Efficient and scalable detection of overlapping communities in big networks

Tianshu Lyu*, Lidong Bing†, Zhao Zhang*, Yan Zhang*
*Department of Machine Intelligence, Peking University, Beijing, China
{llyutianshu,1501214514,zhyzhy001}@pku.edu.cn
†Machine Learning Department, Carnegie Mellon University, Pittsburgh, USA
lbing@cs.cmu.edu

Abstract—Community detection is a hot topic for researchers in the fields including graph theory, social networks and biological networks. Generally speaking, a community refers to a group of densely linked nodes in the network. Nodes usually have more than one community label, indicating their multiple roles or functions in the network. Unfortunately, existing solutions aiming at overlapping-community-detection are not capable of scaling to large-scale networks with millions of nodes and edges.

In this paper, we propose a fast overlapping-community-detection algorithm — FOX. In the experiment on a network with 3.9 millions nodes and 20 millions edges, the detection finishes in 14 minutes and provides the most qualified results. The second fastest algorithm, however, takes ten times longer to run. For another network with 22 millions nodes and 127 millions edges, our algorithm is the only one that can provide an overlapping community detection result and it only takes 238 minutes. Our algorithm draws lessons from potential games, a concept in game theory. We measure the closeness of a node to a community by counting the number of triangles formed by the node and two other nodes form the community. Potential games ensure that the algorithm can reach convergence. We also extend the exploitation of triangle to open-triangle, which enlarges the scale of the detected communities.

Index Terms—Community detection, Potential Games, Heuristic

I. INTRODUCTION

Community detection is a fundamental and important work. Communities are groups of densely connected nodes in the network. Birds of a feather flock together and nodes in the same group may have certain characteristics in common in different fields, while those characteristics sometimes are not apparent to the researchers. Community detection gives the researchers a chance to gain insight into the related field. Overlapping communities allow nodes to belong to more than one community. In a social network, it is well-understood that people are naturally characterized by multiple community memberships, for instance, family circles and workmate circles. In [1], the authors show that overlap is indeed a significant feature of many real-world social networks.

However, no existing algorithm gives a fast resolution for overlapping community detection on large-scale networks. When the numbers of nodes and edges add up to several millions, the algorithm [2]–[6] either can only detect a rather small number of communities, or has to spend over a day giving a seemingly satisfactory result as our experiments show. Another problem is that most community detection algorithms have no relation to a systematic theory of the emergence of communities [5]. Algorithms based on modularity optimization or dense subgraph detection exploit the structure of the network and aim to maximize the value of a global or local function (e.g. modularity [7], conductance [8]). They try to define the patterns of community structure, but the fact is that there is no acknowledged definition of community.

In this paper, we explore the problem of overlapping community detection on big networks from the perspective of community formation mechanisms. As we all know, the emergence of community is the consequence of humans’ interactions. People have conflict and cooperation and tend to be with the best match friends at any time. Each player decides which community to join independently, but instead, the choice is determined by other players’ choices. In mathematics, this dynamic process can be modeled by Potential Games in game theory. Furthermore, we find that there is a connection between Potential Games and heuristic algorithms. On the basis of Potential Games, we propose FOX (Fast Overlapping Community Search) framework, which is a principle, neat and adequate solution for community detection task. FOX is capable to process both weighted and unweighted graphs. It can also help improve the detection results provided by other algorithms.

In the experiments, we evaluate our algorithms from two aspects. One is on the networks with ground-truth and evaluating the results by F1-score and NMI. The other is on two large-scale networks: (1) the phone-call data provided by a mobile phone company spanning 3 months, and (2) the Google+ social network. The phone-call dataset is composed of over 3.9 million users and their 20 million communication patterns. Google+ includes 22 millions users and 127 million mutual followed relationships. Baselines include overlapping and non-overlapping-community detection algorithms. FOX achieves the best performance in the networks with ground-truth. We also use FOX as post process to improve the detection results provided by non-overlapping-community detection algorithms. As for the large-scale networks, our algorithm has distinct speed advantages over any other state-of-the-art algorithms and the quality of the detection results is also the best. For the Google+ social network, our algorithm is the only overlapping-community detection algorithm that can produce
results. Give the above, our algorithm can tackle large-scale networks quickly, and give a surprisingly good detection result at the same time.

We conclude three main contributions of our research:
- We explore the connection between Potential Games and heuristic algorithm and confirm that heuristic algorithm is an adequate solution for overlapping-community-detection task.
- We develop a heuristic function and its corresponding approximation on the basis of existing works. The approximation can guarantee the efficiency and quality of the detection results. Our algorithm is the fastest overlapping-community-detection algorithm to the best of our knowledge.
- Our algorithm can be used on both unweighted and weighted big graph for community detection task. Moreover, it can also be used to improve other algorithms' detection results.

II. RELATED WORK AND BACKGROUND

A. Overlapping Community Detection

Community detection is a growing field of interest in many areas. Most researches focus on uncovering disjoint communities. And many disjoint community detection algorithms are now available for large networks [9]–[12]. But for overlapping-community detection, more common in real networks, the scalability of which is unsatisfactory. Most overlapping-community-detection algorithms are based on finding a predefined community structure or maximizing a mathematical criterion. The problem is that both of these two ways can’t reveal the process of community emergence. And the algorithm quality and efficiency mainly rely on how to define the community structure or the objective function. Clique Percolation [13] treats adjacent cliques as communities. Generative models include Mixed Membership Stochastic Block (MMSB) Model [3], [14] and Nonnegative Matrix Factorization (NMF) [2]. The major limitation of NMF is the high cost of time and memory due to the matrix multiplication. Algorithms based on expanding the community locally from the seed (Louvain [15], OSLOM [16]) use a benefit function (modularity, conductance and etc.) to decide which node will be absorbed into the communities. They have a good performance in scalability, while on large-scale network, the partition quality is disappointing.

