# Proximity-Aware Local-Recoding Anonymization with MapReduce for Scalable Big Data Privacy Preservation in Cloud

Xuyun Zhang, Wanchun Dou, Jian Pei, *Fellow, IEEE*, Surya Nepal, *Member, IEEE*, Chi Yang, Chang Liu, and Jinjun Chen, *Senior Member, IEEE*

**Abstract**—Cloud computing provides promising scalable IT infrastructure to support various processing of a variety of big data applications in sectors such as healthcare and business. Data sets like electronic health records in such applications often contain privacy-sensitive information, which brings about privacy concerns potentially if the information is released or shared to third-parties in cloud. A practical and widely-adopted technique for data privacy preservation is to anonymize data via generalization to satisfy a given privacy model. However, most existing privacy preserving approaches tailored to small-scale data sets often fall short when encountering big data, due to their insufficiency or poor scalability. In this paper, we investigate the local-recoding problem for big data anonymization against proximity privacy breaches and attempt to identify a scalable solution to this problem. Specifically, we present a proximity privacy model with allowing semantic proximity of sensitive values and multiple sensitive attributes, and model the problem of local recoding as a proximity-aware clustering problem. A scalable two-phase clustering approach consisting of a *t*-ancestors clustering (similar to *k*-means) algorithm and a proximity-aware agglomerative clustering algorithm is proposed to address the above problem. We design the algorithms with MapReduce to gain high scalability by performing data-parallel computation in cloud. Extensive experiments on real-life data sets demonstrate that our approach significantly improves the capability of defending the proximity privacy breaches, the scalability and the time-efficiency of local-recoding anonymization over existing approaches.

**Index Terms**— Big Data; Cloud Computing; MapReduce; Data Anonymization; Proximity Privacy

## 1 INTRODUCTION

CLOUD computing and big data, two disruptive trends at present, pose a significant impact on current IT industry and research communities [1, 2]. Today, a large number of big data applications and services have been deployed or migrated into cloud for data mining, processing or sharing. The salient characteristics of cloud computing such as high scalability and pay-as-you-go fashion make big data cheaply and easily accessible to various organizations through public cloud infrastructure. Data sets in many big data applications often contain personal privacy-sensitive data like electronic health records and financial transaction records. As the analysis of these data sets provides profound insights into a number of key areas of society (e.g., healthcare, medical, government services, e-research), the data sets are often shared or released to third party partners or the public. The privacy-sensitive information can be divulged with less effort by an adversary as the coupling of big data with public cloud environments disables some traditional privacy protection measures in cloud [3, 4]. This can bring considerable economic loss or severe social reputation impairment to data owners. As such, sharing or releasing privacy-sensitive data sets to third-parties in cloud will bring about potential privacy concerns, and therefore requires strong privacy preservation.

Data anonymization has been extensively studied and widely adopted for data privacy preservation in non-interactive data sharing and releasing scenarios [5]. Data anonymization refers to hiding identity and/or sensitive data so that the privacy of an individual is effectively preserved while certain aggregate information can be still exposed to data users for diverse analysis and mining tasks. A variety of privacy models and data anonymization approaches have been proposed and extensively studied recently [5, 6, 7, 8, 9, 10, 11, 12]. However, applying these traditional approaches to big data anonymization poses scalability and efficiency challenges because of the "3Vs", i.e., Volume, Velocity and Variety. The research on scalability issues of big data anonymization has come to the picture [10, 13, 14, 15], but they are only applicable to the sub-tree or multidimensional scheme. Following this line, we investigate the local-recoding scheme herein and attempt to identify a scalable solution to big data local-recoding anonymization. Recently, differential privacy has attracted plenty of attention due to its robust privacy

- *X. Zhang, C. Yang, C. Liu and J. Chen are with the Faculty of Engineering and IT, University of Technology, Sydney, Australia, NSW2007. E-mail: {xyzhanggz, chiyangit, changliu.it, jinjun}@gmail.com.*
- *W. Dou is with the State Key Laboratory for Novel Software Technology, the Dept. of Computer Science and Technology, Nanjing University, China, 210023. E-mail: douwc@nju.edu.cn.*
- *J. Pei is with the School of Computing Science, Simon Fraser University, Canada. E-mail: jpei@cs.sfu.ca.*
- *S. Nepal is with CSIRO Computational Informatics, CSIRO, Australia, NSW2122. E-mail: Surya.Nepal@csiro.au.*

*Manuscript received (insert date of submission if desired). Please note that all acknowledgments should be placed at the end of the paper, before the bibliography.*

guarantee regardless of an adversary's prior knowledge [16]. However, besides the drawbacks pointed in [16], differential privacy also loses correctness guarantees because it produces noisy results to hide the impact of any single individual [17]. Hence, syntactic anonymity privacy models still have practical impacts in general data publishing and can be applied in numerous real-world applications.

The local-recoding scheme, also known as cell generalization, groups data sets into a set of cells at the data record level and anonymizes each cell individually. Existing approaches for local recoding [9, 11, 18, 19] can only withstand record linkage attacks by employing $k$-anonymity privacy model [6], thereby falling short of defending proximity privacy breaches [12, 20, 21]. In fact, combining local recoding and proximity privacy models together is interesting and necessary when one wants an anonymous data set with both low data distortion and the ability to combat proximity privacy attacks. However, this combination is a challenge because most proximity privacy models have the property of non-monotonicity [21] and local recoding fails to be accomplished in a top-down way, which will be detailed in Subsection 3.3 (Motivation and Problem Analysis).

In this paper, we model the problem of big data local recoding against proximity privacy breaches as a proximity-aware clustering problem, and propose a scalable two-phase clustering approach accordingly. Specifically, we put forth a proximity privacy model based on [12] by reasonably allowing multiple sensitive attributes and semantic proximity of categorical sensitive values. As the satisfiability problem of the proximity privacy model is proved to be NP-hard, it is interesting and practical to model the problem as a clustering problem of minimizing both data distortion and proximity among sensitive values in a cluster, rather than to find a solution satisfying the privacy model rigorously. Technically, a proximity-aware distance is introduced over both quasi-identifier and sensitive attributes to facilitate clustering algorithms. To address the scalability problem, we propose a two-phase clustering approach consisting of the $t$-ancestors clustering (similar to $k$-means [22]) and proximity-aware agglomerative clustering algorithms. The first phase splits an original data set into $t$ partitions that contain similar data records in terms of quasi-identifiers. In the second phase, data partitions are locally recoded by the proximity-aware agglomerative clustering algorithm in parallel. We design the algorithms with MapReduce in order to gain high scalability by performing data-parallel computation over multiple computing nodes in cloud. We evaluate our approach by conducting extensive experiments on real-world data sets. Experimental results demonstrate that our approach can preserve the proximity privacy substantially, and can significantly improve the scalability and the time-efficiency of local-recoding anonymization over existing approaches.

The major contributions of our research are fourfold. Firstly, an extended proximity privacy model is put forth via allowing multiple sensitive attributes and semantic proximity of categorical sensitive values. Secondly, we model the problem of big data local recoding against proximity privacy breaches as a proximity-aware clustering problem. Thirdly, a scalable and efficient two-phase clustering approach is proposed to parallelize local recoding on multiple data partitions. Fourthly, several innovative MapReduce jobs are designed and coordinated to concretely conduct data-parallel computation for scalability.

The remainder of this paper is organized as follows. The next section reviews related work. In Section 3, we briefly introduce some preliminary and analyze the problems in detail. Section 4 models the proximity-aware clustering problem formally, and Section 5 elaborates the two-phase clustering approach and the MapReduce jobs. We empirically evaluate our approach in Section 6. Finally, we conclude this paper and discuss future work in Section 7.

## 2 Related Work

Recently, data privacy preservation has been extensively investigated [5]. We briefly review existing approaches for local-recoding anonymization and privacy models to defense against attribute linkage attacks. In addition, research on scalability issues in existing anonymization approaches is shortly surveyed.

Recently, clustering techniques have been leveraged to achieve local-recoding anonymization for privacy preservation. Xu et al. [9] studied on the anonymization of local recoding scheme from the utility perspective and put forth a bottom-up greedy approach and the top-down counterpart. The former leverages the agglomerative clustering technique while the latter employs the divisive hierarchical clustering technique, both of which pose constraints on the size of a cluster. Byun et al. [19] formally modeled local-recoding anonymization as the $k$-member clustering problem which requires the cluster size should not be less than $k$ in order to achieve $k$-anonymity, and proposed a simple greedy algorithm to address the problem. Li et al. [18] investigated the inconsistency issue of local-recoding anonymization in data with hierarchical attributes and proposed KACA (K-Anonymization by Clustering in Attribute hierarchies) algorithms. Aggarwal et al. [11] proposed a set of constant factor approximation algorithms for two clustering based anonymization problems, i.e., $r$-GATHER and *r-CELLULAR CLUSTERING*, where cluster centers are published without generalization or suppression. However, existing clustering approaches for local-recoding anonymization mainly concentrate on record linkage attacks, specifically under the $k$-anonymity privacy model, without paying any attention to privacy breaches incurred by sensitive attribute linkage. On the contrary, our research takes both privacy concerns into account. Wong et al. [23] proposed a top-down partitioning approach based on the Mondrian algorithm in [24] to specialize data sets to achieve $(\alpha, k)$-anonymity which is able to defend certain attribute linkage attacks. However, the data utility of the resultant anonymous data is heavily influenced by the choice of splitting attributes and values, while local recoding does not involve such factors. Our approach leverages clustering to accomplish local recoding because it is a natural and effective way to anonymize data sets at a cell level.

To preserve privacy against attribute linkage attacks, a variety of privacy models have been proposed for both categorical and numerical sensitive attributes. $l$-diversity and its variants [7] require each QI-group to include at least $l$ well-represented sensitive values. Note that $l$-diverse data sets are already $l$-anonymous. Some privacy models, such as $(\alpha, k)$-anonymity [23], extend $k$-anonymity with the confidence bounding principle that requires the confidence of associating a quasi-identifier to a sensitive value to be less than a user-specified threshold. In general, these integrated models are more flexible than $l$-diversity. Since the models above handle categorical attributes only, they fail to thwart proximity privacy breach in numerical sensitive attributes. As a result, several privacy models such as $(k, e)$-anonymity [20], variance control [10], $(\varepsilon, m)$-anonymity [21] and $t$-closeness [8] are put forth. $(k, e)$-anonymity requires that the difference of the maximum and minimum sensitive values in a QI-group must be at least $e$, while the variance control principle demands that the variance of the sensitive values must be not less than a threshold. $(\varepsilon, m)$-anonymity, a stronger model, requires that for any sensitive value in a QI-group, at most $1/m$ of records can have sensitive values similar to the value, where $\varepsilon$ determines the similarity. A stringent privacy model $t$-closeness [8], which mainly combats data distribution skewness attacks by requiring the distribution of sensitive values in any QI-group should be close to the distribution of the entire data set, incorporates semantics through the kernel smoothing technique to mitigate the proximity breaches to a certain extent. Moreover, $t$-closeness is applicable to both categorical and numerical attributes as it only demands a pre-defined distance matrix. But $t$-closeness is insufficient to protect against proximity attacks as pointed in [21]. To cope with both categorical and numerical attributes, Wang et al. [12] proposed a general proximity privacy model, namely, $(\epsilon, \delta)^k$-dissimilarity, where $\epsilon$ determines the similarity threshold, $\delta$ controls the least number of dissimilar sensitive values for any sensitive value in a QI-group, and $k$ means $k$-anonymity is integrated. The privacy model we proposed herein can be regarded as an extended form of $(\epsilon, \delta)^k$-dissimilarity. Nevertheless, our model differs from it in that multiple sensitive attributes are taken into account and categorical sensitive attributes have semantic proximity in terms of their taxonomy trees.

