World Wide Web (2021) 24:1551–1561
https://doi.org/10.1007/s11280-021-00922-2

Data privacy preservation algorithm with *k*-anonymity

Waranya Mahanan[1] · W. Art Chaovalitwongse[2] · Juggapong Natwichai[3]

Received: 21 May 2020 / Revised: 15 June 2021 / Accepted: 5 July 2021 /
Published online: 28 July 2021

Abstract
With growing concern of data privacy violations, privacy preservation processes become more intense. The *k*-anonymity method, a widely applied technique, transforms the data such that the publishing datasets must have at least *k* tuples to have the same linkable attribute, quasi-identifiers, values. From the observations, we found that, in a certain domain, all quasi-identifiers of the datasets, can have the same data type. This type of attribute is considered as an Identical Generalization Hierarchy (*IGH*) data. An *IGH* data has a particular set of characteristics that could utilize for enhancing the efficiency of heuristic privacy preservation algorithms. In this paper, we propose a data privacy preservation heuristic algorithm on *IGH* data. The algorithm is developed from the observations on the anonymous property of the problem structure that can eliminate the privacy constraints consideration. The experiment results are presented that the proposed algorithm could effectively preserve data privacy and also reduce the number of visited nodes for ensuring the privacy protection, which is the most time-consuming process, compared to the most efficient existing algorithm by at most 21%.

Keywords Data privacy preservation · *k*-anonymity · Optimal algorithm

1 Introduction and motivation

In the past decades, an important issue of data releasing for further utilization is data privacy. One of the important privacy preservation models, *k*-anonymity, is proposed in [9]. The

This article belongs to the Topical Collection: *Special Issue on Intelligent Fog and Internet of Things (IoT)-Based Services*
Guest Editors: Farookh Hussain, Wenny Rahayu, and Makoto Takizawa

✉ Juggapong Natwichai
juggapong@eng.cmu.ac.th

[1] Data Engineering Laboratory, Computer Engineering Department, Faculty of Engineering, Chiang Mai University, Chiang Mai, Thailand

[2] Department of Industrial Engineering, University of Arkansas, Fayetteville, AR, USA

[3] Data Engineering Laboratory, Computer Engineering Department, Faculty of Engineering and Data Analytics and Knowledge Synthesis for Healthcare Center, Chiang Mai University, Chiang Mai, Thailand

released data must have at least k identical tuples for guaranteeing data privacy preservation. In the k-anonymity process, the link-able attributes [10], so-called the quasi-identifiers, are to be replaced with the more general data [13]. The distortion, which causes loss of information, occurs after the data is generalized [12]. Thus, to ensure both data privacy and minimize information loss, the optimal k-anonymity is highly desired.

There are a few optimal k-anonymity algorithms, which have been proposed for preserving the privacy of the generalized dataset [1, 3, 4]. The existing algorithms are based on the searching through the generalization lattice [1], the lattice that represents all generalization schemes in the k-anonymity method, for determining the optimal k-anonymity solution. The generalization schemes which satisfy the k-anonymity and loss minimal information will consider the optimal solution. The Flash algorithm [3] determines the optimal k-anonymity solutions by binary search on the lattice of generalization. The algorithm searches through all paths in the generalization lattice and terminates when all paths are traversed completely. The incognito algorithm is proposed, in [4], by dividing the lattice into sub-lattices and searching through each sub-lattice with breadth-first search manners until the search on all sub-lattices is complete. The Optimal Lattice Anonymization (OLA) algorithm is also proposed to address the optimal k-anonymity problem. The algorithm divides the generalization lattice into sub-lattices and determines whether all sub-lattices satisfy the k-anonymity condition until the optimal solution is found. These existing algorithms are proposed for the general dataset, a dataset with various datatype. Therefore, in [6], the authors proposed that there is a special type of datasets that all quasi-identifiers are in the same domain, so-called an Identical Generalization Hierarchy (*IGH*) data. With this type of datasets, the special characteristics of the optimal k-anonymity solution on *IGH* data are discovered. The optimal solution of an IGH dataset is always at the lowest level found k-anonymity satisfied node. Thus, the algorithms for *IGH* data privacy-preserving that could utilize this characteristic was proposed in our previous work, Optimal-IGH [7]. The algorithm is especially proposed for an IGH dataset by performing the pre-order and post-order depth-first search manners. The algorithm can effectively find the optimal k-anonymity solution on an IGH dataset. However, due to the characteristic of the generalization lattice, the generalization schemes which are considered to satisfy the k-anonymity condition, all its children are also considered the k-anonymity condition without visiting it. Thus, the pre-order and post-order manners might not be the most efficient solution for finding the optimal solution.

