IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 4, 2020, accepted June 25, 2020, date of publication June 30, 2020, date of current version July 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3006029

A Novel Community Detection Algorithm Based on Paring, Splitting and Aggregating in Internet of Things

YIZHE LI[1], HUI XIA[2], (Member, IEEE), RUI ZHANG[1], (Student Member, IEEE), BENXU HU[1], AND XIANGGUO CHENG[1]

[1]College of Computer Science and Technology, Qingdao University, Qingdao 266000, China
[2]College of Information Science and Engineering, Ocean University of China, Qingdao 266000, China

Corresponding authors: Hui Xia (xiahui@qdu.edu.cn) and Xiangguo Cheng (15964252399@163.com)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61872205, in part by the Shandong Provincial Natural Science Foundation under Grant ZR2019MF018, and in part by the Source Innovation Program of Qingdao under Grant 18-2-2-56-jch.

ABSTRACT The explosive growth of Internet of Things (i.e., IoT) terminal equipment makes its topology more complex, which leads to the increasing cost of network research. Recently, the implicit community structure is widely used to improve the efficiency of research. However, most of the non-overlapping community detection algorithms have some weakness, such as the large number of community detected and the obvious scale gap between communities. To address these abovementioned problems, we design a novel non-overlapping community detection algorithm, named as *P*airing, *S*plitting and *A*ggregating algorithm (i.e., *PSA*). Firstly, in order to improve the accuracy of community division, a new node similarity index is designed to transform the network into a large number of similar node pairs. Secondly, based on the connected branches composed of similar node pairs, the network is further divided into several similar node sets. Thirdly, to balance the scale gap of different communities, the *G*rasshopper *O*ptimization *A*lgorithm, (i.e., *GOA*) is introduced to combine the local attribute (i.e., conductance) and global attribute (i.e., modularity) together to aggregate similar node sets into potential (or final) communities. Finally, the experimental results show that *PSA* not only controls the difference among communities well, but also outperforms the other four popular algorithms in terms of two metrics. Moreover, we propose a community-based resource discovery method (or scheme), named as *C*ommunity-assisted *S*hort-distance-query *R*esource *D*iscovery algorithm (i.e., *CSRD*) to further verify the efficiency of *PSA*. The results show that the resource discovery efficiency of *CSRD* using *PSA* is better compared with other algorithms.

INDEX TERMS Community detection, grasshopper optimization algorithm, Internet of Things, non-overlapping community, resources discovery.

I. INTRODUCTION

The Internet of Things (i.e., IoT) [1] connects all kinds of sensors, controllers and embedded devices in the physical world to the internet through wired and wireless access, expanding the boundaries and capabilities of information systems to obtain information from the physical world. Based on the traditional information network with human semantics, a new dimension of device with data is introduced to form an abstract digital ecosystem with the fusion of physical world and human information domain. The number of devices in IoT reached 13.7 billion in 2019 and will reach 20.4 billion by 2020 [2]. As a result, we face many challenges in a practical application process, such as a huge number of devices, limitation of resource utilization, various kinds of multi-format real-time data and numerous historical records [3]. Several research teams have used the implicit community structure to reduce the research cost of the network. In graph theory, community structure refers to the sub-graph structure in which the inner connection is compact and the outer connection is sparse [4]. In order to improve the efficiency and quality of IoT, the IoT network can be

The associate editor coordinating the review of this manuscript and approving it for publication was Minho Jo.

subdivided by the community detection algorithm into several blocks with similar function or structure.

The community detection algorithms can be divided into non-overlapping community detection algorithms and overlapping community detection algorithms [5]. The former has advantages in time complexity and space complexity, which is more suitable for the IoT environment. Currently, the widely used non-overlapping community detection algorithms can be divided into the following three categories according to the different reference attributes, as global attribute-based method, local attribute-based method and hybrid method. (*i*) Global attribute-based method can accurately assign nodes with similar characteristics to the same community, while the high computational complexity makes them only suitable for small-scale network environments, not for the IoT environment with such a large number of devices; (*ii*) Compared with the former, the method based on local attributes can reduce the complexity of computation, while it will lead to a large gap between the sizes of communities (i.e., the number of community nodes) due to the local optima dilemma. The limitation will seriously affect the research quality of IoT; (*iii*) The hybrid method is superior to the global attribute-based method in time complexity and suitable for large-scale networks, and superior to the local attribute-based methods in accuracy. However, this method still has much room for improvement;

In order to solve the problems of the large number of community detected and the obvious scale gap among communities, a novel non-overlapping community detection algorithm is proposed in this paper, named as ***P***airing, ***S***plitting and ***A***ggregating algorithm (i.e., *PSA*), which can balance the scale gap of different communities and reduce the number of communities.

The main contributions of this paper are as follows:

(1) In order to assign the nodes with the same characteristics into the same community, a new index used for measuring the attraction among nodes is designed. We use this new index to improve the most commonly used node similarity index.

(2) In order to avoid the local optima dilemma, we introduce the ***G***rasshopper ***O***ptimization ***A***lgorithm, (i.e., *GOA*) in the aggregating phase. This optimization scheme makes it easier to find the optimal merging location for each similar node set, enhances the global searching ability of the algorithm, and improves the accuracy of merging process.

(3) In order to improve the efficiency of resource discovery in IoT environment, we propose a community-based resource discovery method, named as ***C***ommunity-assisted ***S***hort-distance-query ***R***esource ***D***iscovery algorithm (i.e., *CSRD*). This new method redesigns the functions of each node and realizes a new distributed resource discovery mechanism. Terminal devices in IoT can search and interact autonomously according to their requirements, thus forming a distributed, loosely coupled and easily expanded resource discovery system.

This paper is divided into six parts: the first part, we introduce the background of community detection in IoT, and classify the current community detection algorithms. In the second part, we introduce the related work of community detection. In the third part, we first introduce some knowledge about community detection in order to help us understand the rest of the article better, and then introduce the specific operation steps of *PSA*. In the fourth part, we compare the experimental results of *PSA* with other four methods to prove that *PSA*'s stability and accuracy. In the fifth part, we apply the community detection algorithm to the resource discovery domain of the IoT, and carry on the experimental proof. Finally in part six, we briefly summarize this article.

II. RELATED WORK

This section first introduces some related work of swarm intelligence optimization and then focuses on three types of community detection algorithms, and evaluates the relative merits of these algorithms.

A. THE SWARM INTELLIGENCE OPTIMIZATION

The main idea of ***P***article ***S***warm ***O***ptimization(i.e., *PSO*) is study the clustering behavior of birds, using the characteristics of bird population attracted by habitat to guide human decision-making process [6]. Chen *et al.* [7] add two genetic operators based on union-find into *PSO* process, namely crossover and mutation operations, which update the velocity and position of particles and further enhance the search ability. Liu *et al.* [8]–[10] define the Global Routing problem as a multicommodity flow problem, and adopt the partitioning strategy and *PSO* to reduce the size of the Global Routing problem. *GOA* [33] is inspired by the foraging behavior of grasshopper colony and can solve real problems with unknown search spaces which is consistent with the application scenario in this paper.

B. THE OVERLAPPING COMMUNITY DETECTION ALGORITHMS

Cheng *et al.* [11], [12] proposes an novel overlapping community change-point detection based on a signal processing framework and a decision function-based strategy, which can ensure higher accuracy and a lower false positive rate. Guo *et al.* [13] proposed a local community detection algorithm to discover communities accurately based on expanding the seeds by fitness function with internal force between nodes. Liu *et al.* [14] proposed an approach based on coarsening strategy and the local overlapping modularity to quickly detect overlapping communities. Although they are good methods to study the network structure, the high time and space complexity makes it difficult to adapt to the IoT network research.

C. THE NON-OVERLAPPING COMMUNITY DETECTION ALGORITHMS

1) THE GLOBAL ATTRIBUTE-BASED COMMUNITY DETECTION ALGORITHM

There are many algorithms for community detection based on global attributes. Girvan-Newman (*GN*) algorithm [15] performs community detection by gradually deleting edges

with high edge betweenness, which greatly promotes the development of community detection to a certain extent. Although the cost of the Algorithm is very high, the accuracy of the result is improved greatly. Arasteh and Alizadeh [16] proposes a new improved method of *GN* called the proposed methods of *GN* (*GNp*), which optimizes the definition of the edge betweenness and can delete multiple edges in an iteration. Blondel *et al.* [17] maximize the modularity by using a greedy algorithm and merging modules, and propose *Louvain* Algorithm whose time complexity is the square of the network size. Radicchi and Filippo [18] proposed a hierarchical clustering method called Kernighan-Lin (i.e., *KL*), which can divide the network into two parts with similar structure. Lin *et al.* have also combined *KL* with integer programming [19] methods for community detection.

