A Theoretically-Sound Accuracy/Privacy-Constrained Framework for Computing Privacy Preserving Data Cubes in OLAP Environments

Alfredo Cuzzocrea and Domenico Saccà

ICAR-CNR and University of Calabria, Italy
{cuzzocrea,sacca}si.deis.unical.it

Abstract. State-of-the-art *privacy preserving OLAP* approaches lack of *strong theoretical bases* that provide solid foundations to them. In other words, there is not a theory underlying such approaches, but rather, an *algorithmic vision* of the problem. A class of methods that clearly confirm to us the trend above is represented by the so-called *perturbation-based techniques*, which propose to alter the target data cube cell-by-cell to gain privacy preserving query processing. This approach exposes us to clear limits, whose lack of *extendibility* and *scalability* are only the tip of an enormous iceberg. With the aim of fulfilling this critical drawback, in this paper we propose and experimentally assess a *theoretically-sound accuracy/privacy-constrained framework for computing privacy preserving data cubes in OLAP environments*. The benefits deriving from our proposed framework are two-fold. First, we provide and meaningfully exploit *solid theoretical foundations* to the privacy preserving OLAP problem that pursue the idea of obtaining privacy preserving data cubes via *balancing accuracy and privacy of cubes* by means of *flexible sampling methods*. Second, we ensure the *efficiency* and the *scalability* of the proposed approach, as confirmed to us by our experimental results, thanks to the idea of leaving the algorithmic vision of the privacy preserving OLAP problem.

1 Introduction

Given a multidimensional range R of a data cube A, an *aggregate pattern* over R is defined as an aggregate value extracted from R that is able of providing a "description" of data stored in R. In order to capture the privacy of aggregate patterns, in this paper we introduce a *novel notion of privacy OLAP*. According to this novel notion, given a data cube A *the privacy preservation of A is modeled in terms of the privacy preservation of aggregate patterns defined on multidimensional data stored in A*. Therefore, we say that a data cube A is privacy preserving iff aggregate patterns extracted from A are privacy preserving. Contrary to our innovative privacy OLAP notion above, previous privacy preserving OLAP proposals totally neglect this even-relevant theoretical aspect, and, inspired by well-established techniques that focus on the privacy preservation of relational tuples [18,13], mostly focus on the privacy preservation of data cells (e.g., [17]) accordingly. Despite this, OLAP deals with aggregate data, and neglects individual information. Therefore, it makes more sense to deal with the privacy preservation of aggregate patterns rather than the privacy preservation of data cube cells.

Given a multidimensional range R of a data cube A, AVG(R), i.e. the average value of data cells in R, is the simplest aggregate pattern one could think about. It should be noted that this pattern could be inferred with a high degree of accuracy starting from (i) the knowledge about the summation of items in R, SUM(R) (which could be public for some reason – e.g., the total amount of salaries of a certain corporate department) and (ii) the estimation of the number of items in R, COUNT(R) (which, even if not disclosed to external users, could be easily estimated – e.g., the total number of employees of the department). In turn, basic aggregate patterns like AVG can be meaningfully combined to progressively discover (i) more complex aggregate patterns (e.g., [11] describes a possible methodology for deriving complex OLAP aggregates from elementary ones) to be exploited for trend analysis of sensitive aggregations and prediction purposes, or, contrary to this, (ii) aggregations of coarser hierarchical data cube levels until the privacy of individual data cells is breached. From the latter amenity, it follows that results of [17], which conventionally focuses on the privacy preservation of data cube cells, are covered by our innovative privacy OLAP notion.

Inspired by these considerations, in this paper we propose an innovative framework based on *flexible sampling-based data cube compression techniques for computing privacy preserving OLAP aggregations on data cubes while allowing approximate answers to be efficiently evaluated over such aggregations*. This framework addresses an application scenario where, given a multidimensional data cube A stored in a *producer* Data Warehouse server, a collection of multidimensional portions of A defined by a given (range) *query-workload* QWL of interest must be published online for *consumer* OLAP client applications. Moreover, after published, the collection of multidimensional portions is no longer connected to the Data Warehouse server, and updates are handled from the scratch at each new online data delivery. The query-workload QWL is cooperatively determined by the Data Warehouse server and OLAP client applications, mostly depending on OLAP analysis goals of client applications, and other parameters such as business processes and requirements, frequency of accesses, and locality. OLAP client applications wish for retrieving summarized knowledge from A via adopting a *complex multi-resolution query model* whose components are (i) queries of QWL and, for each query Q of QWL, (ii) *sub-queries* of Q (i.e., in a multi-resolution fashion). To this end, for each query Q of QWL, an *accuracy grid* $\mathcal{G}(Q)$, whose cells model sub-queries of interest, is defined. While aggregations of (authorized) queries and (authorized) sub-queries in QWL are disclosed to OLAP client applications, it must be avoided that, by meaningfully combining aggregate patterns extracted from multidimensional ranges associated to queries and sub-queries in QWL, malicious users could infer sensitive knowledge about other multidimensional portions of A that, due to privacy reasons, are hidden to unauthorized users. Furthermore, in our reference application scenario, target data cubes are also massive in size, so that data compression techniques are needed in order to efficiently evaluate queries, yet introducing *approximate answers* having a certain *degree of approximation* that, however, is perfectly tolerable for OLAP analysis goals [5]. In our proposal, the described application scenario with accuracy and privacy features is accomplished by means of the so-called *accuracy/privacy contract*, which determines the *accuracy/privacy constraint* under which client applications must access and process multidimensional data. In this contract, the Data Warehouse server and client OLAP applications play the role of mutual subscribers, respectively. A preliminary version of this work appears online as a copyright-unconstrained short communication [23].

2 Related Work

Despite the above-discussed privacy preserving issues in OLAP, today's OLAP server platforms lack of effective countermeasures to face-off relevant-in-practice limitations deriving from privacy breaches. Contrary to this actual trend, privacy preserving issues in statistical databases, which represent the theoretical foundations of PPOLAP, have been deeply investigated during past years [1], and a relevant number of techniques developed in this context are still waiting to be studied, extended and integrated within the core layer of OLAP server platforms. Basically, privacy preserving techniques for statistical databases can be classified in two main classes: *restriction-based techniques*, and *perturbation-based techniques*. First ones propose restricting the number of classes of queries that can be posed to the target database (e.g., [4]); second ones propose adding random noise at various levels of the target database, ranging from schemas [15] to query answers [5]. *Auditing query techniques* aim at devising intelligent methodologies for detecting *which* queries must be forbidden, in order to preserve privacy. Therefore, these approaches have been studied in the broader context of restriction-based privacy preserving techniques. Recently, some proposals on auditing techniques in OLAP appeared. Among these proposals, noticeable ones are: [22], which makes use of an *information theoretic approach*, and [14], which exploits *Integer Linear Programming* (ILP) techniques. Nevertheless, due to different, specific motivations, both restriction-based and perturbation-based techniques are not effective and efficient in OLAP. Restriction-based techniques are quite ineffective in OLAP since the nature of OLAP analysis is intrinsically *interactive*, and based on a *wide* set of operators and query classes. Perturbation-based techniques, which process one data cube cell at time, are quite inefficient in OLAP since they introduce excessive computational overheads when executed on massive data cubes. It should be noted that the latter drawback derives from the lack of a proper notion of privacy OLAP, as highlighted in Sect. 1.

