Received October 24, 2019, accepted November 11, 2019, date of publication November 15, 2019, date of current version December 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2953470

DynaPro: Dynamic Wireless Sensor Network Data Protection Algorithm in IoT via Differential Privacy

SONGYAN LI[1], ZHAOBIN LIU[1], ZHIYI HUANG[2], HAOZE LYU[1], ZHIYANG LI[1], AND WEIJIANG LIU[1]
[1]School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China
[2]Department of Computer Science, University of Otago, Dunedin 9012, New Zealand

Corresponding author: Zhaobin Liu (zhbliu@gmail.com)

This work was supported in part by the National Nature Science Foundation of China under Grant 61370198, Grant 61370199, Grant 61672379, and Grant 61300187, in part by the ''double-top class'' contruction project (''innovation project'') of China under Grant 0143190518, and in part by the Liaoning Provincial Natural Science Foundation of China under Grant 2019-MS-028.

ABSTRACT With the advent of the era of intelligent Internet of Things (IoT), more and more personal information are collected in the process of deploying IoT in various domains. This brings the problem of data protection in IoT. To address this problem, we propose a data protection algorithm, called DynaPro, based on differential privacy for dynamic wireless sensor networks (WSN), which is the sensor layer of IoT. DynaPro has addressed two issues in data protection of wireless sensor networks. The first issue is the dynamic topology of the network. In order to solve this problem, DynaPro has adopted methods like time window, hierarchical sampling, and graph similarity to process the snapshots of the dynamic network topology. The second issue is addition of noise in differential privacy. To address this issue, DynaPro uses Hierarchical Random Graph (HRG) as middleware to apply differential privacy protection. Instead of adding noise to network data and topology directly, DynaPro adds noise to HRG. In this way, DynaPro can hide the network topology but retain the necessary structural information to support data analysis. Theoretical analysis and experimental results show that DynaPro can preserve the important network features of the original network topology under the premise of the differential privacy protection model.

INDEX TERMS Differential privacy, WSN, IoT, dynamic networks, sensor layer.

I. INTRODUCTION

IoT has received more and more attention in recent years. It is widely used and rapidly developed because it makes data collection more convenient. The wireless sensors in IoT are used to monitor and track various objects, such as animals, vehicles and physical phenomena. With the development of these wireless sensors, IoT applications have proliferated. According to estimation of Ahmed et al [1], there will be 50 billion IoT devices in 2020 [2]. With the wide application of IoT, more and more personal information in people's daily life is collected by wireless sensors and devices.

IoT takes advantage of the benefits of the Internet like data sharing and can achieve more accurate data management [3]. It uses the Internet and other communication technologies to connect sensors, microcontrollers, machines, people and objects together [4]. Generally speaking, the architecture of IoT is divided into three layers: sensor layer, network layer and application layer [1]. The task of the sensor layer is to collect the environmental information. The network layer solves the problem of transmitting the data obtained by the sensor layer. The application layer addresses the problem of information processing and human-machine interface. In this layer, the data are processed by various kinds of information systems and consumed by people through various devices.

With the development of intelligent devices, the sensor nodes in the sensor layer are no longer simple hardware devices. Rather they could be smart devices that are controlled by their users, such as mobile phone, camera, GPS [5]. The information collected by these devices has also become more dynamic and complex, and especially could be personal. Therefore, how to protect these personal information in the

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

dynamic and complex wireless sensor network is a serious concern. It will affect the wide applications of IoT.

In this paper, we propose a data protection algorithm, called DynaPro, for dynamic wireless sensor networks base on hierarchical random graph. DynaPro has addressed two issues in data protection of wireless sensor networks. The first issue is the dynamic topology of the network. Since the network topology collected by the sensor layer is dynamic, we cannot simply treat the network topology as static graph. In order to solve this problem, DynaPro has adopted methods like time window, hierarchical sampling, and graph similarity to process the snapshots of the dynamic network topology.

The second issue is addition of noise in differential privacy. In order to reduce the amount of added noise, we use dynamic community detection to divide the network graph into independent communities and add noise to each community in order to reduce the total amount of noise. In order to achieve differential privacy, it is common to add Laplace noise to the data directly. However, the network data may be very sensitive to small changes in network topology. Adding noise directly to data often leads to excessive noise, and makes it impossible to carry out effective data mining and analysis. To address this issue, DynaPro uses HRG [6] as middleware to apply differential privacy protection. Our network topology is represented by HRG. Instead of adding noise to network data and topology directly, DynaPro adds noise to HRG. In this way, DynaPro can hide the network topology but retain the necessary structural information to support data analysis.

In summary, this paper has the following contributions:

1) We proposes a differential privacy algorithm that can balance between network dynamics, data privacy, and data availability in the sensor layer of IoT.
2) We uses hierarchical sampling and graph similarity to guarantee the accuracy of DynaPro as well as time efficiency.
3) Our experimental results show feasibility and advantages of DynaPro.

The rest of this article is organized as follows. Section II introduces the related work. Section III presents the background knowledge. Section IV describes DynaPro in detail and analyzes its privacy protection. The experimental results are shown and discussed in Section V. Finally, Section VI concludes the article.

II. RELATED WORK

In this section, we introduce the related work from the following three aspects: data privacy protection in IoT, differential privacy protection of network data and dynamic community detection.

A. DATA PRIVACY PROTECTION IN IoT

In IoT applications, environmental information is collected by wireless sensor networks and transmitted to the network layer. In this process, the information is easy to leak. Due to the increased intelligence of IoT devices, WSN collect more and more personal information, which may result in more leakage of personal information. How to protect these personal information in wireless sensor networks is a great challenge.

There are three main methods protecting data privacy in IoT. The first is the anonymity technology [2]. By anonymizing the original data, the anonymity technology balances the data between the privacy disclosure risk and the data precision, so as to balance the data availability and data privacy security. The proposed k-anonymity algorithm in [7] directly calculates the quasi-signal utility of sensitive attributes by anonymizing the data. It can protect data privacy while satisfying the user query service. The proposed n-source anonymity in [3] uses the cryptographic tools to unlink the data from its sender among n members. Although anonymity technology has the advantages of simple calculation, small delay and low resource consumption, it causes the loss of original data to a certain degree and thus affects the accuracy of data processing.

The second method is based on cryptography technology [8], [9]. A lightweight encryption scheme is one of the most important methods in privacy protection of IoT. It commonly uses encryption methods like homomorphic encryption [10] and secure multi-party computation (SMC) [11]. The invisibility of the original data and losslessness of the data are realized by the cryptography mechanism. It ensures not only the confidentiality of the data but also the privacy of the data. However, the computational time is long and the complexity of the method is high.

The third method is the popular differential privacy model, which we use in this article. Differential privacy was proposed in 2006 by Dwork et al. [12]. It has two significant advantages. Firstly, it makes the strictest definition of the attacker's background knowledge, ensuring an individual's privacy even when the attacker has access to all the relevant information. Secondly, it has a rigorous mathematical model, which facilitates quantitative theoretical analysis and proof of privacy levels. In [13], the application of differential privacy in data protection and data mining is demonstrated. Differential privacy is now used in social networks, recommendation systems, network tracking analysis and many other fields. We will discuss differential privacy in detail in Section III.