The algorithms which only rely on global attributes often take the whole network structure as a parameter until all community structures are detected, which leads to the high complexity and it is not suitable for large-scale networks with complex structures like IoT [20], [21].

2) THE LOCAL ATTRIBUTE-BASED COMMUNITY DETECTION ALGORITHM

Random walk based algorithms are also effective local attribute-based methods. It is assumed that a random walker prefers to stay in the interior rather than the exterior of community. Random Walk with Restart [22] uses Markovian random walkers to explore the networks. With certain probability, the walkers jump back to seed nodes. Some community detection algorithms reduce the complexity of detecting community structure by introducing local attributes such as node similarity and adjacency matrix. Raghavan *et al.* [23] put forward a good method called label propagation algorithm (i.e., *LPA*) which has a nearly linear time complexity, but the stability of *LPA* algorithm is not satisfactory. Žalik and Rizman [24] has designed a bottom-up community detection method, which can achieve high accuracy by combining adjacent neighbors into the most similar set of nodes.

The local attribute-based community detection algorithm can achieve local optimal processing of network structure, but from the perspective of global, it turns to make relatively large differences in the number of nodes between communities, which not only causes the imbalance between the communities, but also is disadvantageous to the network research later.

3) THE HYBRID DETECTION ALGORITHM

The hybrid detection algorithm synthesizes the two attributes, so it can detect community structure accurately and has a low time complexity. Query-biased Densest Connect Subgraph method [25] weights nodes by random walk. Then the local community is selected to minimize a predefined goodness function. Colored Random Walk [26] is also a typical random walk based algorithms which find node as seed with global attribute in the beginning and then detect the communities with local random walk. Liu and Ma [27] proposed a hierarchical method called the divide and agglomerate algorithm (i.e., *DA*) which detect the communities by dividing the networks into many part and then merge them refer to the modularity. Although *DA* combine the local and global information, it ignores the node whose degree equals to 1, which is not conducive to practical application.

Although they are good methods to detect communities in complex network, this kind of methods is prone to local optimization and slow convergence because of greedy algorithm, which leads to the detection of a large number of communities, making it difficult to find a particular community structure.

In order to solve the problem of imbalance of community structure, we proposed a hybrid detection algorithm named *PSA*, which can be applied to large-scale networks and improve the community balance compared with other algorithms.

III. PAIRING, SPLITTING AND AGGREGATING ALGORITHM

A. RELEVANT KNOWLEDGE ABOUT COMMUNITY DETECTION

In this section, we systematically describe the community detection problem and review some of its key concepts.

1) PROBLEM STATEMENT

Given a undirected graph $G = (V,E)$ with a node set V and an edge set E, the graph can be represented as an adjacency matrix A, where $A_{ij} = 1$ if there is an edge between node i and node j, or $A_{ij} = 0$ if there is no edge between node i and node j. The purpose of community detection is to find the node set $C_1 C_2 \ldots\ldots C_i$, such that $C_1 \cup C_2 \cup \ldots\ldots \cup C_i = V$, where the node in set C_i has more links with the node in C_i than the node in other set.

2) MEASURES OF NODE SET QUALITY

There are some popular measures for gauging the quality of a node set: Cut, Ncut, and conductance [28]. Let us define $\varphi(A, B)$ to be the number of edges between node sets A and B.

Cut: The cut of set A is defined as the number of edges between set A and its complementary set. which can be write as follow:

$$cut(A) = \varphi(A, V \backslash A) \tag{1}$$

Ncut: The normalized cut of set A is defined by the cut with volume normalization as follow

$$N(A) = \frac{cut(A)}{\varphi(A, V)} \tag{2}$$

Conductance: The conductance is a classical metric to measure the quality of a community, which is defined to be the cut divided by the least number of edges incident on either set A or $V \backslash A$.

$$\vartheta(A) = \frac{cut(A)}{\min(\varphi(A, V), \varphi(V \backslash A, V))} \tag{3}$$

From the definition we can conclude that the conductance of A is always greater than or equal to the Ncut of A, and can avoid that the set A has too much nodes and edges. Farther more, we define the conductance of node set which is made of many connected components.

Definition 1: Given a node set C which has more than one connected components, $C = C_1 \cup C_2 \cup C_3 \cup \ldots \cup C_i$, the conductance of C can be signified as follow:

$$\theta(C) = \max(\vartheta(C_i)) \tag{4}$$

3) MEASURES OF COMMUNITY QUALITY

To evaluate our *PSA*, we need some criterions for measuring the accuracy of community partitions. One of the most wide-used index is the modularity Q proposed by Newman and Girvan [29].

$$Q = \frac{1}{2m}\sum_{vw}\left[A_{vw} - \frac{k_v k_w}{2m}\right]\theta\left(c_v, c_w\right) \tag{5}$$

where $m = |E|$, A is the adjacency matrix of G, and $\theta\left(c_v, c_w\right) = 1$, when node v and node w are in a same community, otherwise, $\theta\left(c_v, c_w\right) = 0$.

If we get the network data with the ground truth, another popular index called normalized mutual information (*NMI*) should be used, which measures how close are the detected communities and the ground truth [30].

$$NMI\left(X|Y\right) = \frac{H\left(X\right) + H\left(Y\right) - H\left(X, Y\right)}{\left[H\left(X\right) + H\left(Y\right)\right]/2} \tag{6}$$

where X is the ground truth, Y is the predicted communities by algorithms. $H(X, Y)$ mean the entropy of community X and the joint entropy of X and Y respectively.

4) SIMILARITY INDEXES

Many indexes, like jaccard, only concentrate on the proportional relation between the same neighbors and the whole neighbors ignoring the differences between each nodes [31], [32]. While the AA index [27] takes the differences between nodes into account, but the scarcity of proportional relationship leads to a great deal of uncertainty in the result, for example, in some cases the result of low-degree node is much more higher than the high-degree node which has more same neighbor [34].

$$\phi_{Jaccard} = \frac{\left|CN_{v_1 v_2}\right|}{\left|N_{v_1} \cup N_{v_2}\right|} \tag{7}$$

$$\phi_{AA} = \sum_{v \in CN_{v_1 v_2}} \frac{1}{\lg\left(K_v\right)} \tag{8}$$

In order to solve this problem, we propose an attractive concept, and design a new similarity standard:

Attraction $\Gamma(v_i, v_j)$ refers to the AA index which outstand the difference between nodes of different degrees. We modified the denominator to make sure that the denominator makes sense when $K_v = 1$.

$$\Gamma(v_i, v_j) = \frac{W(v_i, v_i)}{\lg(K_{v_i} + 1)} \tag{9}$$

where $W(v_i, v_j) = 1$ when there is a connection between v_i and v_j, and $W(v_i, v_j) = 0$ when there is no connection.

We propose a new similarity index by combining the $\Gamma(v_i, v_j)$ and jaccard index.

$$\Phi_{PSA}(v_i, v_j) = \frac{\sum\limits_{v_x \in CN(v_i, v_j)} \left[\Gamma(v_x, v_i) + \Gamma(v_x, v_j)\right]}{\sum\limits_{v_x \in N(v_i, v_j)} \left[\Gamma(v_x, v_i) + \Gamma(v_x, v_j)\right]} \tag{10}$$

Nowadays, the most popular similarity index Jaccard is only concerned with the consistency of neighbors between two nodes, ignoring the differences between neighbors. But in the network, each node has its own influence, the computation of the same value will affect the accuracy of the node similarity. In addition to the proportional relation of jaccard, Φ_{PSA} has improved it by introducing $\Gamma(v_i, v_j)$, which makes every neighbor node participate in similarity calculation to increase the accuracy.

5) GRASSHOPPER OPTIMIZATION ALGORITHM

Like most other swarm intelligence optimization methods, *GOA* has two steps of exploration and development, which ensures that the algorithm has strong global searching ability and can effectively avoid stagnation in local optimization [36]. The mathematical model for simulating grasshopper's behavior in nature is as follows:

$$x_i(t+1) = S_i(t) + G_i(t) + A_i(t) \tag{11}$$

where $x_i(t+1)$ represents the position of the grasshopper i in the *(t+1)* iteration, $S_i(t)$ represents the interaction between grasshoppers, $G_i(t)$ and $A_i(t)$ represent gravitational and wind effects.

