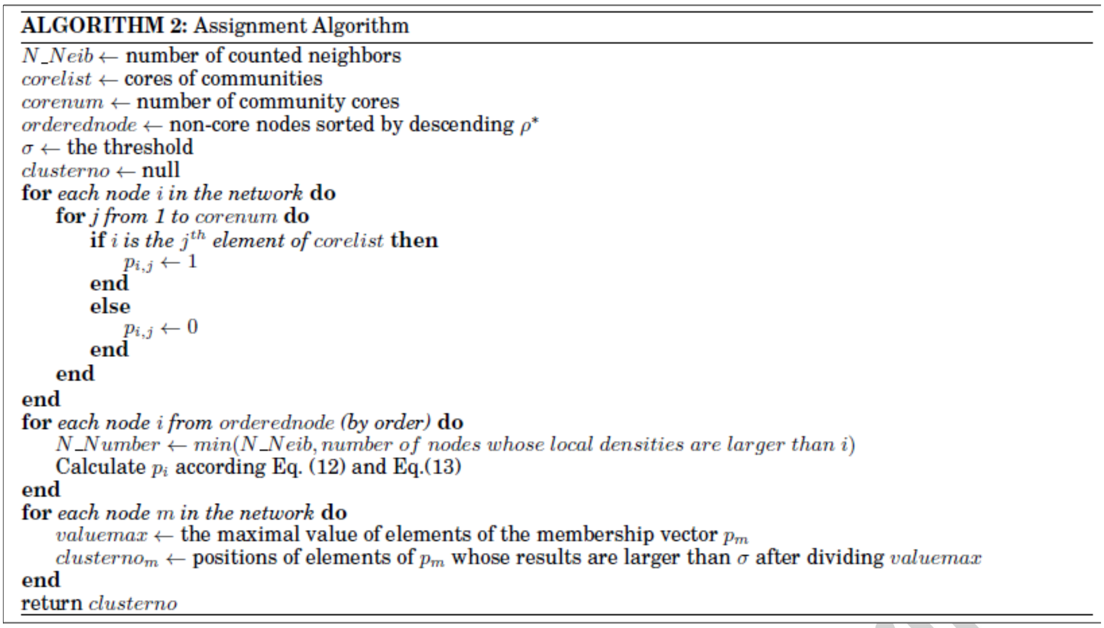
$$p_i = \sum_{j=1}^{\min\{N\_Neib, N\_Li\}} w_{i^{(j)}} p_{i^{(j)}} \tag{12}$$

where *N_Li* is the number of nodes whose local densities are larger than i, $i^{(j)}$denotes the index of the j^{th} “densest” neighbor of node i, $w$is the weight defined as

$$w_{i^{(j)}} = \frac{str_{i,i^{(j)}}}{\sum_{k=1}^{\min\{N\_Neib, N\_Li\}} str_{i,i^{(k)}}} \tag{13}$$

Naturally, node i is allocated to community r if $r=argmax_s\{p_{i,s}, s=1,2,\ldots,c\}$. For overlapping community detection, each node can belong to more than one community. The allocation method is according to the probability vector: node i is also assigned to community s as an overlapping one, if $p_{i,s} / p_{i,r} > \sigma$, where σ is a given threshold.

The pseudo code of assignment algorithm is shown as Fig 6.



```
ALGORITHM 2: Assignment Algorithm
$N\_Neib$ ← number of counted neighbors
$corelist$ ← cores of communities
$corenum$ ← number of community cores
$orderednode$ ← non-core nodes sorted by descending $\rho^*$
$\sigma$ ← the threshold
$clusterno$ ← null
for each node $i$ in the network do
    for $j$ from 1 to $corenum$ do
        if $i$ is the $j^{th}$ element of $corelist$ then
            $p_{i,j} \leftarrow 1$
        end
        else
            $p_{i,j} \leftarrow 0$
        end
    end
end
for each node $i$ from $orderednode$ (by order) do
    $N\_Number \leftarrow min(N\_Neib, number\ of\ nodes\ whose\ local\ densities\ are\ larger\ than\ i)$
    Calculate $p_i$ according Eq. (12) and Eq.(13)
end
for each node $m$ in the network do
    $valuemax$ ← the maximal value of elements of the membership vector $p_m$
    $clusterno_m$ ← positions of elements of $p_m$ whose results are larger than $\sigma$ after dividing $valuemax$
end
return $clusterno$
```