In this paper, we propose an extended heuristic optimal k-anonymity algorithm on the *IGH* which further improves the efficiency. The algorithm enhances the performance of the existing algorithm by performing the heuristic search on the given generalization lattice on the in-order manners. From the observed characteristics of the *IGH* data, the algorithm could leave some nodes unvisited which could faster find the optimal k-anonymity solution. The result shows that our proposed data privacy preservation heuristic algorithm could produce the privacy preserved datasets while minimizing the information loss and also reducing the running time compared with the existing algorithms.

2 Background

For preserving data privacy, the k-anonymity, which is one of the most prominent approaches, can be applied. The privacy preserved dataset must have an equivalent set of quasi-identifiers [9] at least k tuples in order to satisfy the k-anonymity condition, so-called k-anonymous dataset. In order to transform the given dataset into the k-anonymous, some

Table 1 An example of a non-IGH dataset

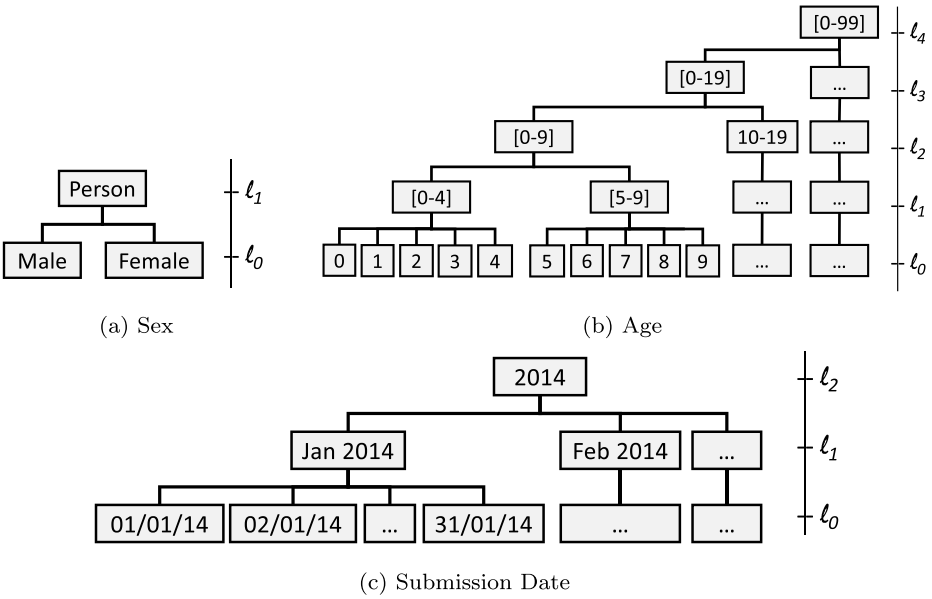
ID	Quasi-identifiers			ID	Quasi-identifiers		
	Sex	Age	Submission date		Sex	Age	Submission date
(a) Original dataset				(b) 2-anonymous dataset			
1	Male	21	01/01/2014	1	Person	[20-29]	Jan 2014
2	Male	25	04/01/2014	2	Person	[20-29]	Jan 2014
3	Male	27	22/01/2014	3	Person	[20-29]	Jan 2014
4	Female	27	4/01/2014	4	Person	[20-29]	Jan 2014
5	Female	28	19/01/2014	5	Person	[20-29]	Jan 2014

quasi-identifiers need to be replaced or generalized by using the generalization hierarchy [10, 12].