More recently, [21] proposes a cardinality-based inference control scheme that aims at finding sufficient conditions for obtaining *safe data cubes*, i.e. data cubes such that the number of known values is under a tight bound. In line with this research, [20] proposes a PPOLAP approach that combines access and inference control techniques [8], being (*i*) first one based on the hierarchical nature of data cubes modeled in terms of *cuboid lattices* [10] and multi-resolution of data, and (*ii*) second one based on *directly* applying *restriction* to coarser aggregations of data cubes, and then *removing* remaining inferences that can be still derived. [21] and [20] are not properly comparable with our work, as they basically combine a technique inspired from statistical databases with an access control scheme, which are both outside the scope of this paper. [12] extends results of [20] via proposing the algorithm *FMC*, which still works on the cuboid lattice to hide sensitive data that cause inference. [3] defines a PPOLAP model over data partitioned across multiple clients using a *randomization approach* on the basis of which (*i*) clients perturb tuples with which they participate to the partition in order to gain *row-level privacy*, and (*ii*) server is capable of evaluating OLAP queries against perturbed tables via *reconstructing* original distributions of attributes involved by such queries. In [3], authors demonstrate that the proposed PPOLAP model is safe against privacy breaches. Being [3] (*i*) focused on a distributed environment rather than a single OLAP server (like ours), and (*ii*) oriented to the privacy preservation of data rows rather than the one of aggregate patterns, the comparison of our work with [3] is outside the scope of this paper. Finally, [17] proposes a *random data distortion technique*, called *zero-sum method*, for preserving the privacy of data cells while providing accurate answers to range-queries. To this end, [17] *iteratively* alters the values of data cells of the target data cube in such a way as to maintain the *marginal sums* of data cells along rows and columns of the data cube equal to zero. According to motivations given in Sect. 1, when applied to massive data cubes, [17] clearly introduces excessive overheads, which are not comparable with low computational requirements due to sampling-based techniques like ours. In addition to this, in our framework we are interested in preserving the privacy of aggregate patterns, rather than the one of data cells, which, however, can be still captured by introducing aggregate patterns at the coarser degree of aggregation of the input data cube, as stated in Sect. 1. In other words, [17] does not introduce a proper notion of privacy OLAP, but only restricts the analysis to the privacy of data cube cells. Despite this, we observe that, from the client side perspective, (*i*) [17] solves the same problem we investigate, i.e. providing privacy preserving (approximate) answers to OLAP queries against data cubes, and, contrary to [20] and [3], (*ii*) [17] adopts a "data-oriented" approach, which is similar-in-nature to ours. For these reasons, in our experimental analysis we test the performance of our framework against the one of [17], which, apart from being the state-of-the-art perturbation-based privacy preserving OLAP technique, will be hereby considered as the comparison technique for testing the effectiveness of the privacy preserving OLAP technique we propose.

3 Basic Constructs and Definitions

3.1 Fundamentals

A *data cube* A defined over a relational data source S is a tuple $A = \langle D,\mathcal{F},\mathcal{H},\mathcal{M}\rangle$, such that: (*i*) D is the data domain of A containing (OLAP) data cells, which are the basic aggregations of A computed over relational tuples stored in S; (*ii*) $\mathcal{F}$ is the set of *dimensions* of A, i.e. the *functional attributes* with respect to which the underlying OLAP analysis is defined (in other words, $\mathcal{F}$ is the set of attributes along which tuples in S are aggregated); (*iii*) $\mathcal{H}$ is the set of *hierarchies* related to the dimensions of A, i.e. hierarchical representations of the functional attributes shaped in the form of general trees; (*iv*) $\mathcal{M}$ is the set of *measures* of A, i.e. the *attributes of interest* for the underlying OLAP analysis (in other words, $\mathcal{M}$ is the set of attributes taken as argument of SQL aggregations whose results are stored in data cells of A). Given these definitions, (*i*) $|\mathcal{F}|$ denotes the number of dimensions of A, (*ii*) $d \in \mathcal{F}$ a generic dimension of A, (*iii*) $|d|$ the cardinality of d, and (*iv*) $H(d) \in \mathcal{H}$ the hierarchy related to d. Finally, for the sake of simplicity, we assume to deal with data cubes having a single measure (i.e., $|\mathcal{M}| = 1$). However, extending schemes, models and algorithms proposed in this paper as to deal with data cubes having *multiple measures* (i.e., $|\mathcal{M}| > 1$) is straightforward.

Given an $|\mathcal{F}|$-dimensional data cube A, an *m-dimensional range-query* Q against A, with $m \le |\mathcal{F}|$, is a tuple $Q = \langle R_{k_0}, R_{k_1}, ..., R_{k_{m-1}}, \mathcal{A} \rangle$, such that: (*i*) R_{k_i} denotes a *contiguous* range defined on the dimension d_{k_i} of A, with k_i belonging to the range $[0, |\mathcal{F}|-1]$, and (*ii*) $\mathcal{A}$ is a SQL aggregation operator. The evaluation of Q over A returns the $\mathcal{A}$-based aggregation computed over the set of data cells in A contained within the multidimensional sub-domain of A bounded by the ranges $R_{k_0}, R_{k_1}, ..., R_{k_{m-1}}$ of Q. Range-SUM queries, which return the SUM of the involved data cells, are trendy examples of range-queries. In our framework, we take into consideration range-SUM queries, as SUM aggregations are very popular in OLAP, and efficiently support summarized knowledge extraction from massive amounts of multidimensional data as well as other SQL aggregations (e.g., COUNT, AVG etc). Therefore, our framework can be straightforwardly extended as to deal with other SQL aggregations different from SUM. However, the latter research aspect is outside the scope of this paper, thus left as future work.

Given a query Q against a data cube A, the *query region* of Q, denoted by $R(Q)$, is defined as the sub-domain of A bounded by the ranges $R_{k_0}, R_{k_1}, ..., R_{k_{m-1}}$ of Q.

Given an m-dimensional query Q, the accuracy grid $\mathcal{G}(Q)$ of Q is a tuple $\mathcal{G}(Q) = \langle \Delta\ell_{k_0}, \Delta\ell_{k_1}, ..., \Delta\ell_{k_{m-1}} \rangle$, such that $\Delta\ell_{k_i}$ denotes the range partitioning Q along the dimension d_{k_i} of A, with k_i belonging to $[0, |\mathcal{F}|-1]$, in a $\Delta\ell_{k_i}$-based (one-dimensional) partition. By combining the one-dimensional partitions along *all* the dimensions of Q, we finally obtain $\mathcal{G}(Q)$ as a *regular multidimensional partition* of $R(Q)$. From Sect. 1, recall that the elementary cell of the accuracy grid $\mathcal{G}(Q)$ is implicitly defined by sub-queries of Q belonging to the query-workload QWL against the target data cube. An example of accuracy grid is depicted in Fig. 2: each elementary data cell corresponds to a sub-query in QWL.

Based on the latter definitions, in our framework we consider the broader concept of *extended range-query* Q^+, defined as a tuple $Q^+ = \langle Q, \mathcal{G}(Q) \rangle$, such that (*i*) Q is a "classical" range-query, $Q = \langle R_{k_0}, R_{k_1}, ..., R_{k_{m-1}}, \mathcal{A} \rangle$, and (*ii*) $\mathcal{G}(Q)$ is the accuracy grid associated to Q, $\mathcal{G}(Q) = \langle \Delta\ell_{k_0}, \Delta\ell_{k_1}, ..., \Delta\ell_{k_{m-1}} \rangle$, with the condition that each interval $\Delta\ell_{k_i}$ is defined on the *corresponding* range R_{k_i} of the dimension d_{k_i} of Q. For the sake of simplicity, here and in the remaining part of the paper we assume $Q \equiv Q^+$.

Given an n-dimensional data domain D, we introduce the *volume* of D, denoted by $\|D\|$, as follows: $\|D\| = |d_0| \times |d_1| \times \ldots \times |d_{n-1}|$, such that $|d_i|$ is the cardinality of the dimension d_i of D. This definition can also be extended to a multidimensional data cube A, thus introducing the volume of A, $\|A\|$, and to a multidimensional range-query Q, thus introducing the volume of Q, $\|Q\|$.

Given a data cube A, a range query-workload QWL against A is defined as a *collection* of (range) queries against A, as follows: $QWL = \{Q_0, Q_1, \ldots, Q_{|QWL|-1}\}$, with $R(Q_k) \subseteq R(A)$ $\forall$ $Q_k \in QWL$. An example query-workload is depicted in Fig. 1.

Given a query-workload $QWL = \{Q_0, Q_1, \ldots, Q_{|QWL|-1}\}$, we say that QWL is *non-overlapping* if there not exist two queries Q_i and Q_j belonging to QWL such that $R(Q_i) \cap R(Q_j) \neq \emptyset$. Given a query-workload $QWL = \{Q_0, Q_1, \ldots, Q_{|QWL|-1}\}$, we say that QWL is *overlapping* if there exist two queries Q_i and Q_j belonging to QWL such that $R(Q_i) \cap R(Q_j) \neq \emptyset$. Given a query-workload $QWL = \{Q_0, Q_1, \ldots, Q_{|QWL|-1}\}$, the *region set* of QWL, denoted by $R(QWL)$, is defined as the *collection* of regions of queries belonging to QWL, as follows: $R(QWL) = \{R(Q_0), R(Q_1), \ldots, R(Q_{|QWL|-1})\}$.