Community is the product of human social activities. Individuals tend to be with people with similar interests. Everyone independently choose the best fit groups of people to join. The community emergence process can be seen as a Game as described in Section I. Game theory is firstly associated with the formation of community in [17]. Instead of optimizing a mathematical criterion, the game-theory-based algorithms are more natural and have relatively stable performance in different kinds of networks [18]. But these algorithms are not doing well on scalability [5], [19]–[22].

B. Community Scoring Functions

As there is no standard definition of community, researchers have different scoring functions to assess communities. In [8], the authors gather thirteen commonly used functions and evaluate their performances from 4 aspects: separability, density, cohesiveness and clustering coefficient. We select four typical scoring functions to present in Fig. 1. Rank 1 is the best performance among all the scoring functions. Results show that Triangle Participation Ratio (TPR) performs best in density, cohesiveness and clustering coefficient, but scores badly on separability. Conductance, on the contrast, does well in separability and badly in other 3 aspects. The authors conclude that conductance suits for well-separated non-overlapping communities and TPR suits for heavily overlapped community.

In our algorithm, we use WCC (Weighted Community Clustering) [23] instead of TPR. In fact, both of them are established from the intuition that community formation is highly relevant to the number of triangle structure. Actually, WCC is put forward on the basis of TPR, not only taking the number of triangles in one community into consideration, but also counting the number of nodes that can compose a triangle. Our experiments show that WCC indeed performs well on all datasets. Section III-A presents how we develop WCC in details.

WCC is used in many community detection algorithms [24]–[26]. SCD [25] is a disjoint community detection algorithm that shows great advantages on large-scale networks. The main reason for the scalability is the approximation of WCC, while there exist some flaws in the approximation. We also believe that the algorithm doesn’t meet the intuition of potential games so that the final result provided by SCD is imperfect. We revise the flaws and the whole iterative process to deal with the overlapping-community-detection task. In [26], the authors present a distributed disjoint community-detection algorithm based on WCC. Designing a distributed algorithm isn’t the focus of this paper, but it indicates a possible direction to improve our algorithm.

III. POTENTIAL GAME BASED SELF-ADAPTATION

We start by developing a community formation model with an intuitive example, and illustrating how it responds to the change of the group membership; with this in place, we can then introduce the game-theoretic aspects of the problem.
We represent a social network by a weighted graph: we consider nodes to be individuals and edges to be the positive relationships (e.g., friendship, sharing the same hobby). To make this concrete, consider the graph in Fig. 2. There are some initial partitions in (a). Phil joins two groups at the same time, as he connects tightly with the movie fans and hiking amateur. Lily is not satisfied with her condition and want to join the hiking group as the hiking group members are more similar to her. So does Jay. Because of their movements, Haley and Luke also join the hiking group. In (b), the new outdoor sports amateur group is formed. For Phil, most members in this new group are interested in cycling, but he isn’t. The connection between Phil and the sports group isn’t as strong as that is in (a). So in (c), he leaves the sports group. At last, every student is satisfied with his/her group members and doesn’t change their group memberships.

The above procedure fits very naturally into the game-theoretic framework. Game players correspond to the nodes and tend to join the best fit group, which means players will always choose the strategy with the best payoff. Note that the payoff to each player depends on the strategies chosen by all players. Just like what we do in daily life, all nodes in our model constantly update their community membership according to the best responses (best-response dynamics). A Nash equilibrium is a list of strategies, one for each player, so that each player’s strategy is a best response to all the others. Therefore, the Nash equilibrium corresponds to the best community memberships.

![Fig. 2. Community formation game.](image)

In this section we will show how we get the Nash equilibrium point for community detection. Potential Game [27] is a special model in game theory, in which it has been proved that the best-response dynamics always converges to a Nash equilibrium when the payoff for each player is related to a global payoff function [28]. To be more specific for our model, every node’s judgement depends on whether its movement will contribute to the closeness from a global view.

If we consider best-response dynamics from the perspective of algorithms, it has the same intuition as heuristics does. The point is, if and only if the heuristic rule is related to improve the whole partition, a heuristic algorithm is adequate to solve the community formation problem. Next, we will discuss about the heuristic rules in detail.

A. Heuristic Function: WCC

WCC [23] is a metric about the closeness between a node and a community. Given a graph \( G(V, E) \), a node \( x \) and a community \( C \),

\[
WCC(x, C) = \begin{cases} 
\frac{t(x, C)}{t(x, V)} \cdot \frac{vt(x, V)}{|C|-|\{x\}|+vt(x, V-C)}, & \text{if } t(x, V) > 0; \\
0, & \text{if } t(x, V) = 0.
\end{cases}
\]

(1)

where \( t(x, C) \) stands for the number of triangles that node \( x \) closes with the nodes in \( C \) and \( vt(x, C) \) stands for the number of nodes in \( C \) that close at least one triangle with node \( x \) and another node in \( V \). \( C-\{x\} \) stands for the remaining part of \( C \) when taking out \( x \). Together, the level of closeness between node \( x \) and community \( C \) is denoted by \( WCC(x, C) \). Simply stated, the first part is about the triangles composed of node \( x \), and the bigger is better. The second part is about the nodes that close a triangle with \( x \). \( C \) should contain as many as possible of this kind of nodes.

