Towards Privacy Preserving Publishing of Set-valued Data on Hybrid Cloud

Hongli Zhang, Zhigang Zhou, Lin Ye, Xiaojiang Du

Abstract—Storage as a service has become an important paradigm in cloud computing for its great flexibility and economic savings. However, the development is hampered by data privacy concerns: data owners no longer physically possess the storage of their data. In this work, we study the issue of privacy-preserving set-valued data publishing. Existing data privacy-preserving techniques (such as encryption, suppression, generalization) are not applicable in many real scenes, since they would incur large overhead for data query or high information loss. Motivated by this observation, we present a suite of new techniques that make privacy-aware set-valued data publishing feasible on hybrid cloud. On data publishing phase, we propose a data partition technique, named extended quasi-identifier-partitioning (EQI-partitioning), which disassociates record terms that participate in identifying combinations. This way the cloud server cannot associate with high probability a record with rare term combinations. We prove the privacy guarantee of our mechanism. On data querying phase, we adopt interactive differential privacy strategy to resist privacy breaches from statistical queries. We finally evaluate its performance using real-life data sets on our cloud test-bed. Our extensive experiments demonstrate the validity and practicality of the proposed scheme.

Index Terms—cloud computing, differential privacy, data partition, data privacy, hybrid cloud, set-valued data

1 INTRODUCTION

CLOUD computing enables organizations or individuals outsource their data for enjoying a number of advantages: location independent data storage, ubiquitous data access and on-demand high quality services [1]. In recent years, several large IT companies have provided their own cloud platform, such as Amazon's EC2 and S3, Google AppEngine, IBM's Blue Cloud, and Microsoft Azure platform. However, it also makes data owners no longer physically possess the storage of their data, which inevitably brings in new security concerns and challenges towards this promising outsourcing service paradigm. The outsourced data maybe contain private information, such as the business financial records, medical data of patients' symptoms, etc. The concern of privacy breaches has hampered the development of cloud computing.

Data encryption with fine-grained data access control is a natural solution, however, data processing based on cryptographic operations (such as homomorphic encryption [2], fine-grained cloud data access [3-5, 34-35]) are still not up to the challenge posed by large data operation, which causes large overhead for publishing, querying, and other data operations. For example, homomorphic encryption was found to be prohibitively expensive for big data query, while the secret-sharing techniques underlying most outsourcing proposals lead to intensive data exchanges among the shareholders on cloud during a computation involving an enormous amount of data, and are therefore hard to scale.

- *H. Zhang is with Harbin Institute of Technology, Harbin, 150001, China. E-mail: zhanghongli@hit.edu.cn.*
- *Z. Zhou is with Harbin Institute of Technology, Harbin, 150001, China. E-mail: zzgisgod@sina.com.*
- *L. Ye is with Harbin Institute of Technology, Harbin, 150001, China. E-mail: yelin@hit.edu.cn.*
- *X. Du is with Temple University, Philadelphia PA 19122, USA. E-mail: dxj@ieee.org.*

In this work we study the issue of privacy-aware set-valued data publishing (e.g., market basket data publishing) on hybrid cloud. To protect data privacy from plaintext data publishing, the research and industrial communities have been investigating many data publication solutions to ensure data confidentiality. Most of these techniques employ generalization [7-9] to reduce the original term domain and eliminate quasi-identifiers (QIs) which are a set of rare term combinations that can identify a record. For instance, they would generalize Beijing to China, so that the infrequent term would be replaced by the more frequent one, which satisfies the anonymous requirement. Alternatively, other techniques rely on adding noise [10-11] (such as fake terms or records) to offer non-interactive differential privacy. However, these methods are not applicable in many real scenes, since they would incur a high irreversible information loss that is not acceptable. There are only few work based on QI-partitioning [12-13] to split QIs into several parts. These methods are based on *k-anonymity* model. Unfortunately, they often get stuck when facing set-valued data with the character of high-dimensional sparse. Relative to data publishing, data querying is the superstructure. Based on the existing data publishing technologies, data querying is such a punchbag. As there are a thousand Hamlets in a thousand people's eyes, differential data privacy protection should be provided for satisfying the various requirements on users with different roles. User roles reflect the commercial agreements between data owner and authorized users. Similar to fine-grained access control, data owner should provide different granular results for authorized users with different roles, even facing the same data set for the same query.

To address these urgent problems, a novel generic secure framework needs to be built, which explicitly takes into consideration both privacy needs on data publishing and data querying. Our system, named Cocktail, is a privacy-aware hybrid cloud framework that includes two phases: data publishing, and data querying. On data publishing phase, our setting of the privacy problem is generic and does not assume, where any term or term combination might form an identifier to reveal private information. Focusing on privacy-preserving data publishing, our solution is based on a data partition strategy that breaks associations among terms, guaranteeing respect of both confidentiality constraints and data availability. Specifically, in our modeling, the privacy model is based on k^m-*anonymity* [13]: An adversary, who knows up to m terms from any record, will not be able to match his knowledge to less than k records in the published data. Confidentiality constraints provide an explicit metrics for EQI-partitioning to express the requirement that any term combination T_i ($|T_i| \leq m$) identifying less than k record should be partitioned into several chunks so that each one satisfies k^m-*anonymity* model. Data availability in turn provides an evaluation criterion that, under the condition of privacy-ensuring, the number of chunks N should be minimal. It minimizes the overhead of data reconstruction that needs $N-1$ times data integration operations. On data querying phase, we adopt the popular MapReduce paradigm which breaks down a query into subtasks and executes them in the public and private clouds respectively. In addition, we employ interactive differential privacy to protect data privacy of the integrated data results. It is different from the existing privacy-preserving mechanisms based on data anonymization models [21, 22] which are vulnerable to privacy attacks based on background knowledge, and it is also not the same as non-interactive differential privacy strategy. Instead of adding noise to the outsourced original data, interactive differential privacy just inserts noise into query result. Also, it provides strong privacy guarantees independent of an adversary's background knowledge, computational power and subsequent behavior. In our research, we strictly prove the privacy guarantee of the proposed mechanism. And the extensive experiments on our cloud test-bed with real-life data set further demonstrate the validity and practicality of the proposed mechanism. The contributions of the paper are summarized as follows:

1) To the best of our knowledge, this is the first study that formalizes the problem of privacy-preserving set-valued data publication over hybrid cloud, and provides a complete system framework. Our design proposes a novel data partition mechanism that splits EQI into different chunks, and it ensures that private information will not be exposed to the public cloud. Moreover, we employ interactive differential privacy into the proposed framework, which provides strong privacy guarantees.

2) We built a new query analysis tool that automatically transforms the structure of a query to optimize data query on hybrid-cloud. The tool not only breaks down a query into sanitized sub-queries that can work on the public and private clouds respectively, but also helps control the amount of the intermediate outcomes to be delivered back to the private cloud, which ensures the data confidentiality.

3) We have implemented our scheme and achieved limited information loss by evaluating it over our cloud testbed on real datasets. Furthermore, our extensive experiments further demonstrate the validity and practicality of our scheme.

The rest of the paper is organized as follows. Section 2 introduces the system and threat model, and the security guarantee metric. Section 3 provides the detailed description of privacy-preserving data publishing scheme, and gives thorough analysis of privacy guarantee. Section 4 presents interactive differential privacy on data query. Section 5 describes the technical details of Cocktail. Section 6 evaluates the data quality and performance of Cocktail. Section 7 overviews related work, followed by the concluding remarks in Section 8. Notations are summarized in Table 1.

TABLE 1
LIST OF NOTATIONS USED IN THIS WORK

Symbol	Explanation
E	set-valued data set
R, r_i	records
T, t_j	terms
k^m	anonymous parameters
P	clusters
C^{Pu}, C^{Pr}	public chunks, private chunks
$\mathbb{U}$, U	concept set, a concept
$\mathbb{I}$	intension domain
$q(I)$	extension domain
S	aggregated clusters
Q	query
M	differential privacy mechanism

2 PROBLEM DEFINITION

The proposed anonymization method focuses on privacy-aware set-valued data publishing on hybrid cloud. In this section, we briefly give some background information on set-valued data and hybrid cloud. In addition, we formally define the threat model and the security guarantees our method targets to.

2.1 Background Information

Set-Valued Data. Set-valued data in the form of high dimensional sparse data is commonly ranging from market basket data to web search query logs, in which a set of values are oftentimes associated with private information. For example, a web search query logs dataset contains records that involve with users' privacy information. Assume that the attacker knows that Bob was interested in booking air ticket to Beijing. If dataset is released and only one record contains both "air ticket" and "Beijing" in dataset, and then attacker can easily infer that this record corresponds to Bob. Figure 2(a) presents an exemplary web search query log consisting of 10 records (each being the web search history of a user). For set-valued data, we do not distinguish between sensitive and non-sensitive terms; rather, we consider the general case that any term might reveal private information and any term can be

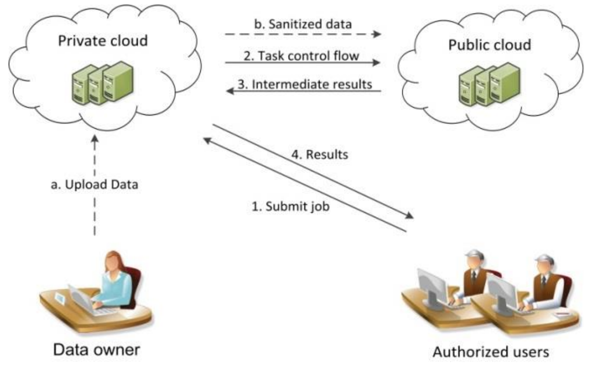


Fig. 1: The high-level architecture for Cocktail.

used as part of a quasi-identifier for identifying a certain user.

Hybrid Cloud. Hybrid cloud framework [6] is a dreamed vision in the family of the cloud computing, which presents a new opportunity that makes secure data publishing possible. It usually involves both the private cloud within an organization and the public commercial cloud, where data owners could first preprocess the outsourcing dataset in the trusted private cloud, such as suppressing, generalizing. In our modeling, data owner could first partition original dataset into several chunks, then send the sanitized chunks to the public commercial cloud infrastructure, while keeping the rest in private cloud, where the data scale can be much smaller than the original data.

2.2 System & Threat Model

We begin by describing a high-level architecture for Cocktail, as illustrated in Fig.1, which involves four types of entities: data owner, authorized users, private cloud, and public cloud.