Traditionally data privacy protection in wireless sensor networks is mainly based on encryption technology. For example, in [14], [15], the end-to-end data aggregation privacy protection used homomorphic encryption technology. In [14], a data aggregation privacy protection method CDA is proposed. The homomorphism encryption method is used to aggregate the encrypted data by the aggregation node. In [15], the homomorphic stream encryption algorithm based on addition is adopted, so the aggregation node can aggregate the encrypted data. The disadvantage of this method is that all nodes share the same key with the base station. If any sensor node is compromised, the attacker can obtain the key and access the encrypted data. So it cannot guarantee the privacy of a single node. In addition, homomorphism encryption method has high complexity and resource consumption.

The trust management method proposed in [16] and [17], as one of the effective methods to defend against internal attacks and identify malicious nodes, is widely used in wireless sensor networks. However, since it is based on trust management, it has shortcomings such as slow speed to identify malicious nodes and ineffectiveness to resist malicious attacks.

In this article, we use differential privacy protection to protect the data in WSN, which is a novelty of our work.

B. DIFFERENTIAL PRIVACY PROTECTION OF NETWORK DATA

Differential privacy is often used to process data mining results to be published, but the problems we face are more challenging than publishing specific network statistics or data mining results. Our goal is to publish the entire network graph in a WSN.

In [18], noise is added to the eigenvalues and eigenvectors of the corresponding adjacent matrices. The disadvantage is that the amount of noise added by this method is $O(\sqrt{n})$, so it will add a lot of noise to large networks.

In [19], according to the value of the element in the fixed probability transformation graph, the eigenvector is constrained. The disadvantage of this method is that the calculation is too complex to deal with large-scale graphs efficiently.

In [6], a model of HRG is used to represent the network graph, and the Laplace noise is added to the HRG in order to achieve the purpose of privacy protection. However, this method can only deal with static network data. In the actual application scenario, the collected network map is dynamic, and this method cannot handle dynamic network graphs.

In this article, we will deal with a dynamic network.

C. DYNAMIC COMMUNITY DETECTION

In this article, we use the function of dynamic community detection to deal with network snapshot map. We divide the map into several communities according to the sparse degree of nodes, and add noise to each community to reduce total noise.

Currently there are three kinds of dynamic network community detection algorithms [20].

The Instant Optimal algorithm matches the new community snapshot map with the old snapshot map, which can ensure that the community structure detected at some point must be the most relevant community to the current point in time. This algorithm is studied in paper [21]–[23]. The advantage of this kind of algorithm is that the traditional static community detection technology can be reused directly without modification, and the community detection process on each independent snapshot can also be executed in parallel to improve the efficiency, the community discovery process on individual snapshot maps can also increase efficiency through parallel execution. The disadvantage is instability, and you can get different results even if you run it twice on the same diagram.

The Temporal Trade-off algorithm weighs the communities found in the current snapshot with those found in previous shots. This kind of algorithm is studied in [24], [25]. The advantage of this kind of algorithm is that it can solve the problem of instability of traditional algorithms. The disadvantage is that the traditional algorithm cannot be used directly here, and the community detection process in different snapshot images cannot be parallelized.

The Cross-time Communities algorithm takes into account all the snapshot graphs in the dynamic network at the same time, and finally obtains a community structure partition scheme. This type of algorithm is studied in [26], [27]. The advantage of such an algorithm is that the problem of instability is eliminated and the degree of polymerization of the detected community is guaranteed. The disadvantage is that the new snapshot cannot be processed, and the changing dynamic network in the real world cannot be dealt with.

The idea of dynamic network community detection used in this paper is the combination of time tradeoff and cross-time community algorithm.

III. BACKGROUND

This section introduces the background knowledge of DynaPro.

A. GRAPH SIMILARITY MODEL

In DynaPro, to ensure time efficiency, we use a graph similarity algorithm to filter similar snapshot maps in a dynamic network topology. Graph similarity algorithms can transform a graph matching problem into a string matching problem by marking the graph with a certain string. In this paper, Depth First Search (DFS) [28] encoding is used as a string to convert a graph into a tag. Then, the similarity between the two graphs is calculated by String Edit Distance (SED) [29]. We use this similarity as the basis for screening similarity graphs.

The details of the algorithm based on DFS and SED are shown in Algorithm 1.

```
Algorithm 1 Graph_Similarity($G_1$, $G_2$)
  Input: Two graphs $G_1$ and $G_2$
  Output: The distance between the two graphs
  1. $N_1$, $N_2$ $\leftarrow$ get the order of the node labels in $G_1$ and $G_2$
  2. $E_1$, $E_2$ $\leftarrow$ get the order of the edge labels in $G_1$ and $G_2$
  3. $C_1$ $\leftarrow$ get $minDFSCode(G_1)$ by $N_1$ and $E_1$
  4. $C_2$ $\leftarrow$ get $minDFSCode(G_2)$ by $N_2$ and $E_2$
  5. Filter the minimum DFS Codes of $C_1$ and $C_2$
  6. distance $\leftarrow$ calculate $SED(C_1, C_2)$
  7. return distance
```

B. COMMUNITY DETECTION IN DYNAMIC NETWORKS

In many applications, some smart devices are closely connected. As a result, in the dynamic network topology acquired by the wireless sensor network, some nodes are closely connected, and others are sparsely connected. In the second step of DynaPro, in order to reduce the added noise, we use a dynamic community detection algorithm to deal with the network snapshot graph. The goal of community detection in the network is to find a set of nodes that are tightly connected within the set, but the connections between the sets are sparse.

Definition 1 (Community Detection): Given graph G = (V, E), V represents the set of nodes, E represents the set of edges. Community detection is to find several communities C= $\{C_1, C_2,\ldots, C_n\}$ in G, where $C_1, C_2,\ldots, C_n$ have disjoint nodes and the set of nodes from $C_1, C_2,\ldots, C_n$ equals V.

Network community detection can be divided into dynamic community detection and static community detection. Static community detection was studied initially, but with the development of various dynamic networks, dynamic community detection becomes an important problem. We divide the dynamic network into network snapshots and process them as static network in DynaPro.

C. HRG

In the third step of DynaPro, we choose HRG as the middleware for differential privacy data protection. HRG proposed by Aaron Clauset et al. [30]. In order to hide the sensitive information of the edges in the graph, HRG uses the tree structure to represent the graph. The nodes in the tree store the probability value and represent the connection of the edges in the graph. For an internal node r, its left and right sub-trees are denoted by L_r and R_r respectively. Let n_{Lr} and n_{Rr} be the respective number of leaf nodes in L_r and R_r. Suppose e_r is the number of edges connecting the leaf nodes in L_r with those in R_r. Then p_r denotes the ratio of the actual edges in L_r and R_r to all possible edges formed by their leaf nodes.

$$p_r = \frac{e_r}{n_{L_r} n_{R_r}} \tag{1}$$

HRG construction method: A graph can have multiple corresponding HRG. We use the method mentioned in [31] to select the optimal HRG according to Markov Monte Carlo sampling method. A primitive graph can be divided in the different forms, which correspond to different HRGs and reflect the structural information of the original graph to varying degrees. Using Bayes' theorem [23], we measure the restoration of the original graph by the HRG model:

$$\Gamma(T, \{p_r\}) = \prod_{r\in T} p_r^{\varepsilon_r}(1-p_r)^{n_{L_r} n_{R_r} - \varepsilon_r} \tag{2}$$

According to the principle of maximum entropy, the larger the value of $\Gamma(T,\{p_r\})$, the more accurately the hierarchical random graph T can reflect the structural information of the original graph G. According to the principle of maximum entropy, the larger the value of $\Gamma(T,\{p_r\})$, the more accurately the hierarchical random graph T can reflect the structural information of the original graph G. The maximum value $\Gamma(T,\{p_r\})$ is obtained when P_r approaches 0 or 1.