We expand this metric further for the sake of overlapping-community detection. The \( WCC \) value of a community \( C_i \) is the sum of its members’ \( WCC \) value.

\[
WCC(C_i) = \sum_{x \in C_i} WCC(x, C_i)
\]

(2)

For a community partition \( P = \{C_1, C_2, \ldots, C_k\} \),

\[
WCC(P) = \sum_{i=1}^{k} WCC(C_i)
\]

(3)

One node in \( P \) changes its community membership and the new partition is \( P' \). We define the heuristic function \( \Delta \) as \( WCC(P') - WCC(P) \).

We further develop the concept of open-triangle, which is a structure composed of 3 nodes and 2 edges. Triangle describes the condition that three people are mutual friends, while open-triangle is the phenomenon that two strangers have one common friend. According to Triadic Closure [29], two people may be friends if they have many mutual friends. Therefore, open-triangle could also be used as scoring function just as triangle does. The difference is that open-triangle can detect the potential relationship between nodes and enlarge the community scale. In the following parts, s-FOX denote as the algorithm using open-triangle.

B. Self-adaptation Strategies

During every iteration, nodes have 4 strategies of movements: (1) do not move, (2) leave the community and be alone, (3) transfer to another community and (4) stay and at the same time join in another community. The last choice makes the community overlapped. The benefit of every
strategy is estimated by the heuristic function, and every node tends to make the best choice to maximize the heuristic function $\Delta$. Next we will derive the payoff of the 4 strategies (denoted as $\Delta_S$, $\Delta_L$, $\Delta_T$ and $\Delta_C$ respectively).

[Strategy 1] Stay and do not move:

The partition doesn’t change at all.

$$\Delta_S = WCC(P) - WCC(P) = 0$$

[Strategy 2] Leave and be alone:

Suppose the original partition is $P = \{C_1, C_2, \ldots, C_k\}$ and when node $x$ leaves its community $C_k$, the partition $P' = \{C_1, C_2, \ldots, C_k', \{x\}\}$, where $C_k' = C_k \cup \{x\}$

$$\Delta_L(x, C_k) = WCC(P') - WCC(P)$$

Especially when community $C_k$ is a singleton community, $\Delta_L(x, C_k) = 0$. $\Delta_L(x, C_k)$ is an important factor and all the following derivations are based on it. In the next Section we will propose its approximation.

[Strategy 3] Transfer to another community:

Suppose that node $x$ transfers from $C_1$ to $C_k$, the original partition is $P = \{C_1, C_2, \ldots, C_k\}$ and the new partition is $P' = \{C_1', C_2, \ldots, C_k\}$, where $C_1' = C_1 \cup \{x\}$ and $C_k' = C_k \cup \{x\}$. This movement is actually a composite transformation of 2 steps. Step 1: node $x$ leaves $C_1$ and doesn’t join any community, $P_m = \{C_1', C_2, \ldots, C_k, \{x\}\}$. And step 2: node $x$ join $C_k$, $P' = \{C_1', C_2, \ldots, C_k\}$. We can easily figure out that step 2 is an inverse transformation of Strategy 2.

$$\Delta_T = (WCC(P') - WCC(P_m)) + (WCC(P_m) - WCC(P))$$

$$\Delta_T = \Delta_L(x, C_1) - \Delta_L(x, C_k)$$

We denote the best transferred community as the community which has the biggest $\Delta_L(x, C_k)$. The WCC improvement of this best transferring choice is $\Delta_T(x, C_{best})$

[Strategy 4] Do not move and at the same time join in another community:

Suppose that node $x$ copies itself to $C_k$, $P = \{C_1, C_2, \ldots, C_k\}$ and $P' = \{C_1, C_2, \ldots, C_k', \{x\}\}$, where $C_k' = C_k \cup \{x\}$. Also, this is a composite transformation. The intermediate state is $P_m = \{C_1, C_2, \ldots, C_k, \{x\}\}$. Similarly, the WCC improvement of the best community is $\Delta_C(x, C_{best})$.

$$\Delta_C = (WCC(P') - WCC(P_m)) + (WCC(P_m) - WCC(P))$$

$$\Delta_C = -\Delta_L(x, C_k') + WCC(x, \{x\})$$

$$\Delta_C = -\Delta_L(x, C_k')$$

For all of these 4 strategies, $x$ will choose the one that can maximize the payoff.

$$\text{Strategy}(x) = \max(\Delta_S(x), \Delta_L(x), \Delta_T(x, C_{best}), \Delta_C(x, C_{best}))$$

Simply stated, in every iteration, if $x$ is negative for its community, it will be removed from the community anyway. If $x$ also hurts all of the other communities, it will be alone. Otherwise $x$ will transfer to the best suitable community. But when $x$ is beneficial to its community, it will stay there and consider whether to join other communities which it can bring the most benefit. Here, the benefit is the WCC improvement namely the enhancement of connectedness. The whole process is shown in Algorithm III-B.

IV. ALGORITHM

FOX and s-FOX are aiming at detecting communities of unweighted or weighted communities by counting triangles and open-triangles respectively. The main steps include partition initialization (pre-treatment), best response dynamics and post-treatment. For simplicity, the following discussion is based on unweighted graph.

A. Pre-treatment

To initialize the first partition, we employ local clustering coefficient, which is well-matched to the main procedure of FOX. The local clustering coefficient (CC) of a node in a graph quantifies how close its neighbors are to being a complete graph. We can infer that the bigger CC of a node is, the larger probability that its neighbors and itself form a community will be.

$$CC(i) = \begin{cases} \frac{k_i}{d_i(d_i-1)/2}, & \text{if } d_i > 1; \\ 0, & \text{if } d_i \leq 1. \end{cases}$$

where $k_i$ denotes the number of edges between two neighbors of node $i$, and $d_i$ denotes the degree of $i$. 