3.2 Handling Overlapping Query-Workloads

When overlapping query-workloads are considered, we adopt a *unifying strategy* that allows us to handle non-overlapping and overlapping query-workloads in the same manner. As we will better discuss in Sect. 5, this approach involves several benefits in all the phases of our privacy preserving OLAP technique. In this respect, given an overlapping query-workload QWL, our aim is to obtain a non-overlapping query-workload QWL' from QWL such that QWL' is *equivalent* to QWL, i.e. (*i*) QWL' provides the *same information content* of QWL, and (*ii*) QWL can be totally reconstructed from QWL'. To this end, for each query Q_i of QWL without any intersection with other queries in QWL, we simply add Q_i to QWL', and remove Q_i from QWL. Contrary to this, for each query Q_j of QWL having at least one non-null intersection with other queries in QWL, we (*i*) extract from Q_j two new sub-sets of queries (defined next), denoted by $\tau_1(Q_j)$ and $\tau_2(Q_j)$, respectively, (*ii*) add $\tau_1(Q_j)$ and $\tau_2(Q_j)$ to QWL', and (*iii*) remove Q_j from QWL. Specifically, $\tau_1(Q_j)$ contains the sub-queries of Q_j defined by intersection regions of Q_j, and $\tau_2(Q_j)$ contains the sub-queries of Q_j defined by regions of Q_j obtained via prolonging the ranges of sub-queries in $\tau_1(Q_j)$ along the

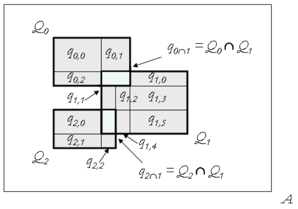


Fig. 1. Building the non-overlapping query-workload (plain lines) from an overlapping query-workload (bold lines)

dimensions of Q_j. As an example, consider Fig. 1, where the non-overlapping query-workload QWL' extracted from an overlapping query-workload $QWL = \{Q_0, Q_1, Q_2\}$ is depicted. Here, we have: (*i*) $\tau_1(Q_0) = \{q_{0\cap 1}\}$ – note that $q_{0\cap 1} \equiv Q_0 \cap Q_1$; (*ii*) $\tau_2(Q_0) = \{q_{0,0}, q_{0,1}, q_{0,2}\}$; (*iii*) $\tau_1(Q_1) = \{q_{0\cap 1}, q_{2\cap 1}\}$ – note that $q_{2\cap 1} \equiv Q_2 \cap Q_1$; (*iv*) $\tau_2(Q_1) = \{q_{1,0}, q_{1,1}, q_{1,2}, q_{1,3}, q_{1,4}, q_{1,5}\}$; (*v*) $\tau_1(Q_2) = \{q_{2\cap 1}\}$; (*vi*) $\tau_2(Q_2) = \{q_{2,0}, q_{2,1}, q_{2,2}\}$. Therefore, $QWL' = \tau_1(Q_0) \cup \tau_2(Q_0) \cup \tau_1(Q_1) \cup \tau_2(Q_1) \cup \tau_1(Q_2) \cup \tau_2(Q_2)$.

3.3 Accuracy Metrics

As accuracy metrics for answers to queries of the target query-workload QWL, we make use of the *relative query error* between exact and approximate answers, which is a well-recognized-in-literature measure of quality for approximate query answering techniques in OLAP (e.g., see [5]).

Formally, given a query Q_k of QWL, we denote as $A(Q_k)$ the exact answer to Q_k (i.e., the answer to Q_k evaluated over the original data cube A), and as $\tilde{A}(Q_k)$ the approximate answer to Q_k (i.e., the answer to Q_k evaluated over the synopsis data cube A'). Therefore, the relative query error $E_Q(Q_k)$ between $A(Q_k)$ and $\tilde{A}(Q_k)$ is defined as follows: $E_Q(Q_k) = \frac{|A(Q_k) - \tilde{A}(Q_k)|}{\max\{A(Q_k), 1\}}$.

$E_Q(Q_k)$ can be extended to the whole query-workload QWL, thus introducing the *average relative query error* $\overline{E}_Q(QWL)$ that takes into account the contributions of relative query errors of all the queries Q_k in QWL, each of them weighted by the volume of the query, $\|Q_k\|$, with respect to the whole volume of queries in QWL, i.e. the *volume of* QWL, $\|QWL\|$. $\|QWL\|$ is defined as follows: $\|QWL\| = \sum_{k=0}^{|QWL'|-1} \|Q_k\|, Q_k \in QWL$.

Based on the previous definition of $\|QWL\|$, the average relative query error $\overline{E}_Q(QWL)$ for a given query-workload QWL can be expressed as a *weighted linear combination* of relative query errors $E_Q(Q_k)$ of all the queries Q_k in QWL, as follows: $\overline{E}_Q(QWL) = \sum_{k=0}^{|QWL|-1} \frac{\|Q_k\|}{\|QWL\|} \cdot E_Q(Q_k)$, i.e.: $\overline{E}_Q(QWL) = \sum_{k=0}^{|QWL|-1} \frac{\|Q_k\|}{\sum_{j=0}^{|QWL|-1} \|Q_j\|} \cdot \frac{|A(Q_k) - \tilde{A}(Q_k)|}{\max\{A(Q_k), 1\}}$, under the constraint: $\sum_{k=0}^{|QWL|-1} \frac{\|Q_k\|}{\|QWL\|} = 1$.

3.4 Privacy Metrics

Since we deal with the problem of ensuring the privacy preservation of OLAP aggregations, our privacy metrics takes into consideration how sensitive knowledge can be discovered from aggregate data, and tries to limit this possibility. On a theoretical plane, this is modeled by the privacy OLAP notion introduced in Sect. 1.

To this end, we first study how sensitive aggregations can be discovered from the target data cube A. Starting from the knowledge about A (e.g., range sizes, OLAP hierarchies etc), and the knowledge about a given query Q_k belonging to the query-workload QWL (i.e., the volume of Q_k, $\|Q_k\|$, and the exact answer to Q_k, $A(Q_k)$), it is possible to infer knowledge about sensitive ranges of data contained within $R(Q_k)$. For instance, it is possible to derive the average value of the contribution throughout which each basic data cell of A within $R(Q_k)$ contributes to $A(Q_k)$, which we name as *singleton aggregation* $I(Q_k)$. $I(Q_k)$ is defined as follows: $I(Q_k) = \frac{A(Q_k)}{\|Q_k\|}$.

In order to provide a clearer practical example of privacy breaches deriving from the proposed singleton aggregation model in real-life OLAP application scenarios, consider a three-dimensional OLAP data cube A characterized by the following set of dimensions: $\mathcal{F}$ = {*Region*, *Product*, *Time*}, and the following set of measures; $\mathcal{M}$ = {*Sale*}. Assume that A stores data about sales performed in stores located in Italy. In the running example, for the sake of simplicity consider again the SQL aggregation operator AVG. If malicious users know the AVG value of the following range R of A: R = 〈*SouthItaly*, [*ElectricProducts* : *OfficeProducts*], [*2008* : *2009*]〉, said AVG(R) = 23,500K€, and the volume of R, said $\|R\|$ = 1,000, they will easily infer that, during the time interval between the 2008 and the 2009, each store located in South Italy has originated a volume of sales of about 23,500€ (suppose that data are uniformly distributed) for what regards electric and office products. This sensitive knowledge, also combined with the knowledge about the hierarchies defined on dimensions of A, e.g. *H*(*Region*): *Italy* ← {*North Italy*, *Central Italy*, *South Italy*, *Insular Italy*} ← … ← *South Italy* ← {*Abruzzo*, *Molise*, *Campania*, *Puglia*, *Basilicata*, *Calabria*} ← …, can allow malicious users to infer sensitive sale data about specific individual stores located in Calabria related to specific individual classes of products and specific individual days, just starting from aggregate values on sale data performed in stores located in the whole Italy during the 2008 and the 2009 (!).

Coming back to the singleton aggregation model, it is easy to understand that, starting from the knowledge about $I(Q_k)$, it is possible to *progressively* discover aggregations of larger range of data within $R(Q_k)$, rather than the one stored within the basic data cell, thus inferring even-more-useful sensitive knowledge. Also, by exploiting OLAP hierarchies and the well-known roll-up operator, it is possible to discover aggregations of ranges of data at higher degrees of such hierarchies. It should be noted that the singleton aggregation model $I(Q_k)$ above represents indeed an *instance* of our privacy OLAP notion target to the problem of preserving the privacy of range-SUM queries (the focus of our paper). As a consequence, $I(Q_k)$ is essentially based on the conventional SQL aggregation operator AVG. Despite this, the underlying theoretical model we propose is general enough to be straightforwardly extended as to deal with more sophisticated privacy OLAP notion instances, depending on the particular class of OLAP queries considered. Without loss of generality, given a query Q_k belonging to an OLAP query class C, in order to handle the privacy preservation of Q_k we only need to define the formal expression of the related singleton aggregation

$I(Q_k)$ (like the previous one for the specific case of range-SUM queries). Then, the theoretical framework we propose works at the same way.