Scalability issues of anonymization over large-scale data sets have drawn the attention of research communities. LeFevre et al. [10] addressed the scalability problem of multidimensional anonymization scheme [24] via introducing scalable decision trees and sampling techniques. Iwuchukwu et al. [13] proposed an R-tree index-based approach by building a spatial index over data sets, achieving high efficiency. Fung et al. [25, 26] proposed a top-down specialization approach, improving the efficiency of sub-tree anonymization scheme by exploiting a data structure named Taxonomy Indexed PartitionS (TIPS). Our previous work [14, 15] addressed the scalability problem of the sub-tree scheme in big data scenarios via leveraging MapReduce paradigm. However, the approaches above aim at either multidimensional scheme or sub-tree scheme, both of which are global recoding, thereby failing to work out for the local-recoding scheme investigated herein.

## 3 Preliminaries and Problem Analysis

### 3.1 Local-Recoding Anonymization Scheme

To facilitate subsequent discussion, we briefly introduce the concept of local-recoding anonymization as background knowledge. Local recoding, also known as cell generalization, is one of the schemes outlined in [5]. Other schemes include full-domain, sub-tree and multidimensional anonymization. Local recoding generalizes a data set at the cell level, while global recoding generalizes them at the domain level. The last three schemes mentioned above are global recoding. Generally, local recoding minimizes the data distortion incurred by anonymization, and therefore produces better data utility than global recoding.

Table 1 lists some basic symbols and notations. Each record in $D$ consists of both quasi-identifier attributes and sensitive attributes. Quasi-identifier attributes are the ones that can be potentially linked to individuals uniquely if combined with external data sets, e.g., age and sex. If a sensitive value is associated to an identified individual, economic loss or reputation damage to the person probably occur. Thus, quasi-identifiers are usually anonymized to preserve privacy, while sensitive values are often kept in the original form for the sake of data mining or data analytics. The consequence of anonymization is that data are partitioned into a set of groups, and each is represented by an anonymous quasi-identifier. Such a group is named as QI-group, denoted by $QIG$ in Table 1. In this way, individual privacy is preserved while aggregate information is still available for data mining or analytics.

We consider both numerical and categorical attributes for local recoding herein, and assume that a taxonomy tree is given for a categorical attribute. To facilitate the discussion, it is assumed that the attributes are arranged in order, i.e., for quasi-identifier attributes, the scheme is $ATT_{QI} = \langle att_1^{QI}, \cdots, att_{l^{QI}}^{QI}, att_{l^{QI}+1}^{QI}, \cdots, att_{m^{QI}}^{QI} \rangle$, where the first $l^{QI}$ attributes are numerical while the rest are categor-

TABLE 1 BASIC SYMBOLS AND NOTIONS

| Symbol | Notations |
|---|---|
| $D$ | A data set containing $n$ data records. |
| $r$ | A data original record, $r \in D$ and $r = \langle v_1, \cdots, v_{m^{QI}}, sv_1, \cdots, sv_{m^S} \rangle$, where $v_i$, $1 \le i \le m^{QI}$, is a quasi-identifier attribute value, and $sv_j$, $1 \le j \le m^S$, is a sensitive attribute value, $m^{QI}$, $m^S$ are the number of the two types of attribute, respectively. |
| $TT_i$ | The taxonomy tree of categorical attribute $att_i$. |
| $DOM_i$ | The set of all domain values in $TT_i$ for categorical attribute $att_i$, or all domain intervals for numerical attribute $att_i$. |
| $V_i$ | The set of attribute values of $att_i^{QI}$. |
| $SV_i$ | The set of sensitive values of $att_i^S$. |
| $qid$ | A quasi-identifier, $qid = \langle q_1, \cdots, q_{m^{QI}} \rangle$, $q_i \in DOM_i$. |
| $QID$ | The set of quasi-identifiers, $QID = \langle DOM_1, \cdots, DOM_{m^{QI}} \rangle$. |
| $QIG$ | The quasi-identifier group containing all records with the same quasi-identifier. |

ical, and for sensitive attributes, the scheme is $ATT_S$ = $\langle att_1^S, \cdots, att_{l^S}^S, att_{l^S+1}^S, \cdots, at\,\square_{m^S}^{S}\rangle$, where the first $l^S$ attributes are numerical while the rest are categorical.

Based on the notions above, the local recoding scheme is formally described as follows. Data records in $D$ can be regarded as data points in a high dimensional space. Local-recoding scheme defines a set of functions on mostly overlapping multidimensional regions which cover $V_1 \times \cdots \times V_m$, where region overlapping means that multiple regions probably contain records with identical quasi-identifiers. Specifically, a function $\varphi_l: R_l \rightarrow QID$, is defined for a region $R_l$, where $l$ is an arbitrary number indexing the region. In fact, a region corresponds to a QI-group. Therefore, the core sub-problems of local recoding are how to construct multidimensional regions and how to choose functions $\{\varphi_l\}$. In our approach, we leverage the clustering technique to build multidimensional regions with keeping proximity privacy of sensitive attributes in mind. For each region, the categorical quasi-identifier attribute values are generalized to their lowest common domain value in the taxonomy tree, and the numerical ones are replaced by an interval that covers them minimally. Unlike local recoding, the sub-tree and full-domain schemes have one function over each attribute, i.e., $f_i: V_i \rightarrow DOM_i$, $1 \leq i \leq m$, and the global multidimensional scheme has a single function over all the attributes, i.e., $f: V_1 \times \cdots \times V_m \rightarrow QID$.

Preserving privacy is one side of anonymization. The other one is retaining aggregate information for data mining or analytics over the anonymous data. Several data metrics have been proposed to capture this [5], e.g., Minimal Distortion (MD) [6], *ILoss* [27] and Discernibility Metric (DM) [7]. With a data metric, the problem of optimal local recoding is to find the local-recoding solution that makes the metric optimal. However, theoretical analyses demonstrate that the problem under most not-trivial data utility metrics is NP-hard [5]. As a result, most existing approaches [9, 11, 18, 19] just try to find the minimal local recoding instead to achieve practical efficiency and a near optimal solution, where the minimal local recoding means that no more partitioning operations are allowed when building multidimensional regions under a certain privacy model. Our proximity-aware two-phase clustering approach herein also follows this line.

So far, only the $k$-anonymity privacy model has been employed to preserve privacy against record linkage attacks in existing clustering based anonymization approaches. The $k$-anonymity privacy model requires that for any $qid \in QID$, the size of $QIG(qid)$ must be zero or at least $k$, so that a quasi-identifier will not be distinguished from other at least $k-1$ ones in the same QI-group [6]. Usually, it is assumed that an adversary already has the knowledge that an individual is definitely in a data set, which occurs in many real-life data like tax data sets. After local recoding, the upper-bound size of a QI-group is $2k-1$ under the $k$-anonymity privacy model. If there were a QI-group of size at least $2k$, it should be split into two groups of size at least $k$ to maximize data utility.

### 3.2 MapReduce Basics

MapReduce [28], a parallel and distributed large-scale data processing paradigm, has been extensively researched and widely adopted for big data applications recently [29]. Integrated with infrastructure resources provisioned by cloud systems, MapReduce becomes much more powerful, elastic and cost-effective due to the salient characteristics of cloud computing. A typical example is the Amazon Elastic MapReduce service.

Basically, a MapReduce job consists of two primitive functions, Map and Reduce, defined over a data structure named key-value pair $(key, value)$. Specifically, the Map function can be formalized as $Map$: $(k_1, v_1) \rightarrow (k_2, v_2)$, i.e., Map takes a pair $(k_1, v_1)$ as input and then outputs another intermediate key-value pair $(k_2, v_2)$. These intermediate pairs are consumed by the Reduce function as input. Formally, the Reduce function can be represented as $Reduce$: $(k_2, list(v_2)) \rightarrow (k_3, v_3)$, i.e., Reduce takes intermediate $k_2$ and all its corresponding values $list(v_2))$ as input and outputs another pair $(k_3, v_3)$. Usually, $(k_3, v_3)$ list is the results which MapReduce users attempt to obtain. Both Map and Reduce functions are specified by users according to their specific applications. An instance running a Map function is called Mapper, and that running a Reduce function is called Reducer, respectively.

### 3.3 Motivation and Problem Analysis

In this section, we analyze the problems of existing approaches for local-recoding anonymization from the perspectives of proximity privacy and scalability. Further, challenges of designing scalable MapReduce algorithms for proximity-aware local recoding are also identified.

Little attention has been paid to the local-recoding anonymization scheme under proximity-aware privacy modes. As mentioned in Section 2, most existing local-recoding approaches concentrate on combating record linkage attacks by employing $k$-anonymity privacy model. As demonstrated in existing work [7, 8, 12, 21], however, $k$-anonymity fails to combat attribute linkage attacks like homogeneity attacks, skewness attacks and proximity attacks. For instance, if the sensitive values of the records in a QI-group of size $k$ are identical or quite similar, adversaries can still link an individual with certain sensitive values with high confidence although the QI-group satisfies $k$-anonymity, resulting in privacy violation. Accordingly, a plethora of privacy models have been proposed to thwart such attacks as shown in Section 2. But these models have been rarely exploited into the local-recoding scheme except the work in [23]. This phenomenon mainly results from two reasons analyzed as follows.

The first one is that, unlike global-recoding schemes, $k$-anonymity based approaches for record linkage attacks cannot be simply extended for attribute linkage attacks. Since global-recoding schemes partition data sets according to domains, they can be fulfilled effectively in a top-down fashion. This property of global-recoding schemes ensures that $k$-anonymity based approaches can be extended to combat attribute linkage attacks though checking extra privacy satisfiability during each round of the top-down anonymization process [5]. However, the local recoding scheme fails to share the same merits because it partitions data sets in a clustering fashion where the top-down anonymization property is inapplicable. Although Wong et al. [23] proposed a top-down approach for local recoding, the approach can only achieve partially local recoding because global recoding is exploited to partition data sets as the first step and local recoding is only conducted inside each partition. Consequently, their approach will incur more data distortion compared with the full potential of the local-recoding scheme.