---

**Algorithm 1 Strategy(x)**

1: remove $\leftarrow 0$
2: transfer $\leftarrow 0$
3: copy $\leftarrow 0$
4: max $\leftarrow 0$
5: $C \leftarrow \text{CurrentCommunity}$
6: if $\Delta_L(x, C) > 0$ then
7: remove $\leftarrow 1$
8: end if
9: for all $C$ in AdjacentCommunities do
10: if remove then
11: if $\Delta_T(x, C) > \text{max}$ then
12: max $\leftarrow \Delta_T(x, C)$
13: bestCommunity $\leftarrow C$
14: transfer $\leftarrow 1$
15: end if
16: else
17: if $\Delta_C(x, C) > \text{max}$ then
18: max $\leftarrow \Delta_C(x, C)$
19: bestCommunity $\leftarrow C$
20: copy $\leftarrow 1$
21: end if
22: end if
23: end for
24: if transfer then
25: remove $\leftarrow 0$
26: end if
We first compute the CC of all nodes and rank them in decreasing order. If two nodes have the same CC, the one has more edges ranks first. Next, for the top-ranked node, all of its neighbors are marked as visited and added into a community. Then in the unvisited node set, pick the top-ranked node and its unvisited neighbors out to form a community. This work will stop when all nodes have been visited. Apparently, the first community partition is disjoint.

B. Best response dynamics

Best response dynamics is the community formation progress. From the analysis above, we can find that the computation of $\Delta_L(x, C)$ plays a significant role. Both $\Delta_T$ and $\Delta_C$ are computed on the basis of $\Delta_L$. But we can imagine the huge consumption of time when counting the triangle of every node, especially for those big scale and densely overlapped communities. Given the number of nodes $n$ and the average node degree $d$, the computation complexity of this step is $O(nd^2)$. In order to apply our algorithm to large-scale network, we propose an approximation from a statistical standpoint and the complexity decreases notably to $O(n)$.

1) Counting triangles approximately: In Section III-A, we present the detailed calculation method of $WCC(x, C)$. It’s about the triangle number between a node $x$ and a group of node $C$. We assume that the more edges between $x$ and $C$, the more triangles between $x$ and $C$. Based on this assumption, we approximately calculate the number of triangles by further assuming that: (1) in a community, any two nodes are connected with the same probability; (2) every edge closes at least one triangle in densely overlapped network; (3) all nodes’ local clustering coefficients have similar values. When node $x$ is outside the community $C$, we approximately calculate the number of triangles as

$$\hat{t}(x, C) = \binom{d_x}{2} \cdot p$$

$$\hat{t}(x, V - C) = \binom{d_x - c}{2} \cdot cc$$

$$\hat{v}(x, V - C) = d_{V - C}$$

where $V$ is the node set of the graph, $p$ is the probability that two random nodes in community $C$ are connected, $d_C$ is the number of edge between $C$ and $x$ and the clustering coefficient of the graph is $cc$. When node $x$ is a member of $C$, the approximation is similar. The approximation is proved to be reasonable in Section IV-D.

The approximation calculation of open-triangle is similar to the calculation of triangles. Suppose that node $x$ is outside the community $C$ and nodes $y$ and $z$ belong to $C$, there are two types of open-triangles that $x \ y \ z$ can form: $x - y - x$, and $x - y - z$.

$$\hat{t}(x, C) = \binom{d_x}{2} + d_C \cdot (|C| - d_C) \cdot p$$

2) Classifying the nodes in $C_k$ into 2 types: Continuing the analysis of Strategy 2, the difference between $P$ and $P'$ is the departure of node $x$. Only node $x$ and the nodes in community $C_k$ get a new $WCC$ after this movement. Therefore, when calculating the difference of $WCC(P)$, we only have to calculate the $WCC$ change of nodes in $C_k$ including $x$. The specific derivation is in Appendix A.

$$\Delta_L(x, C_k) = \sum_{n \in C'_k} (WCC(n, C'_k) - WCC(n, C_k \cup \{x\}))$$

$$- WCC(x, C_k \cup \{x\})$$

(9)

For all the nodes in $C_k'$, they can be divided into 2 categories, node sets $N$ and $M$. Nodes in $N$ are the neighbors of node $x$, and nodes in $M$ are not. Assume that nodes and edges density in every segment of the whole graph are homogeneous. Segments include $N$, $M$ and $G-C'_k$. Next, we can calculate the $WCC$ of nodes in $N$ and $M$ respectively to further simplify Equation 9.

$$\Delta_L(x, C_k) = \sum_{n \in N} (WCC(n, C'_k) - WCC(n, C_k \cup \{x\}))$$

$$+ \sum_{n \in M} (WCC(n, C'_k) - WCC(n, C_k \cup \{x\}))$$

$$- WCC(x, C_k \cup \{x\})$$

$$= |N| \cdot \Delta(a) + |M| \cdot \Delta(b) - WCC(x, C_k \cup \{x\})$$

$\Delta(a)$ denotes the average $WCC$ difference of the nodes in $N$ when $x$ leaves from $C_k$. Or $a$ can just be seen as a random node in $N$. $\Delta(b)$ is the average difference of nodes in $M$.

$$\Delta(n) = WCC(n, C'_k) - WCC(n, C_k \cup \{x\})$$

Next, we discuss these 3 kinds of nodes, nodes in $N$, nodes in $M$ and node $x$, respectively. The statistics we need are

- $d_{in}$: the number of edges between $x$ and $C_k$
- $d_{out}$: the number of edges between $x$ and $G-C_k$
- $p_{in}$: the probability that two nodes in $C_k$ are connected by an edge
- $p_{ext}$: the clustering coefficient of the graph
- $q$: the average number of edges between nodes in $C_k$ and nodes in $G-C'_k\setminus\{x\}$
- $S$: the size of $C_k'$
- $p$: the average degree of the whole graph

When one node finishes implementing its best strategy, these statistics also need to be updated. The computation complexity is $O(d)$, where $d$ is the average degree.