- Data owner. The data owner can be an organization or an individual, who has a collection of set-valued data $E(R,T,F)$ to be outsourced to the private cloud, where $R=\{r_1,\dots,r_n\}$, each $r_i (i \le n)$ denotes a record; $T=\{t_1,\dots,t_h\}$, each $t_j (j \le h)$ denotes a term; $F = R \times T \rightarrow V \in \{0,1\}$ denotes the implication relation between R and T, when $v_{(r_i,t_j)} = 1$, it denotes $(r_i,t_j) \in F$, that is, the file r_i has the term t_j; when $v_{(r_i,t_j)} = 0$, it denotes $(r_i,t_j) \notin F$.
- Private cloud. Private cloud is the control center of Cocktail, which is often built by data owner, and preprocesses original dataset before outsourcing. Here, we assume that the communication channel between data owner and private cloud is secure. When receiving the original dataset, in our modeling, private cloud would use EQI-partitioning strategy to fragment dataset for breaking associations among them (details is shown in Section 3). Eventually, private cloud sends the sanitized data to public cloud. Moreover, in data querying phase, private cloud provides interactive interface to authorized users, which hides the complex business logic between data owner and hybrid cloud.
- Public cloud. Public cloud owns significant storage and computation resources, and offers resource rental services for data owners in pay-as-you-go manner. In data querying phase, public cloud receives data query tasks sent from private cloud, and returns query results to private cloud.
- Authorized users. Authorized users are granted query permissions on a certain outsourced data set. They submit data queries to private cloud, and obtain results directly from private cloud.

To prevent unauthorized users access data, before data owner uploads data to private cloud, data access control strategy has to be applied. We assume that the authorization (access control) to users has been appropriately done. When an authorized user sends a query to the private cloud, Cocktail automatically maps the query into a suit of ordered sub-queries according to data distribution (between private and public clouds). Then these sub-queries are orderly distributed into interrelated clouds based on dependency of the requested data.

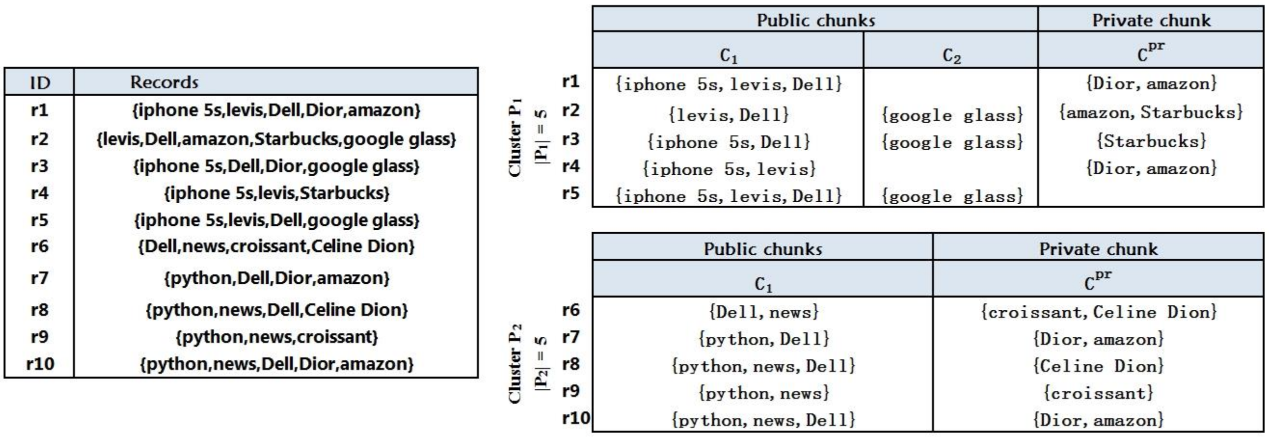
In this paper, we consider public CSP as "honest-but-curious" as [15] does. Specifically, the CSP acts in an “honest” fashion and correctly follows the designated protocol specification. However, it is "curious" to deduce and analyze data so as to learn privacy information. While we assume that the users are dishonest and may collude to obtain privacy information. The adversaries can derive private information from the outsourced data by multi-statistical queries [12], background knowledge attack [26].

2.3 Security Metrics

Many security metrics (such as *k-anonymity* [19, 20], non-interactive differential privacy [10]) have been proposed to prevent an attacker from inferring data privacy. In the context of set-valued data publishing, however, existing solutions, such as generalization, suppression, or adding noise, for satisfying the mentioned security metrics, would cause a huge irreversible information loss. For this reason, we opt for k^m-*anonymity*, which protects data privacy when an adversary knows background knowledge of up to m terms of a record. More formally:

Definition 1 (k^m-*anonymity*) [13]. An anonymized dataset E is k^m-*anonymous* if no adversary that has background knowledge of up to m terms of a record can use these terms to identify less than k candidate records in E.

It is worth noting that, the smaller the m in k^m-*anonymity*, the weaker privacy-preserving k^m-*anonymity* provides. When m equals the number of terms in the maximum length of the records, k^m-*anonymity* is equivalent to *k-anonymity*. Let us consider an interesting issue: in order to achieve k^m-*anonymity* model, any term combination that forms QI should be divided into different chunks; In addition, for those term combinations, they do not form QI but the number of records identified by them is less than k, that is to say, the probability that private information being inferred is greater than $1/k$. Therefore, these term combinations should also be split. For this reason, we adopt extended QI-partitioning as the anonymity



ID	Records
r1	{iphone 5s,levis,Dell,Dior,amazon}
r2	{levis,Dell,amazon,Starbucks,google glass}
r3	{iphone 5s,Dell,Dior,google glass}
r4	{iphone 5s,levis,Starbucks}
r5	{iphone 5s,levis,Dell,google glass}
r6	{Dell,news,croissant,Celine Dion}
r7	{python,Dell,Dior,amazon}
r8	{python,news,Dell,Celine Dion}
r9	{python,news,croissant}
r10	{python,news,Dell,Dior,amazon}

(a) Original data.

		Public chunks		Private chunk
		C_1	C_2	C^{pr}
Cluster P_1 $\|P_1\| = 5$	r1	{iphone 5s, levis, Dell}		{Dior, amazon}
	r2	{levis, Dell}	{google glass}	{amazon, Starbucks}
	r3	{iphone 5s, Dell}	{google glass}	{Starbucks}
	r4	{iphone 5s, levis}		{Dior, amazon}
	r5	{iphone 5s, levis, Dell}	{google glass}	

		Public chunks	Private chunk
		C_1	C^{pr}
Cluster P_2 $\|P_2\| = 5$	r6	{Dell, news}	{croissant, Celine Dion}
	r7	{python, Dell}	{Dior, amazon}
	r8	{python, news, Dell}	{Celine Dion}
	r9	{python, news}	{croissant}
	r10	{python, news, Dell}	{Dior, amazon}

(b) Anonymous data

Fig. 2: Example for EQI-partitioning scheme.

scheme to implement k^m-*anonymity* model.

Definition 2 (EQI). Given a set-valued data $E(R,T,F)$, for $R^{\#} \subseteq R$, $T^{\#} \subseteq T$, $R^{\#} = \{r \in R \mid \cap_{t \in T^{\#}} F(r,t) = 1\}$ and $|R^{\#}| < k$ $(k \geq 1)$, where $|R^{\#}|$ denotes the number of records $R^{\#}$; k is the anonymity threshold, we say $T^{\#}$ is an EQI over E. If and only if $k = 1, T^{\#}$ is an QI over E.

Definition 3 (EQI-partitioning). A set-valued data $E(R,T,F)$ satisfies EQI-partitioning, iff: for each chunk C_i containing data $E^{\#}(R^{\#},T^{\#},F)$, there has no such $T_i(|T_i| \leq m) \subseteq T^{\#}$, which can identify a kind of records $R^{\#} \subseteq R$ with $|R^{\#}| < k$.

Lemma 1 (Confidentiality Constraint). A set-valued data $E(R,T,F)$ satisfying EQI-partitioning is k^m-*anonymous*.

Proof: We prove it by contradiction. Given $T^{\#}(|T^{\#}| \leq m) \subseteq T$, assume that there is a chunk C_i with $T^{\#}$, and the number of records identified by $T^{\#}$ is less than k. In this case, $T^{\#}$ forms an EQI based on the Definition 2. Therefore, the chunk C_i no longer meets the Definition 3, completing the contradiction. □

3 Anonymization on Set-valued Data Publishing

In this paper, we propose a privacy-preserving anonymization scheme based on EQI-partitioning. The proposed scheme partitions the original records into smaller chunks. The objective is to hide infrequent term combinations in the original records by partitioning EQI into different chunks. To describe the crux of the EQI-partitioning scheme, we use an example as illustrated in Fig. 2(a). Our scheme, initially, partitions the original dataset into several clusters P. Although horizontal partitioning is not necessary, it helps the Cocktail achieve the goal of parallel processing when facing big dataset (details are shown in Section 5). Focusing on each cluster, P_1, for example, contains records $r_1 \sim r_5$, where it is prone to be attacked (e.g., r_1 can be identified by iphone 5s and amazon; r_2 can be identified by Dell and Starbucks; etc.). The corresponding transformed dataset is illustrated in Figure 2(b). Here, given $k = 3$ *and* $m = 2$, P_1 is vertically split into two types of chunks C^{Pu} $\{C_1, C_2\}$ and C^{Pr}, where C_1 contains terms $T_1 = \{iphone\ 5s, levis, Dell\}$, C_2 contains $T_2 = \{google\ glass,\}$, and C^{Pr} contains $T_{Pr} = \{Dior, amazon, Starbucks\}$. We can observe that, each chunk in C^{Pu} satisfies k^m-*anonymity* $(k = 3, m = 2)$ as the Definition 3, while each term in C^{Pr} does not meet 3-*anonymity* $(k = 3)$, let alone satisfy 3^2-*anonymity* among them. In our case, C^{Pu} would be outsourced in public cloud, while C^{Pr} would be keep in private cloud for security reasons. In addition, the order of the subrecords in each chunk is disrupted by a one-way Hash function. In other word, the association among different chunks is hidden.

In the following, we present the details of our technique. To facilitate the description, we first present a basic scheme, which performs two steps: EQI-identifying, and EQI-partitioning. Then, to minimize the number of data chunks, we put forward the corresponding improvement strategy.

3.1 A Basic Scheme

3.1.1 EQI-identifying

Terrovitis [8] first proposes an EQI-identifying algorithm for set-valued dataset with the time complexity $O(T^3)$. We named it as EQI-testing. For each term, it tests the existing data chunks with the candidate term one by one to check whether there exists a chunk satisfying k^m-*anonymity* after extending it; If there is such a data chunk, it extends this chunk with the candidate term, otherwise, it generates a new chunk. Obviously, the number of chunks is significantly affected by the order of the candidate term list. Moreover, on average, it needs to scan $O(T)$ times the dataset. Facing the massive volume and high dimensional data, this strategy is not applicable.