The second reason is that most proximity aware privacy models have the property of non-monotonicity [21], which makes such models hard to achieve in a top-down way, even for global-recoding schemes. Formally, monotonicity refers to that if two disjoint data subsets $G_1$ and $G_2$ of a data set satisfy a privacy model, their union $G_1 \cup G_2$ satisfies the model as well. Monotonicity is a prerequisite for top-down anonymization approaches because it ensures to find minimally anonymized data sets. Specifically, if a data set does not satisfy a privacy model, we can infer that any of its subsets will fail to satisfy the model. Thus, when anonymizing data sets in a top-down fashion, we can terminate the process if further partitioning a subset violates the privacy model. However, most proximity-aware privacy models such as $(\varepsilon, m)$-anonymity and $(\epsilon, \delta)^k$-dissimilarity fail to possess the property of monotonicity. As a consequence, most existing anonymization approaches become inapplicable with such privacy models [21]. A two-step approach based on the Mondrian algorithm [24] is presented in [21] to obtain $(\varepsilon, m)$-anonymous data sets, via $k$-anonymizing a data set first, and adjusting partitions to achieve the privacy requirements then. However, this approach targets the multidimensional scheme, rather than local recoding investigated herein. Furthermore, proximity is not integrated into the search metric that guides data partitioning in the two-step approach, potentially incurring high data distortion. Wang and Liu [30] proposed an anonymization model XColor under the $(\epsilon, \delta)^k$-dissimilarity model, yet there is still a gap between its theoretical methodology and a practical algorithm as they acknowledged.

In terms of the analyses above, achieving the local-recoding scheme under proximity-aware privacy models is still a challenging problem. To our best knowledge, no previous work focuses on this problem. Motivated by this challenge, we propose a proximity-aware clustering approach for local-recoding anonymization.

Existing clustering approaches for anonymization are inherently sequential and assume that the data sets processed can fit into memory [9, 18, 19]. Unfortunately, the assumption often fails to hold in most big data applications in cloud nowadays. As a result, the approaches often suffer from the scalability problem when encountering big data applications. Even if a single machine with huge memory could be offered, the I/O cost of reading/writing very large data sets in a serial manner will be quite high. Thus, parallelism is not an option but by far the best choice for big data applications. Utilizing a bunch of small and cheap computation nodes rather a large expensive one is more cost-effective, which also coheres to the spirits of cloud computing where computation is provisioned in the form of various virtual machines.

We attempt to leverage MapReduce in cloud to address the scalability problem of clustering approaches for anonymization. However, designing proper MapReduce jobs for complex applications is still a challenge as MapReduce is a constrained programming paradigm. Usually, it is necessary to consider the problems like which part of an application can be parallelized by MapReduce, how to design Map and Reduce functions to make them scalable, and how to reduce network traffics among worker nodes. The answers to these questions often vary for different applications. Hence, extensive research is still required to design MapReduce jobs for a specific application.

## 4 Proximity-Aware Clustering Problem of Local-Recoding Anonymization

Due to the non-monotonicity property of proximity-aware privacy models and characteristics of local recoding, clustering is a natural and promising way to group both quasi-identifier attributes and sensitive attributes. Hence, we propose to model the problem of local-recoding anonymization under proximity-aware privacy models as a clustering problem in this section. Specifically, a proximity-aware privacy model is formulated in Section 4.1 and the clustering problem is formalized in Section 4.2.

### 4.1 Proximity-Aware Privacy Model

In big data scenarios, multiple sensitive attributes are often contained in data sets, while existing proximity-aware privacy models assume only one single sensitive attribute, either categorical or numerical. Hence, we assume multiple sensitive attributes in our privacy model, including both categorical and numerical attributes. As the discussion of proximity privacy attacks stems from numerical attributes, existing proximity-aware privacy models assume that categorical attribute values have no sense of semantic proximity [12, 21]. That is, categorical values are only examined whether they are exactly identical or different. Also, privacy models for categorical attributes only aims at avoiding exact reconstruction of sensitive values via limiting the number or distribution of sensitive values without considering semantic proximity [7, 8]. However, sensitive categorical values often have the sense of semantic proximity in real-life applications because the values are usually organized in a taxonomy tree in terms of domain specific knowledge. For instance, a taxonomy tree of diseases is presented in [27]. A similar one is depicted in Fig. 1 to facilitate our discussion.

Privacy breaches can still take place even if an anonymous data set already satisfies existing privacy models

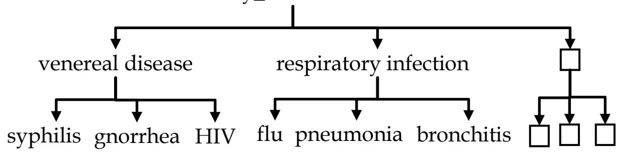

Fig. 1 A taxonomy tree of attribute *Disease*.

like $l$-diversity or $t$-closeness. For instance, an individual identified in a 3-diverse QI-group with sensitive values {syphilis, gnorrhea, HIV}, will be associated with "venereal disease" with 100% confidence based on the taxonomy tree in Fig. 1. This inference can lead to severe privacy breach. We call such an attack as "categorical proximity breach". The core of the breach is the semantic proximity among categorical values defined by the domain specific knowledge, which is ignored in previous privacy models.

With the notion of proximity of categorical sensitive values, we extend the proximity privacy model $(\epsilon,\delta)^k$-dissimilarity in [12] to multiple sensitive attributes including both categorical and numerical ones. Our privacy model is named as $(\epsilon^+,\delta)^k$-dissimilarity, where "+" implies proximity of categorical values is taken into account.

To capture the dissimilarity between sensitive values of two records, the distance metric should be defined first. Let $sv, sv' \in SV_i$ denote two sensitive values from two records, where the meaning of $SV_i$ is already described in Table 1. To establish the distance metric over all sensitive attributes, distance is normalized in subsequent definitions. For a numerical attribute, the normalized distance between $sv$ and $sv'$, $d_N(sv,sv')$, is defined as:

$$d_N(sv,sv') = |sv - sv'|/R, \tag{1}$$

where $R = (sv^{\max} - sv^{\min})$ denotes the domain of the attribute, $sv^{\max}$ and $sv^{\min}$ are the maximum and minimum values of the attribute, respectively. For a categorical attribute, the normalized distance between $sv$ and $sv'$, $d_C(sv,sv')$, is defined as:

$$d_C(sv,sv') = L(sv,sv')/(2\cdot H(TT_i)), \tag{2}$$

where $L(sv,sv')$ is the shortest path length between $sv$ and $sv'$, and $H(TT_i)$ is the height of the taxonomy tree. Note that $2\cdot H(TT_i)$ is the maximum path length between any two nodes in the tree. Our definition is more flexible than that in [19], as it is unnecessary to require that all the leaf nodes of the taxonomy tree should have the same depth, which is quite common in real-life applications. $L(sv,sv')$ can be computed with ease by finding the Lowest Common Ancestor (LCA) of $sv$ and $sv'$.

Let $r^S = \langle sv_1, \cdots, sv_{m^S}\rangle$ denote the vector of sensitive values of a record $r$. For convenience, $r^S$ is named as sensitive vector. With the definitions of single attributes, the distance between sensitive vector $r_x^S = \langle sv_1, \cdots, sv_{m^S}\rangle$ and $r_y^S = \langle sv_1', \cdots, sv_{m^S}'\rangle$, $d(r_x^S, r_y^S)$, is defined as:

$$d(r_x^S, r_y^S) = \sum_{i=1}^{l^S} w_i^S d_N + \sum_{i=l^S+1}^{m^S} w_i^S d_C, \tag{3}$$

Where $d_N$ and $d_C$ are short for $d_N(sv_i, sv_i')$ and $d_C(sv_i, sv_i')$, respectively, and $w_i^S$, $1 \le i \le m^S$, are weights for sensitive attributes. The weights, satisfying $0 \le w_i^S \le 1$, $\sum_{i=1}^{m^S} w_i^S = 1$, indicate the importance of each sensitive attributes and are usually specified by domain experts for flexibility.

Let $SV_{qid} = \{r_1^S, \cdots, r_n^S\}$ be the set of sensitive vectors in a QI-group $QIG(qid)$ of size $n$. Given a parameter $\epsilon^+ \ge 0$, the "$\epsilon^+$-neighborhood" of a sensitive vector $r^S$, denoted as $\Phi_{qid}(r^S,\epsilon^+)$, is defined as: $\Phi_{qid}(r^S,\epsilon^+) = \{r_x^S \mid r_x^S \in SV_{qid}, \text{and } d(r^S, r_x^S) \le \epsilon^+\}$. A sensitive vector is regarded to be similar to $r^S$ if it lies in $\Phi_{qid}(r^S,\epsilon^+)$. Thus, the parameter $\epsilon^+$ controls similarity between two sensitive vectors.

To preserve privacy against proximity attacks, it is expected that the sensitive vectors in a QI-group are dissimilar to each other as much as possible. Accordingly, we define the $(\epsilon^+,\delta)^k$-dissimilarity privacy model based on [12]. The model requires that for any QI-group $QIG(qid)$ in an anonymous data set, its size is at least $k$, and any sensitive vector in $SV_{qid}$ must be dissimilar to at least $\delta\cdot(|QIG(qid)|-1)$ other sensitive vectors in $QIG(qid)$, where $0 \le \delta \le 1$. Parameter $k$ controls the size of each QI-group to prevent record linkage attacks. Parameter $\delta$ specifies constraints on the number of $\epsilon^+$-neighbors that each sensitive vector can own to combat proximity attacks. As $(\epsilon^+,\delta)^k$-dissimilarity is extended from $(\epsilon,\delta)^k$-dissimilarity, it shares the effectiveness and most characteristics of the latter as described in [12]. But monotonicity of $(\epsilon,\delta)^k$-dissimilarity has not been discussed in [12]. We formally establish the non-monotonicity property of the two models by the following theorem.

**Theorem 1.** *Neither $(\epsilon,\delta)^k$-dissimilarity nor $(\epsilon^+,\delta)^k$-dissimilarity is monotonic.*

**Proof.** It suffices to prove this theorem by finding a counter-example for $(\epsilon,\delta)^k$-dissimilarity, where two QI-group $QIG_x$ and $QIG_y$ satisfy $(\epsilon,\delta)^k$-dissimilarity respectively, but their union $QIG_x \cup QIG_y$ does not. A counter-example is shown as follows. Assume a QI-group $QIG_x$ of size 3 has one-dimensional numerical sensitive values {1.0, 3.0, 5.0}, another QI-group $\mathrm{QIG}_y$ of size 3 has sensitive values {2.0, 4.0, 6.0}. Let $\epsilon$, $\delta$ and $k$ be 1.5, 1 and 3, respectively. Both QI-groups satisfy $(1.5, 1)^3$-dissimilarity. After merging the two QI-groups, the set of sensitive values is {1.0, 2.0, 3.0, 4.0, 5.0, 6.0}. For the value 3.0, only values {1.0, 5.0, 6.0} are dissimilar in the sense of $\epsilon = 1.5$. But the model requires $1\cdot(6-1) = 5$ values are dissimilar to 3.0. So, the theorem is proved in terms of this contradiction. □

## 4.2 Proximity-Aware Clustering (PAC) Problem of Local-Recoding Anonymization

Due to the non-monotonicity property, the satisfiability problem of $(\epsilon^+,\delta)^k$-dissimilarity is hard. The satisfiability problem is whether there exists a partition that makes the anonymous data set satisfy $(\epsilon^+,\delta)^k$-dissimilarity. Based on the results in [12], we have the following theorem.