With the help of these statistics and Equation 5 6 7, we can approximately calculate Equation 1 and the value of $\Delta_L(x, C)$. The specific derivation is in Appendix B.

$$\Delta(a) = \frac{0.5(5-1)(S-2)p_{in} + (d_{in} - 1)p_{in} + (q - 1)p_{ext} + (q S + q - 1)p_{ext}}{S + q}$$

$$\Delta(b) = \frac{0.5(5-1)(S-2)p_{in} + 0.5(5-1)(S-2)p_{in} + (q - 1)p_{ext} + (q S + q - 1)p_{ext}}{S + q(S - 1 + q)}$$

$$WCC(x, C'_k \cup \{x\}) = \frac{1}{S + d_{out}}$$

C. Post-treatment

Nodes move in turns and this will inevitably bring a problem of community connectivity. We are curious that this problem isn’t mentioned in [5], [21], [22], [25]. As shown in Fig. 3, nodes with different colors belong to different communities. Fig. 3(a) is the original state in certain iteration. Suppose that
node with bigger ID number moves first, and node 10 and node 9 choose to join into the node 5’s community. However, when it’s node 5’s turn, node 5 decides to join into node 6’s community. Fig. 3(b) shows the final state in this iteration. Node 10 and node 9 have no connection with other nodes in the same community. This kind of misguided node makes some of the detected community become an unconnected graph.

![Graph](image)

**Fig. 3.** An example of how an unconnected community forms.

We put forward two strategies coping with the misguided nodes. First is the order of nodes. Nodes with higher degree move first, as they are more influential (like the role of node 5 in Fig. 3). Second, we conduct a connectivity analysis after the algorithm all the unconnected communities will be marked. All the nodes in unconnected communities will then choose the best community, which must be connected, to join into.

Some small communities may be entirely included in some bigger communities. This kind of small community is ignored in the final partition result.

**D. Is the Approximation Reasonable?**

To prove that our approximation is reasonable, we also develop an algorithm, FOX-naive, in which \( \Delta_L(x, C_k) \) is precisely calculated. The termination criteria of FOX-naive and FOX are also different. In FOX-naive, best response dynamics is perfectly performed, and at last all nodes satisfy with their situations and choose to stay in their communities. But in FOX, the approximation is not precise enough to reach the stable condition as FOX-naive does. In FOX, after each iteration, we compute the exact value of potential function \( WCC(P) \). When the difference between the \( WCC(P) \) of two iterations is less than a threshold \( t \), the algorithm stops. In Fig. 4 and Fig. 5, the experiment on three small datasets, we find that there are no obvious differences between the results provided by FOX-naive and FOX. Therefore the approximate calculation of \( \Delta_L \) is reasonable. Moreover, FOX-naive spends 3 days on the detection of Youtube datasets (over 1 million nodes and 2 million edges). On the contrary, FOX needs only 8 minutes and shows great efficiency.

As stated in Section III-B, nodes actually don’t use the specific \( \Delta \) value of each strategy for judgment, but their relative values instead. The approximation should retain the relative relationships between \( \Delta_S, \Delta_L, \Delta_C \) and \( \Delta_T \) as accurate as possible.

**E. Computational Complexity**

In the pre-treatment, the computational complexity is \( O(n^2) \). If FOX is used to improve a community partition, this step is omitted. The best response dynamics is the main step of FOX and s-FOX. The time cost of this phase depends on the number of nodes and the connection tightness between nodes. The computational complexity for one node computing the WCC improvement when joining into one community is \( O(1) \). Assume that the average degree is \( d \), every node has to consider its \( d \) neighbors’ communities in turn, and the computational complexity is \( O(d) \). Node also needs \( O(d) \) to update its attribution information. In every iteration, all nodes choose a strategy, and the complexity is \( O(dn) = O(m) \) where \( n \) is the number of nodes, and \( m \) is the number of edges. Updating the statistics used in the approximation costs \( O(m) \) too. Altogether, the time cost of one iteration is \( O(m) \).

**V. Experiments**

**A. Experimental set-up**

We test our algorithm on two types of datasets: networks with ground-truth and real large-scale networks, as given in Table I. The configuration of our computer is: two Intel(R) Xeon(R) CPU E5-2620 at 2.00GHz, 64GB of RAM. In FOX and s-FOX, the threshold \( t \) is set to 0.1%.

1) **Datasets: Network with ground-truth.** Most overlapping-detection algorithms use the benchmark datasets provided by SNAP [30]. We employ the collaboration network of DBLP, Amazon product co-purchasing network, and Youtube social network.

Clustering coefficient (CC) measures the tendency of a network to have highly connected clusters. As shown in Table I, DBLP has the best CC score, as each paper assigns to all its authors a fully connected clique. As for Youtube network, users in the same ground-truth community may not contact each other closely. So the CC of Youtube network is very poor.

Real large-scale networks. We analyze 2 big networks: a mobile communication network and the Google+ mutual followed relation network.

The mobile communication network is a phone-call record of a city, including 3.9 million users and lasting for 3 months. We draw an edge between 2 nodes only if the two users call each other more than one time. Google+ network is much bigger. The authors of [31] collected a weakly connected component of Google+, which includes over 70% of all Google+ users from Jul. 2011 to Oct. 2011. We only reserve the mutual followed relationships in the raw data.