In this paper, we ignore the influence of wind, set the modularity Q as the gravity factor $G_i(t)$, and set the conductance variation of two clusters as the interaction $S_i(t)$ between two clusters. So this model is changed to:

$$x_i(t+1) = \max(\Delta\theta_{i\to j}(t) + \Delta Q_{i\to j}(t)) \tag{12}$$

where *ci* and *cj* is the position of grasshopper *i* and *j*. cluster *cj* is one of the neighbors of cluster *ci*, cluster *ci* is always turn to merge with the cluster *cj* which can improve the conductance and modularity, and c is a parameter to control the process which is defined as follow:

$$c = n - t\frac{n}{T} \tag{13}$$

where n is the number of nodes in graph G, and T is the maximum number of iterations which is equal to (n-2) so that the number of communities is no less than 2 in this paper. We can generalize the above equation as follow:

$$x_i\left(t+1\right) = \max\begin{bmatrix} (n - t\frac{n}{T}) \\ *(\theta\left(ci{+}cj\right)\left(t\right) - \theta\left(ci\right)\left(t\right)) \\ +\Delta Q_{ci\to cj}\left(t\right) \end{bmatrix} \tag{14}$$

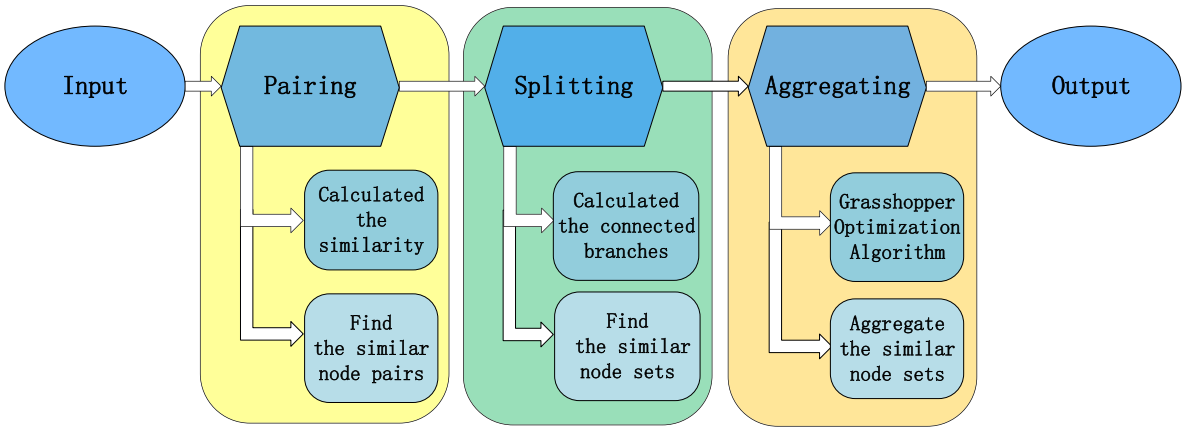


FIGURE 1. The framework of *PSA*.

B. PAIRING, SPLITTING AND AGGREGATING ALGORITHM

Our proposed method *PSA* is divided into three phases: Pairing Phase, Splitting Phase and Aggregating Phase. In the Pairing Phase, we simplify the network G by computing the similar node pairs, and in the Splitting Phase, we compute the connected branches by Depth-first Algorithm to form the similar node set C, in the Aggregating Phase, we aggregate the similar node set C with *GOA* to improve the modularity.

Zachary's karate club [37] is the most popular empirical network with ground truth in the domain of community detection, which is compiled by Zachary when he was studying the social relationships of karate members at American universities from 1970 to 1972. The network has 34 nodes and 78 edges, each node represents a member of the club and each edge represents the interaction between members. This club is split into two parts by the coach Mr. Hi and the club manager John.

Fig. 1 shows the overview of *PSA*. Fig. 2 shows the processes of *PSA*, where 2(a) is the original network of Karate, 2(b), 2(c) and 2(d) show the three phases of *PSA* in community detection.

1) PAIRING PHASE

Each network has a number of nodes that are extremely similar but not connected to each other, such as node 18 and node 20 in Karate, which, taking the whole network into account, must belong to one and the same community. So our task in this phase is to find the ***M***ost ***S***imilar ***N***ode (i.e., *MSN*), which is defined as follow:

Definition 2: Given a graph $G = (V, E), v_i \in V$, the most similar node v_j is the node which can reach the maximum value of $\Phi_{PSA}(v_i, v_x)$, which can be represent as $MSN\ (v_i) = v_x$.

Definition 3: Given a graph $G = (V, E), v, w \in V$, the hops between node v and node w is defined as the number of node from node v to node w.

Proposition 1: Given a graph $G = (V, E), v_i \in V$, the distance between v_i and $MSN\ (v_i) = v_x$ is no more than two hops.

Proof: The $MSN\ (v_i)$ must have more than one common neighbor with v_i, and there maybe have edge between v_i and $MSN\ (v_i)$, so the Proposition 1 is true.

Proposition 2: Given a graph $G = (V, E), v_i \in V$, N_i is the neighbor node set of v_i. We can get $MSN\ (v_i)$ by traversing N_i, who is the neighbor nodes of the node in set N_i (the two-hops nodes of v_i).

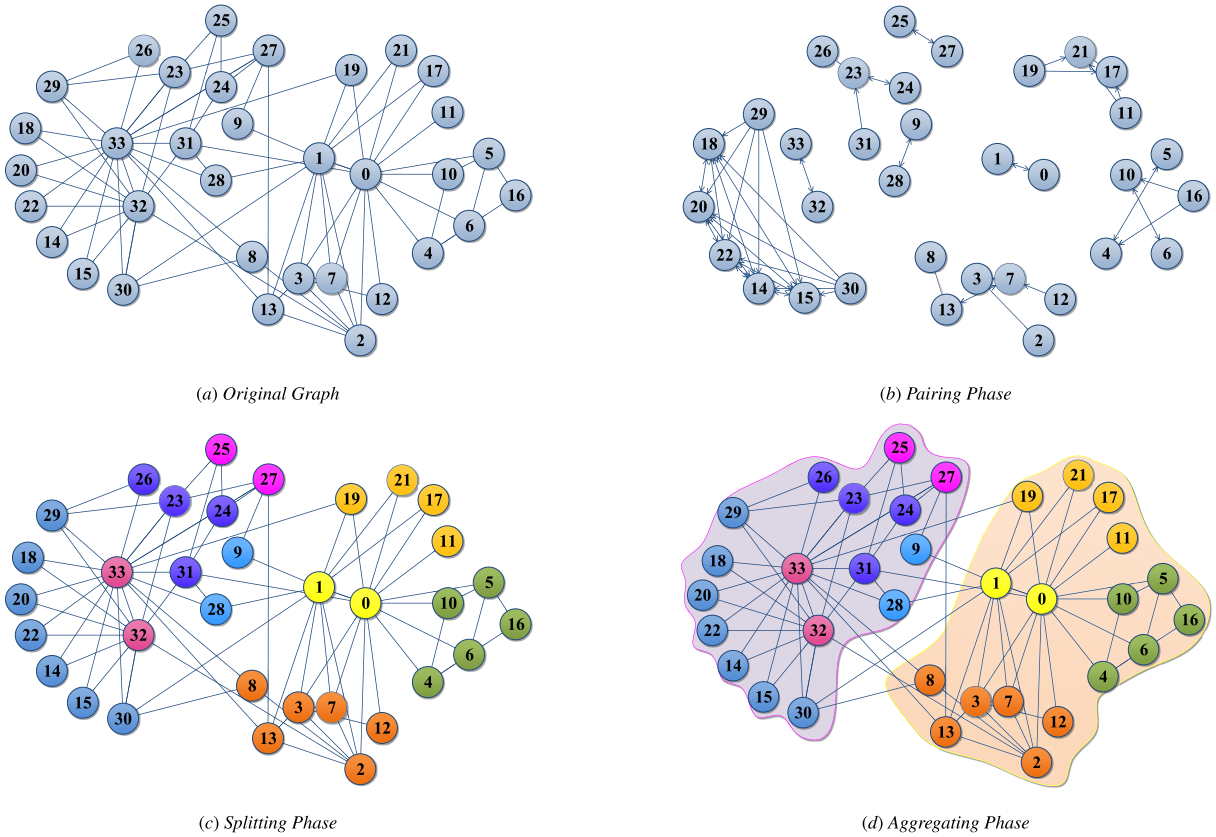
Proof: The distance between $MSN\ (v_i)$ and vi is no more than two hops, so it's not necessary to traversing all the node in V to find the $MSN\ (v_i)$, otherwise, we only need to traversing the two-hops nodes of v_i.

Algorithm 1 Pairing Phase Algorithm

Input: Graph $G = (V, E)$, the adjacency matrix A,the similarity matrix SM
Output: node pair set S
01: Initialize $SM_{i,j} = 0$ for $i, j \in v$
02: For each *node* in v:
03: For i, j in the neighbors of *node*:
04: If $SM_{i,j} == 0$:
05: Update $SM_{i,j} = \phi(i, j)$
06: Find the node j for every node i which can reach the max value of $SM_{i,j}$
07: Add (i, j) to the node pair set S

We can draw a conclusion by proving the proposition above that we can find the most similar node by traversing all its two-hops nodes, it may be too expensive for seeking the *MSN* for a single node, but when it comes to the whole graph, we can avoid a lot of repetitive computation by marking the node pairs calculated.