Therefore, we propose a novel EQI-identifying sheme. To facilitate our discussion, we first introduce a data structure $U(I,\ q(I))$, called a concept, where $I \subseteq T$ is called the intension of concept, and $|I|$ denotes the dimension of the intension; $q(I) \subseteq R$, called the extension of concept, is composed of the record that contain the term set I. We classify concept sets $\mathbb{U}$ over $E(R,T,F)$ by the size of $|I|$, e.g., $\mathbb{U}_{|I|}$ is made up of the whole concepts with the intension dimension $|I|$. The Lemma 2 shows an apriori property of concept, which can be used in reducing the search space of constructing higher-dimensional concept set.

Lemma 2 (Apriori Property). Given a concept $U\big(I, q(I)\big)$ over $E(R,T,F)$, where $I \neq \emptyset$ and $|q(I)| \geq k$, for any concept $U_i\big(I_i,\ q(I_i)\big)$ with $I_i \subseteq I (I_i \neq \emptyset)$, we have $|q(I_i)| \geq k$.

It is obvious that if a concept $U_i\big(I_i,\ q(I_i)\big)$ does not satisfy the anonymous threshold k, all of the higher-dimensional concepts $U\big(I,\ q(I)\big)$ with $(I \supset I_i)$ will not satisfy the anonymous threshold as well. According to the Lemma 2, generating of the i-dimensional concept set $\mathbb{U}_i (i > 1)$ just needs the previous concept set $\mathbb{U}_{i-1}$. Specifically, given the anonymous parameters (k,m), we iteratively generate all of i-dimensional concept set $(i \leq m)$ as follows:

1) Generating candidate 1-dimensional concept set. Each term constitutes the intension of a candidate 1-dimensional concept. The algorithm scans all of the records in the target cluster, recording the corresponding extension and the size of the extension domain.

2) Generating $\mathbb{U}_1$. Based on the anonymous threshold k, the 1-dimensional concept set $\mathbb{U}_1$ can be determined. It consists of the candidate 1-dimensional concept set, where $\left|q\big(I_j\big)\right|$ of each concept U_j is equal or greater than k.

3) Generating i-dimensional concept set $\mathbb{U}_i$ $(i > 1)$ based on the concept set $\mathbb{U}_{i-1}$. The algorithm first uses the join $\mathbb{I}_{i-1} \bowtie \mathbb{I}_{i-1}$ to generate a candidate i-dimensional intension set. Then, based on the apriori property (Lemma 2) that all sub-concept of a higher-dimensional concept satisfying the anonymous threshold also satisfy the anonymous threshold, we can prune the candidate intension sets that do not satisfy the apriori property. For each of the rest candidate intension sets, we compute the intersection of extension sets corresponding with $(i-1)$-dimensional concepts. Then, the i-dimension concept set can be determined. It consists of the concepts, where $|q(\mathbb{I}_i)|$ of $\mathbb{U}_i$ is equal or greater than k.

4) Jumping to step 3) until $\mathbb{U}_i = \emptyset$ or $i = m$.

We can easy to see that, given the anonymous parameters (k,m), the EQI-identifying can be transferred to the m rounds concept-generating. For each round $i (1 \leq i \leq m)$, the algorithm generates concept set $\mathbb{U}_i$. The intension sets in $\mathbb{U}_i$ represent all possible i-dimensional term combinations that satisfy k^m-*anonymity* in the target cluster.

Lemma 3. The proposed EQI-identifying scheme can identify all i-dimension term combinations $(i \leq m)$.

Proof: First, for 1-dimensional terms, it is obvious, which detailedly scans dataset to generate all 1-dimensional concepts with the number of extension satisfying the anonymous threshold k. Next, since the mechanism is iterative, we just prove the one-to-one mapping relationship between the i-th dimension concept set $(i \leq m)$ and i-dimensional term combinations satisfying k^m-*anonymity*. *1)* Assuming that there exists a concept U_{i_j} with I_{i_j}, which does not belong to i-dimensional term combinations. That is, the $\left|q\left(I_{i_j}\right)\right|$ is less than k. It forms the contradiction with the step 3) of the mechanism. *2)* Assuming that there exists a i-dimensional term combination T_{i_j} satisfying k^m-*anonymity* but not mapping into any one of $\mathbb{U}_i$. We have the whole $(i-1)$-dimensional combinations T_{i-1} generated by T_{i_j} also satisfy k^m-*anonymity*. It forms the contradiction with the Lemma 2. □

3.1.2 EQI-partitioning

Before describing the EQI-partitioning strategy, we first introduce two concept operations: concept covering, and concept reducing.

Definition 4 (Concept Covering). Given two concepts U_i, U_j, we say that U_i covers U_j $(U_i \mid_c U_j)$, if and only if: $q(I_i) \subseteq q\big(I_j\big)$, $|q(I_i)| \geq k$, and $I_j \subseteq I_i$. More formally:

$$U_i \mid_c U_j = \begin{cases} U_i; & |q(I_i)| \geq k, \text{and } I_j \subseteq I_i \\ U_j; & otherwise \end{cases} \quad (1)$$

Definition 5 (Concept Reducing). Given two concepts U_i, U_j, we get $U^*\left(I_j - I_i, q\big(I_j - I_i\big)\right)$ by performing "U_i reduces U_j $(U_i \mid_r U_j)$". More formally:

$$U_i \mid_r U_j = U^*\left(I_j - I_i, q\big(I_j - I_i\big)\right) \quad (2)$$

Property 1 (Closure). Concept covering and concept reducing are closed on the proposed EQI-identifying mechanism.

Proof: The proof follows easily from the Lemma 2 and is omitted for brevity. □

EQI-partitioning, intuitively, is a vertical partitioning strategy that applies on each cluster and divides it into chunks. According to chunks whether satisfy k^m-*anonymity,* The partitioned chunks are classified into two classes: public chunk C^{Pu} and private chunk C^{Pr}. In our modeling, C^{Pu} satisfying k^m-*anonymity* is outsourced in public cloud, while C^{Pr} is made up of a few of terms, in which any non-empty subset does not satisfy k^m-*anonymity*, and is kept in private cloud for security reasons. According to this description, we can see that, C^{Pr} is an essential component when all terms in the target chunk satisfy k^m-*anonymity*; otherwise, C^{Pr} uniquely exists.

Next, we describe a feasible EQI-partitioning strategy based on the EQI-identifying scheme.

Algorithm 1 EQI-partitioning

```
INPUT:  $\mathbb{U}$ /* all i-dimensional concept set identified by EQI-
               identifying scheme $(i \leq m)$*/
        $P$ /* the target cluster*/
OUTPUT: $C^{Pu}$, $C^{Pr}$
1: while $\mathbb{U} \neq \emptyset$ do
2:     Sort $\mathbb{U}$ in intension size descending order. Let the
       sorted concept list in $\mathbb{U}$ be $[U|\mathbb{U}_{\bar{U}}]$, where $U$ is the
       first element and $\mathbb{U}_{\bar{U}}$ is the remaining list;
3:     $C^{Pu} = C^{Pu} \cup U$;
4:     $\mathbb{D} = \emptyset$; $\mathbb{G} = \emptyset$;
5:     for each $U_i$ in $\mathbb{U}_{\bar{U}}$ do
6:         $\mathbb{D} = \mathbb{D} \cup (U \mid_c U_i)$;
7:     for each $U_i$ in $\mathbb{D}$ do
8:         $\mathbb{G} = \mathbb{G} \cup \big(U \mid_r U_i\big)$;
9:     $\mathbb{U} = \mathbb{G}$;
```

```
10: $C^{Pr} = P - C^{Pu}$;
11:return $C^{Pu}$, $C^{Pr}$;
```

Correctness of the Algorithm 1. The input $\mathbb{U}$ of the algorithm comes from EQI-identifying scheme, which satisfy k^m-*anonymity* based on the Lemma 3. The concept covering (line 6) and concept reducing (line 8) can refine a set of non-overlapping concept set from $\mathbb{U}$. Therefore, each chunk of C^{Pu} satisfies k^m-anonymity. Due to C^{Pu} is complete, which contains all of terms in $\mathbb{U}$, the operation in line 10 generates the C^{Pr}, which contains the rest terms that do not satisfy k-*anonymity*.

Although the Algorithm 1 can generate a set of feasible EQI partitions, in many practical applications, we hope that the number of partitions is minimal, which helps reduce the data reconstruction overhead. That is, the data partition strategy should meet the data availability rule.

Data Availability Rule: The number of the data chunks satisfying the confidentiality constraint should be minimized.

Unfortunately, the minimal EQI-partitioning (Min-EP) problem is NP-comolete, as illustrated in Theorem 1.

Theorem 1. The Min-EP problem based on EQI-identifying scheme is NP-complete.

Proof: See Appendix B. □

3.2 Improvement Strategy

As showed in the Theorem 1, Min-EP by directly using EQI-identifying scheme is NP-complete. The reason is that, EQI-identifying does not provide the support of concept compatibility. That is, there may be more than m-dimensional concepts which makes less partitions possible. Therefore, we first propose a concept expansion operation.

Lemma 4 (Concept Expanding). Given a set of concept set $\mathbb{U}$ satisfying k^m-*anonymity* over a target cluster $P(R,T,F)$, consider a concept $U_j(I_j, q(I_j)) \in \mathbb{U}$, if $\exists t_i \in T - I_j$, we say U_j are extensible $\left(U_j \mid_e t_i\right)$, if and only if: we can construct a concept set $\mathbb{U}^*$ over $P(R,T,F)$, where $\mathbb{I}^*$ is the intension set containing all m-dimensional combination of I ($I = I_j \cup t_i$). More formally:

$$U_j \mid_e t_i = U^*\left(I_j \cup t_i, q\left(I_j \cup t_i\right)\right) \quad (3)$$

Proof: We can see the precondition of concept expansion is consistent with the Definition 1. Therefore, the expanded concept also satisfies k^m-anonymity. □

According to the Lemma 4, we can get an iterative i-dimensional concept construction algorithm ($i > m$), as illustrated in the Algorithm 2.

```
Algorithm 2  Constructing $i$-dimension concept ($i > m$)
INPUT:  $\mathbb{I}_m$ // the intension set of $m$-dimension concept
        $\mathbb{U}_{i-1}$
OUTPUT:  $\mathbb{U}_i$
1: $\mathbb{I}_{i-1} \bowtie \mathbb{I}_{i-1}$ to generate $i$-dimensional intension set $\mathbb{I}_i$;
2: for each $I_j$ in $\mathbb{I}_i$ do
3:     Generating all $m$-dimension combinations $\mathbb{I}_m^*$ of $I_j$;
4:     if $\mathbb{I}_m^* \in \mathbb{I}_m$ then
5:         $\mathbb{U}_i = \mathbb{U}_i \cup U_j$;
6: return $\mathbb{U}_i$;
```

By iteratively calling the Algorithm 2, we can get all dimension concept set $\mathbb{U}$. Taking it as input, we can get a set of non-covered concept set $\mathbb{U}^c$ ($\mathbb{U}^c \subseteq \mathbb{U}$) by performing the concept covering operation among $\mathbb{U}$. It is easy to prove that the dimension size of each concept in $\mathbb{U}^c$ is maximal.