We calculate the coverage of the maximal connected graph for each network to check if the above preprocessing hurts connectivity. Results show that the processed datasets are well-connected, refer to the fifth column in Table I.

2) **Baseline:** Compared baseline algorithms are given in Table II, including both disjoint-community (in italic text) and overlapping-community (in normal text) detection algorithms. The underlying theory of each algorithm is given in the third
column, and the time complexity is given in the fourth column. Refer to the citations for more details of these algorithms.

3) Metrics: As for the datasets with ground-truth, the main goal of the experiment is to evaluate the similarity between the ground-truth and the detected result. We use two evaluation metrics: Average F1-Score (F1) [2] and Normalized Mutual Information (NMI) [6]. Both of these two values are in [0,1], with 1 standing for perfect matching.

The phone-call record network and Google+ network have no ground-truth communities. Therefore, we evaluate the quality of the detected communities with the following metrics:

- **Density**: This is the average probability of the nodes in the same community being connected.

\[
\text{Density} = \frac{1}{k} \sum_{i=0}^{k} \frac{2m_i}{n_i(n_i - 1)}
\]

where \(k\) is the number of communities, \(n_i\) and \(m_i\) are the size of the \(i^{th}\) community and the edge number in the \(i^{th}\) community respectively.

- \(w_c/w_i\) is proposed to qualify community partition of phone-call record network [32]. There is no ground-truth community for phone-call record network. But the duration of one call can measure the degree of closeness between two users. We define the edge weight as the cumulative time of the phone calls between 2 users. \(w_c\) denotes the average edge weight of the edges inside the community, and \(w_i\) denotes the average weight of inter-community edges of the community. The higher the ratio of \(w_c/w_i\) is, the more intensive the communication within a community is, compared with the inter-community communication. This metric can be seen as the ground-truth of the phone-call record to certain extent.

- **Modularity**: The modularity measure, proposed in [7], was adapted to measuring overlapping community by [33].

\[
Q_{ov} = \frac{1}{2m} \sum_{c \in C} \sum_{i,j \in V} [r_{ijc} A_{ij} - s_{ic} k_i s_{jc} k_j] / 2m
\]

Please refer to [33] for the detail information about this metric.

### B. Experiments using ground-truth

1) Comparing with overlapping-community-detection algorithm: Fig. 4 and Fig. 5 present the performance of our two algorithms (i.e. FOX and s-FOX), FOX-naive and three baseline algorithms on three networks with ground-truth. Game run for over 3 days on Amazon dataset and achieved less than 5% of the total progress, so we terminated it and do not include its results in these two figures. FOX and FOX-naive perform very similar on these three datasets under NMI and F1 score. But, take the experiment on the Youtube dataset as an example, FOX-naive took 3 hours to finish the detection, while FOX only took 8 mins. It shows that FOX achieves encouraging efficiency improvement without sacrificing the effectiveness of community detection, which demonstrates the reasonability of our approximation in FOX. s-FOX and OSLOM get close scores, following FOX. Communities detected by s-FOX is bigger than that by FOX, as it contains more periphery nodes. So the results are not as good as FOX does. We have no acknowledged definitions of communities and the boundaries of communities are actually blurred. SVI and BigCLAM require the number of communities as input. We tend to set it as the number of ground-truth communities. But it’s too large for SVI to finish the detection task. Therefore, we set the community number to 1000 for SVI to finish the detection. Unfortunately SVI performs very badly.

The reason why people want to stay in the same group is because of their common identity or common bond [34]. For the ground-truth community, the group members share the common identity but they may not have high degree of structural similarity. On the other hand, all of the community detection algorithms use the structural information and the partition results are all bond-based communities. So it is no wonder that all algorithms perform poor on the Youtube network, as its communities are more likely identity-based, rather than bond-based. In our viewpoints, we think that the so-called ground-truth is just a community partition accepted by most people. There may not be any definitively correct

### TABLE I

Basic information of our datasets. \(N\): number of nodes. \(E\): number of edges. \(D\): average degree. \(D_{max}\): maximum degree. \(C\): clustering coefficient. \(M\): million. \(K\): thousand.

<table>
<thead>
<tr>
<th>Data Sets</th>
<th>(N)</th>
<th>(E)</th>
<th>(D)</th>
<th>(D_{max})</th>
<th>(C)</th>
<th>(N_c)</th>
<th>(CC)</th>
<th>Node</th>
<th>Edge</th>
<th>Community</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBLP</td>
<td>317K</td>
<td>1M</td>
<td>6.62</td>
<td>343</td>
<td>100%</td>
<td>13K</td>
<td>0.63</td>
<td>author</td>
<td>co-author</td>
<td>publication</td>
</tr>
<tr>
<td>Amazon</td>
<td>335K</td>
<td>926K</td>
<td>5.53</td>
<td>549</td>
<td>100%</td>
<td>75K</td>
<td>0.40</td>
<td>product</td>
<td>co-purchased</td>
<td>products category</td>
</tr>
<tr>
<td>Youtube</td>
<td>1.1M</td>
<td>3.0M</td>
<td>5.27</td>
<td>28754</td>
<td>100%</td>
<td>8K</td>
<td>0.08</td>
<td>user</td>
<td>follow</td>
<td>interest group</td>
</tr>
<tr>
<td>Phone-call</td>
<td>3.9M</td>
<td>20.5M</td>
<td>10.25</td>
<td>438</td>
<td>94%</td>
<td>-</td>
<td>0.12</td>
<td>user</td>
<td>make phone call</td>
<td>-</td>
</tr>
<tr>
<td>Google+</td>
<td>22.5M</td>
<td>127.3M</td>
<td>9.98</td>
<td>7347</td>
<td>94%</td>
<td>-</td>
<td>0.24</td>
<td>user</td>
<td>follow</td>
<td>-</td>
</tr>
</tbody>
</table>