**Theorem 2.** *In general, the $(\epsilon^+,\delta)^k$-dissimilarity satisfiability problem is NP-hard.*

**Proof.** Since the $(\epsilon,\delta)^k$-dissimilarity satisfiability problem is proved to be NP-hard in [12], the conclusion holds as well for the $(\epsilon^+,\delta)^k$-dissimilarity satisfiability problem. The reason is that $(\epsilon,\delta)^k$-dissimilarity can be regarded as $(\epsilon^+,\delta)^k$-dissimilarity in a specific setting where categorical proximity is ignored, and a single sensitive attribute rather than multiple ones is considered. □

In terms of the complexity result in Theorem 2, it is interesting and practical to find an efficient and near-optimal solution that can make the proximity among records in a QI-group as low as possible. Due to the non-monotonicity of $(\epsilon^+,\delta)^k$-dissimilarity, top-down partitioning at domain levels fails to work for this privacy model. But clustering records with low proximity of sensitive values is still a promising way to make the proximity as low as possible. As clustering is also a natural and effective way for the local-recoding scheme, we propose a novel clustering approach for local recoding by integrating proximity among sensitive values. Specifically, our approach attempts to minimize both data distortion and proximity among sensitive values in a QI-group when conducting clustering, unlike all the existing $k$-member clustering approaches that consider the former only. The clustering problem we attempt to address is referred to as Proximity-Aware Clustering (PAC) problem, which is essentially a two-objective clustering problem.

As minimizing proximity is one objective of the PAC problem, we define proximity index formally to capture the proximity between two records. According to (1), the proximity index between two numerical sensitive values $sv$ and $sv'$ can be defined as $d_{N^-}(sv,sv') = 1 - d_N(sv,sv')$. Likely, the index between two categorical values can be defined as $d_{C^-}(sv,sv') = 1 - d_C(sv,sv')$. Then, the proximity index between two sensitive vectors $r_x^S$ and $r_y^S$, denoted by $prox(r_x^S, r_y^S)$, is defined as:

$$prox(r_x^S, r_y^S) = \sum_{i=1}^{l^S} w_i^S d_{N^-} + \sum_{i=l^S+1}^{m^S} w_i^S d_{C^-}, \quad (4)$$

where $d_{N^-}$ and $d_{C^-}$ are short for $d_{N^-}(sv,sv')$ and $d_{C^-}(sv,sv')$, respectively. With the notion of proximity index between two sensitive vectors, we define the proximity index of a QI-group $QIG(qid)$ by:

$$prox(QIG(qid)) = \max_{r_x \neq r_y \in QID(qid)} prox(r_x^S, r_y^S). \quad (5)$$

Here we use the maximizing function rather than the sum function as the former can capture the risk of proximity breach for a QI-group according to risk analyses in [12].

Similar to existing approaches, the other objective of the PAC problem is to minimizing data distortion caused by generalization operations. We define the notion of information loss to measure the amount of data distortion. Recall that the quasi-identifiers of all records in a final cluster will be generalized to a more general one. Specifically, all quasi-identifier values of a categorical attribute are generalized to their lowest common ancestor in the taxonomy tree, and numerical values are generalized to a minimum interval covering all the values. Let $R_i^{QI}$ be the domain of attribute $att_i^{QI}$. The information loss of a record $r \in QIG(qid)$, denoted as $IL(r)$, is defined by:

$$IL(r) = \sum_{i=1}^{l^{QI}} w_i^{QI} \frac{v_i^{\max} - v_i^{\min}}{R_i^{QI}} + \sum_{i=l^{QI}}^{m^{QI}} w_i^{QI} \frac{L(v, v^{\mathrm{LAC}})}{H(TT_i)}, \quad (6)$$

where $v_i^{\max}$ and $v_i^{\min}$ are the maximum and minimum values of the attribute within $QIG(qid)$ respectively, and $L(v, v^{\mathrm{LAC}})$ is the length of the path from $v$ to their LCA $v^{\mathrm{LAC}}$. Weights $w_i^{QI}$, $1 \le i \le m^{QI}$, are exploited to indicate the importance of each attribute, and satisfy $0 \le w_i^{QI} \le 1$ and $\sum_{i=1}^{m^{QI}} w_i^{QI} = 1$. These weights above can bring flexibility of the definition of information loss for different applications. Usually, these weights are specified by domain-specific experts. Further, we define the information loss of $QIG(qid)$ is defined by:

$$IL(QIG(qid)) = \sum_{r \in QIG(qid)} IL(r). \quad (7)$$

Similar to the $k$-member clustering problem defined in [19], the PAC problem is formally defined as follows.

**Definition 1** (*Proximity-Aware Clustering (PAC) Problem*) Given a data set $D$, the Proximity-Aware Clustering (PAC) problem is to find a set of clusters, denoted as $\mathbb{C} = \{C_1, \cdots, C_m\}$, such that each cluster contains at least $k$ ($k > 0$) data records, the proximity of sensitive values in each cluster and the overall data distortion are minimized. Formally, the PAC problem is formulated as:

$$\text{Minimize} \begin{cases} \max_{C \in \mathbb{C}} Prox(C), \\ \sum_{C \in \mathbb{C}} IL(C), \end{cases} \text{s.t.:}$$

1). $\cup_{i=1,\cdots,m} C_i = D$, and $C_i \cap C_j = \emptyset$, $1 \le i \ne j \le m$,
2). $\forall C \in \mathbb{C}$, $|C| \ge k$.

The constraint 2) makes clustering approaches for anonymization different from traditional ones that usually have constraints on the number of clusters rather than the size of a single cluster. Moreover, the PAC problem differs from the $k$-member clustering problem in essence, as the former has one more objective that minimizes $\max_{C \in \mathbb{C}} Prox(C)$ in terms of Def. 1. To explore the computational complexity of the PAC problem, the proximity-aware clustering decision problem is defined as follows.

**Definition 2** (*Proximity-Aware Clustering (PAC) Decision Problem*) Given a data set $D$, the decision problem is to find a set of clusters $\mathbb{C} = \{C_1, \cdots, C_m\}$, where $\cup_{i=1,\cdots,m} C_i = D$ and $C_i \cap C_j = \emptyset$, $1 \le i \ne j \le m$, subject to:
1). $\forall C \in \mathbb{C}$, $|C| \ge k$;
2). $\max_{C \in \mathbb{C}} Prox(C) \le c_1$, $c_1 > 0$;
3). $\sum_{C \in \mathbb{C}} IL(C) \le c_2$, $c_2 > 0$.

**Theorem 3.** *The PAC decision problem is NP-hard.*

**Proof.** It suffices to prove that this problem for a specific setting is NP-hard. We observe that the $k$-member clustering decision problem articulated in [19] is a special case of the PAC decision problem by setting $c_1 = +\infty$, i.e., no constraints are posed on sensitive attributes. Byun et al. [19] proved that the $k$-member clustering decision problem which is NP-hard. It therefore follows that the PAC decision problem is NP-hard as well. □

In terms of Theorem 3, the PAC clustering problem is also NP-hard. To address this problem, we transform it to a single-objective clustering problem. To this aim, a distance measure between two records should be defined with considering the proximity between sensitive values. Unlike distance metrics employed in existing $k$-member clustering approaches, ours is required to consider both the similarity of quasi-identifiers and the proximity of sensitive values. Intuitively, we desire that records tend to go into the same clusters if their quasi-identifiers are similar, and the proximity of sensitive values among them is low, i.e., the sensitive values are dissimilar. Thus, it is reasonable to integrate proximity into the distance measure, and hence the distance measure herein is defined over both quasi-identifier and sensitive attributes.
As to quasi-identifiers, the distance metric can be utilized directly to capture similarity among them. Similar to (1) and (2), the normalized distance between two numerical quasi-identifier values $v$ and $v'$ is defined as $d_N(v,v') = |v - v'|/R$, and the distance between two categorical quasi-identifier values is defined as $d_C(v,v') = L(v,v')/(2 \cdot H(TT_i))$. Then, the distance between two quasi-identifiers $r_x^{QI}$ and $r_y^{QI}$ is defined by:

$$d(r_x^{QI}, r_y^{QI}) = \sum_{i=1}^{l^{QI}} w_i^{QI} d_N + \sum_{i=l^{QI}}^{m^{QI}} w_i^{QI} d_C, \quad (8)$$

where $d_N$ and $d_C$ are short for $d_N(v,v')$ and $d_C(v,v')$, respectively. Weights $w_i^{QI}$, $1 \le i \le m^{QI}$, are the same as (6).

Combining the distance function (8) and the proximity index function (4), a proximity-aware distance measure between two data records $r_x = \langle r_x^{QI},\ r_x^S \rangle$ and $r_y = \langle r_y^{QI},\ r_y^S \rangle$, denoted as $dist(r_x, r_y)$, is defined by:

$$dist(r_x, r_y) = \omega^{QI} d(r_x^{QI}, r_y^{QI}) + \omega^s prox(r_x^S, r_y^S), \quad (9)$$

where the two weights, $0 \le \omega^{QI},\ \omega^s \le 1$ and $\omega^{QI} + \omega^s = 1$, control how much quasi-identifier attributes or sensitive attributes contribute to the distance measure. Note that if $\omega^{QI} = 1$ and $\omega^s = 0$, the distance measure is the same as that in $k$-member clustering approaches. However, the distance measure $dist(\cdot,\cdot)$ is not a distance metric if $\omega^s \neq 0$ because it fails to satisfy coincidence axiom. The coincidence axiom, one of conditions that a distance metric must satisfy, requires $dist(r_x, r_y) = 0$ if and only if $r_x = r_y$. However, $dist(\cdot,\cdot)$ can still capture the degree of similarity records and guide them into proper clusters when we conduct clustering. In fact, $dist(r_x, r_y) = 0$ holds if and only if the quasi-identifiers are the same and the proximity between sensitive values is the lowest, while $dist(r_x, r_y)$ gets the maximum value if and only if the distance of quasi-identifiers reaches the maximum and the sensitive values are the same. So, the distance measure $dist(r_x, r_y)$ possesses the desired property that guides records with similar quasi-identifier but dissimilar sensitive values into the same clusters. Armed with the distance measure, we model the Single-objective Proximity-Aware Clustering (SPAC) problem as follows.

**Definition 3** (*Single-objective Proximity-Aware Clustering (SPAC) Problem*) Given a data set $D$, the Single-objective Proximity-Aware Clustering (SPAC) problem is to find a set of clusters of size at least $k$, denoted as $\mathbb{C} = \{C_1, \cdots, C_m\}$, such that the sum of all intra-cluster distances is minimized. Formally , the SPAC problem is formulated as:

Minimize $\sum_{l=1,\cdots,m} |C_l| \cdot \max_{r_x, r_y \in C_l} dist(r_x, r_y)$, s.t.:

1). $\bigcup_{i=1,\cdots,m} C_i = D$, and $C_i \cap C_j = \emptyset$, $1 \le i \neq j \le m$,

2). $\forall C \in \mathbb{C}, |C| \ge k$.

In fact, $\max_{r_x, r_y \in C_l} dist(r_x, r_y)$ in the objective function is the diameter of the cluster $C_l$. Intuitively, smaller diameter tends to incur lower data distortion, because it implies that the least common ancestors for categorical attributes lie in lower levels and the minimum covering intervals for numerical attributes are shorter. It also tends to produce lower proximity of sensitive values. The factor $|C_l|$ indicates that smaller clusters are preferred, because less data distortion will be incurred with smaller clusters.