Question: Taking $\mathbb{U}^c$ as input, whether the Algorithm 1 will get the Min-EP over a target cluster.

Theorem 2. Let $\left[U_d|\mathbb{U}^c_{\overline{U_d}}\right]$ be the concept list in $\mathbb{U}^c$, where U_d is any concept in $\mathbb{U}^c$, and $\mathbb{U}^c_{\overline{U_d}}$ is the remaining list. The Min-EP can be obtained by a greedy algorithm, if and only if: $I_d \nsubseteq \bigcup_{U_i \in \mathbb{U}^c_{\overline{U_d}}} I_i$.

Proof: See Appendix C. □

This theorem tells us that the Min-EP problem can be solved by using greedy strategy (such as the Algorithm 1) when $\mathbb{U}^c$ meets the necessary and sufficient condition.

Discussion. Now, we discuss two cases, in which the target concept set $\mathbb{U}^c$ does not satisfy the premise of the Theorem 2. Case *1)* $\exists\, \mathbb{U}^* \subseteq \mathbb{U}^c$, for any $U_d \in \mathbb{U}^*$, there is $I_d \subseteq \bigcup_{U_i \in (\mathbb{U}^c - \mathbb{U}^*)} I_i$. It means that concepts in $\mathbb{U}^*$ is independent of each other. That is, eliminating any concept in $\mathbb{U}^*$ will not change the fact that the remaining concepts in $\mathbb{U}^*$ still do not satisfy the Theorem 2. Therefore, we can easy to construct a greedy algorithm with polynomial time complexity for eliminating all the concepts in $\mathbb{U}^*$, which makes the remaining concepts satisfy the Theorem 2. The corresponding algorithm is omitted due to space limitation. Case *2)* $\exists\, \mathbb{U}^* \subseteq \mathbb{U}^c$, for any $U_d \in \mathbb{U}^*$, $I_d \subseteq \bigcup_{U_i \in \mathbb{U}^c_{\overline{U_d}}} I_i$, and $\exists U_b \in \mathbb{U}^*$, $t \in I_b \cap \left(\bigcup_{I_i \in \mathbb{I}^*_{\overline{I_b}}} I_i\right)$ and $t \notin \bigcup_{U_i \in (\mathbb{U}^c - \mathbb{U}^*)} I_i$. It means that there exists a set of concepts $\mathbb{U}_{x_1}, \mathbb{U}_{x_2}, \dots, \mathbb{U}_{x_n} \in \mathbb{U}^*$, making $t_1 \in \mathbb{I}_{x_1} \cap \mathbb{I}_{x_2}, \dots, t_{n-1} \in \mathbb{I}_{x_{n-1}} \cap \mathbb{I}_{x_n}, t_n \in \mathbb{I}_{x_n} \cap \mathbb{I}_{x_1}$ and $t_1, \dots, t_{n-1} \notin \bigcup_{U_i \in (\mathbb{U}^c - \mathbb{U}^*)} I_i$. We vividly call this case as the deadlock phenomenon. Eliminating any concepts $\mathbb{U}_{x_i}$ from $\left\{\mathbb{U}_{x_1}, \dots, \mathbb{U}_{x_n}\right\}$, it maybe make t_{i-1} and t_i become the special terms in $\mathbb{U}_{x_{i-1}}$ and $\mathbb{U}_{x_{i+1}}$, respectively. That is, $\left\{\mathbb{U}_{x_{i-1}}, \mathbb{U}_{x_{i+1}}\right\} \cup (\mathbb{U}^c - \mathbb{U}^*)$ satisfies the Theorem 2. Therefore, there does not exist an intuitive greedy strategy to maximize eliminate these concepts in $\mathbb{U}^*$ making the remaining concepts satisfy the Theorem 2, since these concepts in $\mathbb{U}^*$ are not independent of each other. For case *2)*, we design an approximate algorithm as illustrated in the Algorithm 3. First, we introduce a metric called concept weight.

Definition 6 (Concept Weight). Let $[U|\mathbb{U}_{\overline{U}}]$ be the concept list in $\mathbb{U}$, where U is any concept in $\mathbb{U}$, and $\mathbb{U}_{\overline{U}}$ is the remaining list. We write U^W as the concept weight of U over $\mathbb{U}$, and

$$U^W = \sum_{U_i \in \mathbb{U}_{\overline{U}}} |U \cap U_i|. \quad (4)$$

Assumption. Given a set of concept set $\mathbb{U}$ in a deadlock state, the higher the weight of the concept is, the greater contribution for unlocking over the whole concept set is.

The Algorithm 3 is based on the above assumption. Although in theory the assumption does not hold, in practical it is an acceptable tradeoff between efficiency and the number of chunks, and usually able to get the optimal solution, as illustrated in Fig. 5 and Fig. 6 of Section 6.

Algorithm 3 Eliminating redundant concepts

```
INPUT: $\mathbb{U}$ // a set of non-covered concept set
OUTPUT: $\mathbb{U}^c$
1: for each $U_i$ in $\mathbb{U}$ do
2:   if $I_i \subseteq \cup_{U_j \in \mathbb{U}^c_{\overline{U_i}}} I_j$ then
3:       Create $\mathbb{D}_i$ to record all the concept sets covering $U_i$;
4:       $\mathbb{U}^* = \mathbb{U}^* \cup U_i$;
5: $\mathbb{U}^c = \mathbb{U} - \mathbb{U}^*$;
6: for each $U_i$ in $\mathbb{U}^*$ do
7:     Create $U_i^W$ based on Equation 3;
8: while $\mathbb{U}^* \neq \emptyset$ do
9:     Sort the concept set in $\mathbb{U}^*$ according to the descending order of $U^W$.
10:    $\mathbb{U}^c = \mathbb{U}^c \cup top(\mathbb{U}^*)$; // $top(\mathbb{U}^*)$ is the first concept
                                            //in $\mathbb{U}^*$
11:    $\mathbb{U}^* = \mathbb{U}^* - top(\mathbb{U}^*)$;
12:    Modify $\mathbb{D}$ by deleting $top(\mathbb{U}^*)$. If exists $\mathbb{D}_i = \emptyset$,
         deleting the corresponding concept $U_i$ from $\mathbb{U}^*$;
13:return $\mathbb{U}^c$;
```

4 Differential Privacy for Statistical Queries

In the above EQI-partitioning mechanism, the association relationship among terms has been removed. However, there are two challenges: *1)* Standing in the view of data publishing, it is a successful solution since the real dataset is contained in the Cartesian Product formed by the related data chunks, but for authorized users, the query result with *k*-anonymity is not an acceptable trade-off; *2)* Even with hybrid cloud, where the query result could be reconstructed in private cloud, we consider the general case: assuming that the channel between user and hybrid cloud is not reliable, CSP could steal background knowledge (such wiretap channel), and infer data privacy by background knowledge attack or statistical attack. Therefore, Cocktail just adopts EQI-partitioning mechanism as the static data release policy, while in data query stage, Cocktail employs interactive differential privacy.

4.1 Differential Privacy Mechanism

Differential privacy, independent of an adversary's background knowledge, provides stronger privacy guarantees than traditional anonymity-preserving schemes (such as *k*-anonymity and its variants). Differential privacy in the interactive model is defined as follow.

Definition 7 (α-Differential Privacy). A privacy mechanism M gives α–differential privacy if for any neighbor data sets E_1 and E_2 differing on exactly one record, and for any possible sanitized file set $\tilde{E}$ satisfying M,

$$Pr[M(E_1) = \tilde{E}] \leq \exp(\alpha) \times Pr[M(E_2) = \tilde{E}] \quad (5)$$

where α specifies the degree of privacy offered.

The Definition 7 implies that the mechanism M always returns similar results on neighbor file sets. Dwork et al. [16] presented a general protocol to implement α-differential privacy utilizing the concept of sensitivity.

Definition 8 (Sensitivity). For any neighbor file sets E_1 and E_2, a query set $Q \in \mathbb{Q}$, the sensitivity Δs is the maximal $\mathcal{L}_1$ distance between the exact query results on E_1 and E_2,

$$\Delta s = \max_{E_1,E_2} \|Q(E_1) - Q(E_2)\|_1 \quad (6)$$

Theorem 3 [16]. For any query Q over a data set E, the mechanism M

$$M(Q,E) = Q(E) + Lap(\Delta s/\alpha) \quad (7)$$

gives α–differential privacy.

Let λ denote $\Delta s/\alpha$, $Lap(\Delta s/\alpha) = Lap(\lambda) = \frac{1}{2\lambda} \times exp(-|x|/\lambda)$. Here, $Q(E)$ is the exact query results over E, and $Lap(\Delta s/\alpha)$ denotes the corresponding noise. Allocation of the privacy budget α is beyond the scope of this work. Interested readers are referred to the detailed technical report [10] for privacy budget allocation.

4.2 Statistical Queries

In this paper, we mainly study three kinds of common statistical queries: counting query, linear query, and batch linear queries.

Counting query. Counting query is a base of many complex statistical queries. Considering the case that the candidate terms in one chunk, for example, for a counting query Q over E, returning $Q(E) + Lap(1/\alpha)$ maintains α-differential privacy since a counting query has a sensitivity 1. For the case of the candidate terms across multiple chunks, private cloud first maps original query Q into a series of sub-queries, then sends them into the corresponding VMs in the public cloud. These sub-queries are implemented respectively, and the intermediate results are being returned to the private cloud. In reduce step, these intermediate results are reconstructed, which is an intersect operation, and then appending $Lap(1/\alpha)$ maintains data privacy. Data query distribution and data reconstruction are elaborated further in Section 6.