As the Algorithm 1 shows that the Pairing Phase will divide the whole graph into plenty of node pairs which are consisted of the most similar nodes. Firstly, the similar matrix *SM* is initialized, and then all the nodes are traversed: for each node traversed, the similarity of any two nodes i and j in its neighbor node will be calculated and the result will be recorded to $SM_{i,j}$. Finally, the most similar node pairs of each



(a) *Original Graph* (b) *Pairing Phase*

(c) *Splitting Phase* (d) *Aggregating Phase*

FIGURE 2. **The community detection in Karate with *PSA*.**

node are calculated according to the matrix *SM*. Noted that a node may have more than one nodes which can reach the max value of similarity. For example, node 14 in karate network has four most similar nodes, 15, 18, 20 and 22. In this phase, we split the entire network into multiple similar node sets with the references of the previously discovered similar node pairs.

Algorithm 2 Splitting Phase Algorithm

Input: The node pair set S
Output: The similar node set C
Initialize the similar node set C, temporary set count and *stack*
Constructing undirected graph $G' = (V', E')$ with set S
For each *node* in V':
    If *node* is not in *count*:
    Initialize temporary set *temp*
    Add *node* to *count, stack, temp*
    While *stack* is not empty:
    For *element* in *stack*:
        Add all the neighbors of *element* to count and stack
        Remove *element* from stack
    Add *temp* to set C

2) SPLITTING PHASE

In Algorithm 2, we construct a new topological graph using similar node pairs, and then combine similar node pairs with the same node to form a similar node set. In the first step, we take an unlabeled node in the graph and add it to the temporary set temp and stack. In the second step, we label and delete the nodes in the stack and add their neighbors to the stack, when the number of nodes in count is zero, the nodes in temp is moved to set C. Repeat this step until there are no unlabeled nodes in the graph. At this point, the whole network is split into multiple similar node set C.

Due to the particularity of Algorithm 1, there may be no connection between similar node pairs, so the nodes in the similar node set output in this phase are not necessarily connected.

3) AGGREGATING PHASE

We get multiple similar node sets (clusters) after the algorithm 1 and 2, and in this phase, we will aggregate these clusters through *GOA* and finally get the community structure.

In the Aggregating Phase, we set the number of similar node sets as the number of the grasshoppers, and then traverse all the grasshoppers to calculate its location of the next iteration. When the grasshoppers move from i to j, the cluster i

Algorithm 3 Aggregating Phase Algorithm

Input: Graph $G = (V, E)$, the similar node set C
Output: Communities
Do{
  For *cluster_A* in similar node set C:
    Remove *cluster_A* from set C
    Find the clusters nearby as *NeighborSet*
    For *cluster_B* in *NeighborSet*:
      $cluster = cluster\_A + cluster\_B$
      Calculate *cluster_B* with the minimum $\theta(cluster)$
      Remove *cluster_B* from set C
      Add *cluster* to set C
} While the modularity of the set C is improving
Output set C as Communities

will aggregate with the cluster j. The Algorithm 3 will stop when each grasshopper can not move.

As shown in Fig. 1(c) and Fig. 1(d), the 9 clusters output by Algorithm 2 are merged into two communities by Algorithm 3, which is the same as the ground truth of Karate.

IV. SIMULATION METHODOLOGY

To evaluate our *PSA*'s performance, both the real world networks and the synthetic networks with ground truth are used, and four other community detection algorithms include *DA*, *GNp*, *Louvain*, *LPA* are compared with *PSA* method. All the experiments are taken on a PC with an intel (R) 3.6GHZ Xeon(R) W-2133CPU and 32G RAM.

A. THE REAL-WORLD NETWORKS

Six real-world networks whose nodes ranges from tens to tens of thousands are used to evaluate *PSA* as well as other four community detection methods. The results of community detection and the network description are shown in Table 1, 2 separately.

TABLE 1. Comparisons of modularity values on real-world networks.

Networks	Algorithms									
	PSA		*DA*		*GNp*		*Louvain*		*LPA*	
	Q	*Nc*	*Q*	*Nc*	*Q*	*Nc*	*Q*	*Nc*	*Q*	*Nc*
Karate	0.37	2	0.38	3	0.36	3	0.41	4	0.37	2
Football	0.58	12	0.57	12	0.58	11	0.6	10	0.56	10
Polbooks	0.52	6	0.52	5	0.51	5	0.52	5	0.5	5
Email	0.51	12	0.49	46	0.46	51	0.54	11	0.09	4
Facebook	0.81	18	0.73	9	0.72	77	0.83	17	0.82	60
hep-th	0.64	462	0.71	1258	0.76	1332	0.85	1380	0.79	1243
DBLP	0.43	15186	0.39	21435	-	-	0.4	27193	-	-

Football network (the American collages' football network) is divided into 12 communities by *PSA*, which are almost the same as the reality. Although, it is not much difference between *PSA* and other four methods in modularity, *PSA* get a better result in *NMI* value compared to the ground truth.

TABLE 2. The statistics of real-world networks.

Networks	*n*	*m*	$< k >$	*Description*
Karate	34	78	4.59	Zackary's Karate Club [37]
Football	230	613	5.33	US collage football
Polbooks	105	441	8.4	Books about US politics
Email	1133	5451	9.62	Email networks [38]
Facebook	4039	88234	43.69	Facebook dataset [39]
hep-th	11204	117649	21	Arxiv HEP-TH collaborators [40]
DBLP	317080	1049866	6.62	Co-authorship network of the DBLP

Polbooks network (the political books network) was collected by the Krebs where the nodes represent 1005 books about American politics which can be divide into 3 classifications according to the attitude including conservative, liberal and neutral. None of the five methods get the correct number of *Nc*, but *PSA* get the preferable consequence in both the modularity and the *NMI* value.

Email network (the E-mail network) was collected by Alexandre Arenas, which describes the E-mail interchanges between members of the University Rovira i Virgili (Tarragona). The *Louvain* algorithm performs best, while the *LPA* can hardly accomplish the community detection, and *PSA* gets the second best value of modularity.

Facebook network (the Facebook network dataset) was collected and compiled by Mcauley and Leckovec which is comprised of 4039 nodes and 88234 edges. From the results in table 5, there is no much difference between *Louvain* and *PSA*, while *Louvain* get the best result. As a big network with a high average degree, the node in Facebook often has a complicated relationship with other nodes, although *LPA* get the second high value of modularity, but the number of communities it detected is three to seven times than that of other methods.

The hep-th network (the High Energy Physics-Theory collaboration network) was compiled by Leskovec etc. *Louvain* got the best results out of the five algorithms, but there was a problem with too many communities. And the number of communities with fewer than 5 nodes is 10% of the total number of communities *Louvain* found, which is 5% in *DA*, 7% in *GNp* and 3% in *LPA*. Although the modularity of *PSA* is not as high as that of other algorithms, the community structure of *PSA* is well balanced, and there is no such community.

DBLP network represent the co-authorship network of the DBLP computer science bibliography. Nodes are authors and there is an undirected edge between the two nodes if the corresponding authors have published at least one paper together. *PSA* get a best result in modularity and least number in communities. *Louvain* and *DA* also perform well in DBLP, while *GNp* and *LPA* have difficulty in detect the community structure.

B. THE SYNTHETIC NETWORKS

The LFR benchmarks [41] is used as the synthetic networks which have power-law distributions of both node degree and