Fig 6. Pseudo code of node assignment algorithm

3.5. Time Complexity of the Algorithm

Denote the number of nodes as *n*, the average number of edges as *m*. In each iteration, sab_{ij} for node *i* and any other node *j* can be calculated by a breadth-first search algorithm in exactly two steps. The time complexity at the worst case is $O(n(m+n))$. In most cases the time complexity is far less than this. The time complexity of the calculation of *str* is $O(n^2)$. So it costs about $O(\alpha n(\mathrm{m}+n\ (\alpha+1)/\ \alpha))$ time calculating the distance matrix. It costs about $O(n(n\text{-}1))$ and $O(n(n\text{-}1)/2)$ time calculating local densities and distances from nodes with higher local densities. During the regularization, it takes $O(2n\log_2 n)$ times to sort local densities and δs, $O(3n)$ to calculate ρ^*, δ^* and γ^*. It costs $O(N\_Cores \cdot n\log_2 n)$ time to exclude nodes from the corelist. The assignment algorithm costs $O(n^2\log_2 n)$ time. The overall time complexity is about

$$O\left(\alpha n\left[m+\left(1+\frac{3}{2\alpha}n\right)\right]+n^2\log_2 n\right). \tag{14}$$

4. Experiments

To test the proposed algorithm, we conduct experiments on synthetic networks and social networks, respectively. For comparison, algorithms SLPA [17], COPRA [20], CFinder [9], SpeakEasy [22], the Multiscale algorithm [18] (we call it Multiscale here) are also conducted as competitive algorithms. The average performances over ten repetitions are reported for SLPA and COPRA on each network.

4.1. Experiments on Synthetic Networks

For synthetic networks, we use networks generated by LFR benchmark [23], which process properties found in real networks [8, 23]. We set parameters of LFR benchmark following the method raised in [8]. The networks are with sizes from {1000, 5000}, the average degree k is from {5, 10}.Exponents of the degree distribution and community size distribution are τ_1=2 and τ_2=1, respectively. The maxim degree is 20 or 50, community sizes vary in both small range s=(10,50) and large range b=(20,100) [8]. The mixing parameter μ is set to be 0.1 or 0.3, which is the expected fraction of links through which a node connects to other nodes in the same community [8, 23].

Parameter O_m is the number of communities an overlapping node belongs to, and O_n is the fraction of overlapping nodes [8, 23]. For each set of parameters, we generate 10 networks and calculate the average result of tests on them. Results are measured by Normalized Mutual Information (NMI) [15] and Omega Index [24].

4.1.1 Parameters *t* and *σ*

Different values of the weight t in the *str* calculation and the given threshold σ in the assignment of nodes have different impacts on detecting results of different networks. To study the effect of t and σ, we did experiments on 4 networks with different combinations of k, O_m, community size and μ. Other parameters are taken as: α=1, network size n=1000, On=10%.

As shown in Fig 7, performances of our algorithm are robust with respect to different values of σ. Therefore we set σ=0.9 in the following experiments. However, values of t are diverse to achieve best performances on different types of networks. In Fig 7(a) and Fig 7(g), when t=0.5 networks with large community sizes achieve best results of NMI, while on networks with small community sizes that value of t is 0.2. In Fig 7(c) and Fig 7(e), on networks with small and large degrees values of t to get best NMI results are 0 and 0.5. In Fig 7(a) and Fig 7(e), on those networks values of t for best NMI performances are both 0.5. That means on networks with different parameters Om and μ, it’s not necessary to change the value of t to get best performances.

From Fig 7, it could be found that the fluctuations of NMI are similar with those of Omega. So, due to limitations on space, only NMI results are discussed in the rest of experiments.