Linear query. Linear query can be seen as the algebraic sum of limited counting queries. A linear query Q is in the form of a linear function over dataset E. Given a weight vector $\{w_1, w_2, \ldots, w_m\}$, the linear query Q returns the dot product between the weight vector and counting query vector, i.e.,

$$Q(E) = w_1 Q(t_1) + w_2 Q(t_2) + \cdots + w_m Q(t_m) \quad (8)$$

Based on the Laplace mechanism, the corresponding differential privacy mechanism $M(Q,E)$ is shown as follow.

$$M(Q,E) = Q(E) + \textstyle\sum_{i=1}^{m} Lap(\Delta s_i/\alpha) \quad (9)$$

Batch linear query. Considering a case that a batch of m linear queries $Q = \{Q_1, Q_2, \ldots, Q_n\}$, is submitted at the same time. The query set Q is thus represented by a weight matrix W with n rows and m columns. Each entry W_{ij} in W is the j-th coefficient for query Q_i on term t_j. Given $T = \{t_1, t_2, \ldots, t_m\}$, the query batch Q can be exactly answered by calculating:

$$Q(E) = WT = \left(\textstyle\sum_j W_{1j} Q(t_j), \ldots, \sum_j W_{nj} Q(t_j)\right) \quad (10)$$

Based on the Laplace mechanism, two solutions can be

TABLE 2
DATASET

	T_1	T_2	T_3	T_4
R_1	1	0	0	0
R_2	0	1	0	0
R_3	0	0	1	0
R_4	0	0	0	1

enforced α–differential privacy on a batch linear query.

A naive solution, called noise on queries (*NOQ*), is to process each query independently. Since the queries are linear queries, let the sensitivity of $Q(t_j)$ beΔs_j, the sensitivity of the query set Q is $\Delta s = max_j \sum_i |W_{ij}| \Delta s_j$, i.e., the highest column absolute sum [17]. Therefore, the differential privacy mechanism $M(Q,E)$ is shown.

$$\begin{aligned} M(Q,E) &= Q(E) + 2m\Delta s^2 \alpha^{-2} \\ &= Q(E) + 2m\ max_j \textstyle\sum_i W_{ij}{}^2 \Delta s_j^2\ \alpha^{-2} \end{aligned} \qquad (11)$$

Another solution, called noise on terms (*NOT*), is to process each counting query under differential privacy and combine them to answer the given linear counting queries. Since the sensitivity of $Q(t_j)$ is Δs_j, the expected squared error on the differential privacy mechanism can be denoted as follows.

$$M(Q,E) = Q(E) + 2\alpha^{-2}\left(\textstyle\sum_i \sum_j W_{ij}{}^2\ \Delta s_j^2\right) \qquad (12)$$

For example, consider the following query set $\{Q_1, Q_2, Q_3\}$ over Table 2.

$$\begin{aligned} Q_1 &= 2Q(T_1) + Q(T_2) + Q(T_3) \\ Q_2 &= Q(T_1) + 2Q(T_3) \\ Q_3 &= 2Q(T_2) + 2Q(T_3) + Q(T_4) \end{aligned}$$

Using *NOQ* as the solution, it incurs a sensitivity of 5 (e.g., a record of state $Q(T_3)$ affects Q_1 by 1, and Q_2 and Q_3 by 2 each). Thus, processing $\{Q_1, Q_2, Q_3\}$ directly incurs a noise variance of $50/\alpha^2$ for each query; While using *NOT* as the solution, The sensitivity of *NOD* remains 1, and it answers $\{Q_1, Q_2, Q_3\}$ with noise variance $\{12/\alpha^2, 10/\alpha^2, 18/\alpha^2\}$ respectively.

Note that, both of approaches to answer the batch linear queries could lead to sub-optimal result accuracy. For *NOD* solution, continuing the example above, one possible optimizing strategy is like this: constructing $Q_1' = Q(T_4)/3 + Q(T_2)$ and $Q_2' = 2Q(T_4)/3$, the query set $\{Q_1, Q_2, Q_3\}$ can be written as follows.

$$\begin{aligned} Q_1 &= Q_1' + 2Q(T_1) + Q(T_3) - Q_2'/2 \\ Q_2 &= Q(T_1) + 2Q(T_3) \\ Q_3 &= 2Q_1' + 2Q(T_3) + Q_2'/2 \end{aligned}$$

The sensitivity of the above is also 1, and it answers $\{Q_1, Q_2, Q_3\}$ with noise variance $\{12.5/\alpha^2, 10/\alpha^2, 16.5/\alpha^2\}$, which is less noise than previous strategy. As well for *NOQ* solution, when existing queries in the batch linear queries, which can be expressed as a linear combination of others, the sensitivity of queries could be significantly reduced as the rank of the matrix composed by the batch linear queries, i.e., the low-rank mechanism [18]. Due to an infinite solution space of optimization strategy, searching for the best one is an open problem, which is our future work.

5 TECHNOLOGY FOR COCKTAIL

In this section, we address the partitioning scalability issue for Cocktail. We propose a term-based overlay horizontal partitioning scheme, which is scalable and offers term lookup guarantee.

5.1 A Distributed Overlay-to-Underlay Mapping Scheme

To achieve efficient data querying, we exploit a distributed overlay-to-underlay mapping scheme. The core idea is to segregate the entire term-space into subspaces and assign dataset corresponding with each subspace to one or a set of physical nodes. The benefits of our mapping scheme are four-fold. First, it places similar records together in the same partition. Therefore, it reduces the false positive of the information quality of the published dataset. It greatly increases the probability that the local EQI in each cluster is the global EQI. Second, it provides an upper-bound on the number of partitions. Third, it helps Cocktail achieve the goal of data parallel processing (vertical partitioning, data querying). And fourth, it creates non-overlapping partitions.

Since data is retrieved by terms, we can classify all the data records in separate partitions, based on the presence or absence of terms in a record. At first, we create some partitioning clusters (P) over the term set based on the expected frequency of terms retrieval. Then we create a partitioning tree by associating P_i with i-th level in the tree and by expanding each i-th level node using the permutations of term-presence in each P_i. Leaf nodes in this tree represent the partitions. The mapping scheme is exemplified in Fig. 3, where the tree has been grown upto height three. The first, second and third level partitioning sets are $T_1 = \{a, c\}$; $T_2 = \{e\}$ and $T_3 = \{d\}$, respectively. For level 1 we get four parallel clusters $P_{11}(ac)$, $P_{12}(\bar{a}c)$, $P_{13}(a\bar{c})$ and $P_{14}(\overline{ac})$ based on the terms in P_1. In general, $2^{|P_i|}$ branches will leave a node at level i. Here, $|P_i|$ denotes the number of terms in P_i. In this partitioning strategy, tree height will grow with the volume of data set and the maximum tree height is restricted by the involved term set in the outsourced data set.

It is worth mentioning that the proposed overlay indexing architecture is a task distribution accelerator, which would not influence the real distribution of data set load across the network. Of course, the fatter the tree is, the

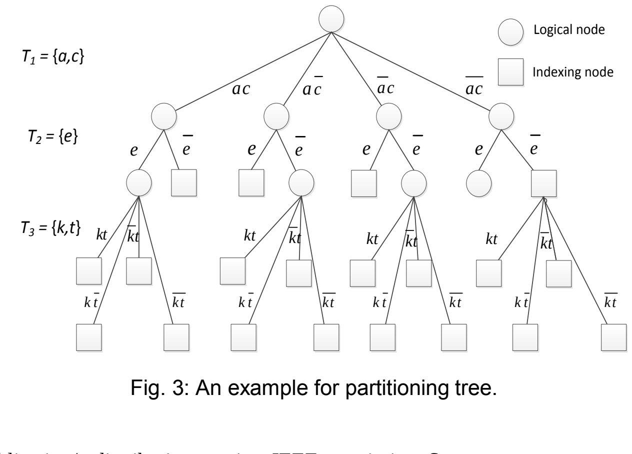


Fig. 3: An example for partitioning tree.

higher the degree of task parallelization. The tree is shorter, the higher task execution efficiency. To fulfill this requirement, we need to ensure that for any node the frequency of each combination leading to its children is roughly equal. For example, in Fig. 3 the joint frequencies of clusters $P_{11}(ac)$, $P_{12}(\bar{a}c)$, $P_{13}(a\bar{c})$ and $P_{14}(\overline{ac})$ should be similar. To this end, the partitioning set is generated by a three step process: *1)* a corpus of terms are parsed based on the previous statistic information of term frequencies, *2)* all possible combinations of terms upto a fixed length threshold (maybe 3 in our experience) are generated, and *3)* the term combination that produces the most balanced branching is selected. For example, we consider that the tree is grown upto height 2 with the terms $\{a, c, e\}$. Now, as data volume increases, we want to extend the tree height, we first compute the term query frequency of the dataset that is partitioned under the 8 nodes at height 2. Then we generate all possible character combinations (leaving out $\{a, c, e\}$) of length at most 3 and select the combination that produces the most balanced branching at height 3, which is $\{k, t\}$ in the figure. This is a trivial task. Fortunately, it can be precomputed by data domain experts.

In the underlay, data owner needs to upload data chunks to cloud file management system, e.g., Hadoop Distributed File System (HDFS), which further places the data chunks to the nodes across the cloud space rented by data owner. Specifically, HDFS has two kinds of nodes: *namenode* that maintains the meta information of the whole file system, including the *inode* for each chunk that records its attributes (such as chunk ID, modification time), and *datanode* that keeps the actual data chunks. The data chunks stored on the *datanode* are organized as blocks, each containing 64 MB by default. The locations of the blocks that belong to the same chunk are recorded by the *BlockInfo* array within the chunk's *inode*.

To protect sensitive subrecords from the public cloud, we use the privacy label to mark all of the private chunks. Similar to [6], we modified the Hadoop client and HDFS to ensure that private chunks are kept in private cloud. Specifically, we extended Hadoop's *INodeFile* class by adding a Boolean attribute, *priLabel*, where "true" denotes the target file is a private chunk, "false" denotes it is a public chunk. When data owner contacts the namenode on the private cloud to build an *inode* for a chunk with privacy label, the namenode would allocate a data block from the private cloud to the chunk.

In order to provide effective data query, we have to map the nodes in the proposed overlay tree to the real data chunk storage topology. The interface is located in the leaf nodes of the overlay tree. Each leaf node in the tree documents the *inodes* of the whole data chunks in the path from the root to the leaf.

5.2 Further Partitioning

Although the horizontal partitioning has many benefits, it also introduces the false positives. Considering a target term t_i which appears in more than k records, that is, t_i satisfies k *-anonymity* in the global dataset, unfortunately, these records containing t_i are divided into several clusters by horizontal partitioning, which makes the number of t_i to be less than k in each one. Therefore, false positive occurs. For example in Figure 2(b), the term set $\{Dior, amazon\}$ is as sensitive information held in private cloud, since it does not satisfy anonymous threshold in clusters P_1 and P_2, respectively. However, we can observe that, given $k = 3, m = 2$, the term set $\{Dior, amazon\}$ satisfy k^m*-anonymity* in the original dataset (see Figure 2(a)). According to our goals, subrecords containing term set $\{Dior, amazon\}$ should be held in public cloud.