### TABLE II

Baseline methods. \((n): node number. \(m\): edge number. \(c\): community number. \(k\): iteration number.\)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Principle</th>
<th>Complexity</th>
<th>Cite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Louvain</td>
<td>modularity</td>
<td>(O(n^2))</td>
<td>[15]</td>
</tr>
<tr>
<td>Infomap</td>
<td>information theory</td>
<td>(O(n^2 \log n))</td>
<td>[4]</td>
</tr>
<tr>
<td>SCD</td>
<td>heuristic</td>
<td>(O(mk))</td>
<td>[25]</td>
</tr>
<tr>
<td>SVI</td>
<td>MMSB</td>
<td>(O(cn_k))</td>
<td>[3]</td>
</tr>
<tr>
<td>GAME</td>
<td>game theory</td>
<td>(O(m^2))</td>
<td>[5]</td>
</tr>
<tr>
<td>BigCLAM</td>
<td>NMF</td>
<td>(O(cn + m))</td>
<td>[2]</td>
</tr>
<tr>
<td>OSLOM</td>
<td>local optimization</td>
<td>(O(n^2))</td>
<td>[16]</td>
</tr>
<tr>
<td>FOX-naive</td>
<td>counting triangles</td>
<td>(O(kmd))</td>
<td>Sec. IV-D</td>
</tr>
</tbody>
</table>
TABLE III
FOX IMPROVES THE PERFORMANCE OF DISJOINT ALGORITHMS

<table>
<thead>
<tr>
<th></th>
<th>NMI</th>
<th>F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DBLP</td>
<td>Amazon</td>
</tr>
<tr>
<td>Louvain+FOX</td>
<td>38%</td>
<td>22%</td>
</tr>
<tr>
<td>SCD+FOX</td>
<td>24%</td>
<td>0.7%</td>
</tr>
<tr>
<td>Infomap+FOX</td>
<td>18%</td>
<td>-</td>
</tr>
</tbody>
</table>

Fig. 4. NMI with ground-truth

Fig. 5. F1-score with ground-truth

2) Improving other algorithms’ result: Given a community partition by a third-party algorithm, FOX can also be used to improve the partition by applying the best-response dynamics. To test this, we apply FOX to the postprocessing step of these three disjoint community detection algorithms. The improvement ratio is showed in Table III. For SCD on Youtube dataset, FOX can’t improve the result anymore, so the improvement ratio is 0. Infomap gives very poor results on the Amazon dataset, none of the partitions is a connected graph. Therefore, FOX fails to improve its partition results. For other cases, FOX generally achieved remarkable improvements.

C. Experiments on big networks

There are a variety of problems for the compared methods to run on big networks, so that we have to make some compromise. SVI cannot handle the phone-call dataset because of its huge memory consumption. Game, as stated before, takes too much time to finish. BigCLAM can decide how many communities to detect automatically. But in this mode, BigCLAM cannot provide a result in three days. We set several community numbers for testing, and 150,000 is the biggest one that BigCLAM can finish detecting with an acceptable running time (38 hours). For all the community partition results, we only reserve the communities that have more than two nodes. The results of the phone-call network are given in Table IV.

We define the overlapping ratio as the average number of community labels per node. In BigCLAM, each node belongs to 4.5 communities on average, while the maximum is 453, which is rather unreasonable. In OSLOM, the community membership for every node is 1.2, which indicates that the overlapping isn’t apparent in this detection result. Our algorithms provide more reasonable community membership, 2 or 3 circles for each of the users, corresponding to the fact that we play a few roles in the society.

Coverage is defined as the ratio of nodes in the network that are partitioned into at least one community. It cannot be used as a metric for community quality evaluation, but it still show some important information of an algorithm. Low coverage shows that an algorithm neglects many nodes, but on the other hand, it may suggest that the community quality is very high.

Table V shows the algorithm performances on phone-call dataset. FOX is the fastest overlapping-community-detection algorithm. Although FOX and s-FOX do not take edge weight into account, their detection results are still better than BigCLAM and OSLOM on \( w_i/w_1 \). In the communities provided by OSLOM, the percentage of triangle sums up to 13.1%. However, in the communities provided by s-FOX the percentage is 2.5%. Triangle contributes a lot to edge density, as the edge density of triangle always equals 1.

The detection results for Google+ network is given in Table VI. The volume of Google+ data is about six times more than the volume of phone-call record. Our algorithms finish the de-
tection work for about 4 hours, which is the only overlapping-community-detection algorithm that can accomplish this work. OSLOM aborted when an exception occurred after running over 3 days. As shown in Table I, the Google+ network has the similar average degree as the phone-call network has, which shows Google+ is a quite dense network with large number of nodes, thus it is a very challenging dataset for community detection algorithms. As shown in Table VI, our algorithms’ the run time for Google+ is about fifteen times more than that for the phone-call data.

VI. CONCLUSION AND FUTURE WORK

In this paper, a fast overlapping-community-detection algorithm is proposed. When the scale of data is over 10 millions, it can provide a reasonable community partition within hours, which is the best performance on both accuracy and time-cost. This heuristic algorithm learns from potential games in game theory. Moreover, the approximation of the heuristic function ensures the quality of community and the speed of the detection. Our algorithm is also appropriate to the weighted graph, only if the edge weight is proportional to the closeness of relationship.

An interesting direction to pursue in future work is community evolution. FOX gives us a chance to quickly find communities in sequential snapshots and we can track the communities to uncover the evolution characteristics.