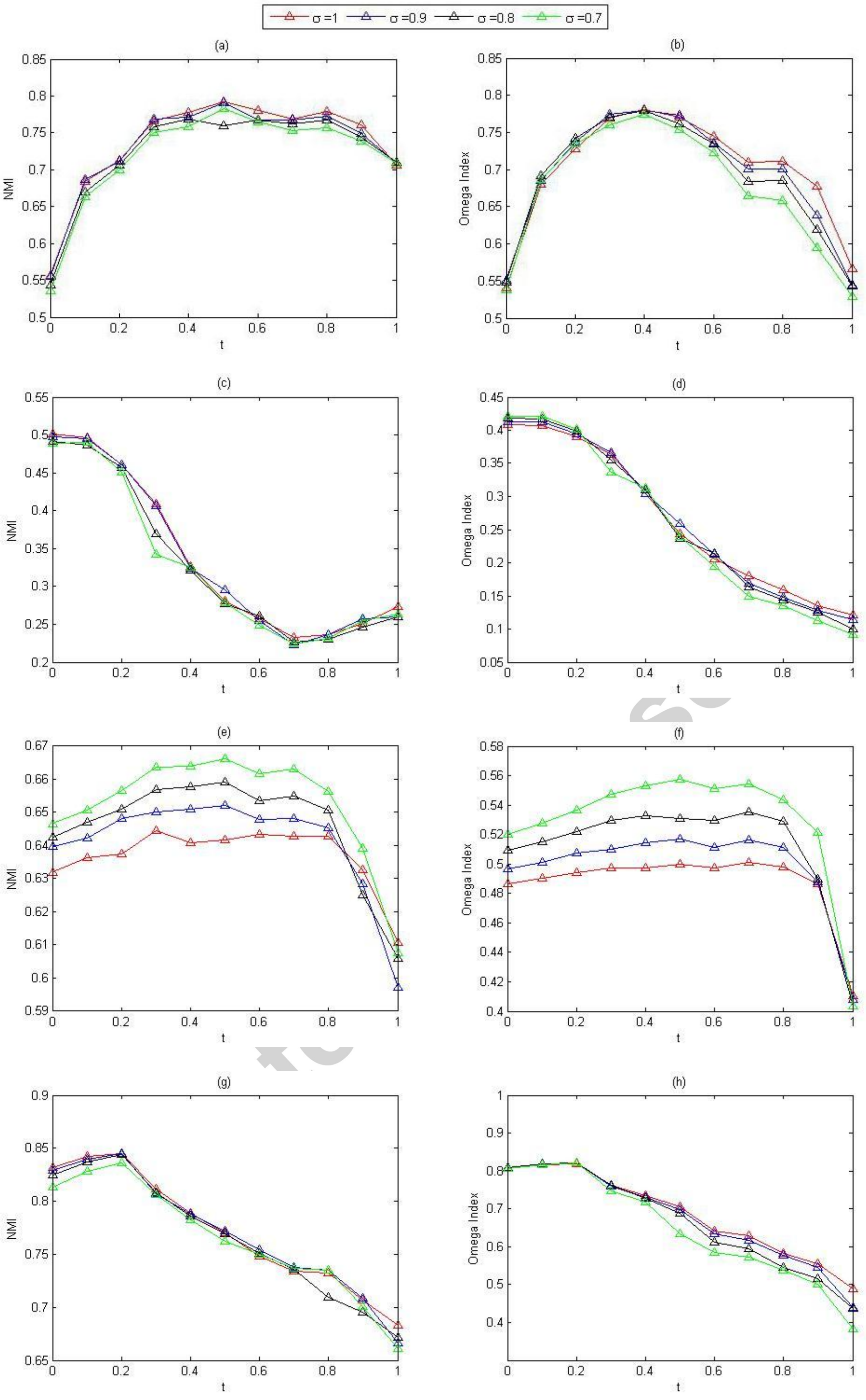


Fig 7. Comparison results for parameters t and σ: (a) (b). NMI and Omega results on network with k=10, large range of community size b=(20,100), μ=0.3, O_m=2. (c) (d). NMI and Omega results on network with k=5, large range of community size b=(20,100), μ=0.1, O_m=8. (e) (f). NMI and Omega results on network with k=10, large range of community size b=(20,100), μ=0.1, O_m=8. (g) (h). NMI and Omega results on network with k=10, large range of community size s=(10,50), μ=0.3, O_m=2.