To move as much data to public cloud as possible, focusing on all the chunks in private cloud, we introduce the notion of aggregated chunk that allows to joint different clusters for further refining terms that satisfies k^m-*anonymity* among the joint clusters. Combining with the partitioning tree, we use the bottom-up recursive refinement strategy, where aggregated clusters can be recursively generated. The core idea is that, one of the lowest-level aggregated cluster can be formed by refining private chunks (located in leaf nodes) from the same parent node, while the higher-level aggregated cluster can be generated by two parts: *1)* combining the lower-level aggregated clusters, and *2)* refining private chunks from the same ancestor node. Specifically, we introduce a type of node called aggregated node in partitioning tree to represent the aggregated cluster. To form one aggregated cluster S_{ij} at height i, *1)* when the child nodes of P_{ij} node is not leaf nodes and the sum of data volume among lower-level aggregated cluster is still less than the predefined horizontal partitioning threshold, we would combine these lower-level aggregated clusters. Lemma 6 explains the privacy guarantee for multichannel data merging. Then, *2)*

	Public chunks		Private chunk	Shared cluster
Cluster P1	{iphone 5s, levis, Dell}		{Starbucks}	{Dior, amazon}
	{levis, Dell}	{google glass}	{Starbucks}	{Dior}
	{iphone 5s, Dell}	{google glass}		{amazon}
	{iphone 5s, levis}		{Starbucks}	
	{iphone 5s, levis, Dell}	{google glass}		
Cluster P2	{Dell, news}		{croissant, Celine Dion}	
	{Dell, python}			{Dior, amazon}
	{python, news, Dell}		{Celine Dion}	
	{python, news}		{croissant}	
	{python, news, Dell}			{Dior, amazon}

Fig. 4: An example for partitioning tree.

we refine the term intersection among any two private chunks which are covered by the logical node P_{ij}. The refining operation is similar to that of EQI-partitioning strategy described in Section III and is omitted for brevity. Fig. 4 illustrates an example of an aggregated cluster, generated by refining from private chunks of Fig. 2(b).

Lemma 5. The lowest level shared cluster is k^m-*anonymous*.

Proof: Based on the generation mechanism of the lowest level shared cluster described in above, we know that, one of the lowest level shared cluster can be formed by a refining operation for private chunks (located in leaf nodes) from the same parent node. We see data records from these private chunks as the target data set, while the refining operation is to refine data chunks that satisfy k^m-*anonymity* by implementing the EQI-partitioning strategy on the target data set. According to the Lemma 3, EQI-partitioning strategy guarantees k^m-*anonymity*. Therefore, the Lemma 5 is proved. □

Lemma 6. Given any aggregated cluster set $\mathbb{S}\{S_{ij}, S_{il}, \dots\}$ located in height i of the partitioning tree, and for any $S_{i*} \in \mathbb{S}$, S_{i*} satisfies k^m-*anonymity*, the combining part of the higher-level aggregated cluster S_{h*} is k^m-*anonymous*.

Proof: See Appendix D. □

Theorem 4. The aggregated clusters satisfy k^m-*anonymity*.

Proof: The proof follows easily from the Lemma 5 and 6, and is omitted for brevity. □

Next, we consider a theoretical problem: How far aggregated cluster can eliminate false positives caused by horizontal partitioning.

Theorem 5. Given k and m as the anonymous parameters, the upper bound of false positives caused by the proposed mechanism is $\sum_{i=1}^{|T|}\left(\left\lceil\frac{\lfloor|q(t_i)|/k\rfloor}{\lceil|q(t_i)|/k\rceil}\right\rceil (k-1)\right)$, where $q(t_i)$ denotes the subrecords containing term t_i, $\lceil * \rceil$ and $\lfloor * \rfloor$ are the ceil operation and floor operation, respectively.

Proof: See Appendix E. □

5.3 Data Querying

After uploading data chunks to HDFS, data owner needs to submit a series of operation functions, particularly the mapper and the reducer. The mapper works with a local inverted index file and an input query to search the file *inodes* matched with the target term. The reducer receives the output of a map task as its input. It combines the intermediate results from mapping tasks of the same query job, and adds the noise corresponding to the authorized user. Eventually, it sends the result back to the user.

Specifically, when receiving a query job from an authorized user, the Cocktail first calls *JobTracker* that breaks the job into tasks. Each task is a *TaskInProgress* object. Then it creates a task execution scheduling that assembles *TaskProgress* objects into one *JobInProgress* queue according to the relationship of the tasks. The current Hadoop does not offer privacy-preserving data query. Our framework provides such capability. The details are given below. For Hadoop, we use one node to run the *JobTracker* processes which is responsible for parsing user query and distributing tasks. Specifically, to protect data privacy, *JobTracker* checks to see whether the query contains the terms in private chunks. If such private terms exist in the query, the target query would be divided into two sub-queries, where the one that just contains private terms would search the private chunks in private cloud. Next, for the sub-query without private terms, *JobTracker* further separate it into a series of map tasks based on the partitioning tree, which mainly contains three steps: *1)* scans the overlay tree with breadth-first traversal, from the root of the tree down a layer to start; *2)* compares the terms in the sub-query with each partitioning set P_i. If some terms match, finds the corresponding nodes and views them as the roots of subtrees, then jumps to step *1)* until these nodes are leaf nodes; *3)* creates map tasks over the whole matched leaf nodes. Then, for each map task, it executes an instance of the mapper object to search the corresponding data chunk *inodes*. On the reduction phase, it also contains three steps: *1)* reduces all the map tasks produced by the sub-query without private terms. Since all of the partitions are not overlapping, the reduce operation just merges all the output sets from the map tasks; *2)* combines the result sets from two sub-queries in private cloud. The combine operation obtains the intersection between the two result set. Note that, instead of transmitting the data itself, the intermediate result just contains subrecords' IDs; *3)* adds the corresponding noise to the result, and sends it back to the authorized user.

6 Evaluation

6.1 Experimental Setting

Environment. We built a public cloud testbed including 3 nodes connected by WS-C3750X-24T-S switch. Each node has Xeon E5506 core processors running at 2.13GHz CPUs, 16G dual-channel 1333GHz memory, and 500GB 7200 RPM disk. We used Hive version 0.7.1 running on Hadoop version 0.20.203. We used two extra nodes as private cloud that has the same configuration with the public cloud, where one node runs the *namenode* and the *JobTracker* processes. We configured Hadoop to run 8 map tasks and 8 reduce tasks per node (a total of 24 map slots and 24 reduce slots in the public cloud testbed). The operating system was Ubuntu 11.04.

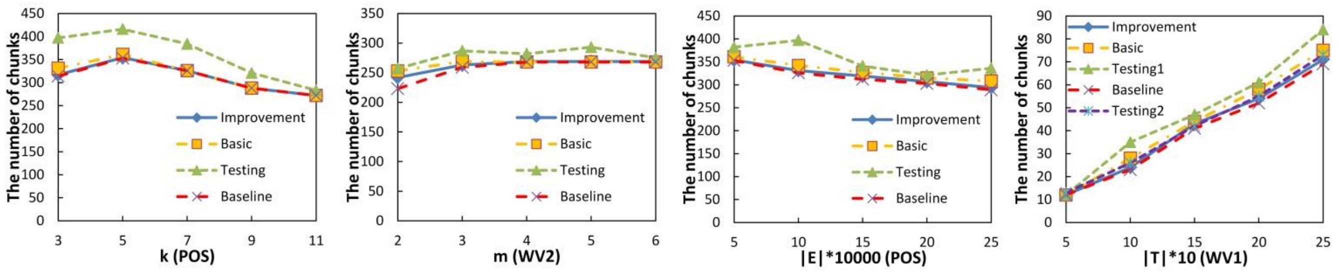
Datasets. We use three real datasets POS, WV1 and WV2 described in Table 3, which were introduced in [33].

TABLE 3
DESCRIPTIONS OF DATASETS

Dataset	$\|E\|$	$\|T\|$	max rec. size	avg rec. size
POS	515,597	1,657	164	6.5
WV1	59,602	497	267	2.5
WV2	77,512	3,340	161	5.0

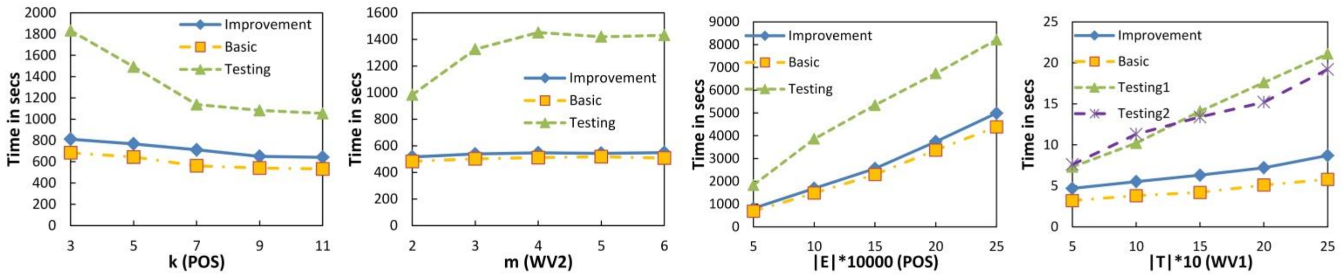
6.2 Experimental Result

The first experiment (Fig. 5(a-d)) uses all datasets to in-



(a) (b) (c) (d)

Fig. 5: Effect of dataset characteristics on the data quality of the algorithms.



(a) (b) (c) (d)

Fig. 6: Effect of dataset characteristics on the performance of the algorithms.

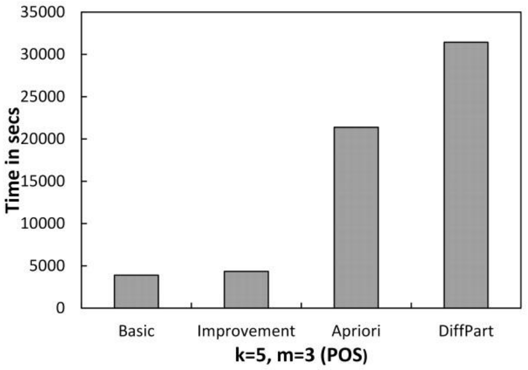


Fig. 7: Comparison with other methods.