Secondly, we study how OLAP client applications can discover sensitive aggregations from the knowledge about approximate answers, and, similarly to the previous case, from the knowledge about data cube and query metadata. Starting from the knowledge about the synopsis data cube *A'*, and the knowledge about the answer to a given query Q_k belonging to the query-workload *QWL*, it is possible to derive an *estimation* on $I(Q_k)$, denoted by $\tilde{I}(Q_k)$, as follows: $\tilde{I}(Q_k) = \frac{\tilde{A}(Q_k)}{S(Q_k)}$, such that $S(Q_k)$ is the *number of samples* effectively extracted from $R(Q_k)$ to compute *A'* (note that $S(Q_k) < \|Q_k\|$). The relative difference between $I(Q_k)$ and $\tilde{I}(Q_k)$, named as *relative inference error* and denoted by $E_I(Q_k)$, gives us a metrics for the privacy of $\tilde{A}(Q_k)$, which is defined as follows: $E_I(Q_k) = \frac{|I(Q_k) - \tilde{I}(Q_k)|}{\max\{I(Q_k),1\}}$.

Indeed, while OLAP client applications are aware about the definition and metadata of both the target data cube and queries of the query-workload *QWL*, the number of samples $S(Q_k)$ (for each query Q_k in *QWL*) is not disclosed to them. As a consequence, in order to model this aspect of our framework, we introduce the *user-perceived singleton aggregation*, denoted by $\tilde{I}_U(Q_k)$, which is the *effective* singleton aggregation *perceived* by external applications based on the knowledge made available to them. $\tilde{I}_U(Q_k)$ is defined as follows: $\tilde{I}_U(Q_k) = \frac{\tilde{A}(Q_k)}{\|Q_k\|}$.

Based on $\tilde{I}_U(Q_k)$, we derive the definition of the *relative user-perceived inference error* $E_I^U(Q_k)$, as follows: $E_I^U(Q_k) = \frac{|I(Q_k) - \tilde{I}_U(Q_k)|}{\max\{I(Q_k),1\}}$.

Since $S(Q_k) < \|Q_k\|$, it is trivial to demonstrate that $\tilde{I}_U(Q_k)$ provides a better estimation of the singleton aggregation of Q_k rather than that provided by $\tilde{I}(Q_k)$, as $\tilde{I}_U(Q_k)$ is evaluated with respect to *all* the items contained within $R(Q_k)$ (i.e., $\|Q_k\|$), whereas $\tilde{I}(Q_k)$ is evaluated with respect to the effective number of samples extracted from $R(Q_k)$ (i.e., $S(Q_k)$). In other words, $\tilde{I}_U(Q_k)$ is an *upper bound* for $\tilde{I}(Q_k)$. Therefore, in our framework we consider $\tilde{I}(Q_k)$ to compute the synopsis data cube, whereas we consider $\tilde{I}_U(Q_k)$ to model inference issues on the OLAP client application side.

$E_I^U(Q_k)$ can be extended to the whole query-workload *QWL*, by considering the *average relative inference error* $\overline{E}_I(QWL)$ that takes into account the contributions of relative inference errors $E_I(Q_k)$ of all the queries Q_k in *QWL*. Similarly to what done for the average relative query error $\overline{E}_Q(QWL)$, we model $\overline{E}_I(QWL)$ as follows:

$$\overline{E}_I(QWL) = \sum_{k=0}^{|QWL|-1} \frac{\|Q_k\|}{\|QWL\|} \cdot E_I(Q_k), \text{ i.e.: } \overline{E}_I(QWL) = \sum_{k=0}^{|QWL|-1} \frac{\|Q_k\|}{\sum_{j=0}^{|QWL|-1}\|Q_j\|} \cdot \frac{|I(Q_k) - \tilde{I}_U(Q_k)|}{\max\{I(Q_k),1\}},$$

under the constraint: $\sum_{k=0}^{|QWL|-1} \frac{\|Q_k\|}{\|QWL\|} = 1$.

Note that, as stated above, $\overline{E}_I(QWL)$ is defined in dependence on $\tilde{I}_U(Q_k)$ rather than $\tilde{I}(Q_k)$. For the sake of simplicity, here and in the remaining part of the paper we assume $E_I(Q_k) \equiv E_I^U(Q_k)$.

Concepts and definitions above allow us to introduce the *singleton aggregation privacy preserving model* $\mathcal{X} = \langle I(\bullet), \tilde{I}(\bullet), \tilde{I}_U(\bullet)\rangle$, which is a fundamental component of the privacy preserving OLAP framework we propose. $\mathcal{X}$ properly realizes our privacy OLAP notion.

Given a query $Q_k \in QWL$ against the target data cube A, in order to preserve the privacy of Q_k under our privacy OLAP notion, we must *maximize the inference error $E_I(Q_k)$ while minimizing the query error $E_Q(Q_k)$*. While the definition of $E_Q(Q_k)$ can be reasonably considered as an *invariant* of our theoretical model, the definition of $E_I(Q_k)$ strictly depends on $\mathcal{X}$. Therefore, given a particular class of OLAP queries C, in order to preserve the privacy of queries of kind C, we only need to *appropriately* define $\mathcal{X}$. This nice amenity states that the privacy preserving OLAP framework we propose is orthogonal to the particular class of queries considered, and can be straightforwardly adapted to a large family of OLAP query classes.

3.5 Thresholds

Similarly to related proposals appeared in literature recently [17], in our framework we introduce the accuracy threshold Φ_Q and the privacy threshold Φ_I. Φ_Q and Φ_I give us an *upper bound* for the average relative query error $\overline{E}_Q(QWL)$ and a *lower bound* for the average relative inference error $\overline{E}_I(QWL)$ of a given query-workload QWL against the synopsis data cube A', respectively. As stated in Sect. 1, Φ_Q and Φ_I allow us to meaningfully model and treat the accuracy/privacy constraint by means of rigorous mathematical/statistical models.

In our application scenario, Φ_Q and Φ_I are cooperatively negotiated by the Data Warehouse server and OLAP client applications. The issue of determining how to set these parameters is a non-trivial engagement. Intuitively enough, for what regards the accuracy of answers, it is possible to (*i*) refer to the widely-accepted *query error threshold* belonging to the interval [15, 20] % that, according to results of a plethora of research experiences in the context of approximate query answering techniques in OLAP (e.g., see [6]), represents the current state-of-the-art, and (*ii*) use it as baseline to trade-off the parameter Φ_Q. For what regards the privacy of answers, there are not immediate guidelines to be considered since privacy preserving techniques for advanced data management (like OLAP) are relatively new hence we cannot refer to any widely-accepted threshold like happens with approximate query answering techniques. As a result, the parameter Φ_I can be set according to a *two-step approach* where *first* the accuracy constraint is accomplished in dependence of Φ_Q, and *then* Φ_I is *consequently* set by trying to maximize it (i.e., augmenting the privacy of answers) *as much as possible, thus following a best-effort approach*.

4 Handling Accuracy and Privacy via Accuracy Grids and Multi-resolution Accuracy Grids

From Sect. 1 and Sect. 3, it follows that, in our framework, the synopsis data cube *A'* stores OLAP aggregations satisfying the accuracy/privacy constraint with respect to queries of the target query-workload *QWL*. Hence, computing *A'* via sampling the input data cube *A* is the most relevant task of the privacy preserving OLAP framework we propose.