4.1.3 Performance of the Algorithm

In this section, the performance of OCDDP is tested on 4 networks with different overlapping community number O_m, network size n and mixing parameter μ, respectively. The experimental results are shown in Fig 8. It is easy to find that the performance of the algorithm (NMI) reduces along with the growth of O_m. The possible reason might be that a bigger O_m means an overlapping node belongs to more communities and therefore the structure of the whole network is more complicated, which leads to the performance reducing. It is similar for parameter μ: the results of μ=0.3 are consistently worse than those of μ=0.1.The result is also in line with our expectations, for it is well known that the bigger μ, the more complicated network [8, 23]. However, OCDDP performs better on networks with 5000 nodes than those with 1000 nodes, though one may think the former network is more “complicated”. This result shows that the number of nodes doesn’t affect the topological complexity. In contrast, more nodes could provide more information for the algorithm and help it to get better results.

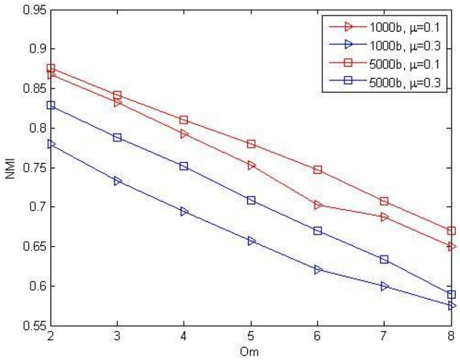


Fig 8. NMI results on networks whose k =10, t=0.5, community sizes are (20,100), O_n=10%.

For comparison, SLPA, COPRA and CFinder are also executed on the same networks along with OCDDP. Fig 9 shows the results on two network groups, both of which are with 5000 nodes, k=10, μ=0.3 and O_n=10%, and community sizes varying from 10 to 50. In the first group shown in Fig 9(a), the average degree and mixing parameter are taken as k=10, μ=0.3, while in the second group shown in Fig 9(b), they are k=5, μ=0.1, respectively. From Fig 9(a), we could find that the NMIs of OCDDP, SLPA and SpeakEasy are significantly higher than those of COPRA, Multiscale and CFinder, and OCDDP performs better than SLPA before O_m = 5. OCDDP has similar results to those of SpeakEasy before O_m = 6. As we know, in most cases of the real world, overlapping nodes belong to no more than 5 communities. Therefore, in this sense, OCDDP is comparable to SLPA on networks shown in Fig 9(a). In Fig 9(b), it is easy to find that OCDDP achieves the best performance on all the O_m cases. From Fig 9, it could be concluded that OCDDP outperforms Multiscale, COPRA and CFinder on all the cases, when the networks are “simple”, saying about with low O_m, k, or μ, OCDDP is also better than SLPA and SpeakEasy, while for the “complicate” networks, OCDDP is little inferior to SLPA and SpeakEasy.

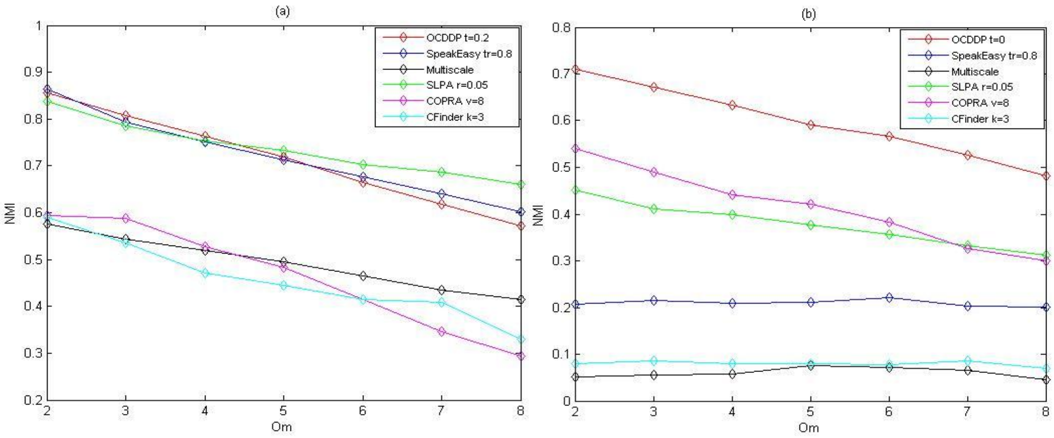


Fig 9. Comparative NMI results with SLPA, COPRA, CFinder, SpeakEasy and Multiscale: (a). On networks whose k=10, μ=0.3. (b). On networks whose k =5, μ=0.1.