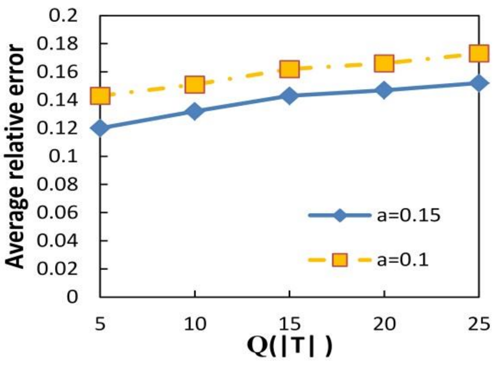


Fig. 8: Average relative error vs. privacy budget.

vestigate how the result of data partition is affected, when the dataset characteristics. The number of data partition is a very important metric to evaluate candidate data partition methods. On the premise of meeting the anonymity, too many chunks will not provide higher privacy guarantee, instead, it will greatly increase the overhead of data reconstruction. Fig. 5(a), using the POS dataset, shows how data partition quality is affected with the increase of the anonymous parameter k, after fixing the remaining parameters to $m = 3, |E| = 50{,}000$. As shown in Fig. 5(a), *1)* the number of data chunks renders a trend of decrease after the first increase along with the increase of k in all candidate data partition methods, the reason is that, when k increases within a certain scope, many term combinations first are divided into several smaller chunks, however, with the continuous increase of k, many chunks with single term do not satisfy the requirement of anonymity so that joins the private chunk; *2)* Compared with the EQI-testing scheme (Testing, for short) proposed in [8], Our schemes containing the basic scheme and the improvement scheme have a better data quality, where the baseline is the optimal value satisfying the anonymity, which is obtained by offline. Fig. 5(b), using the WV2, shows how data quality changes as the power of the guarantee, expressed by the m parameter, grows, after fixing $k = 5, |E| = 50{,}000$. The results show that, when m increases within a certain scope, it has a significant impact on the data quality, but as the continuous increase of m, it does not work. This is associated with the characteristics of high-dimensional sparse data. We note that there is no such keyword combination with size greater than 5 in WV2. Fig. 5(c) shows that the number of data chunks has a downward trend along with the increase of $|E|$ in POS, where $k = 5, m = 3$. Fig. 5(d) depict the effect of term domain after fixing $k = 5, m = 3$. We use WV1 with the average record size 2.5, which is the sparsest dataset. When we keep the dataset size and the anonymous parameters constant and we only increase the term domain, the number of data chunks significantly increases. By repeating the experiment several times, we noticed that the Testing scheme is associated with the order of terms. We do not report all detailed results due to space limitations.

Fig. 6(a-d) illustrate the corresponding performance overhead of Fig. 5. The result shows that, *1)* our schemes containing the basic scheme and the improvement scheme are hardly affected by the value of the anonymous parameters (k, m), and at the same time they scale linearly to the dataset and the term domain size; *2)* from the perspective of performance overhead, the basic scheme was always the least, improved scheme followed by, and finally the Testing scheme; *3)* focusing on Fig. 6(d), we show two very different results generated by the Testing scheme, which just change the order of terms. Fig. 7 on POS demonstrates the performance overhead of our schemes compared to other state-of-the-art methods (such as Apriori [13], DiffPart[10]),where $k = 5, m = 3$. For the DiffPart algorithm we used privacy budgets 0.5. In summary, the experiments on real datasets illustrate the validity and practicality of our scheme. Moreover, using POS, Fig. 8 reports the average relative error with different values of the privacy budget α at different dataset size, which decreases with the increase of α.

7 Related Work

Privacy protection in set-valued data has been acknowledged as an important problem in the data privacy security literature. Y. He et al. [7] propose an efficient algorithm for classical *k-anonymity* in a set-value context. [8, 23] introduce the k^m*-anonymity* guarantee, which is used and extended in this paper. The authors provide algorithms for anonymizing the data that employ both local and global recoding. [24] studies multi-relational k-anonymity, which still relies on generalization as the anonymization procedure. [25] demonstrates that the information loss, when providing *k-anonymity*, can be reduced by using natural domain generalization hierarchies. To address the problem of attribute disclosure, where a person can be associated with a sensitive value, the concept of ℓ-diversity [14] was introduced. Anatomy [12] provides ℓ-diversity which does not generalize or suppress the data, but instead it disassociates them by publishing them separately. V. Ciriani et al. [27] propose using fragmentation lattice to split the connection among data attributes, which builds self-contained attributes lattice sets. M. Terrovitis et al. [13] propose an anonymous strategy for set-valued data based on disassociation, which extends the idea of Anatomy [12] to provide protection against identity disclosure by separating terms of the original data. In perturbation-based approaches, differential privacy has been gaining considerable attention. Xiao et al. [28] propose a two-step algorithm for relational data, which first issues queries for possible combination of attribute values to the PINQ interface and then produces a generalized output using the perturbed dataset returned by PINQ. Apparently, this approach is computationally expensive in the context of set-valued data due to the high dimensionality. To minimize the error of linear queries under differential privacy requirements, several methods try to build a synopsis of the original data set, such as wavelets [30] and hierarchical trees [10].

On another hand, MapReduce [31] framework with the advantages of large-scale distributed computation has been successfully applied to the cloud. However, the data privacy-preserving approaches above mentioned are hard to extend to the MapReduce. Airavat [32] ensures mandatory access control and differential privacy when MapReduce operations are performed on sensitive data. Different from Airavat, Sedic [6] aims at protecting sensitive data from the public cloud. Our approach and Sedic share the idea that achieves a security mechanism designed with MapReduce framework on the hybrid-cloud platform.

8 Conclusion and Future Work

In this paper, we present Cocktail, a privacy-aware set-valued data publishing system. In data publishing stage, we propose an EQI-partitioning strategy that disassociates record terms that participate in identifying combinations. This way the cloud server cannot associate with high probability a record with rare term combinations. We strictly prove the privacy guarantee of our mechanism. In data querying stage, we employ differential privacy strategy to resist privacy breaches from counting query, linear query, and batch linear query. In addition, our extensive experiments demonstrate the validity and practicality of the proposed scheme. As mentioned earlier, due to an infinite solution space of optimization strategy, batch linear query with the minimum noise remains an open question. In future work, we would focus on exploring an approximation algorithm that not only protects data privacy but also reduces the loss of information.

Acknowledgment

This work was partially supported by the National Basic Research Program of China (973 Program) under grant No. 2011CB302605, the National High Technology Research and Development Program of China (863 Program) under grant No. 2011AA010705, the National Science Foundation of China (NSF) under grants No. 61100188 and No. 61173144.

Appendix

A. Proof of Lemma 2

We prove it by contradiction. Given a concept $U(I,\ q(I),\ count)$ over $E(R, T, F)$, where $I \neq \emptyset$ and $count \geq k$, we assume that, there exists a concept $U_i(I_i,\ q(I_i),\ count_i)$ with $I_i \subseteq I$ and $\text{count}_\text{i} < \text{k}$. We discuss it by two cases. 1) When $I_i = I$, we have $count = count_i < k$, it forms the contradiction with the hypothesis. 2) When $I_i \subset I$, there exists a term set $I_j \in I - I_i$. Since $|q(I_i)| < k$, $count = |q(I)| = |q(I_i \cup I_j)| = |q(I_i) \cap q(I_j)| \leq min(|q(I_i)|, |q(I_j)|) < k$, it forms the contradiction with the hypothesis. □

B. Proof of Theorem 1

The Min-EP can be reduced to the NP-complete problem of minimum hypergraph coloring, which is formulated as follows: given a hypergraph $G(V,E)$, determine a minimum coloring of G, that is, assign to each vertex in V a color such that adjacent vertices have different colors, and the number of colors is minimized. The correspondence between the Min-EP and the minimum vertex coloring is given below. Given a set of concept set $\mathbb{U}$ satisfying k^m-*anonymity* over a target cluster $P(R,T,F)$, where $\mathbb{I}$ is the corresponding intension set in $\mathbb{U}$, $\mathbb{I}^c$ is the complementary set of $\mathbb{I}$ over $\{T^1, T^2, \dots, T^m\}$ (T^i denotes the i-dimensional Cartesian product over T). A vertex v_i of hypergraph G translates to an intensive I_i, and any edge $e_i(v_{i_1}, \dots, v_{i_n})$ in G translates to an element in $\mathbb{I}^c$. A partition strategy $C = \{C_1, \dots, C_m\}$ corresponds to m colors over T. Hence, the Min-EP problem based on EQI-identifying scheme is NP-complete. □

C. PROOF OF THEOREM 2

We first proof the necessity by using a counterexample. Considering the following example, there exists a set of non-covered concept set $\mathbb{U}^{\#}(U_1, U_2, U_3, U_4)$ over a target cluster $P(R, \{a, b, \dots, i\}, F)$, where $I_1(a,b,c)$, $I_2(d,e,f)$, $I_3(g,h,i)$, and $I_4(a,d,g,h)$. We observed that, $I_4 \in I_1 \cup I_2 \cup I_3$. Based on the greedy partition strategy, we get a partition set $\mathbb{C} = \{C_1(a,d,g,h), C_2(b,c), C_3(e,f), C_4(i)\}$ with $|\mathbb{C}| = 4$. While we can construct another partition set $\mathbb{C}^* = \{C_1(a,b,c),\ C_2(d,e,f),\ C_3(g,h,i)\}$ with $|\mathbb{C}^*| = 3$. It is easy to see that $\mathbb{C}$ and $\mathbb{C}^*$ satisfy the requirements of anonymity, and $|\mathbb{C}^*|$ is smaller than $|\mathbb{C}|$. Thus, the necessity is proved. Next, we proof the sufficient by contradiction. For any $U_d \in \mathbb{U}^{\#}$, $I_d \notin \bigcup_{U_i \in \mathbb{U}^{\#}_{\overline{U_d}}} I_i$ means that $\exists t_i \in I_d$ and $t_i \notin \bigcup_{U_i \in \mathbb{U}^{\#}_{\overline{U_d}}} I_i$. That is, for any U_d in $\mathbb{U}^{\#}$, there are at least one term which just exists in I_d. Based on the greedy partition strategy, we get a partition set $\mathbb{C}$ with $|\mathbb{C}| = |\mathbb{U}^{\#}|$. We assume that there exists a partition strategy $\mathbb{C}^*$ with $|\mathbb{C}^*| < |\mathbb{U}^{\#}|$. Based on the pigeonhole principle, it makes at least two terms that just exist in their respective concepts be divided into the same partition. Therefore, the partition strategy $\mathbb{C}^*$ does not satisfy k^m-*anonymity*, since the dimension size of each concept in $\mathbb{U}^{\#}$ is maximal. This completes the proof. □