An example dataset, from patients medical data, is illustrated in Table 1(a). The quasi-identifiers are Sex, Age, and Submission date. Using the generalization hierarchy of Sex, Age and Submission date quasi-identifiers in Figure 1(a), (b), and (c) respectively, the 2-anonymous dataset shows in Table 1(b).

2.1 Identical generalization hierarchy data

Typically, the given dataset for privacy preservation has diverse types of data and also has different generalization hierarchies. In [7], the authors discovered that in some datasets all quasi-identifiers could be the same data type. As an example in Table 2, the quasi-identifiers are the satisfaction scores from 0 to 5 that each user gave to the taxi drivers.



(a) Sex

(b) Age

(c) Submission Date

Figure 1 The generalization hierarchy of a non-IGH dataset

Table 2 An example of an *IGH* dataset

ID	Quasi-identifiers			ID	Quasi-identifiers		
	Taxi 1	Taxi 2	Taxi 3		Taxi 1	Taxi 2	Taxi 3
(a) Original dataset				(b) 2-anonymous dataset			
1	4	0	1	1	[0-5]	[0-2]	[0-2]
2	4	0	1	2	[0-5]	[0-2]	[0-2]
3	1	1	1	3	[0-5]	[0-2]	[0-2]
4	1	4	3	4	[0-5]	[3-5]	[3-5]
5	1	5	5	5	[0-5]	[3-5]	[3-5]
6	0	4	4	6	[0-5]	[3-5]	[3-5]
7	0	4	3	7	[0-5]	[3-5]	[3-5]
8	0	4	5	8	[0-5]	[3-5]	[3-5]

Clearly, the quasi-identifiers use only one generalization hierarchy, shown in Figure 2, in order to produce the k-anonymous dataset. This special type of dataset is referred to as an Identical Generalization Hierarchy (*IGH*) data.

Definition 1 (Identical generalization hierarchy data) Let $H = \{H_1, H_2, \ldots, H_m\}$ be the set of the generalization hierarchy function of attributes $\{A_1, A_2, \ldots, A_m\}$ in a dataset T . A dataset T is an IGH data if and only if $\bigcup_{i=1}^{m} H_i = H_1 = H_2 = \ldots = H_m$.

The generalized data that satisfy the 2-anonymity condition presents in Table 2(b). The dataset can be generalized using the same generalization hierarchy for all quasi-identifiers as they are in the same domain.

2.2 Optimal *k*-anonymity on *IGH*

Since the information loss could occur when the dataset is generalized to satisfy the k-anonymity condition, the optimality must be concerned. To determine optimal k-anonymity solutions, the generalization lattice could be applied for representing all generalization schemes. An example of the generalization lattice is shown in Figure 3, where each node represents a generalization scheme, and edges are the direct generalization of each scheme. For instance, node $<102>$ is the generalization scheme of the generalized dataset in which the quasi-identifier Taxi 1, Taxi 2, and Taxi 3 are generalized to level 1, 0, and 2 respectively. The nodes that satisfy the k-anonymity constraint, or k-anonymous node, are presented