**Theorem 4.** *The SPAC problem is NP-hard.*

**Proof.** It suffices to prove that this problem for a specific setting is NP-hard. The conventional $k$-member clustering problem [19] is a special case of the SPAC problem by setting $\omega^{QI} = 1$ and $\omega^s = 0$, i.e., the distance is determined by quasi-identifier attributes only, the same as $k$-member clustering problem. The $k$-member clustering problem is NP-hard as its decision problem is NP-hard [19]. It thus follows that the SPAC problem is NP-hard as well. □

Given the hardness of the SPAC problem, it is practical and interesting to time-efficiently find a near-optimal solution in big data scenarios, rather than to seek the optimal one. The next section will show how to achieve this.

## 5 Two-Phase Proximity-Aware Clustering using MapReduce

Except where otherwise noted, the proximity-aware clustering problem refers to the Single-objective Proximity-Aware Clustering (SPAC) problem hereafter. To address the SPAC problem in big data scenarios, we propose a two-phase clustering approach where agglomerative clustering method and point-assignment clustering method are employed in the two phases, respectively. We outline the sketch of the two-phase clustering approach in Subsection 5.1. Then, the algorithmic details of the two phases are elaborated in Subsection 5.2 and 5.3, respectively. We illustrate the execution process of our approach and analyze the performance in Subsection 5.4.

### 5.1 Sketch of Two-Phase Clustering

In order to choose proper clustering methods for the SPAC problem, some observations of clustering problems for data anonymization should be taken into account. Firstly, the parameter $k$ in the $k$-anonymity privacy model is relatively small compared with the scale of a data set in big data scenarios. Since the upper-bound of the size of a cluster for local-recoding anonymization is $2k - 1$, the size of clusters is also relatively small. Accordingly, the number of clusters will be quite large. Secondly, under the condition that the size of any cluster is not less than $k$, the smaller a cluster is, the more it is preferred. The reason is that this tends to incur less data distortion. Ideally, the size of all clusters is exactly $k$. Thirdly, the intrinsic clustering architecture in a data set is helpful for local-recoding anonymization, but building such an architecture is not the final purpose.

Given the observations above, the agglomerative clustering method is suitable for local-recoding anonymization, as the stopping criterion can be set as whether the size of a cluster reaches to $k$. Moreover, the agglomerative clustering method can achieve minimum data distortion in the sense of the defined distance measure. Most existing approaches for $k$-anonymity mentioned in Section 2 employ greedy agglomerative clustering approaches. But they construct clusters in a greedy manner rather than combine the two clusters that have the minimum distance in each round, which results in more data distortion. But the optimal agglomerative clustering method suffers the scalability problem when handling large-scale data sets. Its time complexity is $O(n^2 \log n)$ with utilizing a priority queue. Worse still, the agglomerative method is serial, which makes it difficult to be adapted to parallel environments like MapReduce.

From the perspective of scalability, the point-assignment method seems to be ideal for local-recoding anonymization in MapReduce. The point-assignment process is to initialize a set of data records to represent the clusters, one for each, and assign the rest records into these clusters. The process is repeated until certain conditions are satisfied. Point assignment can be conducted in a scalable and parallel fashion in MapReduce. However, the set of representative records will become quite large according the observations above. Approximately, its size will be $1/k$ of an original data set. This fact makes it is a challenge to distribute such representative records to MapReduce workers who conduct point assignment according to the representative records independently. Another problem is that the size of each cluster is uncontrollable in the point-assignment process. Thus, the size of a cluster can exceed the upper-bound $2k-1$ or be less than $k$, especially when a data set has high skewness. Extra effort is often required to adjust clusters to proper size.

Given the pros and cons of the two clustering methods for local-recoding anonymization, we propose a two-phase approach that combines both methods based on MapReduce. In the first phase, we leverage point-assignment clustering method to partition an original data set into $t$ clusters, where $t$ is not necessarily relevant to $k$. For convenience, a cluster produced in the first phase is named as $\beta$-cluster. In the second phase, the agglomerative clustering method is run on each $\beta$-cluster simultaneously as 'plug-ins' , which is similar to [31]. In this way, our approach shares the merits of both methods but avoids the drawbacks. Specifically, the two-phase approach can produce quality anonymous data sets with the agglomerative clustering method and gain high scalability with the point-assignment method. In addition, no extra adjustment is required.

Algorithm 1 describes the main steps in the two-phase clustering approach. Similar to the spirit of $t$-means family [22, 32], we propose the $t$-ancestor clustering algorithm for point-assignment method. To avoid confusion, we employ the term '$t$-means' rather than '$k$-means' which is commonly used in literature. The details of the $t$-ancestor algorithm and the agglomerative algorithm will be presented in Subsection 5.2 and Subsection 5.3, respectively.

As $t$ is usually required in advance, we roughly estimate it and demonstrate that the two-phase clustering algorithm is scalable in big data scenarios. Let $N$ be the capacity of a MapReduce task worker, i.e., either a Mapper or a Reducer. Concretely, the capacity of $N$ here means that the worker can accomplish the agglomerative clustering on a data set of size $N$ within an acceptable time. The value of $t$ is estimated as $t \geq |D|/N$. Then, the expected maximum size of $\beta$-clusters $|D|/t$ can be less than the worker capacity $N$. In fact, the skewness in a data set will affect the maximum size of $\beta$-clusters. Thus, the higher the degree of skewness is, the larger $t$ should be. As $k \ll N$ according to the aforementioned observations, $t$ will be much smaller than $|D|/k$ which is the case if the point-assignment clustering method is exploited directly on anonymization. Hence, the set of $t$ representative records is relative small and can be distributed to each MapReduce worker efficiently. As such, our approach can handle large-scale data sets in a linear manner with respect to the number of MapReduce workers, which can be accomplished with ease in cloud environments due to their scalability.

ALGORITHM 1. **SKETCH OF TWO-PHASE CLUSTERING.**

```
Input: Data set $D$, anonymity parameter $k$.
Output: Anonymous data set $D^*$.
1: Run the $t$-ancestor clustering algorithm on $D$, get a set of $\beta$-
   clusters: $\mathbb{C}^\beta = \{C_1^\beta, \cdots, C_t^\beta\}$;
2: For each $\beta$-cluster $C_i^\beta \in \mathbb{C}^\beta$, $1 \leq i \leq t$: run the agglomerative
   clustering algorithm on $C_i^\beta$, get a set of clusters $\mathbb{C}_i =$
   $\{C_{i1}, \cdots, C_{im_i}\}$;
3: For each cluster $C_j \in \mathbb{C}$, where $\mathbb{C} = \bigcup_{i=1}^t \mathbb{C}_i$, generalize $C_j$ to $C_j^*$
   by replacing each attribute value with a general one;
4: Generate $D^* = \bigcup_{j=1}^{m_j} C_j^*$, where $m_j = \sum_{i=1}^t m_i$.
```

## 5.2 *t*-ancestor Clustering for Data Partitioning

One core problem in the point-assignment method is how to represent a cluster. Similar to $t$-medians [32], We propose to leverage the 'ancestor' of the records in a cluster to represent the cluster. More precisely, an ancestor of a cluster refers to a data record whose attribute value of each categorical quasi-identifier is the lowest common ancestor of the original values in the cluster. Each numerical quasi-identifier of an ancestor record is the median of original values in the cluster. The notion of ancestor record also attempt to capture the logical centre of a cluster like $t$-means/medians, but $t$-ancestors clustering is more suitable for anonymization due to categorical attributes in the clustering problem herein.

To facilitate $t$-ancestors clustering, we take quasi-identifier attributes but sensitive ones into consideration. This will rarely affect the proximity of sensitive values in a final cluster, because the clustering granularity in the first phase is rather coarse. Accordingly, we leverage the distance measure (8) to calculate the distance between a data record and an ancestor. Usually, an ancestor is not a real data record in the data set, but the (8) can still be employed to calculate the distance between two vectors of attribute values. Except where otherwise noted, a record $r$ in this subsection refers to the quasi-identifier part.

Initially, the $t$ ancestors in the first round of point assignment are $t$ records which are dedicatedly selected as seeds. In general, the selection of such $t$ records influences the quality of clustering to a certain extent. To obtain a good set of seeds, we pick data records that are as far away from each other as possible. Concretely, we accomplish seed selection via a MapReduce job *SeedSelection* which outputs a set of seeds: $S = \{r_1, \cdots, r_t\}$. The Map and Reduce functions of *SeedSelection* are described in Algorithm 2. In this job, only one Reducer is utilized for seed selection due to the serial nature of the algorithm. To make it scalable to big data, we sample an original data set by emitting a record to the Reducer with probability $N/|D|$ in the Map function, so that only about $N$ records in total go to the Reducer. In the Reduce function, the first seed is picked at random, and then we repeatedly pick the record whose minimum distance to the existing seeds is the largest until the number of seeds reaches $t$.

The $t$-ancestors clustering algorithm exploits Lloyd-style iteration refinement technique [22]. Each round of

ALGORITHM 2. **SEEDSELECTION MAP AND REDUCE.**

```
Input: Data record $r$, $r \in D$.
Output: A set of seeds $S = \{r_1, \cdots, r_t\}$.
Map: Generate a random value $rand$, where $0 \leq rand \leq 1$; if
   $rand \leq N/|D|$, emit $(1, r)$.
Reduce: 1: Select a random record $r$ from $list(r)$, $S \leftarrow r$;
        2: While $|S| < t$:
              Find $r \in list(r)$ that maximizes $\min_{r' \in S} d(r, r')$;
              $S \leftarrow r$;
        3: Emit (null, $S$).
```

iteration consists of two steps, namely, expectation ($E$) and maximization ($M$). In the $E$ step, data records are assigned to their nearest ancestor and constitute a $\beta$-cluster. In the $M$ step, the ancestor of a $\beta$-cluster is recomputed according to the records in the cluster. The new set of ancestors is used in the $E$ step of the next round. Ideally, it is expected that the iteration converges, i.e., the assignments no longer change after a finite number of rounds. However, a Lloyd-style clustering algorithm using a different distance measure other than Euclidean distance fails probably to converge, or is very slow to converge. In practice, two widely-adopted stopping criteria are employed together: 1) the difference of ancestors between two continuous rounds is smaller than a predefined threshold; 2) the rounds of iteration arrive at predefined number. Formally, let $S^i$ and $S^{(i+1)}$ be the two sets of seeds in round $i$ and $(i+1)$, respectively. The difference between them, denoted by $d(S^i, S^{(i+1)})$, is defined as the average distance between their records:

$$d\left(S^i, S^{(i+1)}\right) = \left(\sum_{j=1}^{t} d(r_j^i, r_j^{(i+1)})/t\right). \quad (10)$$

The first stopping criterion is quantified by $d\left(S^i, S^{(i+1)}\right) < \tau$, where $\tau$ is a predefined threshold. Let $\theta$ denote the predefined maximum number of iteration rounds. The $t$-ancestors clustering algorithm stops if either of the criteria above is satisfied. Ultimately, the algorithm is described in Algorithm 3.