REFERENCES


APPENDIX

We can use Equation 2 and Equation 3 to expand $\Delta_L(x, C_k)$ and deduce the calculation formula to make it only related with nodes in $C_k$. Here, $\Delta_L(x, C_k)$ depicts the situation that $x$ leave $C_k$ and $C_k = C_k' \cup \{x\}$.

$$\Delta_L(x, C_k) = WCC(P') - WCC(P)$$

$$= \left( \sum_{i=1}^{k-1} (\sum_{n \in C_i} WCC(n, C_i)) + \sum_{n \in C_k'} WCC(n, C_k') \right) - \left( \sum_{i=1}^{k-1} (\sum_{n \in C_i} WCC(n, C_i)) + \sum_{n \in C_k} WCC(n, C_k) \right)$$

$$= \sum_{n \in C_k'} WCC(n, C_k') - \sum_{n \in C_k} WCC(n, C_k)$$

$$= \sum_{n \in C_k'} (WCC(n, C_k') - WCC(n, C_k \cup \{x\}))$$

$$- WCC(x, C_k' \cup \{x\})$$

A. Approximation of $\Delta(a)$

As shown in Equation 5 and 6, the triangle numbers between $a$, $C_k'$ and $C_k' \cup \{x\}$ can be calculated as

$$t(a, C_k') = \frac{1}{2} (S-1)(S-2)p_{in}^3$$

$$t(a, C_k' \cup \{x\}) = \frac{1}{2} (S-1)(S-2)p_{in}^3 + (d_{in} - 1)p_{in}$$

$V$ is the node sets of the graph. Besides $C_k' \cup \{x\}$, node $a$ can also close triangles with (1) a node in $C_k'$ and a node outside $C_k'$, (2) 2 nodes outside $C_k'$, (3) a node outside $C_k'$ and node $x$.

$$t(a, V) = t(a, C_k' \cup \{x\}) + (S-1)p_{in}p_{ext} + \frac{1}{2}q(q-1)p_{ext} + d_{out}p_{ext}$$

As we assume that every edge close at least one triangle in densely overlapped network, $vt(a, V)$ actually accounts for the number of $a$’s neighbors, including neighbors in $C_k'$, outside $C_k'$ and node $x$.

$$vt(a, V) = (S-1)p_{in} + q + 1$$

$$|C_k' - \{a\}| + vt(a, V - C_k') = (S-1) + (1+q) = S + q$$

$$|C_k' \cup \{x\} - \{a\}| + vt(a, V - C_k' \cup \{x\}) = S + q$$

Now we get the approximation of every multiplier in equation 1. We can further calculate the value of $\Delta(a)$.

The calculation of open-triangles are also listed as followed. There must be an edge between a and x. The third node p can be chosen from $N-\{a\}$ or M. If p belongs to N, whether node a and p are connected doesn’t matter. Node p can be any nodes in $N-a$. But if p belongs to M, there must be an edge between a and p.

$$t_s(a, C_k') = \frac{1}{2} (S-1)(S-2)p_{in}^3 * 3$$

$$t_s(a, C_k' \cup \{x\}) = t_s(a, C_k') + (S_{in} - 1) + (S - d_{in})p_{in}$$

The calculation of $t_s(a, V)$ is more complicated. (1)If the second node is x and the third node p is outside the community, the number of open-triangle equals $q + d_{out}$. (2)If a chooses two nodes outside the community to form a community, the number is $\frac{1}{2}q(q-1) + qp|V|-|C_k'|$. (3)If the second node is in $C_k'$ and the third one is outside $C_k$, the number is $2q(S-1)p_{in} + q |C_k'|/|V|$. The approximation of $\Delta(b)$ is almost the same as the approximation of $\Delta(a)$. As node b has no connection with node x, the biggest difference between $\Delta(a)$ and $\Delta(b)$ is the items related to x.

$$t(b, C_k') = \frac{1}{2} (S-1)(S-2)p_{in}^3$$

$$t(b, C_k' \cup \{x\}) = t(b, C_k')$$

$$t(b, V) = t(b, C_k') + (S-1)p_{in}q_{ext} + \frac{1}{2}q(q - 1)p_{ext}$$

$$vt(b, V) = (S-1)p_{in} + q$$

$$t_s(b, C_k') = \frac{1}{2} (S-1)(S-2)p_{in}^3$$

$$t_s(b, C_k' \cup \{x\}) = t_s(b, C_k')$$

$$t_s(b, V) = t_s(b, C_k' \cup \{x\}) + 2(S-1)p_{in}q + qp + \frac{1}{2}q(q - 1)$$

B. Approximation of $\Delta(x)$

Node x randomly chooses 2 nodes in $N$.

$$t(x, C_k' \cup \{x\}) = \frac{1}{2} d_{in}(d_{in} - 1)p_{in}$$

Besides $C_k' \cup \{x\}$, node x can also form a triangle with (1) 2 nodes outside $C_k'$, or (2) one in $C_k'$ and one outside $C_k'$.

$$t(x, V) = t(b, C_k' \cup \{x\}) + \frac{1}{2} d_{out}(d_{out} - 1)p_{ext} + d_{out}d_{in}p_{ext}$$

Node x chooses a node in $N$ and a node in $C_k'$ randomly.

$$vt(x, V) = d_{in} + d_{out}$$

$$t_s(x, C_k' \cup \{x\}) = d_{in}(S-1)p_{in}$$

Node x chooses a node outside $C_k'$ and another node in $G$ randomly.

$$t_s(x, V) = t_s(x, C_k' \cup \{x\}) + d_{out}p$$