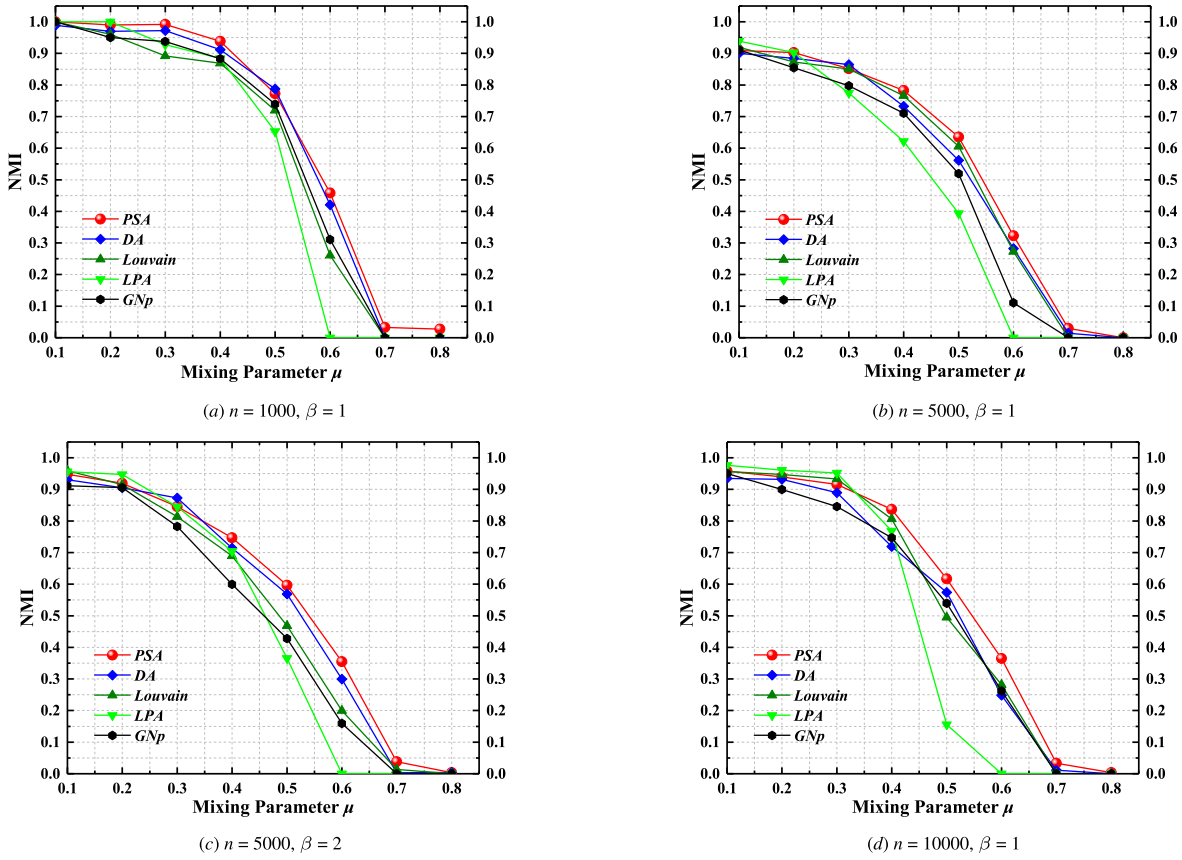


FIGURE 3. Comparisons of *NMI* values of algorithms on LFR networks with different scales of μ.

the community size. Therefore, it is always considered as a substitution of real-world network with community structure and is duly to evaluate the performance of community detection algorithms. The networks LFR benchmarks produced have some parameters which include the number of nodes n, the average degree $< k >$, the power-law exponents for the degree α and the community size distribution parameter β, the mixing parameter μ, where the mixing parameter μ represents each vertex shares a fraction μ of its links with the other vertices of its community and a fraction μ with the other vertices of the network. The higher value of μ, the more ambiguous community structure. the parameters of LFR networks in following calculation are set as follows.

(1) The number of nodes n: Set $n = 1000, 5000, 10000$ respectively.

(2) The average degree $< k >$: Set $< k > = 15$ and the upper bound of the degree of the nodes in LFR networks $kmax = 0.1n$.

(3) The power-law exponents for the degree α: Set $\alpha = 2$.

(4) The size of the communities β: Set $\beta = 1$ or 2 respectively.

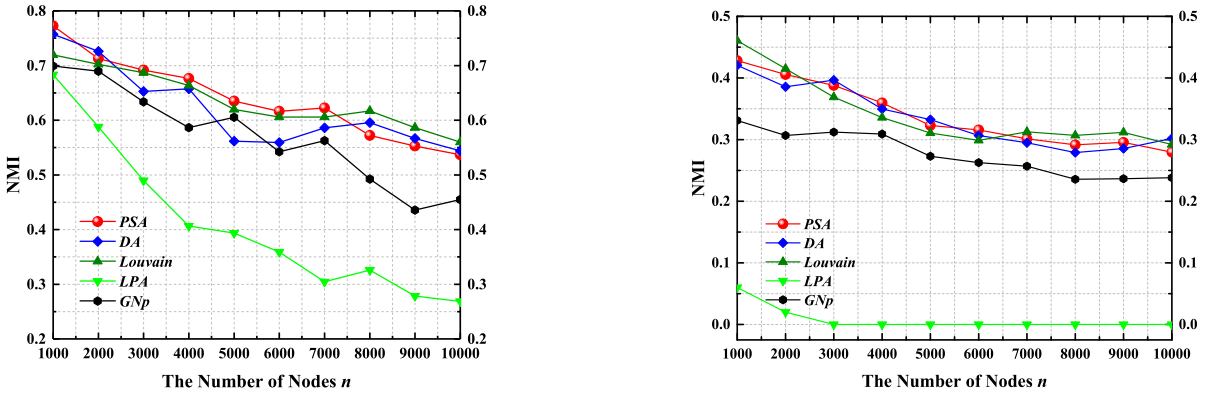
(5) The mixing parameter β: Set $\beta = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8$.

(6) The maximum and minimum for the community size $maxc$ and $minc$: Set $maxc = 0.1n$, $minc$ is set to $minc = 10$ for $n = 1000$, and $minc = 20$ for $n = 5000$ and $n = 10000$.

For each set of parameters, 72 networks are generated to evaluate the performance of *PSA* and the result is shown in Fig. 3, 4 and 5. Fig. 3 and Fig. 5 show that with the mixing parameter μ increases, the community detection is getting more and more difficult for the algorithm we tested, because the greater μ means the more ambiguous network structure.

Fig. 3(a), 3(b) and 3(d) shows the result for LFR networks with the same community distribution and different networks' size respectively. *PSA* is represented by the red dotted line with balls are almost get the best result when μ is range from 0.4 to 0.6, other algorithms have better performance when μ is less than 0.3 whose value of *NMI* is almost reach to 1 for *NMI*. When μ equals to 0.5, *LPA* begins to have a sharp decrease. When $\mu = 0.6$, all the results have a same downward trend. The *NMI* value of *LPA* fall to 0 when $\mu = 0.7$ while another four algorithms behave normally. When $\mu = 0.8$, all the algorithms don't perform very well while the *NMI* value of *PSA* is a little higher than others.

Fig. 3(b) and 3(c) exhibit the results of increasing community size distribution parameter β from 1 to 2. The result in Fig. 3(b) and 3(c) is nearly the same, which means the



(a) $\mu = 0.5, \beta = 1$ (b) $\mu = 0.6, \beta = 1$

FIGURE 4. Comparisons of *NMI* values of algorithms on LFR networks with different scales of *n*.

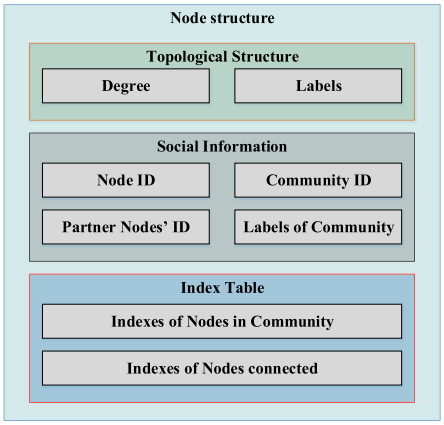


FIGURE 5. The structure of node.

difference in community sizes distribution parameter β have no impact on the accuracy of the algorithms.

In order to further compare those algorithms for discovering communities in blurred networks, we set networks' size range from 1000 to 10000 with $\mu = 0.5, 0.6$ and $\beta = 1$, and the results are shown in Fig. 4. Fig. 4(a) shows the results for $\mu = 0.5$, where *PSA* and *Louvain* perform better than others. When N ranges from 3000 to 7000, the *NMI* results of *PSA* is a little better than *Louvain*'s. With the increasing of n, all algorithm perform steadily except *LPA*, who has difficulty to find the community structure in this condition. Fig. 4(b) depicts the comparison for $\mu = 0.6$. In this condition, *PSA*, *Louvain* and *DA* have a analogical performance, and *PSA* get the better values of *NMI* when $n = 4000, 6000$.

C. COMPARISON OF REAL-WORLD NETWORKS AND SYNTHETIC NETWORKS WITH GROUND TRUTH

We collect some real-world data and generate four LFR networks with ground truth in order to evaluate *PSA*. The results is shown in Table 3 and Table 4 show the statistics of LFR networks. The consequences show that *PSA* outperforms others or have a same performance with other algorithm.

TABLE 3. Comparisons of *NMI* values on networks with ground truth.

Networks	Algorithms									
	PSA		*DA*		*GNp*		*Louvain*		*LPA*	
	NMI	*Nc*	*NMI*	*Nc*	*NMI*	*Nc*	*NMI*	*Nc*	*NMI*	*Nc*
Karate	1	2	0.86	3	0.81	3	0.59	4	1	2
Football	0.79	12	0.76	12	0.77	11	0.77	10	0.78	10
Polbooks	0.52	6	0.5	4	0.51	5	0.51	4	0.51	4
LFR128	0.88	5	0.82	5	0.84	6	0.88	6	0.85	6
LFR256	0.92	22	0.89	21	0.85	27	0.91	16	0.89	19
LFR512	0.88	33	0.87	30	0.81	43	0.87	30	0.86	37
LFR1024	0.67	90	0.65	89	0.58	104	0.66	98	0.62	120

TABLE 4. The statistics of LFR networks.