In each round of the while-loop in Algorithm 3, a MapReduce job named as *AncestorUpdate* is designed to fulfill the $E$ and $M$ steps. Specifically, the Map function of the job is responsible for point assignments in the $E$ step, while the Reduce function accomplishes the recomputation of ancestors in the $M$ step. The Map and Reduce functions are described in Algorithm 4. Two subroutines, $Median()$ and $Ancestor()$, are utilized in the Reduce function to calculate the medians of numerical attributes and ancestors of categorical attributes, respectively. Note that the Reduce function is scalable with setting $t$ appropriately, and one Reducer can process more than one $\beta$-clusters in sequence if $t$ is large enough.

## 5.3 Proximity-Aware Agglomerative Clustering

Unlike Subsection 5.2, we leverage the proximity-aware distance measure (9) for the agglomerative clustering in this subsection. In the agglomerative clustering method, each data record is regarded as a cluster initially, and then two clusters are picked to be merged in each round of iteration until some stopping criteria are satisfied. Usually, two clusters with the shortest distance are merged.. Thus, one core problem of the agglomerative clustering method is how to define the distance between two clusters. To coincide with the objective in the SPAC problem, we leverage the complete-linkage distance in our agglomerative clustering algorithm, i.e., the distance between two clusters equals to the weighted distance between those two records (one in each cluster) that are farthest away from each other. In fact, after merging such two clusters, the distance between them is the diameter of the new cluster. Formally, the distance between clusters $C_x$ and $C_y$, denoted as $d(C_x, C_y)$, is calculated by:

$$d\left(C_x, C_y\right) = \left(|C_x| + |C_y|\right)\max_{r_x \in C_x, r_y \in C_y} dist\left(r_x, r_y\right). \quad (11)$$

Dissimilar to traditional agglomerative clustering algorithms, a cluster in our algorithm will not be considered for further merging if its size is equal to or larger than $k$. If no two clusters are of size less than $k$, the merging process stops. Accordingly, the maximum size of a cluster after merging is $2k-2$. It is possible that a single cluster of size less than $k$ remains after merging. We assign each data record of the left cluster to a cluster of size less than $2k-1$ who is the nearest to the record. In an extreme case that all clusters are already $2k-1$, we randomly pick a cluster and assign certain records from it to the left cluster to make the size of the latter be $k$. Note that there are only at most $k-1$ clusters if the extreme case takes place. Finally, every cluster resulting from the proximity-aware agglomerative clustering algorithm has at least $k$ records, but no more than $2k-1$ records.

Based on the analyses above, Algorithm 5 presents the proximity-aware agglomerative clustering algorithm formally. We leverage a priority queue $PQueue$ to retain distance between any two clusters, which aims at improving the performance of the agglomerative method. In the while-loop, the two clusters with the shortest distance are merged in step 3.1, and then the $PQueue$ as well as the set of clusters $\mathbb{C}^{(i+1)}$ are adjusted in step 3.2 and 3.3. In step 4, we cope with the remaining cluster mentioned above without considering the extreme case.

A MapReduce job named as *AgglomertiveClustering* is designed to wrap Algorithm 5. Specifically, Algorithm 5

ALGORITHM 3. **T-ANCESTORS CLUSTERING**

```
Input: Data set $D$; parameter $t$; thresholds $\tau$, $\theta$.
Output: $t$ $\beta$-clusters $\mathbb{C}^\beta = \{C_1^\beta, \cdots, C_t^\beta\}$.
1: Run job SeedSelection, get initial seeds $S^0$; $i \leftarrow 0$;
2: Run job AncestorUpdate, get ancestors $S^{(i+1)}$ ; $i \leftarrow i+1$;
   While $d\left(S^i, S^{(i+1)}\right) \geq \tau$ and $i \leq \theta$, repeat Step 2;
3: Return $\beta$-clusters with ancestors $S^{(i+1)}$.
```

ALGORITHM 4. **ANCESTORUPDATE MAP AND REDUCE.**

```
Input: Data record $r$, $r \in D$; seeds of round $i$, $S^i = \{r_1, \cdots, r_t\}$.
Output: Seeds of round $i$, $S^{(i+1)} = \{r_1^{(i+1)}, \cdots, r_t^{(i+1)}\}$.
Map: 1: $d^{\min} \leftarrow +\infty$;
     2: For $j$: 1 to $t$
            If $d(r, r_j) < d^{\min}$, then $d^{\min} \leftarrow d(r, r_j)$ and $j^{\min} \leftarrow j$;
     3: Emit $(j^{\min}, r)$.
Reduce: 1: For $l$: 1 to $m^{QI}$
            If $att_l^{QI}$ is numerical, then $v_l \leftarrow Median(list(r), l)$;
            Else $v_l \leftarrow Ancestor(list(r), l)$;
        2: Emit $(j, r_j^{(i+1)} = \langle v_1, \cdots, v_{m^{QI}} \rangle)$.
```

ALGORITHM 5. **PROXIMITY-AWARE AGGLOMERATIVE CLUSTERING**

```
Input: Data set $C^\beta$; $k$-anonymity parameter $k$.
Output: Clusters $\mathbb{C} = \{C_1, \cdots, C_n\}$.
1: Initialize each record in $C^\beta$ as a cluster, $\mathbb{C}^0 = \{C_1^0, \cdots, C_{n_0}^0\}$; $\mathbb{C} \leftarrow \emptyset$; $i \leftarrow 0$;
2: $\forall C_x^0, C_y^0 \in \mathbb{C}^0, x \neq y, PQueue \leftarrow \langle C_x^0, C_y^0, d(C_x^0, C_y^0) \rangle$;
3: While $PQueue$ is not empty
    3.1: $\langle C_x', C_y', d(C_x', C_y') \rangle \leftarrow PQueue$; $C_z' \leftarrow C_x' \cup C_y'$;
    3.2: $\mathbb{C}^{(i+1)} \leftarrow \mathbb{C}^i \backslash \{C_x', C_y'\}$;
         Delete entries involving $C_x'$ or $C_y'$ in $PQueue$;
    3.3: If $|C_z'| \geq k$, then $\mathbb{C} \leftarrow \mathbb{C} \cup \{C_z'\}$;
         Else $\mathbb{C}^{(i+1)} \leftarrow \mathbb{C}^{(i+1)} \cup \{C_z'\}$;
              $\forall C' \in \mathbb{C}^{(i+1)}, PQueue \leftarrow \langle C', C_z', d(C', C_z') \rangle$;
4: If $|\mathbb{C}^{(i+1)}| = 1$ and $C'' \in \mathbb{C}^{(i+1)}$, then $\forall r \in C''$, find a cluster $C \in \mathbb{C}$ and $|C| \leq 2k-2$, minimizing $d(\{r\}, C)$, and $C \leftarrow C \cup \{r\}$.
```

is plugged in the Reduce function of the job. After a Reducer collects all data records of a $\beta$-cluster, Algorithm 5 is executed to generate final clusters (QI-groups). The Map function is relatively simple, which just emits a record and its corresponding cluster.

### 5.4 Execution Process and Performance Analysis

In order to demonstrate the proposed two-phase proximity-aware clustering approach visually, its execution process overview is illustrated in Fig. 2. The bold solid arrow line in the right indicates the timeline of the process.

Seen from Fig. 2, the three MapReduce jobs are coordinated together to accomplish the local-recoding anonymization. From the perspective of control flow, our approach is partially parallel because the first phase is sequential with iteration of the *SeedUpdate* job, while the second phase is parallel. However, our approach is fully parallel from the perspective of data flow. The light solid arrow lines in Fig. 2 represent data flows in the canonical MapReduce framework, while the dashed arrow lines stand for data flows of dispatching seeds to distributed caches and the data flow of updating seeds. An Original data set is read by Map functions and its splits are processed in a parallel manner. As such, the two-phase clustering approach can handle large-scale data sets. Note that the amount of seeds (or ancestors) in the SeedUpdate job is relatively small with proper parameter $t$, so that they can be delivered to distributed caches efficiently.

To evaluate our approach theoretically, we analyze the performance of each MapReduce job in the two phases from five aspects in terms of [33], namely, time cost, space cost, task workers, network traffics and execution rounds. Table 2 illustrates the complexity results of the five aspects for each job. For conciseness, we use the number 1, 2 and 3 to stand for the jobs *SeedSelection*, *SeedUpdate* and *AgglomerativeClustering*, respectively. M and R are short for Map and Reduce functions, respectively. Most symbols mean the same as aforementioned. Parameter $l$ denotes the number of iteration rounds of the job *SeedUpdate*, and $P$ denotes the size of a data split fed to a Mapper. The number of all data records $n = |D|$.

Note that $t \geq n/N$ according to the analyses in Subsection 5.1. Thus, the time and space costs of Map and Reduce functions have a constant upper bound determined by $N$. The number of task workers varies in a linear manner with respect to the scale of the data set. As such, our approach is scalable with appropriate valuing of $t$. The first and third jobs are one-pass, while the second one is iterative with $O(l)$ rounds, where $l$ is determined by the stopping criteria discussed in Subsection 5.2.

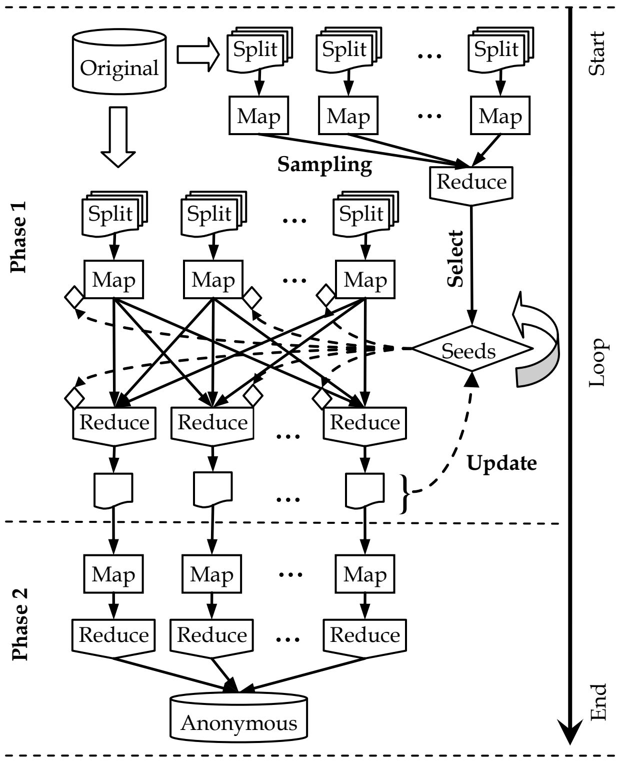

Fig. 2. Execution process overview of two-phase clustering.

TABLE 2 PERFORMANCE ANALYSIS OF MAPREDUCE JOBS

| Jobs | | Time | Space | Workers | Traffics | Rounds |
|---|---|---|---|---|---|---|
| 1 | M | $O(1)$ | $O(1)$ | $O(n/P)$ | $O(N)$ | $O(1)$ |
| | R | $O(t^2(N-t))$ | $O(N)$ | $O(1)$ | | |
| 2 | M | $O(t)$ | $O(t)$ | $O(n/P)$ | $O(n)$ | $O(l)$ |
| | R | $O(m^{QI}(n/t)^2)$ | $O(n/t)$ | $O(n/N)$ | | |
| 3 | M | $O(1)$ | $O(1)$ | $O(n/P)$ | $O(n)$ | $O(1)$ |
| | R | $O((n/t)^2 \log n/t)$ | $O((n/t)^2)$ | $O(n/N)$ | | |

## 6 EXPERIMENT EVALUATION

### 6.1 Overall Comparison

To evaluate the effectiveness and efficiency of the Proximity-Aware Clustering approach (PAC), we compare it with the $k$-Member Clustering approach (kMC) proposed in [18], which also represents the approaches for local recoding in [9, 19]. The kMC approach is the state-of-the-art approach for local-recoding anonymization with clustering techniques. As to effectiveness, we consider three factors, namely, the resistibility to proximity breaches, data distortion and scalability.