To this end, we adopt the *strategy of sampling query regions according to the partitioned representation defined by their accuracy grids*. This strategy is named as *accuracy-grid-constrained sampling*. On the basis of this strategy, *samples are extracted from cells of accuracy grids*, according to *a vision that considers the elementary cell of accuracy grids as the atomic unit of our reasoning*. This assumption is well-founded under the evidence of noticing that, given a query Q_k of *QWL*, and the collection of its sub-queries $q_{k,0}$, $q_{k,1}$, …, $q_{k,m-1}$ defined by the accuracy grid $\mathcal{G}(Q_k)$ of Q_k, sampling the (sub-)query regions $R(q_{k,0})$, $R(q_{k,1})$, …, $R(q_{k,m-1})$ of $R(Q_k)$ allows us to (*i*) efficiently answer sub-queries $q_{k,0}$, $q_{k,1}$, …, $q_{k,m-1}$, as sampling is accuracy-grid-constrained, and, at the same time, (*ii*) efficiently answer the super-query Q_k, being the answer to Q_k given by the summation of the answers to $q_{k,0}$, $q_{k,1}$, …, $q_{k,m-1}$ (recall that we consider range-SUM queries). It is a matter of fact to note that the alternative solution of sampling the super-query Q_k directly, which we name as *region-constrained sampling*, would expose us to the flaw of being unable to efficiently answer the sub-queries $q_{k,0}$, $q_{k,1}$, …, $q_{k,m-1}$ of Q_k, since there could exist the risk of having (sub-)regions of Q_k characterized by *high density* of samples, and (sub-)regions of Q_k characterized by *low density* of samples.

It is important to further highlight that, similarly to the privacy OLAP notion (see Sect. 3), the proposed sampling strategy depends on the particular class of OLAP queries considered, i.e. range-SUM queries. If different OLAP queries must be handled, different sampling strategies must be defined accordingly.

Similarly to what done with the accuracy of answers, we exploit amenities offered by accuracy grids to *accomplish the privacy constraint as well*. In other words, just like accuracy, privacy of answers is handled by means of the granularity of accuracy grids, and still considering the elementary cell of accuracy grids as the atomic unit of our reasoning.

To become convinced of the benefits coming from our sampling strategy, consider Fig. 2, where a data cube *A* and a query Q_k are depicted along with 63 samples (represented by blue points) extracted from $R(Q_k)$ by means of two different strategies: (*i*) accuracy-grid-constrained sampling (left side of Fig. 2), and (*ii*) region-constrained sampling (right side of Fig. 2). As shown in Fig. 2, accuracy-grid-constrained sampling allows us to avoid that "favorite" regions in the synopsis data cube *A'* are obtained, i.e. regions for which the total amount of allocated space (similarly, the total number of extracted samples) is much greater than the one of other regions in *A'*. It is easy to understand that the latter circumstance, which could be caused by the alternative strategy (i.e., region-constrained sampling), arbitrary originates regions in *A'* for which the accuracy error is low and the inference error is high (which is a desiderata in our framework), and regions for which the accuracy error is very high and the inference error is very low (which are both undesired effects in our framework). The final, global effect of such a scenario results in a limited capability of answering queries by satisfying the accuracy/privacy constraint. Contrary to the latter scenario, the accuracy-grid-constrained sampling aims at obtaining a *fair* distribution of samples across *A'*, so that a *large* number of queries against *A'* can be accommodated by satisfying the accuracy/privacy constraint.

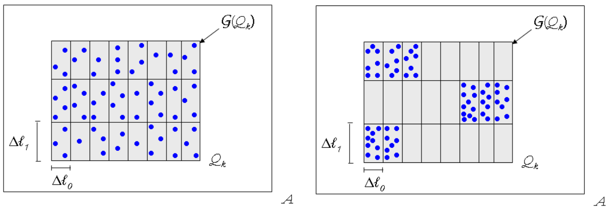


Fig. 2. Two sampling strategies: accuracy-grid-constrained sampling (left) and region-constrained sampling (right)

When overlapping query-workloads are considered, intersection (query) regions pose the issue of *dealing with the overlapping of different accuracy grids*, which we name as *multi-resolution accuracy grids*, meaning that such grids partition the *same* intersection region of multiple queries by means of *cells at different granularities*. As much as granularities of such cells are different, we obtain "problematic" settings to be handled where sub-queries have very *different volumes*, so that, due to geometrical issues, handling both accuracy and privacy of answers as well as dealing with the sampling phase become more questioning. It should be noted that, contrary to what happens with overlapping queries, accuracy grids of non-overlapping queries originate sub-queries having volumes that are equal one to another, so that we obtain facilities at both modeling and reasoning tasks.

To overcome issues deriving from handling multi-resolution accuracy grids, we introduce an innovative solution that consists in *decomposing non-overlapping and overlapping query-workloads in sets of appropriately-selected sub-queries*, thus achieving the amenity of treating both kinds of query-workload in a unified manner. The baseline operation of this process is represented by the decomposition of a query Q_k of the target query-workload *QWL*. Given a $I_{0,k} \times I_{1,k}$ query Q_k and its accuracy grid $\mathcal{G}(Q_k) = \langle \Delta\ell_{0,k}, \Delta\ell_{1,k} \rangle$, the query decomposition process generates a set of sub-queries $\zeta(Q_k)$ on the basis of the nature of Q_k. In more detail, if Q_k is non-overlapping, then Q_k

is decomposed in $\frac{I_{0,k}}{\Delta\ell_{0,k}} \cdot \frac{I_{1,k}}{\Delta\ell_{1,k}}$ sub-queries given by cells in $\mathcal{G}(Q_k)$, such that each sub-query has volume equal to $\Delta\ell_{0,k} \times \Delta\ell_{1,k}$. Otherwise, if Q_k is overlapping, i.e. there exists another query Q_h in *QWL* such that $R(Q_k) \cap R(Q_h) \neq \emptyset$, the query decomposition process works as follows: (*i*) the intersection region of Q_k and Q_h, denoted by $R^I(Q_k,Q_h)$ is decomposed in the set of sub-queries $\zeta(Q_k,Q_h)$ given by the overlapping of $\mathcal{G}(Q_k)$ and $\mathcal{G}(Q_h)$; (*ii*) let $\pi(Q_k)$ be the set of sub-queries in $\zeta(Q_k)$ (computed considering Q_k as non-overlapping) that partially overlap $R^I(Q_k,Q_h)$, and $\pi(Q_h)$ be the analogous set for Q_h, $\pi(Q_k)$ and $\pi(Q_h)$ are decomposed in sets of sub-queries obtained by considering portions of sub-queries in $\pi(Q_k)$ and $\pi(Q_h)$ that are completely contained by the regions $R(Q_k) - R^I(Q_k,Q_h)$ and $R(Q_h) - R^I(Q_k,Q_h)$, respectively, thus obtaining the sets of *border queries* $\mu_1(Q_k)$ and $\mu_1(Q_h)$; (*iii*) let $\omega(Q_k)$ be the set of sub-queries in $\zeta(Q_k)$ that are completely contained by the region $R(Q_k) - R^I(Q_k,Q_h) - \mu_1(Q_k)$, and $\omega(Q_h)$ be the analogous set for Q_h, $\omega(Q_k)$ and $\omega(Q_h)$ are decomposed in sets of sub-queries given by $\mathcal{G}(Q_k)$ and $\mathcal{G}(Q_h)$, respectively, thus obtaining the sets $\mu_2(Q_k)$ and $\mu_2(Q_h)$; (*iv*) finally, the set of sub-queries originated by the decomposition of the overlapping queries Q_k and Q_h, denoted by $\zeta^I(Q_k,Q_h)$, is obtained as follows: $\zeta^I(Q_k,Q_h) = \zeta(Q_k,Q_h) \cup \mu_1(Q_k) \cup \mu_1(Q_h) \cup \mu_2(Q_k) \cup \mu_2(Q_h)$. Finally, given a query-workload *QWL*, *QWL* is decomposed by iteratively decomposing its queries Q_0, Q_1, …, $Q_{|QWL|-1}$ according to the query decomposition process described above.

5 Computing the Synopsis Data Cube

From the Sections above, it follows that our privacy preserving OLAP technique, which is finally implemented by greedy algorithm `computeSynDataCube`, encompasses three main phases: (*i*) allocation of the input storage space *B*, (*ii*) sampling of the input data cube *A*, (*iii*) refinement of the synopsis data cube *A'*. In this Section, we present in detail these phases, and finally conclude with algorithm `computeSynDataCube`.