D. DIFFERENTIAL PRIVACY

In the third step of DynaPro, we also select the differential privacy model to protect the privacy of data.

Differential privacy protection model aims to perturb the data by random noise before it is published. Thus, even if an attacker knows all the other records except the target one, the individual's private information cannot be inferred through data mining and analysis.

Definition 2 (ε -DP): A randomized algorithm M gives ε-DP, for any pair of neighboring datasets D1 and D2, and for every set of outcomes SM (SM ∈Range(B)), M satisfies equation (3):

$$P_r[M(D_1) \in SM] \leqq \exp(\varepsilon)^* P_r[M(D_2) \in SM] \tag{3}$$

Then it is said that algorithm M provides ε-difference privacy protection. The parameter ε is called privacy protection budget, which reflects the level of privacy protection. The smaller the value of ε, the higher the privacy protection level. Conversely, the greater the value of ε, the lower the level of privacy protection [32].

Laplace mechanism [10] and exponential mechanism [33] are two basic differential privacy protection mechanisms. Among them, Laplace mechanism is suitable for the protection of numerical results, and exponential mechanism is suitable for non-numerical results.

The Laplace mechanism realizes ε-differential privacy protection by adding random noise obeying Laplace distribution to the exact query results. Note that the position parameter is 0, The Laplace distribution with scale parameter b is Lap (b), then its probability density function is

$$p(x) = \frac{1}{2b}\exp(-\frac{|x|}{b}) \tag{4}$$

Definition 3 (Laplace Mechanism): Given any query function f: D→R^d, where D is the input dataset, the sensitivity of the function f is Δf, ε is the privacy budget, f satisfies Laplace mechanism when

$$L(D) = f(D) + Lap(\Delta f/\varepsilon) \tag{5}$$

It can be seen from the Laplace distribution of different parameters (see Fig. 1) that the smaller the ε, the greater the noise introduced.

The main idea of the exponential mechanism is to use a scoring function to screen the results with higher scores from the output results.

Definition 4 (Exponential Mechanism): The scoring function U of dataset D is u(D, r)→R. If algorithm A is proportional to the probability of $\exp(\frac{\varepsilon^* u(D,r)}{2\Delta u})$ from the result space ξ select r as the output, the algorithm A will satisfy ε-differential privacy.

E. TRIANGULAR MATRIX

The application of triangular matrix is the fourth step of DynaPro. In order to observe the average value of network graph, the HRG is transformed into triangular matrix. Fig. 2 is

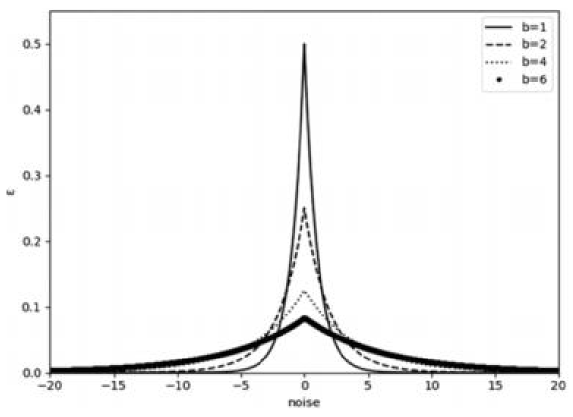
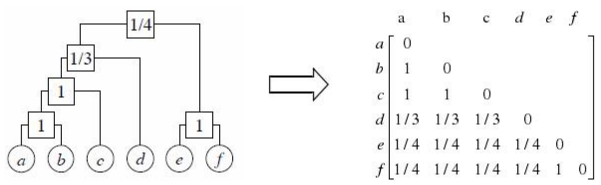


FIGURE 1. Laplace distribution function with different value b.



$$\begin{array}{c|cccccc} & a & b & c & d & e & f \\ \hline a & 0 & & & & & \\ b & 1 & 0 & & & & \\ c & 1 & 1 & 0 & & & \\ d & 1/3 & 1/3 & 1/3 & 0 & & \\ e & 1/4 & 1/4 & 1/4 & 1/4 & 0 & \\ f & 1/4 & 1/4 & 1/4 & 1/4 & 1 & 0 \end{array}$$

FIGURE 2. HRG converted to triangular matrix.

a schematic diagram of the transformation of a HRG into a triangular matrix.

The horizontal and longitudinal coordinates in the triangular matrix represent the nodes of the network graph, and the value of the transverse and longitudinal coordinates is the connection probability of the two nodes stored in the hierarchical random graph. Compared with other adjacent matrices, a triangular matrix can save storage space.

IV. DynaPro

As mentioned before, DynaPro is divided into four steps: division of dynamic network into snapshots, community detection, HRG processing, optimization and publishing of the network graph. In this section, we first introduce the steps of DynaPro in detail. Then we present the privacy analysis of DynaPro. The flowchart of DynaPro is shown in Fig. 3.

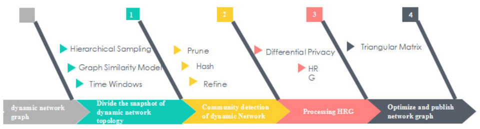


FIGURE 3. Flowchart of DynaPro.

A. *THE STEPS OF DynaPro*

1) DIVISION OF DYNAMIC NETWORK

This step solves the problem of how to deal with continuously changing topology of a dynamic network. The dynamic nature of the network requires continuous updates of the graphs with added noises, otherwise the network topology cannot be accurately represented. The real-time nature of the dynamic network requires that the noise-added graphs be released in a timely manner. Once a new snapshot is acquired, it should be processed and released immediately.

We analyze the dynamic network topology according to a time window. In order to ensure the time efficiency, we use stratified sampling when processing the data in each time window: select the snapshot graph from each layer time window according to the corresponding sampling ratio. At this point, we simplify the dynamic network diagram into a collection of extracted static snapshots. This approach saves time while ensuring data availability. We notice that the snapshots generated by the adjacent time windows have a similar structure. To save time while ensuring accuracy, we use the graph similarity model to remove similar snapshot graphs, and then output the snapshot graphs.

2) COMMUNITY DETECTION OF DYNAMIC NETWORK

Before protecting the dynamic network topology, we use community detection to control the amount of noises added. As can be seen from Fig. 4, in a network graph, there are often some sets the nodes of which are tightly connected inside each set, but the nodes crossing the sets are sparsely connected. We use this feature to add different noises to the edges inside a set and the edges between the sets to control the amount of noises added. Community detection like the PHASR approach [35] is used to find the aforementioned sets.

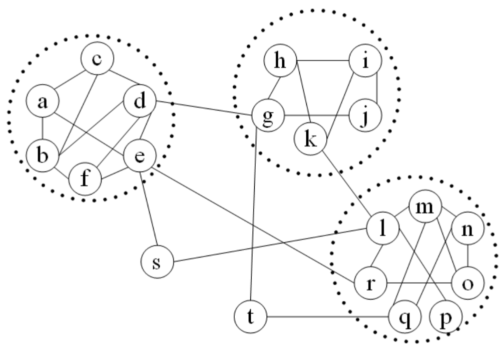


FIGURE 4. Communities in a network graph.