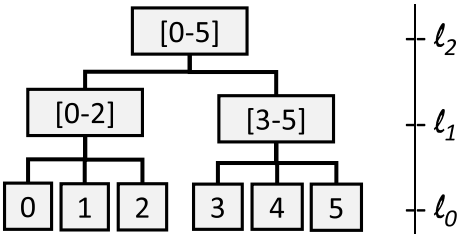


Figure 2 The generalization hierarchy of an *IGH* dataset

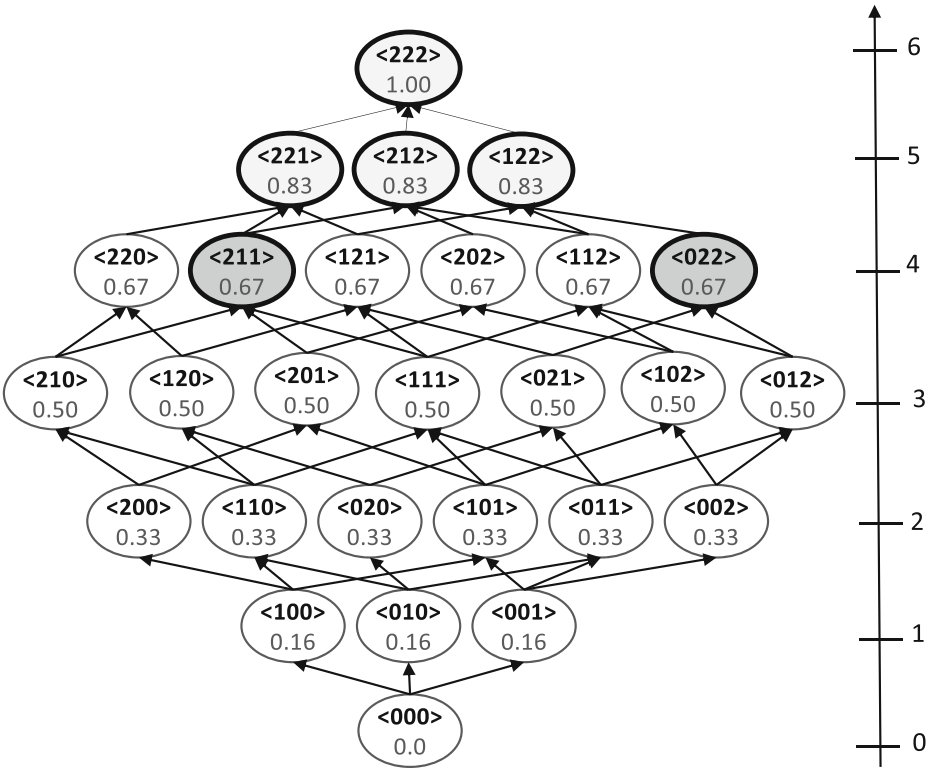


Figure 3 The generalization lattice

as the shaded nodes in the figure. The lower line value in the nodes represents the information loss, calculated by the precision [11] shown in (1). The height(H) denoted as the highest height of the generalization of each quasi-identifier and m be the number of quasi-identifiers. The higher precision means higher information loss.

$$Prec(GT) = \frac{1}{m} \cdot \sum_{i=1}^{m} \frac{h_i}{height(H_i)} \tag{1}$$

From the original dataset in Table 2(a) and the generalization hierarchy in Figure 2, the optimal solution which is the k-anonymous nodes with the lowest precision, is node $<211>$ and $<022>$ of the generalization lattice in Figure 3.

$$C_{avg}(GT) = \frac{|GT|}{count(E) \cdot k} \tag{2}$$

Since we need only one answer, the Average Class Size (C_{avg}) information loss metric [5] is used among the k-anonymous node with the lowest precision. The equation of the C_{avg} shown in (2) while $|GT|$ be a number of tuples and E be a number of the equivalence classes. Therefore, the optimal solution is node $<022>$ with the lowest $C_{avg} = 0.89$.

3 Algorithm description

3.1 Extended-OIGH algorithm

For an *IGH* data, we observe that the special characteristics of the generalization lattice are as follows.

1. All direct generalization nodes in the higher generalization lattice level of a k-anonymous node are considered a k-anonymous node.
2. All direct generalization nodes in the lower generalization lattice level of a non-anonymous node are considered a non-anonymous node.
3. The precision of the nodes at the same generalization lattice level is identical.
4. The precision of the node at a higher generalization lattice level will always higher than at the lower generalization lattice level.

Therefore, with all these special characteristics, the optimal k-anonymous node is always among the nodes in the lowest k-anonymous level, the level of the lattice which has the k-anonymous nodes.

Our proposed Extended-OIGH algorithm is designed based on the tree search, by performing the in-order depth-first search to find the k-anonymous node with the lowest precision. From the special characteristics on the optimal k-anonymity of an *IGH* data, the optimal solution would always be at the lowest level found the k-anonymous node. Thus, we design the algorithm using the benefit of these characteristics.