5.1 The Allocation Phase

Given the input data cube *A*, the target query-workload *QWL*, and the storage space *B*, in order to compute the synopsis data cube *A' the first issue to be considered is how to allocate B across query regions of QWL*. Given a query region $R(Q_k)$, allocating an amount of storage space to $R(Q_k)$, denoted by $B(Q_k)$, corresponds to assign to $R(Q_k)$ a certain number of samples that can be extracted from $R(Q_k)$, denoted by $N(Q_k)$. To this end, during the allocation phase of algorithm `computeSynDataCube`, we *assign more samples to those query regions of QWL having skewed* (*i.e., irregular and asymmetric*) *data distributions* (*e.g., Zipf*)*, and less samples to those query regions having Uniform data distributions*. The idea underlying such an approach is that few samples are enough to "describe" Uniform query regions as data distributions of such regions are "regular", whereas we need more samples to "describe" skewed query regions as data distributions of such regions are, contrary to the previous case, not “regular”. Specifically, we face-off the deriving allocation problem by means of a *proportional storage space allocation scheme*, which allows us to efficiently allocate B across query regions of QWL via assigning a *fraction* of B to each region. This allocation scheme has been preliminarily proposed in [7] for the different context of approximate query answering techniques for two-dimensional OLAP data cubes, and, in this work, it is extended as to deal with multidimensional data cubes and (query) regions.

First, if QWL is overlapping, we compute its corresponding non-overlapping query-workload QWL' (see Sect. 3.2). Hence, in both cases (i.e., QWL is overlapping or not) a set of regions $R(QWL) = \{R(Q_0), R(Q_1), \ldots, R(Q_{|QWL|-1})\}$ is obtained. Let $R(Q_k)$ be a region belonging to $R(QWL)$, the amount of storage space allocated to $R(Q_k)$, $B(Q_k)$, is determined according to a proportional approach that considers (i) the nature of the data distribution of $R(Q_k)$ and geometrical issues of $R(Q_k)$, and (ii) the latter parameters of $R(Q_k)$ in proportional comparison with the same parameters of all the regions in $R(QWL)$, as follows: $B(Q_k) = \left\lfloor \frac{\varphi(R(Q_k)) + \Psi(R(Q_k)) \cdot \xi(R(Q_k))}{\sum_{h=0}^{|QWL|-1} \varphi(R(Q_k)) + \sum_{h=0}^{|QWL|-1} \Psi(R(Q_k)) \cdot \xi(R(Q_k))} \cdot B \right\rfloor$,

such that [7]: (i) $\Psi(R)$ is a Boolean *characteristic function* that, given a region R, allows us to decide if data in R are Uniform or skewed; (ii) $\varphi(R)$ is a factor that captures the *skewness* and the *variance* of R in a combined manner; (iii) $\xi(R)$ is a factor that provides the ratio between the skewness of R and its standard deviation, which, according to [16], allows us to estimate the *skewness degree* of the data distribution of R. Previous formula can be extended as to handle the overall allocation of B across regions of QWL, thus achieving the formal definition of our proportional storage space allocation scheme, denoted by $\mathcal{W}(A,R(Q_0),R(Q_1),\ldots,R(Q_{|QWL|-1}),B)$, via the following system:

$$\begin{cases} B(Q_0) = \left\lfloor \frac{\varphi(R(Q_0)) + \Psi(R(Q_0)) \cdot \xi(R(Q_0))}{\sum_{k=0}^{|QWL|-1} \varphi(R(Q_k)) + \sum_{k=0}^{|QWL|-1} \Psi(R(Q_k)) \cdot \xi(R(Q_k))} \cdot B \right\rfloor \\ \ldots \\ B(Q_{|QWL|-1}) = \left\lfloor \frac{\varphi(R(Q_{|QWL|-1})) + \Psi(R(Q_{|QWL|-1})) \cdot \xi(R(Q_{|QWL|-1}))}{\sum_{k=0}^{|QWL|-1} \varphi(R(Q_k)) + \sum_{k=0}^{|QWL|-1} \Psi(R(Q_k)) \cdot \xi(R(Q_k))} \cdot B \right\rfloor \\ \sum_{k=0}^{|QWL|-1} B(Q_k) \leq B \end{cases} \tag{1}$$

In turn, for each query region $R(Q_k)$ of $R(QWL)$, we further allocate the amount of storage space $B(Q_k)$ across the sub-queries of Q_k, $q_{k,0}$, $q_{k,1}$, …, $q_{k,m-1}$, obtained by decomposing Q_k according to our decomposition process (see Sect. 4), via using the *same* allocation scheme (1). Overall, this approach allows us to obtain a storage space allocation for each *sub-query* $q_{k,i}$ of QWL in terms of the maximum sample number $N(q_{k,i}) = \left\lfloor \frac{B(q_{k,i})}{32} \right\rfloor$ that can be extracted from $q_{k,i}$[1], being $B(q_{k,i})$ the amount of storage space allocated to $q_{k,i}$.

[1] Here, we are assuming that an integer is represented in memory by using 32 bits.

It should be noted that the approach above allows us to achieve an extremely-accurate level of detail in handling accuracy/privacy issues of the final synopsis data cube *A'*. To become convinced of this, recall that the granularity of OLAP client applications is *the one of queries* (see Sect. 1), which is *much greater* than the one of sub-queries (specifically, the latter depends on the degree of accuracy grids) we use as atomic unit of our reasoning. Thanks to this difference between granularity of input queries and accuracy grid cells, which, in our framework, is made "conveniently" high, we finally obtain a crucial *information gain* that allows us to efficiently accomplish the accuracy/privacy constraint.

5.2 The Sampling Phase

Given an instance of our proportional allocation scheme (1), $\mathcal{W}$, during the second phase of algorithm `computeSynDataCube`, we sample the input data cube *A* in order to obtain the synopsis data cube *A'*, in such a way as to satisfy the accuracy/privacy constraint with respect to the target query-workload *QWL*. To this end, we apply a different strategy in dependence on the fact that query regions characterized by Uniform or skewed distributions are handled, according to similar insights that have inspired our allocation technique (see Sect. 5.1). Specifically, for a skewed region $R(q_{k,i})$, given the maximum number of samples that can be extracted from $R(q_{k,i})$, $N(q_{k,i})$, we *sample the* $N(q_{k,i})$ *outliers of* $q_{k,i}$. It is worthy to notice that, for skewed regions, *sum of outliers represents an accurate estimation of the sum of all the data cells contained within such regions*. Also, it should be noted that this approach allows us to gain advantages with respect to approximate query answering as well as the privacy preservation of sensitive ranges of multidimensional data of skewed regions. Contrary to this, for a Uniform region $R(q_{k,i})$, given the maximum number of samples that can be extracted from $R(q_{k,i})$, $N(q_{k,i})$, let (*i*) $\overline{C}_{R(q_{k,i})}$ be the average of values of data cells contained within $R(q_{k,i})$, (*ii*) $\mathcal{U}(R(q_{k,i}),\overline{C}_{R(q_{k,i})})$ be the set of data cells *C* in $R(q_{k,i})$ such that $value(C) > \overline{C}_{R(q_{k,i})}$, where $value(C)$ denotes the value of *C*, and (*iii*) $\overline{C}^{\uparrow}_{R(q_{k,i})}$ be the average of values of data cells in $\mathcal{U}(R(q_{i,k}),\overline{C}_{R(q_{k,i})})$, we adopt the strategy of extracting $N(q_{k,i})$ samples from $R(q_{k,i})$ by selecting them as the $N(q_{k,i})$ *closer-to-*$\overline{C}^{\uparrow}_{R(q_{k,i})}$ *data cells C in* $R(q_{k,i})$ *such that* $value(C) > \overline{C}_{R(q_{k,i})}$. Just like previous considerations given for skewed regions, it should be noted that the above-described sampling strategy for Uniform regions allows us to meaningfully trade-off the need for efficiently answering range-SUM queries against the synopsis data cube, and the need for limiting the number of samples to be stored within the synopsis data cube.

In order to satisfy the accuracy/privacy constraint, the sampling phase aims at accomplishing (decomposed) accuracy and privacy constraints *separately*, based on a two-step approach Given a query region $R(Q_k)$, we *first* sample $R(Q_k)$ in such a way as to satisfy the accuracy constraint, and, *then*, we check if samples extracted from $R(Q_k)$ *also* satisfy, beyond the accuracy one, the privacy constraint. As mentioned in Sect. 3.5, this strategy follows a best-effort approach aiming at minimizing computational overheads due to computing the synopsis data cube, and it is also the conceptual basis of guidelines for setting the thresholds Φ_Q and Φ_I.