3) HRG PROCESSING

The third step of DynaPro is to use the HRG model and the differential privacy to process the communities returned by the second step. Our privacy protection of graph structure is divided into two parts: the first part is the privacy protection of the edge between the community, and the second part is the privacy protection of the edges within the community.

In the process of privacy protection within the community, if we add noises to the nodes, it will lead to significant changes in the structure of the network graph [36]. Therefore, as a novelty, we use HRG [6] as the middleware for data privacy protection in our differential privacy model. Using the community found in the second step, the HRG is constructed and noises are added to the internal nodes using the Laplacian mechanism. The internal nodes in the HRG represent the probability values of the connected edges between the leaf nodes in the left and right subtrees. Therefore, adding noise to the probability values stored by the nodes of HRG can protect the sensitive information of the edges of the network. Comparing Fig. 4 with Fig. 5, it can be found that in a better HRG, nodes within the same community tend to be concentrated in the same subtree.

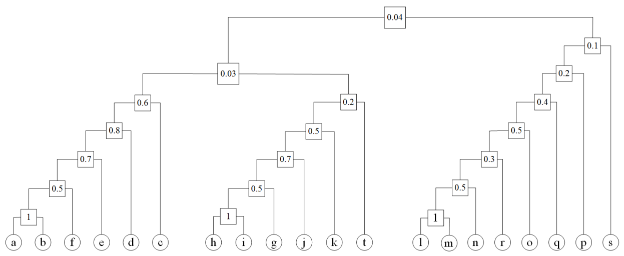


FIGURE 5. **Figure 4 communities corresponding HRG.**

4) OPTIMIZATION AND PUBLISHING OF NETWORK GRAPH

In the process of constructing HRG sampling, the exponential mechanism is used to provide differential privacy protection between communities. At the end of the step, we merge the child HRGs to facilitate the next step.

Through the previous steps, we get a number of HRGs for each time window, and we are looking for an overall network feature in the time window. In order to extract the average value of multiple HRGs, we use triangular matrix [20] by which the average characteristics are obtained and the final published network graph is reduced.

B. GENERATION OF SNAPSHOTS

Algorithms 2-3 present our approach for generating the snapshots of the dynamic network topology. In Algorithm 2, a time window is denoted as W, and contains the snapshot graph of Δt. In order to reduce the time complexity of the algorithm and more accurately reflect the latest structure of the network, we further divide a W into k layers in accordance with the time stamps. The newer the time stamp is, the higher the proportion of layered sampling is. So we can achieve the goal of accurately reflecting the latest structure of the network. R is the basic sampling ratio. The proportion of stratified sampling in layer j is j/k*r (1≤j≤k).

Next we carry out similarity filtering in Algorithm 3, which traverses the set S output by Algorithm 2 and filters the elements with low similarity to the filtered snapshot set S' until the traversal is complete.

Algorithm 2 Stratified_Sampling(W, k, r)

```
Input: Time window $W$, number of strata $k$, basic sampling rate $r$
Output: Set of the chosen snapshots $S$
1. Initialize an empty set $S = \{\}$
2. for $j$ from 1 to $k$ do
3.   for each snapshot $s$ in the stratum $W[j]$ do
4.     if $rand(0, 1) >=$ j/k*r then
5.       append $s$ to $S$
6.     end if
7.   end for
8. end for
9. return $S$
```

Algorithm 3 Similar_Graph_Filtering(S, x)

```
Input: Set of the chosen snapshots $S$, threshold of similarity $x$
Output: Filtered set of the snapshots $S'$
1. Initialize an empty set $S' = \{\}$
2. $i = 1$
3. while true do
4.   if $i >= len(S)$ then
5.     break
6.   end if
7.   append $S[i]$ to $S'$
8. $j = i + 1$
9. while $j < len(S)$ and $Graph\_Similarity(S[i], S[j]) < x$ do
10.    $j++$
11.  end while
12.   $i = j$
13.  end while
14.  return $S'$
```

C. COMMUNITY DETECTION

The community detection algorithm PHASR is divided into three steps, as shown in Algorithm 4. First, we compute composite bounds of individual intervals and attempt to prune them (Line 1-4 in Algorithm 4). Second, we use Hash function to find neighborhoods N_u^t (Line 5-11). Finally, we maintain and report multiple top communities (Line 12-20).

D. HRG PROCESSING

The hierarchical random graph is constructed with the Markov Monte Carlo sampling method [30]. Algorithm 5 shows this process.

In Algorithm 5, T_{t-1} is the current state of the Markov chain, and multiple T' of the next state is generated by replacing an internal node n. We use the exponential mechanism to screen the final T', and the scoring function of the exponential mechanism is $\exp(\frac{\varepsilon_1}{2\Delta u}\log\Gamma(T'))$, the transfer probability of Markov chain is $\min(\frac{\exp(\frac{\varepsilon_1}{2\Delta u}\log\Gamma(T'))}{\exp(\frac{\varepsilon_1}{2\Delta u}\log\Gamma(T_{t-1}))}, 1)$. In this step, we use the exponential mechanism to make differential privacy protection on the edges that are between the communities. The privacy protection budget is ε_1. The output is T_t.

Next, we use Algorithm 6 to add noise to the inside community, mainly using the Laplace mechanism in the differential privacy. If the privacy budget of the root node is set to ε_2, the number of layers is 1, α is the total number of

Algorithm 4 PHASR

```
Input: S', $\alpha$, LSH rows r, bands b, pruning res. l
Output: Communities of each snapshot
1. Compute bounds $\Phi$ at scales $l^i$, i = 0 ... $\lceil$ log(T)
2. Compute an estimate $\phi^*$ using $\Phi$
3. Prune intervals [t, t'] $\in \tau$ based on $\varphi c(G^{\tau}) \geq \varphi*$
4. Prune remaining intervals [t, t'] based on $\varphi c\ (G^{[t,t']}) \geq \varphi*$
5. for all (u, t) $\in$ (V, [1 ...T]) do
6.   for all scales $s \in$ 1 ...T/2 do
7.     if $\exists$ an unpruned [l, r] $\in$ [t − s, t + s], then
8.       Hash($N_u^t$, k∗(s), r, b)
9.     end if
10.  end for
11. end for
12. for $\forall$ Buckets B sorted by fill-factor do
13.   [l, r] = interval of B
14.   if $\varphi c(G^{[l,r]}) < \varphi*$ then
15.     (C, t, t' ) =Refine(B)
16.     $\varphi* = \min(\varphi*, \varphi$(C, t, t' ))
17.     add (C, t, t') to C
18.   end if
19.  end for
20. RETURN C
```

layers of the hierarchical random graph, and β is the number of layers where the node is located, the privacy budget of a node is $\frac{\alpha-\beta}{\alpha-1}\varepsilon_2$.

After adding noises, we use Algorithm 7 to merge self-hierarchical random graphs:

E. OPTIMIZATION AND PUBLISHING OF NETWORK GRAPH

Algorithms 8-9 show the last step of DynaPro. For N hierarchical random graphs obtained in the same time window, we convert them into N trigonometric matrices $A_i(1 \leq i \leq N)$. By finding the average value of N matrices, the average value of triangular matrix corresponding to time window is recorded as A_{avg}(Algorithm 8).

Finally, the trigonometric matrix is converted into the network diagram to be published (Algorithm 9).

F. PRIVACY ANALYSIS OF DynaPro

DynaPro mainly uses differential privacy protection in processing HRG. We discuss and analyze the privacy budget in this section.