4.2. Social networks

To further test our proposed method, OCDDP is also conducted on 10 real social networks. These typical social networks were frequently used as test networks in [8, 17, 20]. From the above section, it could be found that CFinder is inferior obviously to SpeakEasy, Multiscale, SLPA, COPRA, and our proposed OCDDP. Therefore, CFinder are not executed as a comparison algorithm in this experiment.

To measure the performance of the algorithms on real overlapping networks, two widely accepted modularity functions are selected as the metrics in this experiment, namely that EQ and Q_{ov}. EQ is a measure proposed by Shen in 2009 [25] which uses the number of communities a node belongs to as a weight of Q [8, 25]. Qov is a link-based extended modularity raised by Nicosia in 2009 [8, 26]. In Qov tests, we adopt the function $f(x) = 60x – 30$, which is the same as that proposed in [20]. Results are shown in Table 1 and Table 2. For there are only part of Q_{ov} results of SLPA and COPRA are listed in Ref [17, 20], we implement SLPA and COPRA by ourselves. And the Q_{ov} results of SLPA and COPRA are also listed in Table 2. In all tests, we set σ = 0.9. The selection of parameter tr in SpeakEasy refers to results in [27]. Other parameters are shown in Table 1.

From Table1, EQ(O), EQ(CO), EQ(S), EQ(Sp), EQ(M) represents the EQ results of OCDDP, COPRA, SLPA, SpeakEasy, Multiscale respectively, r(S) represents the parameter r in the algorithm SLPA. It is easy to find that EQ results of OCDDP are better than those of COPRA and Multiscale in most cases. SLPA achieves better results than our algorithm in networks Football (by 0.003), Pol.blogs (by 0.005), Jazz (by 0.06) and Email (by 0.02). SpeakEasy get better results in Karate (by 0.01), Lesmis (by 0.016), Football (by 0.003) and Jazz (by 0.006). In 4 networks OCDDP has better performance, especially in the network Power (by 0.2). OCDDP gets the highest average EQ value among the three methods, which is 20.2% higher than that of COPRA, 5.0% higher than that of SLPA and SpeakEasy.

Table 1.Results of the *EQ* test

Data Set	*n*	*m*	*k*	*α*	*t*	*EQ*(O)	*r*(S)	*EQ*(S)	*v*	*EQ*(CO)	*tr*	*EQ*(Sp)	*EQ*(M)
Karate	34	78	4.6	1	0	0.4063	0.33	0.3924	3	0.1654	0.8	**0.4198**	0.2599
Dolphin	62	159	5.1	1	0.5	**0.5202**	0.45	0.4833	4	0.3759	0.8	0.4971	0.3270
Lesmis	77	254	6.6	1	0	0.5310	0.45	0.5270	2	0.4903	0.8	**0.5479**	0.3866
Football	115	613	10.6	1	0.5	0.5958	0.45	0.5984	2	0.5863	0.8	**0.5985**	0.3840
Pol.books	105	441	8.4	1	0.4	**0.5095**	0.45	0.4831	2	0.4802	0.8	0.5052	0.2158
Jazz	198	2742	27.7	1	0	0.3848	0.45	0.4427	1	0.3903	0.8	**0.4447**	0.1562
Email	1133	5254	9.6	1	0.4	0.5023	0.45	**0.5163**	2	0.2896	0.8	0.4574	0.1513
Netscience	1461	2742	4.8	2	0	**0.9460**	0.45	0.9039	6	0.8221	0.8	0.8856	—
Power	4941	6593	2.7	2	0.6	**0.8848**	0.45	0.6563	8	0.7182	0.8	0.6745	0.5502
Pol.blogs	1224	19022	27.3	1	0	0.4222	0.45	**0.4275**	1	0.4274	0.8	0.4072	—
Average						**0.5703**		0.5431		0.4746		0.5438	0.3039