Moreover, our sampling strategy aims at obtaining a *tunable* representation of the synopsis data cube *A'*, which can be *progressively refined* until the accuracy/privacy constraint is satisfied as much as possible. This means that, given the input data cube *A*, we first sample *A* in order to obtain the *current* representation of *A'*. If such a representation satisfies the accuracy/privacy constraint, then the *final* representation of *A'* is achieved, and used at query time to answer queries instead of *A*. Otherwise, if the current representation of *A'* does not satisfy the accuracy/privacy constraint, then we perform "corrections" on the current representation of *A'*, thus refining such representation in order to obtain a final representation that satisfies the constraint, on the basis of a best-effort approach. What we call the *refinement process* (described in Sect. 5.3) is based on a greedy approach that *"moves"*[2] *samples from regions of QWL whose queries satisfy the accuracy/privacy constraint to regions of QWL whose queries do not satisfy the constraint, yet ensuring that the former do not violate the constraint.*

Given a query Q_k of the target query-workload *QWL*, we say that Q_k satisfies the accuracy/privacy constraint iff the following inequalities simultaneously hold: $\begin{cases} E_Q(Q_k) \leq \Phi_Q \\ E_I(Q_k) \geq \Phi_I \end{cases}$.

In turn, given a query-workload *QWL*, we decide about its *satisfiability* with respect to the accuracy/privacy constraint by inspecting the satisfiability of queries that compose *QWL*. Therefore, we say that *QWL* satisfies the accuracy/privacy constraint iif the following inequalities simultaneously hold: $\begin{cases} \overline{E}_Q(QWL) \leq \Phi_Q \\ \overline{E}_I(QWL) \geq \Phi_I \end{cases}$.

Given the target query-workload *QWL*, the criterion of our greedy approach used during the refinement process is the *minimization* of the average relative query error, $\overline{E}_Q(QWL)$, and the *maximization* of the average relative inference error, $\overline{E}_I(QWL)$, within the *minimum* number of movements that allows us to accomplish both the goals simultaneously (i.e., minimizing $\overline{E}_Q(QWL)$, and maximizing $\overline{E}_I(QWL)$). Furthermore, the refinement process is bounded by a *maximum occupancy of samples moved across queries of QWL*, which we name as *total buffer size* and denote as $\mathcal{L}_{A',QWL}$. $\mathcal{L}_{A',QWL}$ depends on several parameters such as the size of the buffer, the number of sample pages moved at each iteration, the overall available swap-memory etc.

5.3 The Refinement Phase

In the refinement process, the third phase of algorithm `computeSynDataCube`, given the current representation of *A'* that does *not* satisfy the accuracy/privacy constraint with respect to the target query-workload *QWL*, we try to obtain an alternative representation of *A'* that satisfies the constraint, according to a best-effort approach.

[2] In Sect. 5.3, we describe in detail the meaning of "moving" samples between query regions.

To this end, the refinement process encompasses the following steps: (*i*) sort queries in *QWL* according to their "distance" from the satisfiability condition, thus obtaining the ordered query set $QWL^{\mathcal{P}}$; (*ii*) select from $QWL^{\mathcal{P}}$ a pair of queries $Q^{\mathcal{T}}$ and $Q^{\mathcal{F}}$ such that (*ii.j*) $Q^{\mathcal{T}}$ is the query of $QWL^{\mathcal{P}}$ having the *greater positive distance* from the satisfiability condition, i.e. $Q^{\mathcal{T}}$ is the query of $QWL^{\mathcal{P}}$ that has the greater *surplus* of samples that can be moved towards queries in $QWL^{\mathcal{P}}$ that do not satisfy the satisfiability condition, and (*ii.jj*) $Q^{\mathcal{F}}$ is the query of $QWL^{\mathcal{P}}$ having the *greater negative distance* from the satisfiability condition, i.e. $Q^{\mathcal{F}}$ is the query of $QWL^{\mathcal{P}}$ that is in most need for new samples; (*iii*) move enough samples from $Q^{\mathcal{T}}$ to $Q^{\mathcal{F}}$ in such a way as to satisfy the accuracy/privacy constraint on $Q^{\mathcal{F}}$ while, at the same time, ensuring that $Q^{\mathcal{T}}$ does not violate the constraint; (*iv*) repeat steps (*i*), (*ii*), and (*iii*) until the current representation of *A'* satisfies, as much as possible, the accuracy/privacy constraint with respect to *QWL*, within the maximum number of iterations bounded by $\mathcal{L}_{A',QWL}$. For what regards step (*iii*), moving ρ samples from $Q^{\mathcal{T}}$ to $Q^{\mathcal{F}}$ means: (*i*) removing ρ samples from $R(Q^{\mathcal{T}})$, thus obtaining an *additional* space, said $B(\rho)$; (*ii*) allocating $B(\rho)$ to $R(Q^{\mathcal{F}})$, (*iii*) re-sampling $R(Q^{\mathcal{F}})$ by considering the additional number of samples that have became available – in practice, this means extracting from $R(Q^{\mathcal{F}})$ further ρ samples.

Let $S^*(Q_k)$ be the number of samples of a query $Q_k \in QWL$ satisfying the accuracy/privacy constraint. From the formal definitions of $E_Q(Q_k)$ (see Sect. 3.3), $I(Q_k)$, $\tilde{I}(Q_k)$ and $E_I(Q_k)$ (see Sect. 3.4), and the satisfiability condition, it could be easily demonstrated that $S^*(Q_k)$ is given by the following formula: $S^*(Q_k) = \frac{(1-\Phi_Q)}{(1-\Phi_I)} \cdot \|Q_k\|$.

Let $S_{eff}(Q^{\mathcal{F}})$ and $S_{eff}(Q^{\mathcal{T}})$ be the numbers of samples *effectively* extracted from $R(Q^{\mathcal{F}})$ and $R(Q^{\mathcal{T}})$ during the previous sampling phase, respectively. Note that $S_{eff}(Q^{\mathcal{F}}) < S^*(Q^{\mathcal{F}})$ and $S_{eff}(Q^{\mathcal{T}}) \geq S^*(Q^{\mathcal{T}})$. It is easy to prove that the number of samples to be moved from $Q^{\mathcal{T}}$ to $Q^{\mathcal{F}}$ such that $Q^{\mathcal{F}}$ satisfies the accuracy/privacy constraint and $Q^{\mathcal{T}}$ does not violate the constraint, denoted by $S_{mov}(Q^{\mathcal{T}},Q^{\mathcal{F}})$, is finally given by the following formula: $S_{mov}(Q^{\mathcal{T}},Q^{\mathcal{F}}) = S^*(Q^{\mathcal{F}}) - S_{eff}(Q^{\mathcal{F}})$, under the constraint: $S_{mov}(Q^{\mathcal{T}},Q^{\mathcal{F}}) < S_{eff}(Q^{\mathcal{T}}) - S^*(Q^{\mathcal{T}})$.

Without going in details, it is possible to demonstrate that, given (*i*) an *arbitrary* data cube *A*, (*ii*) an *arbitrary* query-workload *QWL*, (*iii*) an arbitrary pair of thresholds Φ_Q and Φ_I, and (*iv*) an *arbitrary* storage space *B*, it is not always possible to make *QWL* satisfiable via the refinement process. From this evidence, our idea of using a best-effort approach makes sense perfectly.

5.4 Algorithm `computeSynDataCube`

Main greedy algorithm `computeSynDataCube` is described by the pseudo-code listed in Fig. 3, wherein: (*i*) `allocateStorageSpace` implements the proportional storage space allocation scheme (1), (*ii*) `sampleDataCube` implements the sampling strategy, (*iii*) `check` tests the satisfiability of the target query-workload against the current representation of the synopsis data cube, (*iv*) `refineSynDataCube` is in charge of refining the current representation of the synopsis data cube.

```
computeSynDataCube($A$,$QWL$,$\Phi_Q$,$\Phi_I$,$B$,$\mathcal{L}_{A',QWL}$)
 $\mathcal{W} \leftarrow$ allocateStorageSpace($A$,$QWL$,$B$)
 $A' \leftarrow$ sampleDataCube($A$,$QWL$,$\mathcal{W}$)
 while (!check($A'$,$QWL$,$\Phi_Q$,$\Phi_I$) && $\mathcal{L}_{A',QWL} > 0$){
   $\langle A'$,currSwapMemorySize$\rangle \leftarrow$ refineSynDataCube($A'$,$QWL$,$\Phi_Q$,$\Phi_I$,$\mathcal{W}$,$\mathcal{L}_{A',QWL}$)
   $\mathcal{L}_{A',QWL} \leftarrow \mathcal{L}_{A',QWL}$ – currSwapMemorySize
 }
 return $A'$
```

Fig. 3. Algorithm `computeSynDataCube`

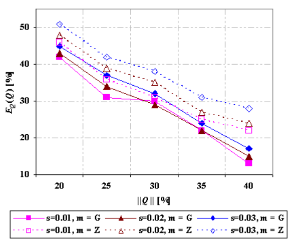
6 Experimental Evaluation

In order to test the effectiveness of our framework throughout studying the performance of algorithm `computeSynDataCube`, we conducted an experimental evaluation where we tested how the relative query error (similarly, the accuracy of answers) and the relative inference error (similarly, the privacy of answers) due to the evaluation of populations of randomly-generated queries, which model query-workloads of our framework, over the synopsis data cube range with respect to the volume of queries. The latter is a relevant parameter costing computational requirements of any query processing algorithm (also referred as *selectivity* – e.g., see [6]). According to motivations given in Sect. 2, we considered the zero-sum method [17] as the comparison technique.