In the third step of DynaPro, we have differential privacy protection between the communities, and the privacy budget is ε_1. The differential privacy protection is carried out by using Laplace mechanism to protect the nodes and edges within a community, and the privacy budget is ε_2. We set $\varepsilon = \varepsilon_1 + \varepsilon_2$, and $\varepsilon_1 < \varepsilon_2$, because the community's internal edges are more tightly connected and can take more noise;

Algorithm 5 Construct_HRG(G,ε_1)

```
Input: Subgraph $G$, privacy budget $\varepsilon_1$
Output: HRG $T$
1. initialize the Markov chain by generating an HRG $T_0$ according to
    $G$ randomly
2. set $t = 1$
3. while the Markov chain is not converged do
4.   choose an internal node $n$ in $T_{t-1}$ randomly
5.   generate $T'$ by replacing $n$ with its neighboring HRG in $T_{t-1}$
6.   if accept the transition with the probability of $\min(\frac{\exp(\frac{\varepsilon_1}{2\Delta u}\log\Gamma(T'))}{\exp(\frac{\varepsilon_1}{2\Delta u}\log\Gamma(T_{t-1}))}, 1)$ then
7.     $T_t = T\prime$
8.   else
9.     $T_t = T_{t-1}$
10.  end if
11.  $t = t + 1$
12. end while
13. return $T = T_t$
```

Algorithm 6 Add_Noise(T, ε, n, α, β)

```
Input: HRG $T$, privacy budget $\varepsilon$, internal node $n$, total layers count $\alpha$, number of layer $\beta$
Output: HRG $T'$ with noise
1. $T' = T$
2. $\varepsilon' = \frac{\alpha-\beta}{\alpha-1}\varepsilon$
3. $P_n = \min\{\frac{e_n+Lap(1/\varepsilon')}{n_{L_r}n_{P_r}}, 1\}$
4.  if $n$'s left sub-tree is not null then
5.    Add_Noise($T'$, $\varepsilon_2$, $n$'s left child, $\alpha$, $\beta$+1)
6.  end if
7.  if $n$'s right sub-tree is not null then
8.    Add_Noise($T'$, $\varepsilon_2$, $n$'s right child, $\alpha$, $\beta$+1)
9.  end if
10. return $T'$
```

the edges between the communities are more sparse and can withstand less noise.

Moreover, DynaPro is also in line with the constraints of the differential privacy model. The proof is as follows. At the end of the second step, the edge set E in graph G is divided into the edge between the community (E_1) and the edge within the community (E_2). We use privacy budget ε_1 to process E_1, and use privacy budget ε_1 and $\varepsilon_2$to process E_2. On the processing of E2, according to the sequence combination property of the differential privacy model, these two stages satisfy ε-difference privacy ($\varepsilon = \varepsilon_1 + \varepsilon_2$). Because $E=E_1+E_2$, and according to the parallel combination property of differential privacy, the algorithm generally satisfies ε-differential privacy between the community (E_1) and the edge within the community (E_2). We use privacy budget $\varepsilon_1$to process E_1, and use privacy budget ε_1 and $\varepsilon_2$to process E_2. On processing E2, according to the sequence combination

Algorithm 7 Aggregate_HRG(H, ε_2)

```
Input: HRG set $H$, privacy budget $\varepsilon_2$
Output: HRG $T$
1. initialize the Markov chain by generating an HRG $T_0$ according to root nodes in $H$ randomly.
2. set $t = 1$
3. while the Markov chain is not converged do
4.   choose an internal node $n$ in $T_{t-1}$ randomly
5.   generate $T'$ by replacing $n$ with its neighboring HRG in $T_{t-1}$
6.   if accept the transition with the probability of $\min(\frac{\exp(\frac{\varepsilon_1}{2\Delta u}\cdot\log\Gamma(T'))}{\exp(\frac{\varepsilon_1}{2\Delta u}\cdot\log\Gamma(T_{t-1}))}, 1)$ then
7.     $T_t = T'$
8.   else
9.     $T_t = T_{t-1}$
10.  end if
11.  $t = t + 1$
12. end while
13. return $T = T_t$
```

Algorithm 8 HRG_to_Tri_Matrix ($T_1, T_2, \ldots, T_N$)

```
Input: $N$ HRGs, $T_1, T_2, \ldots, T_N$
Output: The average triangular matrix $A_{avg}$
1. for $i$ from 1 to $N$ do
2.   initialize matrix, $A_i[j][j] = 0$
3.   for every two leaf nodes $m$ and $n$ $(m < n)$ in $T_i$ do
4.     find the lowest common ancestor of $m$ and $n$
5.     get the value of the ancestor in HRG, denoted as $p_{mn}$
6.     set the matrix element $A_i[m][n] = p_{mn}$
7.   end for
8. end for
9. calculate the average matrix of $A_1, A_2, \ldots, A_N$, denoted as $A_{avg}$
10. return $A_{avg}$
```

property of the differential privacy model, these two stages satisfy ε- differential privacy ($\varepsilon = \varepsilon_1 + \varepsilon_2$). Because E=E$_1$+E$_2$, according to the parallel combination property of differential privacy, the algorithm generally satisfies ε-differential privacy (max $(\varepsilon, \varepsilon_2) = \varepsilon$).

V. EXPERIMENTAL EVALUATION

In this section, we will use experiments to verify the effectiveness of DynaPro. It includes experimental setup and design, evaluation metrics, and analysis of results.

A. EXPERIMENTAL SETUP

The configuration of our computing system is shown in TABLE 1.

In order to show the effectiveness of DynaPro, we select two real dynamic network graphs from the SNAP graph database [19]. There are two reasons to choose these two datasets. First, Both AS-Internet and AS-Caida have a certain number of dynamic network snapshot graphs, which meet the requirements of this paper. Second, from TABLE 2, we can see that the two datasets are very different in the number of nodes, the number of edges, the number of snapshot graphs and so on, which is convenient for us to do comparative experiments and test the processing ability and stability of the algorithm. The information about the two datasets is shown in TABLE 2.

Algorithm 9 Reconstruct_Graph (G, A_{avg})

```
Input: Original graph $G$, average triangular matrix $A_{avg}$
Output: Sanitized graph $G'$
1. initialize $G'$ by putting all vertices of $G$ in it
2. for each pair of vertices $i, j \in G'$ do
3.   find the value $p_{ij}$ of $A_{avg}[i][j]$ in $A_{avg}$
4.   place an edge between $i$ and $j$ in $G'$ according to the probability stored in $p_{ij}$
5. end for
6. return $G'$
```

TABLE 1. Configuration of computing system.

name	Configuration details
CPU	16 Intel Xeon 2.27GHz
memory	12GB
disk	500GB SSD
Systemimage	Ubuntu 16.04.1 64-bit server

TABLE 2. Dataset summary.

dataset	AS-Internet	AS-Caida
Minimum number of nodes	103	8020
Maximum number of nodes	6474	26475
Minimum number of edges	243	36406
Maximum number of edges	13233	106762
Number of snapshot graphs	733	122
Type of edge	undirected	directed

B. EXPERIMENTAL DESIGN

We set two sets of comparative experiments.