Networks	n	m	$< k >$	μ
LFR128	128	351	5.48	0.3
LFR256	256	700	5,46	0.4
LFR512	512	2514	4.91	0.4
LFR1024	1024	2740	5.35	0.3

D. THE COMPLEXITY ANALYSIS

PSA is a three-phases algorithm where the pairing phase first forms the most similar node pairs and then the splitting phase and the aggregating phase will merges them into communities. In the pairing phase, calculating similarity is a important step of finding the similar node pairs, and the similarity calculation will reduce to O(1) with the help of hash table. Therefore, in the overall network, the comparison number is $\sum_{(v,w\in E)} \min(d_v, d_w) < d_{\max}|V|$and the similarity calculation's complexity is $O(d_{\max}|V|)$, where $d_{\max}$ is the max degree in the graph. In the splitting phase, the calculation's complexity is $O(k-1)$ where k is the number of the similar node pairs. In the aggregating phase, the calculation's complexity is $O(e(m-1)^2)$, where m is the number of the connected brunches, and e is the maximal exterior edges

of a connected brunch. So the calculation's complexity is $O(d_{\max}|V| + (k-1) + e(m-1)^2)$.

V. APPLICATION

A. NETWORK STRUCTURE FOR RESOURCE DISCOVERY IN IoT

In this chapter, we simulate a network structure of IoT based on 4 social behaviors, and then demonstrate the feasibility of applying community detection to resource discovery in IoT.

1) SOCIAL BEHAVIOR IN IoT

As a social network, the network of IoT must meet the characteristics of the social network, in order to better simulate the Internet of things, we set up two social behavior:

Social Behavior 1: *In social networks, people with similar interests tend to form small groups (communities).*

The community structure is used to simulate the small groups in Social Behavior 1, and the community detection algorithm which can divide the whole network into many parts according to the network topology structure can be used to discover the community structure in the network.

Social Behavior 2: *People in social networks make new friends through community or friend referrals. They also tend to leave their contact information after socializing so that they can contact each other quickly afterwards.*

In order to simulate this social behavior, each node in the IoT network has to set aside some memory space as Index Table to record other nodes' address. Once a node has some resource requirements, the node can access the resource directly through the Index Table.

To prevent infinite propagation of request, we introduce the concept of time-to-live (*TTL*) [42]. If the propagation distance of resource request exceed *TTL*, the request could not be distributed by the other node.

The main objects of the IoT are all sorts of devices and sensors which can be supposed to be honest and uninquisitive. So base on this peculiarity we designed Social Behavior 3.

Social Behavior 3: *The people in the social networks are honest, and the information passed between different communities is authentic and reliable.*

In order to simulate the resource discovery process, each node in the network is required to be honest, not to provide false information to other nodes, and give feedbacks after receiving the resource request [43], [44]. This is the basis for various interactions between nodes and communities.

Social Behavior 4: *Each person in the community is well informed about the information of others, and people in the community tend to help each other when a request for resources arises within the community.*

After dividing the whole network into communities using the community detection algorithm, we set each node to store not only the index of its neighbors, but also the index of the node which is in the same community [45]. In addition, each node records the type of label within the community in order to respond to a request for resource discovery in a timely manner.

In reality, there is no device has unlimited space for storage [46], [47], so we limit the number of indexes to three times of the average degree. If there are too many nodes in a community, the Index Table of a single node will be difficult to cover the whole community, which will hamper the query [48], [49].

Based on the above four social behaviors, we put forward the node structure which is shown in Fig. 5.

2) INDEX TABLE INITIALIZATION

When the community detection algorithm divides the network into several parts, each node will record the address of the node in its community into its Index Table. In addition, if the Index Table has extra space, the node will randomly copy portions of the Index Table of its neighbor nodes until reaching its maximum capacity.

3) COMMUNITY-ASSISTED SHORT-DISTANCE-QUERY RESOURCE DISCOVERY

We propose the *CSRD* to apply community to resource discovery which is shown in Algorithm 4.

Algorithm 4 Community-Assisted Short-Distance-Query Resource Discovery Method

Input: *Communities*, *nodeA*, *labels* required, *TTL*
Output: The index of nodes
Initialize *Coms* = null set, *Targets* = *labels* required
Find the nodes of *nodeA*'s community as *ComA*
If *label* in *Targets* can be searched in *ComA*
  Removed the *label* searched from *Targets*
  Return the index of the node with the label
Else:
  Add the *ComA* to the *Coms*
  Do{
    For *Community* in *Coms*:
      Find the Communities nearby as *NeighborCom*
      Ask the nodes of *NeighborCom*
      If *label* in *Targets* can be searched in *NeighborCom*:
        Removed the *label* searched from *Targets*
        Return the index of the node with the label
        add the *Communities* with label to *Coms*
      Else:
        Add all the *Communities* in *NeighborCom* to *Coms*
  }While the times of delivery is less than *TTL* and *Targets* has more than one element

To better explain our Algorithm 4, we set $TTL = 6$ for resource discovery, and simulated the resource discovery process with Fig. 6 as an example. Each node in Fig. 6 represents a community, and the line between the two nodes indicates

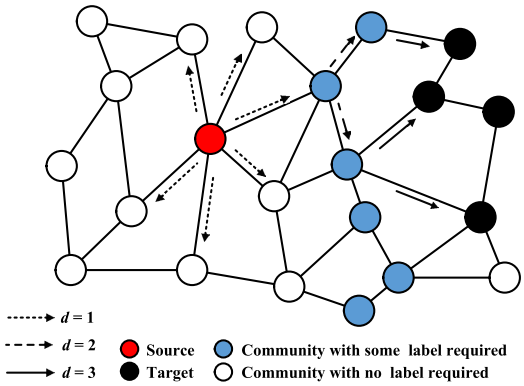


FIGURE 6. An example of *CSRD*.

that there are some indexed links between the two communities. The red node represents the source community with labels required, black node represents the target community with all of the labels required, and the blue node represents the transition community with only some of the labels required, the white nodes represent the communities that can not meet the requirements of the label.

(*i*) $d = 0$ and d<*TTL*: In the beginning, a node in the red community with a resource request checks whether the label request can be satisfied within the community, and if not, initiates a query broadcast to the surrounding community with the help of other nodes in the community;

(*ii*) $d = 1$ and d<*TTL*: After the first step, the nodes in the red communities will request the nodes in the blue communities to conduct the second query after receiving the responses.

(*iii*) $d = 2$ and d<*TTL*: After the second step, the nodes in the red community received responses from the nodes in the blue community, and in the same way as (*ii*), the nodes in the blue community continued to be requested for the next round of queries.

(*iv*) $d = 3$ and d<*TTL*: After the third step, the node in red community gets a response that the black communities can meet its labels requirements.

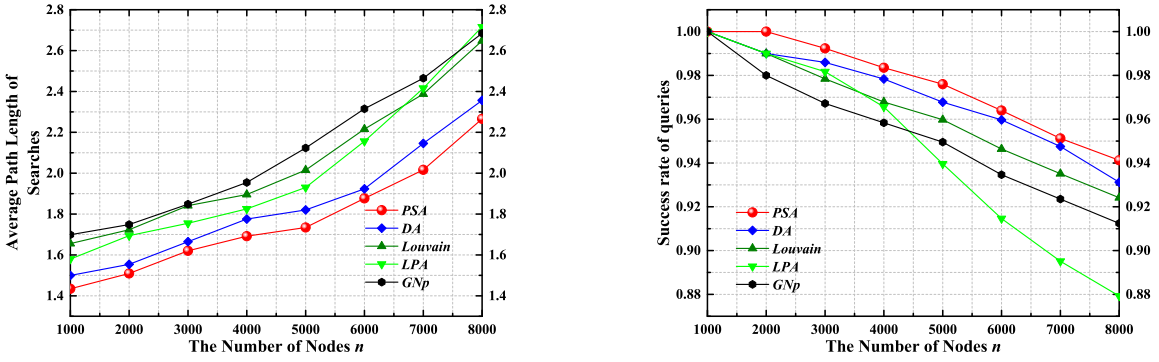
B. EXPERIMENTAL SIMULATION AND PERFORMANCE COMPARISON

1) LABEL GENERATION AND NETWORK CONFIGURATION

Due to the variety and complexity of IoT devices [50], [51], we set the number of nodes n range from 1000 to 8000. In the network generation phase of the simulation experiment, we assume that there is a ***D***ata ***S***haring ***P***latform (i.e., *DSP*) that could provide data interaction [52], [53]. First, we generated 100 labels and randomly assigned them to $0.1n$ data where each data assigned two labels, then we randomly assign $0.1n$ data to n nodes. After that each node uploads its address and data to the *DSP* [54], [55], and tries to download exactly or partially the same data with a probability $P = 90\%$, try to download data which is completely different with a probability 1-$P = 10\%$. When the data is downloaded by the node, the address is also downloaded in passing. To ensure that the data of each node can be queried by the others, we assume that data of zero download will be displayed preferentially on the *DSP*. In reality, the resources of devices in IoT are not completely balanced [56], [57], so the data with the most downloads is more likely to be download by other node when each type of data has been downloaded at least once. The *DSP* will close the data upload and download channel when there is no 0 download data in the network and all data associated with a label reaches $0.1n$ downloads. Then each node builds a connection based on the address it downloaded, eventually forming a network structure. Table 5 shows some properties of the test network.