Table 2.Results of the *Qov* test

Data Set	Q_{ov}(O)	Q_{ov}(S)	Q_{ov}(S)*	Q_{ov}(CO)	Q_{ov}(CO)*	Q_{ov}(Sp)	Q_{ov}(M)
Karate	**0.7036**	0.6980	0.65	0.3786	0.44	0.7022	0.2482
Dolphin	**0.7735**	0.7602	0.76	0.6616	0.70	0.6970	0.3766
Lesmis	0.7376	**0.7767**	0.78	0.7412	0.72	0.7247	0.4259
Football	**0.7201**	0.6989	0.70	0.6923	0.69	0.6910	0.4593
Pol.books	**0.8395**	0.8299	0.83	0.8281	0.82	0.6841	0.1354
Jazz	0.6261	0.7042	0.70	**0.7151**	0.716	0.6401	0.1041
Email	**0.6421**	0.6312	0.64	0.5193	0.506	0.5107	0.0693
Netscience	**0.9728**	0.9174	0.85	0.8365	0.812	0.8894	—
Power	**0.9077**	0.6586	—	0.4762	—	0.6750	0.5416
Pol.blogs	**0.8002**	0.7998	—	0.7998	0.748	0.7660	—
Average	0.7723	0.7475	0.7388	0.6649	0.6836	0.6980	0.2951

* The recorded results in [17, 20]

Table 2 gives the *Qov* results, *Qov*(O), *Qov*(CO), *Qov*(S), *Qov*(Sp), *Qov*(M) represents the *Qov* results of OCDDP, COPRA, SLPA, SpeakEasy and Multiscale respectively. From the table, we could find that OCDDP occupies 8 of 10 best values, while COPRA and SLPA occupy 1 for each. On the average *Qov* value, OCDDP outperforms COPRA by 16.2%, SLPA by 3.3% and SpeakEasy by 10.6% respectively. Therefore, it could be claimed that OCDDP is better than COPRA, SLPA and SpeakEasy on the real social networks.

5. Conclusion and Discussion

In this paper, based on the density peak clustering algorithm, a novel overlapping community detection algorithm (OCDDP) is developed. In the algorithm, a distance matrix computation method is firstly proposed, and then a three-step process to select cores of communities is designed, at last, a node allocation method based on membership vectors is developed. Numerical results on synthetic networks and social networks show the effectiveness of OCDDP. In general, OCDDP performs better when the networks are “simple” compared with the existing methods. On those “complicate” networks, OCDDP still works well, and is comparable with the top existing method SLPA and SpeakEasy. Especially for the real social networks, OCDDP outperforms all the compared methods. OCDDP is a useful algorithm in overlapping community detection of networks.

Acknowledgments

The authors are grateful to the support of the National Natural Science Foundation of China (61373050) and the Science Technology Development Project from Jilin Province of China (20130206003SF, 20130101070JC).

Reference

[1] Newman M.E.J. 2010. Networks: An Introduction. ISBN 978-0-19-920665-0

[2] Newman M.E.J. 2006. Modularity and communities structure in networks. Proc. of the National Academy of Science103 (23): 8577−8582.

[3] Kernighan, B. W. and Lin, S. 1970. An efficient heuristic procedure for partitioning graphs. Bell System Technical Journal 49, 291-307.

[4] Flake G.W, Lawrence S, Giles C.L, Coetzee F.M. 2002. Self-Organization and identification of Web communities. IEEE Computer, 35(3):66−71.

[5] Girvan M, Newman M.E.J. 2002. Community structure in social and biological networks. Proc. of the National Academy of Science, 9(12):7821−7826.

[6] L.R. Ford Jr., D.R. Fulkerson.1956. Maximal Flow through a Network, Canadian J. Math, vol. 8, no.3, pp. 399-404.

[7] Newman M.E.J. 2012. Communities, modules and large-scale structure in networks. Nature Physics, 8, 25-31

[8] Xie J, Kelley S, Szymanski BK. 2013. Overlapping community detection in networks: the state of the art and comparative study. Acm Computing Surveys 45 (4): 115-123

[9] Palla G, Derenyi I, Farkas I, Vicsek T. 2005. Uncovering the overlapping community structures of complex networks in nature and society. Nature. 435(7043):814−818.