In the first step of DynaPro, we use hierarchical sampling and graph similarity model to ensure the time efficiency. In order to show DynaPro is effective, we set up a set of comparative experiments. We choose AS-Internet dataset to observe the results of privacy protection with DynaPro and a DCP algorithm. The DCP (Dynamic Network Community Discovery Protection) algorithm only uses steps two, three and four of DynaPro to protect data directly.

In the second group of our comparative experiments, we compare DynaPro with DPDHRG [37] and PrivHRG [6] using datasets AS-Internet and AS-Caida.

C. EVALUATION METRICS

Three evaluation metrics are introduced in our evaluation.

1) Degree Distribution: The degree of a node is the number of edges between a node and the nodes connecting to it. It can reflect the structural information of the network.

2) Shortest Path Length Distribution: It calculates the length of the path between each node in the original graph and the publishing diagram, and gives the distribution of several shortest paths.
3) Overlap of Top-K Vertices: The central performance of the nodes reflects the propagation characteristics of the network. The centrality of nodes is judged by eigenvector centrality [6].

For the first group of experiments, we select AS-Internet dataset, and take the number of snapshot graphs processed as independent variables to verify the influence of hierarchical sampling and graph similarity algorithm on the time and accuracy of the results. The dependent variables referenced are the distribution of time and node degree.

For the second set of experiments, we will verify the effectiveness of the proposed algorithm on AS-Internet and AS-Caida datasets by verifying the influence of the overall budget of differential privacy on the results. The other variables used in the experiment are shown in Table 3.

TABLE 3. Parameter settings.

Parameter	Value
r	0.6
Time window length w (to AS-Internet)	50
Time window length w (to AS-Caida)	20

D. ANALYSIS OF RESULTS

The first set of experimental results is shown in Fig. 6 and Fig. 7. For comparison, we take the average relative error between the experimental results and the original graph on the degree distribution nodes.

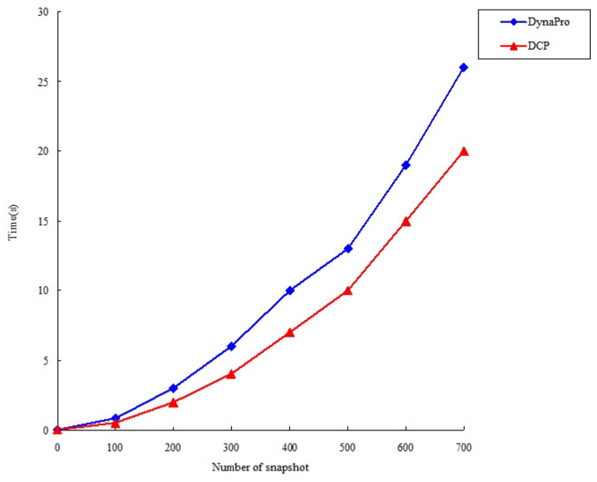


FIGURE 6. Comparison of algorithm efficiency (AS-Internet).

As can see from Fig. 6, DynaPro is superior to DCP at any stage in the figure in terms of completion time, because DynaPro filters similar snapshots and saves time for later processing.

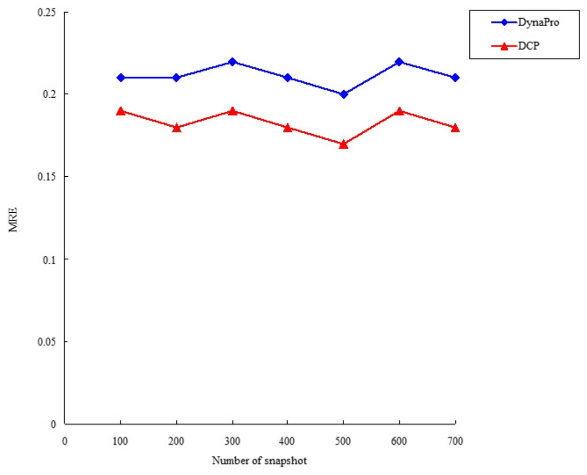


FIGURE 7. Comparison of algorithm accuracy (AS-Internet).

From Fig. 7, in the index of node degree distribution, the node degree distribution of DCP is lower than that of DynaPro in most cases. It shows that the network structure of DCP is better retained than DynaPro after adding irritability, because DCP retains more dynamic network snapshot diagrams. However, the error is within an acceptable range.

In the second set of experiments, because the PrivHRG algorithm has no processing strategy for the dynamic network, it processes all snapshot graphs in a time window one by one, and finally finds the average value as the result for publishing. In order to facilitate the comparison, we use the MRE between the result and the original graph in the display of the first two indices. Fig. 8 and Fig. 9 are the comparison results of three algorithms on node degree distribution. Fig. 10 and Fig. 11 are the comparison results of three algorithms on the shortest path length distribution. Fig. 12 and Fig. 13 are the comparison results of three algorithms on Overlap of Top-K Vertices.

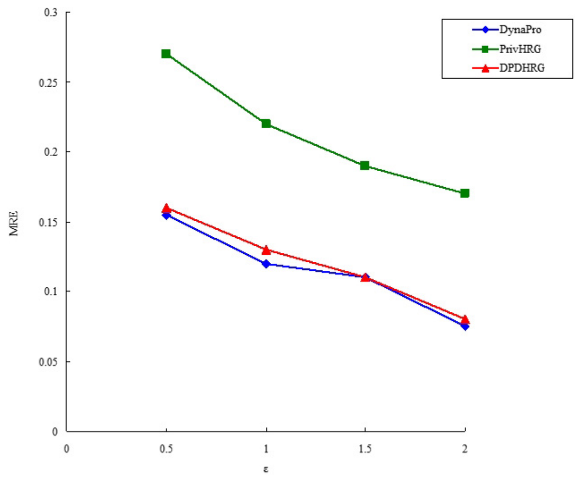


FIGURE 8. MRE of degree distribution (AS-Internet).

Degree Distribution: It can be seen from Fig. 8 and Fig. 9, the MRE of the results obtained by DynaPro is lower than the

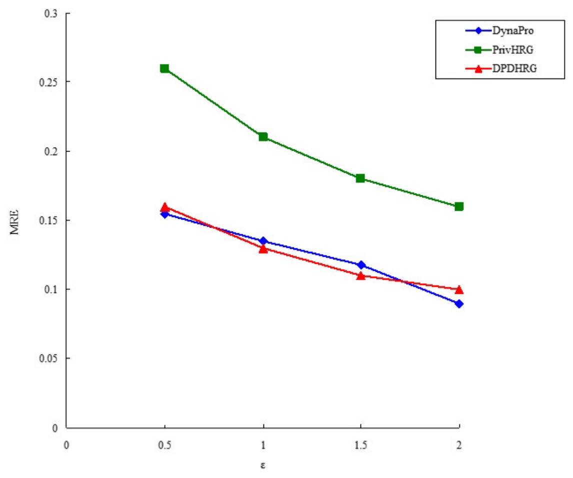


FIGURE 9. MRE of degree distribution (AS-Caida).

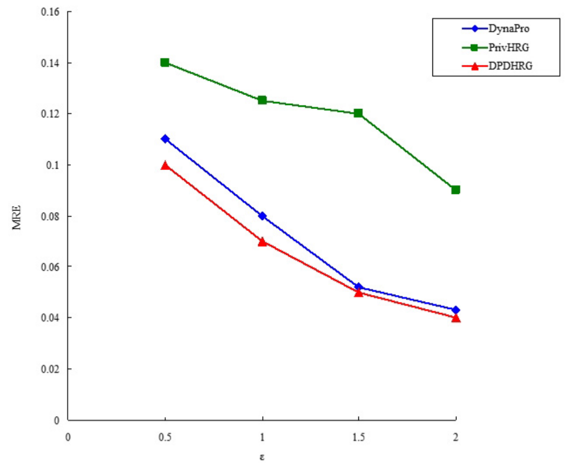


FIGURE 10. MRE of shortest path length distribution(AS-Internet).