Algorithm 1 $Extended\text{-}OIGH(L)$.

```
Input: lowest level found k-anonymous node L
Output: optimal k-anonymity node op
1  begin
2      R = TraversalRoute(L)
3      if !R then
4          OP ← an optimal node among the k-anonymous nodes in level L
5          Return OP
6      else
7          foreach node in R do
8              if node is not tagged then
9                  if node is k-anonymous then
10                     Mark node as k-anonymous node
11                     Tag all successor nodes as k-anonymous
12                     L ← node.level
13                     Exit foreach loop
14                 else
15                     Mark node as non-anonymous node
16                     Tag all predecessor nodes as non-anonymous
17                 end
18             end
19         end
20         Extended-OIGH(L)
21     end
22 end
```

The algorithm first begins at the Extended-OIGH algorithm, as shown in Algorithm 1, with the lowest level found k-anonymous node L as an input. At first, the input L will be set as the highest level of the generalization lattice. Then, the TraverseRoute sub-algorithm, in Algorithm 2, will be called for providing the route from the root node $<000>$ at level 0 to a node at level L. From the route, the algorithm iterative determines the k-anonymity of each node in the route started from the node at the highest level. If the node is a k-anonymous node then all direct generalization nodes in the higher level are tagged as a k-anonymous node and set level L as a k-anonymous level. If the node is not a k-anonymous node then all direct generalization nodes in the lower level are tagged as a non-anonymous node. The algorithm will continue to perform the Extended-OIGH with the new input L until all nodes at and below the k-anonymous level are tagged.

Since the optimal node is always at the lowest level found k-anonymous nodes, the algorithm will evaluate the nodes only at and below the k-anonymous level. Therefore, it could leave some nodes above the k-anonymous level unvisited.

The TraverseRoute algorithm which performs an in-order traversal since it search the boundary of the generalized lattice between the higher and lower levels alternatively. Thus the number of unvisited nodes in the lattice can be higher than the pre-order or post-order traversals which could waste the execution to traverse up or down until the anonymous node is firstly found.

3.2 Example of extended-OIGH algorithm

We present an example to illustrate our proposed work. Assume that, we want to release an optimal 2-anonymous *IGH* dataset. From the lattice of generalization shown in Figure 5, the shaded nodes with the bold outline are the k-anonymous nodes. The Extended-OIGH algorithm starts evaluating the node $<000>$. Then, with the in-order traversal manners, the node $<222>$, $<221>$, and $<220>$ are handled. The node $<222>$ and $<221>$ are the k-anonymous node, so these two nodes are tagged. The k-anonymous level is now set at 5. The node $<220>$ is a non-anonymous node, then all direct generalization nodes, $<210>$, $<200>$, $<110>$, $<200>$, $<100>$, and $<010>$, are all tagged as a non-anonymous node.

Algorithm 2 *TraversalRoute*.

```
Input: Lattice, Level L
Output: Route R
begin
    if hasRoute then
        R ← a route from root node to node at level L with in-order traversal
        Remove route R from Lattice
        Return R
    else
        Return False
    end
end
```

In the next iteration, the algorithm will traverse through the lattice from the root node to the node at the k-anonymous level at 5. The algorithm continues to check the nodes until all nodes at and below the k-anonymous level are discovered. Thus, the lowest k-anonymous level is 4. There are two k-anonymous nodes in level 4, $<211>$, and $<022>$. For obtaining

only an optimal answer, the C_{avg} is used among the node <211> and <022>. Thus, the node with the minimum C_{avg}, <022>, is returned as the optimal k-anonymity node.

4 Experimental results

4.1 Dataset and configuration

In this section, we evaluate our proposed algorithm, Extended-OIGH, with existing algorithms, i.e. the Optimal-IGH [7], Flash [3] and the depth-first search algorithm on the Uber ride rating [8], Jester [2] and T-drive [14] datasets. All algorithms are implemented based on Java SE 8. The number of visited nodes is reported as the efficiency indicator, the result reported in each configuration is three-times average to obtain stable results.