2) PERFORMANCE COMPARISON

In this chapter, we set $TTL = 6$ and use 8 test networks to examine the efficiency of resource discovery of *PSA*, *DA*, *GNp*, *Louvain*, and *LPA*. We randomly select 100 node pairs which have completely different labels in the network, and one node as the source node and the other node as the target node for resource discovery [58]. Then we define the average path length of searches for resource discovery as the average distance of queries between all the source nodes and their corresponding target nodes, define the success rate of queries



(*a*) Comparisons of average path length of searches

(*b*) Comparisons of success rate of queries

FIGURE 7. Comparisons of different algorithms in resources discovery on the Testworks.

TABLE 5. The statistics of test networks.

Networks	*n*	*m*	<*k*>
Testwork1	1000	7335	14.67
Testwork2	2000	15234	15.23
Testwork3	3000	22681	15.12
Testwork4	4000	34517	17.23
Testwork5	5000	45421	18.17
Testwork6	6000	59442	19.81
Testwork7	7000	71579	20.45
Testwork8	8000	82485	20.62

as the percentage of the total number of queries whose query distance is less than *TTL*.

The data in Fig. 7(*a*) shows that the average path length of searches for resource discovery increases as the number of nodes increases and the structure of the community grows. The average path length of community structure discovered by *PSA* is always at a low level, and the efficiency of *DA* algorithm is closed to *PSA*. However, the community structure generated by *LPA*, *Louvain* and *GNp* algorithm does not perform well in resource discovery due to the excessive number of node and unbalanced community structure.

The data in Fig. 7(*b*) shows that as the number of nodes increases, the network structure becomes more complex and the success rate of resource discovery decreases. However, the community structure discovered by *PSA* is more stable and successful in resource discovery, and the effect of other algorithms is not as good as *PSA* because of the great difference of nodes in the community structure.

VI. CONCLUSION

The community structure generated by the current non-overlapping community detection algorithm often has a obvious gap in the number of nodes between communities, and when the community detection algorithm is applied to resource discovery in the Internet of things, this quantitative gap greatly reduces the efficiency of resource discovery. In order to solve this problem, we put forward the ***P***airing, ***S***plitting and ***A***ggregating algorithm, (i.e., *PSA*), which is designed to balance the differences of communities detected. In the first stage, we raise the Pairing Phase Algorithm to find the most similar nodes for each node in the network to generate multiple similar node pairs. Then in the second stage, we split the whole network into multiple similar node sets with similar node pairs as reference. Finally, we merge these similar node sets into a community structure based on *GOA*. The experimental results show that *PSA* has the same performance as the current popular non-overlapping community detection algorithm, while the community structure detected by *PSA* is more balanced in structure and number of nodes. In addition, we also simulate the resource discovery process in the IoT, and propose the ***C***ommunity-assisted ***S***hort-distance-query ***R***esource ***D***iscovery (i.e., *CSRD*), the experimental results show that the community structure detected by *PSA* is more suitable for resource discovery in IoT.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of this article.

REFERENCES

[1] X. Lu, Z. Pan, and H. Xian, "An integrity verification scheme of cloud storage for Internet-of-Things mobile terminal devices," *Comput. Secur.*, vol. 92, May 2020, Art. no. 101686, doi: 10.1016/j.cose.2019.101686.

[2] T. Qiu, B. Li, W. Qu, E. Ahmed, and X. Wang, "TOSG: A topology optimization scheme with global small world for industrial heterogeneous Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3174–3184, Jun. 2019.

[3] H. Xia, L. Li, X. Cheng, X. Cheng, and T. Qiu, "Modeling and analysis botnet propagation in social Internet of Things," *IEEE Internet Things J.*, early access, Mar. 31, 2020, doi: 11.1109/JIOT.2020.2984662.

[4] R. Barthwal, S. Misra, and M. S. Obaidat, "Finding overlapping communities in a complex network of social linkages and Internet of Things," *J. Supercomput.*, vol. 66, no. 3, pp. 1749–1772, Dec. 2013.

[5] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, nos. 3–5, pp. 75–174, Feb. 2010.

[6] G. Liu, Z. Chen, Z. Zhuang, W. Guo, and G. Chen, "A unified algorithm based on HTS and self-adapting PSO for the construction of octagonal and rectilinear SMT," *Soft Comput.*, vol. 24, no. 6, pp. 3943–3961, Mar. 2020.

[7] X. Chen, G. Liu, N. Xiong, Y. Su, and G. Chen, "A survey of swarm intelligence techniques in VLSI routing problems," *IEEE Access*, vol. 8, pp. 26266–26292, 2020.

[8] G. Liu, X. Huang, W. Guo, Y. Niu, and G. Chen, "Multilayer obstacle-avoiding X-architecture Steiner minimal tree construction based on particle swarm optimization," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 1003–1016, May 2015.

[9] G. Liu, W. Guo, Y. Niu, G. Chen, and X. Huang, "A PSO-based timing-driven octilinear Steiner tree algorithm for VLSI routing considering bend reduction," *Soft Comput.*, vol. 19, no. 5, pp. 1153–1169, May 2015.

[10] G. Liu, W. Guo, R. Li, Y. Niu, and G. Chen, "XGRouter: High-quality global router in X-architecture with particle swarm optimization," *Frontiers Comput. Sci.*, vol. 9, no. 4, pp. 576–594, Aug. 2015.

[11] J. Cheng, X. Wu, M. Zhou, S. Gao, Z. Huang, and C. Liu, "A novel method for detecting new overlapping community in complex evolving networks," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 9, pp. 1832–1844, Sep. 2019.

[12] J. Cheng, M. Chen, M. Zhou, S. Gao, C. Liu, and C. Liu, "Overlapping community change-point detection in an evolving network," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 189–200, Mar. 2020.

[13] K. Guo, L. He, Y. Chen, W. Guo, and J. Zheng, "A local community detection algorithm based on internal force between nodes," *Int. J. Speech Technol.*, vol. 50, no. 2, pp. 328–340, Feb. 2020.

[14] Z. Liu, B. Xiang, W. Guo, Y. Chen, K. Guo, and J. Zheng, "Overlapping community detection algorithm based on coarsening and local overlapping modularity," *IEEE Access*, vol. 7, pp. 57943–57955, 2019.

[15] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002.

[16] M. Arasteh and S. Alizadeh, "A fast divisive community detection algorithm based on edge degree betweenness centrality," *Int. J. Speech Technol.*, vol. 49, no. 2, pp. 689–702, Feb. 2019.

[17] V. D. Blondel, J. L. Guillaume, R. LambIoTte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Express*, vol. 2008, no. 10, pp. 10008–10012, 2008.

[18] F. Radicchi, "Decoding communities in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 97, no. 2, Feb. 2018, Art. no. 022316.

[19] C.-C. Lin, J.-R. Kang, and J.-Y. Chen, "An integer programming approach and visual analysis for detecting hierarchical community structures in social networks," *Inf. Sci.*, vol. 299, pp. 296–311, Apr. 2015.