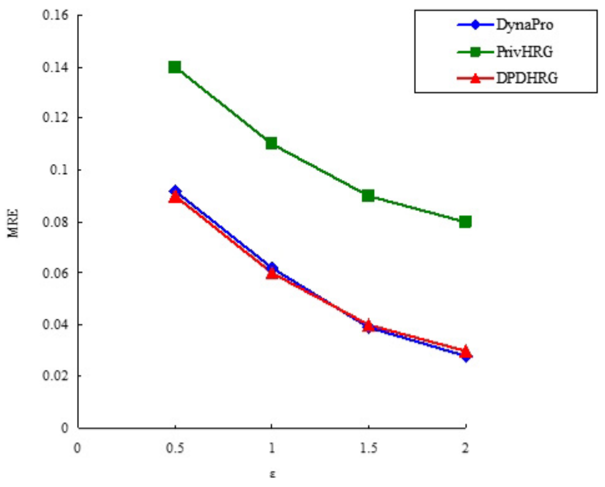


FIGURE 11. MRE of shortest path length distribution (AS-Caida).

PrivHRG algorithm, and slightly lower than the DPDHRG algorithm, in both AS-Internet and AS-Caida datasets.

Shortest Path Length Distribution: As you can see from Fig. 10 and Fig. 11, the MRE of the results obtained by DynaPro is lower than that of the PrivHRG algorithm on the two datasets, It is not much different from the DPDHRG algorithm.

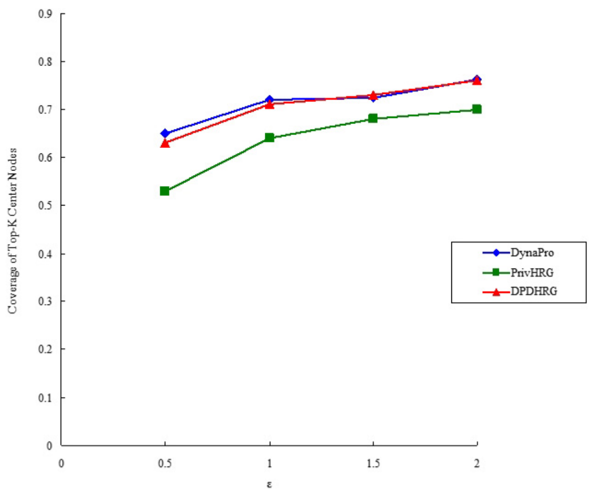


FIGURE 12. Coverage of Top-K center nodes (AS-Internet).

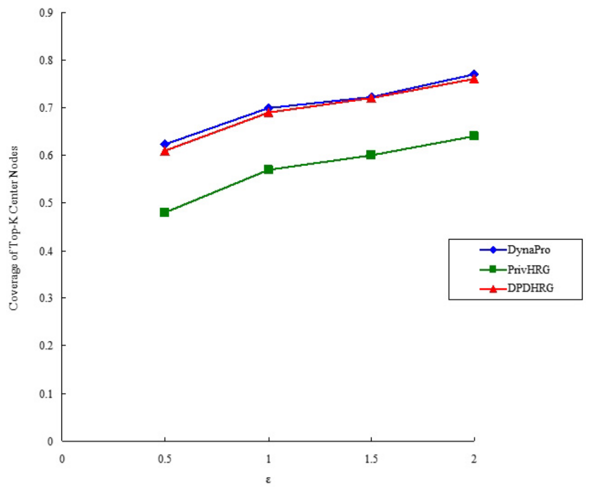


FIGURE 13. Coverage of Top-K center nodes(AS-Caida).

Overlap of Top-K Vertices: On the two datasets, the Top-K center point coverage of DynaPro is higher than that of PrivHRG, and slightly higher than that of DPDHRG. This shows that our algorithm is more effective in retaining the structural features of the original graph.

From the above experimental results, it can be seen that DynaPro is feasible and efficient in terms of node degree distribution, shortest path length distribution, and Overlap of Top-K Vertices.

VI. CONCLUSION

In the deployment of the Internet of things, in order to meet the privacy protection requirements for the dynamic network topology of WSN, we propose DynaPro, a data protection algorithm based on differential privacy for dynamic wireless sensor networks, i.e., the sensor layer of IoT. DynaPro has adopted methods like time window, hierarchical sampling, and graph similarity to process the snapshots of the dynamic network topology. It uses Hierarchical Random Graph (HRG) as middleware to apply differential privacy protection. Instead of adding noise to network data and topology directly, DynaPro adds noise to HRG. In this way, DynaPro can hide the network topology but retain the necessary structural information to support data analysis. We have done experimental evaluation with two real datasets, AS-Internet and AS-Caida. We show that the DynaPro is feasible and efficient in terms of running time, accuracy, and structural preservation of the dynamic network of WSN.

REFERENCES

[1] M. Ammar, G. Russello, and B. Crispo, "Internet of Things: A survey on the security of IoT frameworks," *J. Inf. Secur. Appl.*, vol. 38, pp. 8–27, Feb. 2018.

[2] C. Chiasserini, M. Garetto, and E. Leonardi, "De-anonymizing clustered social networks by percolation graph matching," *ACM Trans. Knowl. Discovery Data*, vol. 12, no. 2, p. 21, Mar. 2018.

[3] Y.-L. Liu, Y.-P. Wang, X.-F. Wang, Z. Xia, and J.-F. Xu, "Privacy-preserving raw data collection without a trusted authority for IoT," *Comput. Netw.*, vol. 148, pp. 340–348, Jan. 2019.

[4] Y. Zhang, Q.-J. Chen, and S. Zhong, "Privacy-preserving data aggregation in mobile phone sensing," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 5, pp. 980–992, May 2016.

[5] B.-T. Mi, X. Liang, and S.-S. Zhang, "A survey on social Internet of Things," *Chin. J. Comput.*, no. 7, pp. 41–47, Jul. 2018.

[6] Q. Xiao, R. Chen, and K.-L. Tan, "Differentially private network data release via structural inference," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 911–920.

[7] P. Samarati, "Protecting respondents' identities in microdata release," *IEEE Trans. Knowl. Data Eng.*, vol. 13, no. 6, pp. 1010–1027, Nov./Dec. 2001.

[8] K. Mershad and H. Artail, "A framework for secure and efficient data acquisition in vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 2, pp. 536–551, Feb. 2013.

[9] X. Dong, L. Wei, H. Zhu, Z. Cao, and L. Wang, "An efficient privacy-preserving data- forwarding scheme for service-oriented vehicular Ad Hoe networks," *IEEE Trans Veh. Technol.*, vol. 60, no. 2, pp. 580–591, Feb. 2011.

[10] T. Shen, F. Wang, K. Chen, K. Wang, and B. Li, "Efficient leveled (multi) identity-based fully homomorphic encryption schemes," *IEEE Access*, vol. 7, pp. 79299–79310, 2019.

[11] F. Bayatbabolghani and M. Blanton, "Secure multi-party computation," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2018, pp. 2157–2159.