4.2 Results

As the algorithms visit the nodes in the generalization lattice to obtain an optimal answer, the number of visited nodes can be used to determine the efficiency of the algorithms. The less visiting number of nodes means we can reduce the evaluating time for obtaining the optimal solution. Furthermore, as the optimal node is the node in the lowest level found k-anonymous nodes in the generalization lattice, the faster finding a k-anonymous node the faster obtaining the optimal answer. Our algorithm traverses through the lattice with an in-order traversal manner, left-current-right node order. The algorithm jumps to evaluate the node at an upper level and the current level, increasing the opportunity to find the k-anonymous node. Therefore, the Optimal-IGH algorithm searches through the lattice with post-order (left-right-current node order) and pre-order (current-left-right node order) manner which determines the node in the same level first. The Optimal-IGH algorithm has the

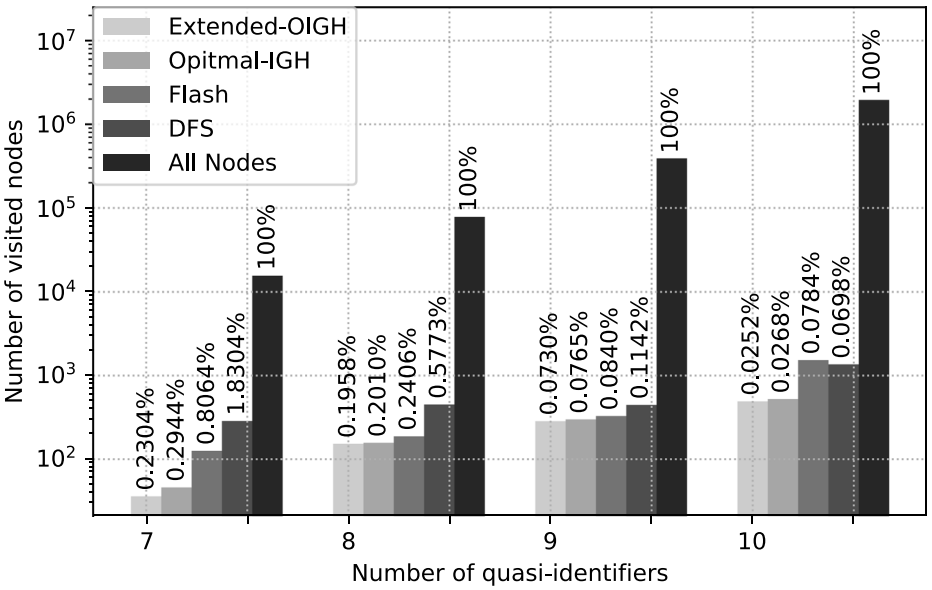


Figure 4 Number of visited nodes per number of quasi-identifiers of T-drive dataset