[20] H. Zhang, J. Yu, C. Tian, L. Tong, J. Lin, L. Ge, and H. Wang, "Efficient and secure outsourcing scheme for RSA decryption in Internet of Things," *IEEE Internet Things J.*, early access, Jan. 30, 2020, doi: 10.1109/JIOT.2020.2970499.
[21] H. Zhang, J. Yu, C. Tian, G. Xu, P. Gao, and J. Lin, "Practical and secure outsourcing algorithms for solving quadratic congruences in Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2968–2981, Apr. 2020.
[22] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *Proc. 6th Int. Conf. Data Mining (ICDM)*, Dec. 2006, pp. 613–622.
[23] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 76, no. 3, Sep. 2007, Art. no. 036106.
[24] K. R. Žalik, "Maximal neighbor similarity reveals real communities in networks," *Sci. Rep.*, vol. 5, no. 1, Nov. 2016, Art. no. 18374.
[25] Y. Wu, R. Jin, J. Li, and X. Zhang, "Robust local community detection: On free rider effect and its elimination," *Proc. VLDB Endowment*, vol. 8, no. 7, pp. 798–809, Feb. 2015.
[26] Y. Yan, Y. Bian, D. Luo, D. Lee, and X. Zhang, "Constrained local graph clustering by colored random walk," in *Proc. World Wide Web Conf.*, 2019, pp. 2137–2146.
[27] Z. Liu and Y. Ma, "A divide and agglomerate algorithm for community detection in social networks," *Inf. Sci.*, vol. 482, pp. 321–333, May 2019.
[28] J. J. Whang, D. F. Gleich, and I. S. Dhillon, "Overlapping community detection using neighborhood-inflated seed expansion," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1272–1284, May 2016.
[29] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 6, Dec. 2004, Art. no. 066111.
[30] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *J. Stat. Mech., Theory Exp.*, vol. 2005, no. 9, Sep. 2005, Art. no. P09008.
[31] Q. Zhao, C. Wang, P. Wang, M. Zhou, and C. Jiang, "A novel method on information recommendation via hybrid similarity," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 3, pp. 448–459, Mar. 2018.
[32] Z. Cai and W. Zhu, "Feature selection for multi-label classification using neighborhood preservation," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 1, pp. 320–330, Jan. 2018.
[33] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017.
[34] Q. Zhang and D. Haglin, "Semantic similarity between ontologies at different scales," *IEEE/CAA J. Automatica Sinica*, vol. 3, no. 2, pp. 132–140, Apr. 2016.
[35] L. A. Adamic and E. Adar, "Friends and neighbors on the Web," *Social Netw.*, vol. 25, no. 3, pp. 211–230, Jul. 2003.
[36] Y. Xiao, Y. Jia, C. Liu, X. Cheng, J. Yu, and W. Lv, "Edge computing security: State of the art and challenges," *Proc. IEEE*, vol. 107, no. 8, pp. 1608–1631, Aug. 2019.
[37] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropol. Res.*, vol. 33, no. 4, pp. 452–473, Dec. 1977.
[38] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, "Models of social networks based on social distance attachment," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 5, Nov. 2004, Art. no. 056122.
[39] J. Mcauley and J. Leskovec, "Discovering social circles in ego networks," *ACM Trans. Knowl. Discovery from Data*, vol. 8, no. 1, pp. 1–28, Feb. 2014.
[40] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, p. 2, Mar. 2007.
[41] L. Prokhorenkova and A. Tikhonov, "Community detection through likelihood optimization: In search of a sound model," in *Proc. World Wide Web Conf.*, 2019, pp. 1498–1508.
[42] L. Liu, N. Antonopoulos, S. Mackin, J. Xu, and D. Russell, "Efficient resource discovery in self-organized unstructured peer-to-peer networks," *Concurrency Comput., Pract. Exper.*, vol. 21, no. 2, pp. 159–183, Feb. 2009.
[43] D. Yu, Y. Zou, J. Yu, X. Cheng, Q.-S. Hua, H. Jin, and F. C. M. Lau, "Stable local broadcast in multihop wireless networks under SINR," *IEEE/ACM Trans. Netw.*, vol. 26, no. 3, pp. 1278–1291, Jun. 2018.
[44] F. Li, D. Yu, H. Yang, J. Yu, H. Karl, and X. Cheng, "Multi-armed-bandit-based spectrum scheduling algorithms in wireless networks: A survey," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 24–30, Feb. 2020.
[45] Z. Cai, X. Zheng, and J. Yu, "A differential-private framework for urban traffic flows estimation via taxi companies," *IEEE Trans. Ind. Informat.*, vol. 15, no. 12, pp. 6492–6499, Dec. 2019.
[46] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 4, pp. 577–590, Aug. 2018.
[47] Z. Su, X. Mu, Y. Dai, and H.-J. Li, "Defectors for high degree with adaptive interactions," *Phys. A, Stat. Mech. Appl.*, vol. 527, Aug. 2019, Art. no. 121132.
[48] C. Tian, J. Yu, H. Zhang, H. Xue, C. Wang, and K. Ren, "Novel secure outsourcing of modular inversion for arbitrary and variable modulus," *IEEE Trans. Services Comput.*, early access, Aug. 26, 2019, doi: 10.1109/TSC.2019.2937486.
[49] Z. Bu, H.-J. Li, C. Zhang, J. Cao, A. Li, and Y. Shi, "Graph K-means based on leader identification, dynamic game, and opinion dynamics," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 7, pp. 1348–1361, Jul. 2020, doi: 10.1109/TKDE.2019.2903712.
[50] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, Jul. 2019, pp. 144–153.
[51] H.-J. Li, Z. Bu, Z. Wang, and J. Cao, "Dynamical clustering in electronic commerce systems via optimization and leadership expansion," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5327–5334, Aug. 2020.
[52] T. Qiu, J. Liu, W. Si, and D. O. Wu, "Robustness optimization scheme with multi-population co-evolution for scale-free wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1028–1042, Jun. 2019.
[53] H.-J. Li and J. J. Daniels, "Social significance of community structure: Statistical view," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 91, no. 1, Jan. 2015, Art. no. 012801.
[54] H. Xia, S.-S. Zhang, Y. Li, Z.-K. Pan, X. Peng, and X.-Z. Cheng, "An attack-resistant trust inference model for securing routing in vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 7108–7120, Jul. 2019.
[55] H.-J. Li and L. Wang, "Multi-scale asynchronous belief percolation model on multiplex networks," *New J. Phys.*, vol. 21, no. 1, Jan. 2019, Art. no. 015005.
[56] H. Xia, L. Li, X. Cheng, C. Liu, and T. Qiu, "A dynamic virus propagation model based on social attributes in city IoTs," *IEEE Internet Things J.*, early access, Apr. 24, 2020, doi: 10.1109/JIOT.2020.2990365.
[57] H.-J. Li, Z. Bu, Z. Wang, J. Cao, and Y. Shi, "Enhance the performance of network computation by a tunable weighting strategy," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 3, pp. 214–223, Jun. 2018.
[58] H. Xia, C.-Q. Hu, F. Xiao, X.-G. Cheng, and Z.-K. Pan, "An efficient social-like semantic-aware service discovery mechanism for large-scale Internet of Things," *Comput. Netw.*, vol. 152, pp. 210–220, Apr. 2019.

YIZHE LI was born in 1996. He received the bachelor's degree in information security from the College of Computer Science and Technology, Qingdao University, in July 2018, where he is currently pursuing the master's degree in cyberspace security with the College of Computer Science and Technology. His research interests include community detection, deep learning, machine learning, information security, and the IoT security. He has participated in many software design competitions at university. He was awarded the academic scholarship for many times.

HUI XIA (Member, IEEE) received the Ph.D. degree in computer science from the School of Computer Science and Technology, Shandong University, China, in June 2013. From July 2013 to May 2020, he worked as a Lecturer and an Associate Professor with the College of Computer Science and Technology, Qingdao University, China. Since June 2020, he is currently a Full Professor and a Ph.D. Supervisor with the College of Information Science and Engineering, Ocean University of China. He was a Visiting Scholar with the Department of Computer Science, The George Washington University, USA, from 2017 to 2018. His current research interests include the IoT security, vehicular ad hoc networks, crowdsourcing, privacy protection, machine learning, and edge computing. He has published over 43 scientific articles. His research was sponsored by the Natural Science Foundation of China (NSFC), the Shandong Provincial Natural Science Foundation, and the Source Innovation Project of Qingdao. He is a member of CCF, ACM, and the IEEE Computer Society.

RUI ZHANG (Student Member, IEEE) was born in 1995. She received the bachelor's degree in software outsourcing and computer science and technology from the School of Medical Information Engineering, Jining Medical College, Shandong, China, in July 2018. She is currently pursuing the master's degree in cyberspace security with the College of Computer Science and Technology, Qingdao University, Shandong. Her research interests include game theory, information security, privacy protection, and the IoT security. She received the First-Class Scholarship for many times in college and the First-Class Academic Scholarship.

BENXU HU was born in 1996. He received the bachelor's degree in Internet of Things Engineering from Shandong Jiaotong University, China, in 2018. He is currently pursuing the master's degree in computer technology with the College of Computer Science and Technology, Qingdao University, China. His research interest includes net communication, the IoT Security, edge computing, and deep learning.

XIANGGUO CHENG was born in 1969. He received the Ph.D. degree from the State Key Laboratory of Comprehensive Business Network, Xidian University, in 2006. He is currently a member of the China Cryptography Society and Qingdao Computer Society. He is currently a Professor with the College of Computer Science and Technology, Qingdao University, China. His main research interests include cloud security computer security and public key cryptosystems. He has participated in the completion of the Natural Science Foundation of China, the National Cryptographic Development Fund of China, the Natural Science Foundation of Shandong Province, the Science and Technology Program of Shandong Education Department, and many other scientific research projects. He has published more than 60 articles in domestic and foreign academic journal or international conferences, among which more than 40 articles have been included by SCI or EI.