Several notions are defined for convenience. To capture the resistibility to proximity breaches, we define two statistics for a QI-group $G$, namely, the minimum distance $\epsilon_G^{MIN}$ and the average distance $\epsilon_G^{AVG}$. Let $n_G = |G|$, then,

$$\epsilon_G^{MIN} = \min_{r_x \neq r_y \in G} d\left(r_x^S, r_y^S\right), \tag{12}$$

$$\epsilon_G^{AVG} = \left(2 \sum_{r_x \neq r_y \in G} d\left(r_x^S, r_y^S\right)\right) / \left(n_G(n_G - 1)\right). \tag{13}$$

Actually, the QI-group here satisfies $(\epsilon_G^{MIN}, n_G - 1)^k$-dissimilarity. Hence, the resistibility can be captured by $\epsilon_G^{MIN}$ directly. As to the entire data set $D$, we measure the distribution of $\epsilon^{MIN}$ to capture the resistibility intuitively, where $\epsilon^{MIN}$ is the minimum intra-cluster distance of $D$, a random variable ranging within $[0, \max_{G \subseteq D} \epsilon_G^{MIN}]$. Technically, we leverage the Relative Cumulative Frequency (RCF, similar to cumulative distribution function) [34] of $\epsilon^{MIN}$ to describe the distribution for comparison. The average distance of QI-groups in $D$ is defined as

$$\epsilon^{AVG} = \frac{1}{|\{G | G \subseteq D\}|} \sum_{G \subseteq D} \epsilon_G^{AVG}. \tag{14}$$

From an overall perspective, the average distance $\epsilon^{AVG}$ can also reflect the vulnerability to proximity breaches to a certain extent. The metric *ILoss* [27] is employed to evaluate how much data distortion incurred by PAC and kMC. The value of *ILoss* is normalized to facilitate comparisons. For scalability, we check whether both approaches can still work and scale over large-scale data sets. Execution time is measured to evaluate efficiency. Experimental parameters are summarized in Table 3.

TABLE 3 SUMMARY OF EXPERIMENTAL PARAMETERS

| Symbol | Notations |
|---|---|
| $\epsilon^{MIN}$ | The minimum intra-cluster distance of a data set. |
| $\epsilon^{AVG}$ | The average intra-cluster distance of a data set. |
| $\omega^s$ | The weight of proximity in the distance measure (9). |
| $k$ | The $k$-anonymity parameter. |
| $t$ | The number of partitions after $t$-ancestors clustering. |
| $\theta, \tau$ | Two parameters for the stopping criteria of $t$-ancestors clustering. $\theta$ is the maximum number of iteration rounds, and $\tau$ is the threshold of the difference of the seeds between two consecutive iteration rounds. |

We conduct an overall comparison between PAC and kMC in terms of the four factors above. Intuitively, PAC will produce more QI-groups with higher $\epsilon_G^{MIN}$ than kMC, and $\epsilon^{AVG}$ of PAC will be larger than kMC. This is because PAC tends to assign dissimilar sensitive values into a cluster while kMC randomly put sensitive values into clusters. Normally, the *ILoss* of PAC will be higher than that of kMC, as there is a tradeoff between data distortion and the capability of defending privacy attacks. Note that this is a common phenomenon for privacy models (e.g., $l$-diversity and $t$-closeness) that fight against attribute linkage attacks [7, 8]. As to scalability, PAC can scale over more computation nodes with the data volume increasing, due to its parallelization capability. kMC will suffer from poor scalability over large-scale data sets because it requires too much memory, while PAC can linearly scale over data sets of any size. Correspondingly, the execution time of PAC is often less than kMC in big data scenarios. But note that the contrary case may occur because of the extra overheads engendered by PAC when the data set or the scale of MapReduce cluster is small. PAC is equivalent to kMC if the weight $\omega^s = 0$ and the parameter $t = 1$. Thus, kMC can be regarded as a special form of PAC.

## 6.2 Experiment Evaluation

### 6.2.1 Experiment Settings

Our experiments are conducted in a cloud environment named U-Cloud [4]. The system overview of U-Cloud has been depicted in Fig. 3. The Hadoop cluster is built on U-cloud. For more details about U-Cloud, please refer to [4].

The data set *Census-Income* (KDD) [35] is utilized in our experiments. Its subset *Adult* data set has been commonly used as a de facto benchmark for testing anonymization algorithms [9, 18, 19, 25, 26]. The data set is sanitized via removing records containing missing values and attributes with extremely skewed distributions. We obtain a sanitized data set with 153,926 records, from which data sets in the following experiments are sampled. Twelve

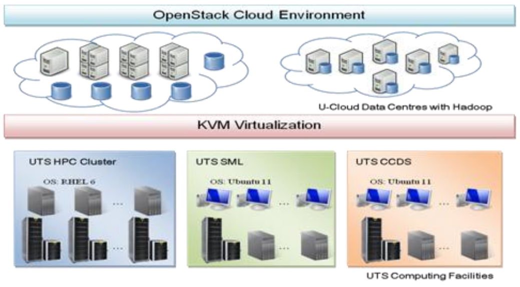

Fig. 3. System overview of U-Cloud.

attributes are chosen out of the original 40 ones, including 9 (4 numerical and 5 categorical) quasi-identifier ones and 3 (2 numerical and 1 categorical) sensitive ones.

Both PAC and kMC are implemented in Java. Further, PAC is implemented with the standard Hadoop MapReduce API and executed on a Hadoop cluster built on OpenStack in U-cloud. The Hadoop cluster consists of 20 virtual machines with type *m1.medium* which has 2 virtual CPUs and 4 GB Memory. kMC is executed on one virtual machine with type *m1.medium*. The maximum heap size of Java VM is set as 4 GB when running kMC. Each round of experiment is repeated 10 times. The mean of the measured results is regarded as the representative.

### 6.2.2 Experiment Process and Results

We conduct two groups of experiments in this section to evaluate the effectiveness and efficiency of our approach. In the first one, we explore whether proximity-aware clustering leads to larger dissimilarity by comparing PAC with kMC from the perspectives of resistibility to proximity breaches and data distortion. The other investigates the scalability and efficiency.

In the first group, we measure the change of the RCF of $\epsilon^{MIN}$, $\epsilon^{AVG}$ and *ILoss* with respect to the weight $\omega^s$. The weight $\omega^s$ varies from 0.0 to 1.0 with step 0.2. PAC is run over two data sets $D_1$ and $D_2$, with size $|D_1| = 1000$ and $|D_2| = 10000$, respectively. Note that the results of kMC are the same as PAC when $\omega^s = 0.0$. The parameter $k = 10$ for $D_1$, and $k = 50$ for $D_2$, respectively. Other parameters are set as follows: $t = 10$, $\theta = 5$, $\tau = 0.001$, and the number of Reducers is set as 10. The selection of some of these specific values is rather random and does not affect our analysis because what we want to see is whether PAC results in larger dissimilarity than kMC. Interested readers can try other values and the conclusion will be similar. The results of the first group are depicted in Fig.4.

Fig. 4(a) and Fig. 4(b) demonstrate the RCF of $\epsilon^{MIN}$ for $D_1$ and $D_2$, respectively, with 20 bins. Both of them show the same trend of RCF when the weight $\omega^s$ changes. The trend is that the curve of RCF of $\epsilon^{MIN}$ shifts right when $\omega^s$ is getting larger, indicating that the percentage of resultant QI-groups with higher $\epsilon^{MIN}$ increases with the growth of $\omega^s$. In particular, the leftmost curve is that of kMC ($\omega^s = 0.0$), which means that kMC produces the highest percentage of QI-groups with low $\epsilon^{MIN}$. Moreover, the percentile of a RCF curve goes up when $\omega^s$ becomes larger, which quantitatively reflects the above tendency. For instance, the median of a RCF curve (the $x$-axis coordinate of the point produced by the intersections of the 50%

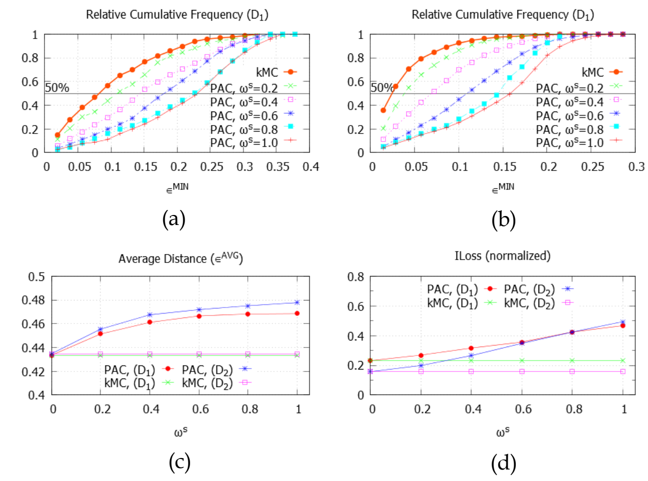

Fig. 4. Change of RCF of $\epsilon^{MIN}$, $\epsilon^{AVG}$ and normalized *ILoss* w.r.t. $\omega^S$.

horizontal line and the curve) is getting larger with the increase of $\omega^s$, as depicted in Fig. 4(a) and Fig. 4(b). From Fig. 4(c), we can see that the average distance between two records within a QI-group also grows over both data sets when $\omega^s$ becomes larger. As such, PAC outperforms kMC in terms of preventing proximity attacks as it can produce an anonymous data set with higher dissimilarity between two records in most QI-groups.

Meanwhile, it can be seen from Fig. 4(d) that the normalized value of *ILoss* rises as well when $\omega^s$ grows, indicating that more data distortion is incurred. In fact, the gain of dissimilarity is at the cost of data utility, which is a common phenomenon as mentioned in Section 6.1. Fortunately, one can choose a proper weight $\omega^s$ to make a good trade-off between the capability of defending proximity attacks and data utility. For example, $\omega^s = 0.6$ seems to be a good choice via observing Fig. 4.

In the second group of experiments, the execution time of PAC and kMC are gauged to investigate scalability and efficiency. Fig. 5(a) describes the change of execution time of PAC and kMC with respect to the number of data records which ranges from 10,000 (10k) to 100,000 (100k). As the scale of data in these experiments is much greater than that in [9, 18, 19], the data sets in our experiments are big enough to evaluate the effectiveness of our approach in terms of data volume. The value of $t$ varies with data sets, roughly making the size of $\beta$-cluster 1000. The $k$-anonymity parameter $k$ is set as 10. Other parameters are set as follows: $\omega^s = 0.5$, $\theta = 5$ and $\tau = 0.001$. Ten computation nodes are employed for PAC.