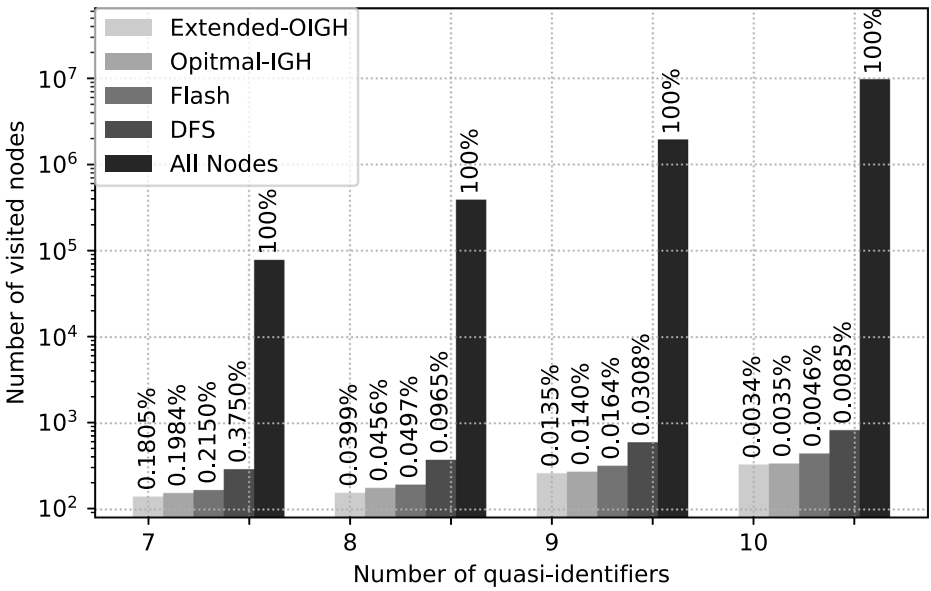


Figure 5 Number of visited nodes per number of quasi-identifiers of the Jester dataset

lower opportunity to obtain a k-anonymous node. Furthermore, our proposed, Extended-OIGH, algorithm leaves some nodes unvisited due to search only in and lower level found k-anonymous node. In contrast, the other existing algorithms, Flash and depth-first search (DFS), need to search and visit all nodes in the generalization lattice to obtain the optimal answer. Thus, our algorithm could find the optimal solution faster than the other algorithms.

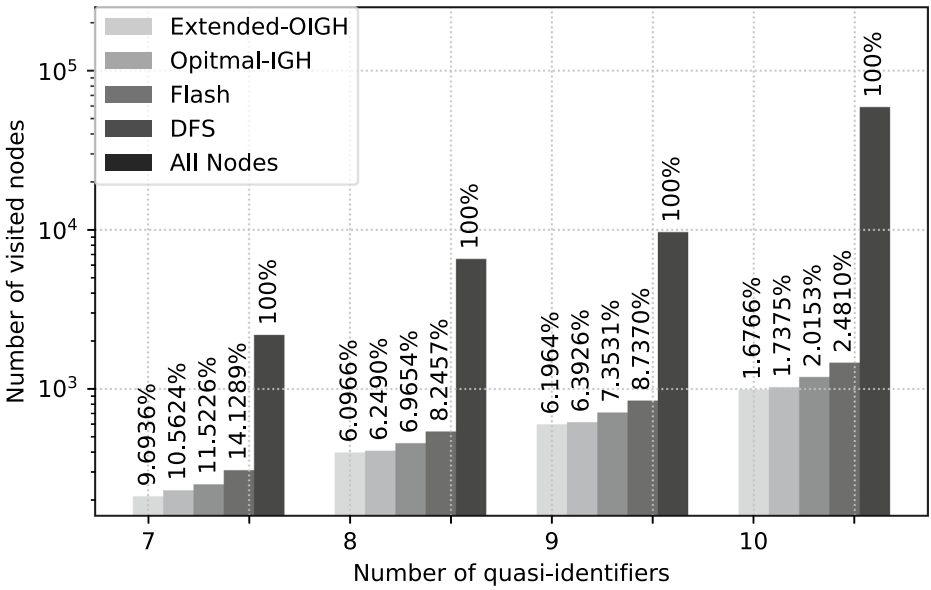


Figure 6 Number of visited nodes per number of quasi-identifiers of Uber dataset

The result is shown in Figures 4, 5, and 6. For all datasets, we vary the number of quasi-identifiers from 7 to 10 which presented in the horizontal axis. While the vertical axis measures the number of visited nodes in the logarithm scales. From the results, it is clear that our algorithm visits fewer nodes than the other algorithms in the various number of quasi-identifiers by visiting only 0.23% of all nodes, for the T-drive dataset, while the fastest existing algorithm, Optimal-IGH, visits 0.29% followed by Flash and DFS at the quasi-identifier at 7. This result means that our proposed algorithm can be faster than the Optimal-IGH by almost 21% When the number of quasi-identifiers is increased, the number of visited nodes of all algorithms increases, since there will be more nodes in the hierarchies to be searched. However, our proposed algorithm is still the most efficient. For the Jester and Uber dataset, the results show a similar trend as the T-drive dataset, our algorithm could also find the optimal solution by visit about 0.01% fewer number of nodes compared with the Optimal-IGH algorithm for the Jester dataset and 0.9% for the Uber dataset.