D. PROOF OF LEMMA 6

Based on the partitioning tree, we know that P_{h*} is the ancestor node of $\mathbb{P}\{P_{ij}, P_{il}, \dots\}$. We use $\mathbb{R}\{R_{ij}, R_{il}, \dots\}$ and $\mathbb{T}\{T_{ij}, T_{il}, \dots\}$ to denote the corresponding record sets and term sets of $\mathbb{P}$, respectively. We get the fact that, for any $R_{im}, R_{in} \in \mathbb{R}$, $R_{im} \cap R_{in} = \emptyset$, since the partitioning tree creates non-overlapping partitions. We prove Lemma 6 by contradiction. Assuming that, there exists term combination T_x which makes S_{h*} does not satisfy k^m-*anonymity*. We discuss it by three cases. For case one, T_x exists in one low-lever aggregated cluster (S_{il}, for example). i.e., $T_x \subseteq T_{il}$, and for any other term sets $T_{i\bar{l}}$, $T_x \cap T_{i\bar{l}} = \emptyset$. This is a contradiction with the prerequisite that S_{il} satisfies k^m-*anonymity*. For case two, T_x exists in multiple low-lever aggregated cluster $\mathbb{S}^*$ ($\mathbb{S}^* \subseteq \mathbb{S}$, $|\mathbb{S}^*| \geq 2$. $\mathbb{T}^*$, $\mathbb{R}^*$ are the corresponding term sets and record sets, respectively). i.e., for any $T_{i*} \in \mathbb{T}^*$, $T_x \subseteq T_{i*}$. Due to the case one meets k^m-*anonymity*, and for any two $R_{im}, R_{in} \in \mathbb{R}^*$, $R_{im} \cap R_{in} = \emptyset$, we have the number of records containing T_x is more than $|\mathbb{S}^*|k$, which forms the contradiction with the hypothesis. For case three, given $\mathbb{T}^* \in \mathbb{T}$, for any $T_j \in \mathbb{T}^*$, $T_x - T_j \neq \emptyset$ and $T_x - \bigcup_{i=1}^{|\mathbb{T}^*|} T_i = \emptyset$. Due to any two $R_{im}, R_{in} \in \mathbb{R}^*$, $R_{im} \cap R_{in} = \emptyset$, we get empty set containing T_x, which forms the contradiction with the hypothesis. The Lemma 6 is proved. □

E. PROOF OF THEOREM 5

For any $t_i \in T$, there are two cases that should be discussed. For case one: $|q(t_i)| < k$, the subrecords containing t_i should be held in private cloud, since the number of these subrecords is less than the anonymous threshold k. i.e., this case cannot cause false positive. It can be expressed as $\left\lceil \frac{\lfloor |q(t_i)|/k \rfloor}{\lceil |q(t_i)|/k \rceil} \right\rceil (k-1)$, since $\left\lceil \frac{\lfloor |q(t_i)|/k \rfloor}{\lceil |q(t_i)|/k \rceil} \right\rceil = 0$; For case two: $|q(t_i)| \geq k$, we prove the upper bound of false positives by contradiction. We assume that, in private chunks, the number of subrecords containing t_i is more than $k-1$. Based on EQI-partitioning, we know that there is no such subrecords $q(t_i)$ in one private chunk, since it satisfies the anonymous threshold k. Based on aggregated-cluster-refining, we know that when subrecords $q(t_i)$ are located in multiple private chunks, it would be recursively refined into aggregated cluster. Therefore, the assumption does not hold. In other words, in private chunks, the subrecords containing t_i are less than k, which can be expressed as $\left\lceil \frac{\lfloor |q(t_i)|/k \rfloor}{\lceil |q(t_i)|/k \rceil} \right\rceil (k-1)$, since $\left\lceil \frac{\lfloor |q(t_i)|/k \rfloor}{\lceil |q(t_i)|/k \rceil} \right\rceil = 1$. □

REFERENCES

[1] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Comput. Commun. Rev., vol. 39, no. 1, pp. 50–55, 2009.

[2] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. Advances in Cryptology - EUROCRYPT 2010, volume 6110 of Lecture Notes in Computer Science, pages 24–43. Springer Berlin / Heidelberg, 2010.

[3] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Overencryption: Management of Access Control Evolution on Outsourced Data. In Proc. of VLDB: 123-134. September 2007.

[4] S. Yu, C. Wang, K. Ren , and Wenjing Lou. Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing. In Proc. of INFOCOM, 2010.

[5] T. Jung, X. Li, Z. Wan and M. Wan. Privacy Preserving Cloud Data Access With Multi-Authorities. In Proc. of INFOCOM, 2013.

[6] K. Zhang, X. Zhou, Y. Chen, X. Wang, and Y. Ruan. Sedic: Privacy-Aware Data Intensive Computing on Hybrid Clouds. In Proc. of CCS, 2011.

[7] Y. He and J. F. Naughton. Anonymization of set-valued data via top-down, local generalization. PVLDB, 2(1):934-945, 2009.

[8] M. Terrovitis, N. Mamoulis, and P. Kalnis. Privacy-preserving anonymization of set-valued data. PVLDB, 1(1):115-125, 2008.

[9] J. Cao, P. Karras, C. Raissi, and K.-L. Tan. ρ-uncertainty: inference-proof transaction anonymization. PVLDB, 3(1-2):1033-

1044, 2010.

[10] R. Chen, M. Noman, B. C. Fung, B. C. Desai, and L. Xiong. Publishing set-valued data via differential privacy. PVLDB, 4(11): 1087-1098, 2011.

[11] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In Proc. of WWW, pp. 171-180, 2009.

[12] X. Xiao and Y. Tao. Anatomy: simple and effective privacy preservation. In Proc. of VLDB, pp. 139-150, 2006.

[13] M. Terrovitis, J. Liagouris, N. Mamoulis, and S. Skiadopoulos. Privacy Preservation by Disassociation. In Proc. of VLDB, pp. 944-955, 2012.

[14] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In Proc. 22nd Intnl. Conf. Data Engg. In Proc. of ICDE, page 24, 2006.

[15] Z. Zhou, H. Zhang, X. Du, P. Li and X. Yu. Prometheus: Privacy-Aware Data Retrieval on Hybrid Clouds. In Proc. of INFOCOM, 2013.

[16] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In Theory of Cryptography Conference, 2006.

[17] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In PODS, pages 123–134, 2010.

[18] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, Z. Hao. LowRank Mechanism: Optimizing Batch Queries under Differential Privacy. In Proc. of VLDB, pp. 1352-1463, 2012.

[19] P. Samarati. Protecting respondents' identities in microdata release. TKDE, 13(6):1010-1027, 2001.

[20] L. Sweeney. *k-anonymity*: a model for protecting privacy. IJUFKS, 10(5):557-570, 2002.

[21] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In SIGMOD, pp. 49-60, 2005.

[22] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In Proc. of ICDE, pp. 25, 2006.

[23] M. Terrovitis, N. Mamoulis, and P. Kalnis. Local and global recoding methods for anonymizing set-valued data. VLDB Journal, 20(1):83-106, 2010.

[24] M. Nergiz, C. Clifton, and A. Nergiz. Multi-relational *k-anonymity*. In Proc. of ICDE, pp. 1417-1421, 2007.

[25] S. De Capitani di Vimercati et al.. Fragments and Loose Associations: Respecting Privacy in Data Publishing. In Proc. of VLDB:1370-1381. 2010.

[26] N. Li, T. Li, S. Venkatasubramanian, "t-Closeness: Privacy Beyond k-Anonymity and l-Diversity," In Proc. of ICDE'07, April 15-20, 2007.

[27] V. Ciriani et al., Fragmentation Design for Efficient Query Execution over Sensitive Distributed Databases, in Proc. of ICDCS'09, Montreal, Quebec, Canada, June 22-26, 2009.

[28] Y. Xiao, L. Xiong, and C. Yuan. Differentially private data release through multidimensional partitioning. In Proc. of VLDB workshop on SDM, 2010.

[29] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proc. 2000 ACM-SIGMOD Int. Conf. on Management of Data (SIGMOD'00), Dallas, TX, May 2000.

[30] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In Proc. of ICDE, pages 225-236, 2010.

[31] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. Commun. of ACM, 51(1):107-113, January 2008.

[32] I. Roy, S. T. V. Setty, A. Kilzer, V. Shmatikov, and E. Witchel. Airavat: Security and privacy for mapreduce. In Proc. of NSDI, pages 297-312. USENIX Association, 2010.

[33] Z. Zheng, R. Kohavi, and L. Mason. Real world performance of association rule algorithms. In Proc. of KDD, pp. 401-406, 2001.

[34] J. Xiong, X. Liu, Z. Yao, J. Ma, Q. Li, K. Geng and P. S. Chen, "A Secure Data Self-Destructing Scheme in Cloud Computing," IEEE Transactions on Cloud Computing, Volume 2, Issue 4, pp. 448-458, 2014.

[35] K. Xue, P. Hong, "A Dynamic Secure Group Sharing Framework in Public Cloud Computing," IEEE Transactions on Cloud Computing, Volume 2, Issue 4, pp. 459-470, 2014.

Hongli Zhang received her B.S. degree in Computer Science from Sichuan University, Chengdu, China in 1994, and her Ph.D. degree in Computer Science from Harbin Institute of Technology (HIT), Harbin, China in 1999. She is a professor in Computer Science and Technology Department of HIT and the vice director of National Computer Information Content Security Key Laboratory. Her research interests include network and information security, network measurement and modeling, and parallel processing.

Zhigang Zhou received the B.S. and M.S. degree in Computer Science from Dalian Jiaotong University, Dalian, China, from 2004 to 2011. From 2011 to now, he has been work as a Ph.D. candidate in Department of Computer Science and Technology at Harbin Institute of Technology (HIT), Harbin, China. His research interests include Cloud computing and security.

Lin Ye received the B.S., M.S., and Ph.D. degrees in Computer Science from Harbin Institute of Technology (HIT), Harbin, China, from 2000 to 2011. In 2012, he was a visiting scholar at Temple University, Philadelphia, PA, USA. He is currently a lecturer in School of Computer Science and Technology in HIT. His research interests include Peer-to-Peer measurement, cloud computing and information security.

Xiaojiang (James) Du is currently an associate professor in the Department of Computer and Information Sciences at Temple University. Dr. Du received his B.E. degree from Tsinghua University, China in 1996, and his M.S. and Ph.D. degrees from the University of Maryland, College Park in 2002 and 2003, respectively, all in Electrical Engineering. His research interests are security, systems, wireless networks and computer networks. Dr. Du has published over 140 journal and conference papers in these areas, and has been awarded more than $5M research grants from the US National Science Foundation and Army Research Office. He serves on the editorial boards of four international journals.