[10]Farkas I, Abel D, Palla G, Vicsek T. 2007. Weighted network modules. New J. Phys. 9, 6, 180.

[11] Ahn, Y. Y., Bagrow, J. P., and Lehmann, S. 2010. Link communities reveal multiscale complexity in networks. Nature 466, 761–764.

[12] Evans, T., Lambiotte, R. 2010. Line graphs of weighted networks for overlapping communities. Eur. Phys. J. B 77, 265.

[13] Kim, Y. and Jeong, H. 2011. The map equation for link community. Phys. Rev. E 84, 026110.

[14] Lancichinetti, A., Fortunato, S., and Kerte´sz, J. 2009. Detecting the overlapping and hierarchical community structure of complex networks. New J. Phys. 11, 033015.

[15] Lancichinetti, A., Radicchi, F., Ramasco, J. J., and Fortunato, S. 2011. Finding statistically significant communities in networks. PLoS ONE 6, 4, e18961.

[16] Ren, W., Yan, G., Liao, X., Xiao, L. 2009. A Simple probabilistic algorithm for

detecting community structure. Phys. Rev. E 79, 3, 036111.

[17] Xie, J., Szymanski, B. K., and Liu, X. 2011. SLPA: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In Proc. ICDM Workshop. 344–349.

[18] Brutz, M., Meyer, F.G. 2015. A flexible multiscale approach to overlapping community detection. Soc. Netw. Anal. Min. (2015) 5: 23. doi:10.1007/s13278-015-0259-z.

[19] Rodriguez A，Laio A. 2014. Clustering by fast search and find of density peaks. Science. 344, 1492.

[20] Gregory, S. 2010. Finding overlapping communities in networks by label propagation. New J. Phys. 12, 10301.

[21] Wang M, Zuo W, Wang Y. 2015. An improved density peaks-based clustering method for social circle. Neurocomputing 179 (2016) 219–227

[22] Chris Gaiteri, Mingming Chen, Boleslaw Szymanski, Konstantin Kuzmin, Jierui Xie, Changkyu Lee, Timothy Blanche, Elias Chaibub Neto, Su-Chun Huang, Thomas Grabowski, Tara Madhyastha and Vitalina Komashko. 2015. Identifying robust communities and multi-community nodes by combining topdown and bottom-up approaches to clustering. *Scientific Reports* **5** Article number: 16361, 2015.

[23] Lancichinetti A, Fortunato S. 2009. Community detection algorithms: a comparative analysis. Phys. Rev. E 80, 056117.

[24] Collins, L. M. and Dent, C. W. 1988. Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. Multivar. Behav. Res. 23, 2 (Feb.), 231–242.

[25] Shen, H., Cheng, X., Cai, K., and Hu, M.-B. 2009. Detect overlapping and hierarchical community structure. Physica A 388, 1706.

[26] Nicosia, V., Mangioni, G., Carchiolo, V., and Malgeri, M. 2009. Extending the definition of modularity to directed graphs with overlapping communities. J. Stat. Mech., 03024.

[27] Chen, M., Szymanski, B.K. 2015. Fuzzy overlapping community quality metrics. Soc. Netw. Anal. Min. (2015) 5: 40. doi:10.1007/s13278-015-0279-8.

Xueying Bai received the Bachelor Degree in 2016 from the College of Computer Science and Technology, Jilin University, China. Bai is currently a graduate student in the University of Virginia, USA. Her research interest mainly focuses on Machine Learning.

Peilin Yang received the Bachelor Degree in 2016 from the College of Computer Science and Technology, Jilin University, China. Yang is currently a graduate student in UC Riverside, USA. His research interest mainly focuses on Data Mining.

Xiaohu Shi received the PhD degree in computer application technology from Jilin University, Changchun, China, in 2006. He is currently a professor in the College of Computer Science and Technology, Jilin University, China. He has published more than 50 journal and conference papers. His current research interests include machine learning and bioinformatics.

Highlights

- An overlapping community detection algorithm based on density peaks is developed.
- Considering node degree, the distance matrix construction algorithm is proposed.
- A three-step process to select community cores is designed.
- Using membership vectors, nodes are allocated to overlapping communities.
- Experiments prove the effectiveness of OCDDP compared with the top existing methods.