5 Conclusion

This paper proposed a heuristic algorithm for preserving the data privacy for the Identical Generalization Hierarchy (*IGH*) data. The proposed work is based on k-anonymity method. The solutions generated from our proposed work is optimal by traversing the generalization lattice to find the minimal information loss. The algorithm employs the benefit of from the characteristics of *IGH* data, the optimal solution is always in the lowest level found k-anonymous nodes of the generalization lattice. The algorithm could leave some nodes unvisited which could make our algorithm find the optimal solution faster than the other algorithms. Additionally, an in-order traversal is applied in the proposed work since it searches the boundary of the generalized lattice between the higher and lower levels alternatively. Thus, the opportunity to find the k-anonymous nodes is higher than the existing work with other traversal. From the experiment results on the benchmark datasets, it is found that the algorithm is the most efficient algorithm compared with the other well-known algorithms in every benchmark dataset. From the experiment, the marginal difference between the efficiency of our proposed algorithm and the most efficient existing algorithm could be as high as most 21%.

In our future work, we will investigate the situation where the quasi-identifier of IGH data can be updated. In which, the solution can be applied in a more practical situation where the data can be changed all the time. Moreover, the algorithm applies the global recoding generalization which causes more information loss due to all values in the same quasi-identifier need to be generalized to the same level in the generalization hierarchy. Thus, in future work, the k-anonymity algorithm on the local recoding will be investigated. The local recoding loses lesser information than the global recoding due to it only generalizes some necessary cells in the dataset.

Acknowledgements This research work was partially supported by Chiang Mai University.

References

1. El Emam, K., Dankar, F., Issa, R., Jonker, E., Amyot, D., Cogo, E., Corriveau, J.P., Walker, M., Chowdhury, S., Vaillancourt, R., Roffey, T., Bottomley, J.: A globally optimal k-anonymity method for the de-identification of health data. J. Am. Med. Inform. Assoc. JAMIA **16**, 670–82 (2009)
2. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Inf. Retr. **4**(2), 133–151 (2001)
3. Kohlmayer, F., Prasser, F., Eckert, C., Kemper, A., Kuhn, K.A.: Flash: Efficient, stable and optimal k-anonymity. In: 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing, pp. 708–717 (2012). https://doi.org/10.1109/SocialCom-PASSAT.2012.52
4. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD '05, pp. 49–60. ACM, New York, NY, USA (2005)
5. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: 22nd International Conference on Data Engineering (ICDE'06), pp. 25–25 (2006)
6. Mahanan, W., Natwichai, J., Art Chaovalitwongse, W.: Characterizations of local recoding method on k-anonymity. In: Barolli, L., Kryvinska, N., Enokido, T., Takizawa, M. (eds.) Advances in Network-Based Information Systems, pp. 648–658. Springer International Publishing, Cham (2019)
7. Mahanan, W., Art Chaovalitwongse, W., Natwichai J: Data anonymization: A novel optimal *k*-anonymity algorithm for identical generalization hierarchy data in iot. Service Oriented Computing and Applications. https://doi.org/10.1007/s11761-020-00287-w (2020)
8. Purvank: Uber ride reviews dataset. https://www.kaggle.com/purvank/uber-rider-reviews-dataset. Accessed 10 Nov 2020 (2017)
9. Samarati, P.: Protecting respondents identities in microdata release. IEEE Trans. Knowl. Data Eng. **13**(6), 1010–1027 (2001)
10. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information. Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems pp. 98 (1998)
11. Sweeney, L.A.: Computational disclosure control: A primer on data privacy protection. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, aAI0803469 (2001)
12. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. Int. J. Uncertain Fuzziness Knowl-Based Syst. **10**(5), 571–588 (2002)
13. Sweeney, L.: k -anonymity: A model for protecting privacy. **10**(5),1–14 (2002)
14. Zheng, Y.: T-drive trajectory data sample. https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/, t-Drive sample dataset (2011)