[12] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. TCC*, 2006, pp. 265–284.

[13] L. C. Zhao, L. H. Ni, S. S. Hu, Y. J. Chen, P. Zhou, F. Xiao, and L. B. Wu, "InPrivate digging: Enabling tree-based distributed data mining with differential privacy," in *Proc. IEEE Int. Conf. Comput. Commun.*, Honolulu, HI, USA, Apr. 2018, pp. 2087–2095.

[14] D. Westhoff, J. Girao, and M. Acharya, "Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation," *IEEE Trans. Mobile Comput.*, vol. 5, no. 10, pp. 1417–1431, Oct. 2006.

[15] C. Castelluccia, A. C.-F. Chan, E. Mykletun, and G. Tsudik, "Efficient and provably secure aggregation of encrypted data in wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 5, no. 3, 2009, Art. no. 20.

[16] Z. Ye, T. Wen, Z. Liu, X. Song, and C. Fu, "An efficient dynamic trust evaluation model for wireless sensor networks," *J. Sensors*, vol. 11, no. 2, Oct. 2017, Art. no. 7864671.

[17] N. Sun, J. Xu, H. Wei, H. Miao, and J. Wang, "A network state based reliability evaluation model for WSNs," in *Proc. SNPD*, Jun. 2017, pp. 303–308.

[18] Y. Wang, X. Wu, and L. Wu, "Differential privacy preserving spectral graph analysis," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*, Gold Coast, Australia, 2013, pp. 329–340.

[19] F. Ahmed, R. Jin, and A. X. Liu, "A random matrix approach to differential privacy and structure preserved social network graph publishing," *CoRR*, vol. 12, no. 4, pp. 13–24, 2013.

[20] D. Kang, "Research on dynamic network publishing algorithm based on differential privacy and game theory," M.S. thesis, Dept. CS. Dalian Maritime Univ., Dalian, China, 2017.

[21] W. Chen, Z. Liu, X. Sun, and Y. Wang, "A game-theoretic framework to identify overlapping communities in social networks," *Data Mining Knowl. Discovery*, vol. 21, no. 2, pp. 224–240, 2010.

[22] Z. Dhouioui and J. Akaichi, "Tracking dynamic community evolution in social networks," in *Proc. IEEE Int. Conf. Adv. Social Netw. Anal. Mining*, Aug. 2014, pp. 764–770.

[23] N. Ilhan and Ş. G. Ögüdücü, "Predicting community evolution based on time series modeling," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, Aug. 2015, pp. 1509–1516.

[24] T. He, L. Cai, T. Meng, L. Chen, Z. Deng, and Z. Cao, "Parallel community detection based on distance dynamics for large-scale network," *IEEE Access*, vol. 6, pp. 42775–42789, 2018.

[25] G. Rossetti, L. Pappalardo, D. Pedreschi, and F. Giannotti, "Tiles: An online algorithm for community discovery in dynamic social networks," *Mach. Learn.*, vol. 106, no. 8, pp. 1213–1241, 2017.

[26] P. Agarwal, R. Verma, A. Agarwal, and T. Chakraborty, "DyPerm: Maximizing permanence for dynamic community detection," in *Proc. PAKDD*, no. 1, 2018, pp. 437–449.

[27] J. Viard, M. Latapy, and C. Magnien, "Computing maximal cliques in link streams," *Theor. Comput. Sci.*, vol. 609, no. P1, pp. 245–252, 2016.

[28] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2002, pp. 721–724.

[29] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Sov. Phys.-Dokl.*, vol. 10, pp. 707–710, Feb. 1966.

[30] A. Clauset, C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, no. 7191, pp. 98–101, 2008.

[31] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 6, no. 10, pp. 2000–2008, 2008.

[32] D. P. Gruska, "Differential privacy and security," *Fundamenta Informaticae*, vol. 143, nos. 1–2, pp. 73–87, 2016.

[33] K. Huang and X. Fu, "Detecting overlapping and correlated communities without pure nodes: Identifiability and algorithm," in *Proc. ICML*, 2019, pp. 2859–2868.

[34] C. Lin, Z. Song, H. Song, Y. Zhou, Y. Wang, and G. Wu, "Differential privacy preserving in big data analytics for connected health," *J. Med. Syst.*, vol. 40, no. 4, p. 97, 2016.

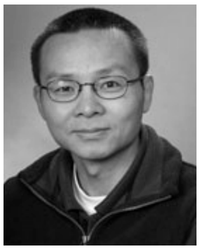
[35] D. J. DiTursi, G. Ghosh, and P. Bogdanov, "Local Community Detection in Dynamic Networks," in *Proc. ICDM*, Nov. 2017, pp. 847–852.

[36] B. P. Nguyen, H. Ngo, and J. Kim, "Publishing graph data with subgraph differential privacy," in *Proc. Int. Workshop Inf. Secur. Appl.*, Jeju Island, South Korea, 2015, pp. 134–145.

[37] S. Y. Li, Z. B. Liu, and K. Dong, "Dynamic network data protection algorithm using differential privacy in Internet of Things," in *Proc. IEEE Int. Conf. Smart Internet Things*, Tianjin, China, Aug. 2019, pp. 306–313.

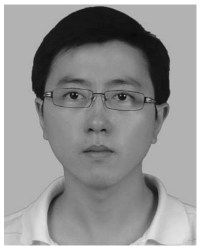
SONGYAN LI received the B.Eng. degree in Internet of Things engineering from Dalian Maritime University, China, in 2018, where she is currently pursuing the master's degree with the Department of Computer Science. Her research focuses on big data and data privacy protection.

ZHAOBIN LIU received the Ph.D. degree in computer science from the Huazhong University of Science and Technology, China, in 2004. He was a Senior Visiting Scientist with The University of Auckland, New Zealand, in 2017, and a Visiting Scholar with the University of Central Florida, USA, in 2014, and University of Otago, New Zealand, in 2008, respectively. He is currently a Professor with the School of Information Science and Technology, Dalian Maritime University, China. His research interests include big data, cloud computing, and data privacy.



ZHIYI HUANG received the B.Sc. and Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), China, in 1986 and 1992, respectively. He was a Visiting Professor with the Swiss Federal Institute of Technology Lausanne (EPFL) and Tsinghua University, in 2005, and a Visiting Scientist with MIT CSAIL, in 2009. He is currently an Associate Professor with the Department of Computer Science, University of Otago, New Zealand. He has published over 150 articles in well-known conferences and journals. His research fields include parallel/distributed computing, multicore architectures, operating systems, green computing, cluster/grid/cloud computing, high-performance computing, signal processing, and computer architectures and networks.

HAOZE LYU is currently with the Department of Computer Science, Dalian Maritime University. His research mainly focuses on big data and privacy protection.



ZHIYANG LI received the bachelor's and Ph.D. degrees from the Dalian University of Technology, China, in 2004 and 2011, respectively. He is currently an Associate Professor with the School of Information Science and Technology, Dalian Maritime University, China. His research interests include mobile computing, information retrieval, and computer vision.

WEIJIANG LIU received the Ph.D. degree in computational mathematics from Jilin University, in 1998. From June 2004 to June 2006, he was a Postdoctoral Fellow of Postdoctoral Station for Computer Science and Technology, Southeast University, China. He is currently a Professor with the School of Information Technology, Dalian Maritime University, China. He has published more than 80 articles. His current research interests include network measurement, network security, and mobile computing.