From Fig. 5(a), we can see the execution time of both PAC and kMC go up when the number of records increases although some slight fluctuations exist. The fluctuations, mainly incurred by the data distribution of each data set, will not affect the trends of execution time. Notably, the execution time of kMC surges from hundreds of seconds to more than 10,000s within only 4 steps, while that of PAC goes linearly and stably. The dramatic increase of PAC execution time illustrates that the intrinsic time complexity of kMC makes it hard to scale over big data. The difference of execution time between kMC and PAC becomes larger and larger when the data size is

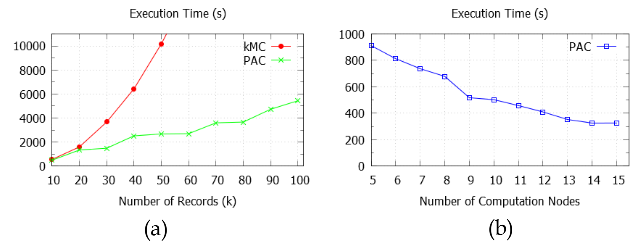

Fig. 5. Change of execution time w.r.t. data size and the number of computation nodes.

growing. This trend demonstrates that our approach becomes much more scalable and efficient compared with kMC in big data scenarios.

Fig. 5(b) exhibits the change of execution time of PAC with respect to the number of computation nodes ranging from 5 to 15. The number of data records set as 10,000, and other settings are the same as Fig. 5(a). It can be seen from Fig. 5(b) that the execution time decreases in a nearly linear manner when the number of computation nodes is getting larger. In terms of the tendency, we maintain that PAC is linearly scalable with respect to the number of computation nodes. Hence, PAC is able to manage to handle big data local recoding in a timely fashion in cloud where computation resources are offered on demand

Above all, the experimental results demonstrate that our approach integrating proximity of sensitive attributes into clustering, significantly improves the capability of defending proximity attacks, the scalability and efficiency of local-recoding anonymization over existing approaches.

## 7 Conclusions and Future Work

In this paper, local-recoding anonymization for big data in cloud has been investigated from the perspectives of capability of defending proximity privacy breaches, scalability and time-efficiency. We have proposed a proximity privacy model, $(\epsilon^+,\delta)^k$-*dissimilarity*, by allowing multiple sensitive attributes and semantic proximity of categorical sensitive values. Since the satisfiability problem of $(\epsilon^+,\delta)^k$-*dissimilarity* is NP-hard, the problem of big data local recoding against proximity privacy breaches has been modeled as a proximity-aware clustering problem. We have proposed a scalable two-phase clustering approach based on MapReduce to address the above problem time-efficiently. A series of innovative MapReduce jobs have been developed and coordinated to conduct data-parallel computation. Extensive experiments on real-world data sets have demonstrated that our approach significantly improves the capability of defending proximity attacks, the scalability and the time-efficiency of local-recoding anonymization over existing approaches.

In cloud environment, the privacy preservation for data analysis, share and mining is a challenging research issue due to increasingly larger volumes of datasets, thereby requiring intensive investigation. Based on the contributions herein, we plan to integrate our approach with Apache Mahout, a MapReduce based scalable machine learning and data mining library, to achieve highly scalable privacy preserving big data mining or analytics.

## ACKNOWLEDGEMENT

This work is partly supported by Australian Research Council Linkage Project LP140100816, National Science Foundation of China under Grant 91318301, an NSERC Discovery grant and a BCIC NRAS Team Project. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

## REFERENCES

[1] S. Chaudhuri, "What Next?: A Half-Dozen Data Management Research Goals for Big Data and the Cloud," *Proc. 31st Symp. Principles of Database Systems (PODS'12)*, pp. 1-4, 2012.

[2] L. Wang, J. Zhan, W. Shi and Y. Liang, "In Cloud, Can Scientific Communities Benefit from the Economies of Scale?," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 2, pp. 296-303, 2012.

[3] X. Wu, X. Zhu, G.-Q. Wu and W. Ding, "Data Mining with Big Data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97-107, 2014.

[4] X. Zhang, C. Liu, S. Nepal, S. Pandey and J. Chen, "A Privacy Leakage Upper Bound Constraint-Based Approach for Cost-Effective Privacy Preserving of Intermediate Data Sets in Cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1192-1202, 2013.

[5] B.C.M. Fung, K. Wang, R. Chen and P.S. Yu, "Privacy-Preserving Data Publishing: A Survey of Recent Developments," *ACM Comput. Surv.*, vol. 42, no. 4, pp. 1-53, 2010.

[6] L. Sweeney, "*K*-Anonymity: A Model for Protecting Privacy," *Int'l J. Uncertain. Fuzz.*, vol. 10, no. 5, pp. 557-570, 2002.

[7] A. Machanavajjhala, D. Kifer, J. Gehrke and M. Venkitasubramaniam, "*L*-Diversity: Privacy Beyond *k*-Anonymity," *ACM Trans. Knowl. Disc. Data*, vol. 1, no. 1, Article No. 3, 2007.

[8] N. Li, T. Li and S. Venkatasubramanian, "Closeness: A New Privacy Measure for Data Publishing," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 7, pp. 943-956, 2010.

[9] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi and A.W.C. Fu, "Utility-Based Anonymization Using Local Recoding," *Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data (KDD'06)*, pp. 785-790, 2006.

[10] K. LeFevre, D.J. DeWitt and R. Ramakrishnan, "Workload-Aware Anonymization Techniques for Large-Scale Datasets," *ACM Trans. Database Syst.*, vol. 33, no.3, pp.1-47, 2008.

[11] G. Aggarwal, R. Panigrahy, T. Feder, D. Thomas, K. Kenthapadi, S. Khuller and A. Zhu, "Achieving Anonymity Via Clustering," *ACM Trans. Algorithms*, vol. 6, no. 3, Article No. 49, 2010.

[12] T. Wang, S. Meng, B. Bamba, L. Liu and C. Pu, "A General Proximity Privacy Principle," *Proc. IEEE 25th Int'l Conf. Data Engineering (ICDE'09)*, pp. 1279-1282, 2009.

[13] T. Iwuchukwu and J.F. Naughton, "K-Anonymization as Spatial Indexing: Toward Scalable and Incremental Anonymization," *Proc. 33rd Int'l Conf. Very Large Data Bases (VLDB'07)*, pp. 746-757, 2007.

[14] X. Zhang, L.T. Yang, C. Liu and J. Chen, "A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using Mapreduce on Cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 2, pp. 363-373, 2014.

[15] X. Zhang, C. Liu, S. Nepal, C. Yang, W. Dou and J. Chen, "A Hybrid Approach for Scalable Sub-Tree Anonymization over Big Data Using Mapreduce on Cloud," *J. Comput. Syst. Sci.*, vol. 80, no. 5, pp. 1008–1020, 2014.

[16] Y. Yang, Z. Zhang, G. Miklau, M. Winslett and X. Xiao, "Differential Privacy in Data Publication and Analysis," *Proc. 2012 ACM SIGMOD Int'l Conf. Management of Data (SIGMOD'12)*, pp. 601-606, 2012.

[17] J. Lee and C. Clifton, "Differential Identifiability," *Proc. 18th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD'12)*, pp. 1041-1049, 2012.

[18] J. Li, R.C.-W. Wong, A.W.-C. Fu and J. Pei, "Anonymization by Local Recoding in Data with Attribute Hierarchical Taxonomies," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 9, pp. 1181-1194, 2008.

[19] J.-W. Byun, A. Kamra, E. Bertino and N. Li, "Efficient K-Anonymization Using Clustering Techniques," *Proc. 12th Int'l Conf. Database Systems for Advanced Applications (DASFAA'07)*, pp. 188-200, 2007.

[20] Q. Zhang, N. Koudas, D. Srivastava and T. Yu, "Aggregate Query Answering on Anonymized Tables," *Proc. IEEE 23rd Int'l Conf. Data Engineering (ICDE'07)*, pp. 116-125, 2007.

[21] J. Li, Y. Tao and X. Xiao, "Preservation of Proximity Privacy in Publishing Numerical Sensitive Data," *Proc. 2008 ACM SIGMOD Int'l Conf. Management of Data (SIGMOD'08)*, pp. 473-486, 2008.

[22] S. Lloyd, "Least Squares Quantization in Pcm," *IEEE Trans. Information Theory*, vol. 28, no. 2, pp. 129-137, 1982.

[23] R.C.-W. Wong, J. Li, A.W.-C. Fu and K. Wang, "($\alpha$, $k$)-Anonymity: An Enhanced *k*-Anonymity Model for Privacy Preserving Data Publishing," *Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD'06)*, pp. 754-759, 2006.

[24] K. LeFevre, D.J. DeWitt and R. Ramakrishnan, "Mondrian Multidimensional *k*-Anonymity," *Proc. 22nd Int'l Conf. Data Engineering (ICDE '06)*, Article No. 25, 2006.

[25] B.C.M. Fung, K. Wang and P.S. Yu, "Anonymizing Classification Data for Privacy Preservation," *IEEE Trans. Knowle. Data Eng.*, vol. 19, no. 5, pp. 711-725, 2007.

[26] N. Mohammed, B. Fung, P.C.K. Hung and C.K. Lee, "Centralized and Distributed Anonymization for High-Dimensional Healthcare Data," *ACM Trans. Knowl. Disc. Data*, vol. 4, no. 4, Article No. 18, 2010.

[27] X. Xiao and Y. Tao, "Personalized Privacy Preservation," *Proc. 2006 ACM SIGMOD Int'l Conf. Management of Data (SIGMOD'06)*, pp. 229-240, 2006.

[28] J. Dean and S. Ghemawat, "Mapreduce: A Flexible Data Processing Tool," *Comm. ACM*, vol. 53, no. 1, pp. 72-77, 2010.

[29] K.-H. Lee, Y.-J. Lee, H. Choi, Y.D. Chung and B. Moon, "Parallel Data Processing with Mapreduce: A Survey," *ACM SIGMOD Record*, vol. 40, no. 4, pp. 11-20, 2012.

[30] T. Wang and L. Liu, "XColor: Protecting General Proximity Privacy," *Proc. IEEE 26th Int'l Conf. on Data Engineering (ICDE'10)*, pp. 960-963, 2010.

[31] R.L. Ferreira Cordeiro, C. Traina Junior, A.J. Machado Traina, J. López, U. Kang and C. Faloutsos, "Clustering Very Large Multi-Dimensional Datasets with Mapreduce," *Proc. 17th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD'11)*, pp. 690-698, 2011.

[32] S. Arora, P. Raghavan and S. Rao, "Approximation Schemes for Euclidean *K*-Medians and Related Problems," *Proc. 30th Annual ACM Symp. Theory of Computing (STOC'98)*, pp. 106-113, 1998.

[33] Y. Tao, W. Lin and X. Xiao, "Minimal Mapreduce Algorithms," *Proc. 2013 ACM SIGMOD Int'l Conf. Management of Data (SIGMOD'13)*, pp. 529-540, 2013.

[34] I.W. Burr, "Cumulative Frequency Functions," *The Annals of Mathematical Statistics*, vol. 13, no. 2, pp. 215-232, 1942.

[35] UCI Machine Learning Repository, "Census-Income (KDD) Data Set," http://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD), accessed on: 30th November, 2013.