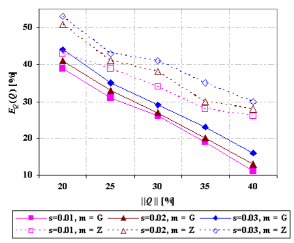
In our experimental assessment, we engineered three classes of two-dimensional data cubes: synthetic, benchmark and real-life data cubes. For all these data cubes, we limited the cardinalities of both dimensions to a threshold equal to 1,000, which represents a reliable value modeling significant OLAP applications (e.g., [6]). In addition to this, data cubes of our experimental framework expose different *sparseness coefficient s*, which measures the percentage number of non-null data cells with respect to the total number of data cells of a data cube. As widely-known since early experiences in OLAP research [2], the sparseness coefficient holds a critical impact on every data cube processing technique, thus including privacy preserving data cube computation as well.

In particular, synthetic data cubes store two kinds of data: Uniform data, and skewed data, being the latter obtained by means of a Zipf distribution. The benchmark data cube we considered has been built from the *TPC-H* data set [19], whereas the real-life one from the *Forest CoverType* (FCT) data set [9]. Both data sets are well-known in the Data Warehousing and OLAP research community. The final sparseness of the TPC-H and FCT data cube, respectively, has been easily *artificially* determined within the same OLAP data cube aggregation routine. The benefits deriving from using different kinds of data cubes are manifold, among which we recall: (*i*) the algorithm can be tested against *different* data distributions, thus stressing the reliability of the collection of techniques we propose (i.e., allocation, sampling, refinement), which, as described in Sect. 5, inspect the nature of input data to compute the final synopsis data cube; (*ii*) parameters of data distributions characterizing the data cubes can be controlled easily, thus obtaining a reliable experimental evaluation. Selectivity of queries has been modeled in terms of a percentage value of the overall volume of synthetic data cubes, and, for each experiment, we considered queries with increasing-in-size selectivity, in order to stress our proposed techniques under the ranging of an increasing input.

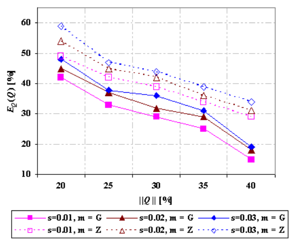
For what regards compression issues, we imposed a *compression ratio r*, which measures the percentage occupancy of the synopsis data cube *A'*, *size*(*A'*), with respect to the occupancy of the input data cube *A*, *size*(*A*), equal to 20%, which is a widely-accepted threshold for data cube compression techniques (e.g., [6]). To simplify, we set the accuracy and privacy thresholds in such a way as not to trigger the refinement process. This also because [17] does not support any "dynamic" computational feature (e.g., tuning of the quality of the random data distortion technique), so that it would have been particularly difficult to compare the two techniques under completely-different experimental settings. On the other hand, this aspect puts in evidence the innovative characteristics of our privacy preserving OLAP technique with respect to [17], which is indeed a state-of-the-art proposal in perturbation-based privacy preserving OLAP techniques.



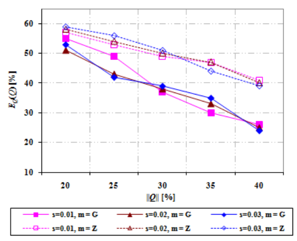
(*a*)



(*b*)



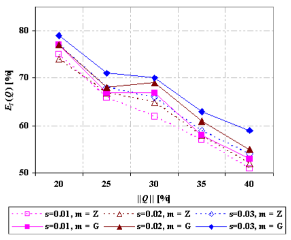
(*c*)



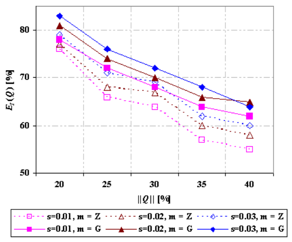
(*d*)

Fig. 4. Relative query errors of synopsis data cubes built from Uniform (*a*), skewed (*b*) TPC-H (*c*) and FCT (*d*) data cubes for several values of *s* (*r* = 20%)

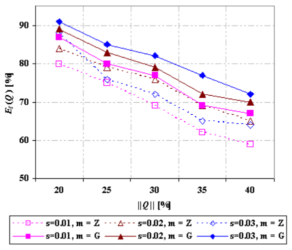
Fig. 4 shows experimental results concerning relative query errors of synopsis data cubes built from Uniform, skewed, TPC-H, and FCT data, respectively, and for several values of *s*. Fig. 5 shows instead the results concerning relative inference errors on the same data cubes. In both Figures, our approach is labeled as *G*, whereas [17] is labeled as *Z*. Obtained experimental results confirm the effectiveness of our algorithm, also in comparison with [17], according to the following considerations. First, relative query and inference errors decrease as selectivity of queries increases, i.e. the accuracy of answers increases and the privacy of answers decreases as selectivity of queries increases. This because the more are the data cells involved by a given query Q_k, the more are the samples extracted from $R(Q_k)$ able to "describe" the original data distribution of $R(Q_k)$ (this also depends on the proportional storage space allocation scheme (1)), so that accuracy increases. At the same time, more samples cause a decrease of privacy, since they provide *accurate* singleton aggregations and, as a consequence, the inference error decreases. Secondly, when *s* increases, we observe a higher query error (i.e., accuracy of answers decreases) and a higher inference error (i.e., privacy of answers increases). In other words, data sparseness influences both accuracy and privacy of answers, with a negative effect in the first case (i.e., accuracy of answers) and a positive effect in the second case (i.e., privacy of answers). This because, similarly to results of [17], we observe that privacy preserving techniques, being essentially based on mathematical/statistical models and tools, *strongly* depend on the sparseness of data, since the latter, in turn, influences the *nature* and, above all,



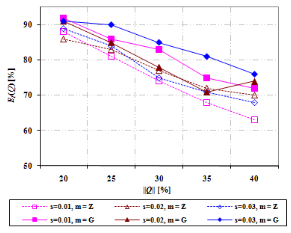
(*a*)



(*b*)



(*c*)



(*d*)

Fig. 5. Relative inference errors of synopsis data cubes built from Uniform (*a*), skewed (*b*) TPC-H (*c*) and FCT (*d*) data cubes for several values of *s* ($r = 20\%$)

the *shape* of data distributions kept in databases and data cubes. Both these experimental evidences further corroborate our idea of trading-off accuracy and privacy of OLAP aggregations to compute the final synopsis data cube. Also, by comparing experimental results on Uniform, skewed, TPC-H, and FCT (input) data, we observe that our technique works better on Uniform data, as expected, while it decreases the performance on benchmark and real-life data gracefully. This is due to the fact that Uniform data distributions can be approximated better than skewed, benchmark, and real-life ones. On the other hand, experimental results reported in Fig. 4 and Fig. 5 confirm to us the effectiveness and, above all, the reliability of our technique even on benchmark and real-life data one can find in real-world application scenarios. Finally, Fig. 4 and Fig. 5 clearly state that our proposed privacy preserving OLAP technique outperforms the zero-sum method [17]. This achievement is another relevant contribution of our research.

7 Conclusions and Future Work

A complete framework for efficiently supporting privacy preserving OLAP aggregations on data cubes has been presented and experimentally assessed in this paper. We rigorously presented theoretical foundations, as well as intelligent techniques for processing data cubes and queries, and algorithms for computing the final synopsis data cube whose aggregations balance, according to a best-effort approach, accuracy and privacy of retrieved answers. An experimental evaluation conducted on several classes of data cubes has clearly demonstrated the benefits deriving from the privacy preserving OLAP technique we propose, also in comparison with a state-of-the-art proposal. Future work is mainly oriented towards extending the actual capabilities of our framework in order to encompass intelligent update management techniques (e.g., what happens when query-workload's characteristics change dynamically over time?), perhaps inspired by well-known principles of *self-tuning databases*.

References

1. Adam, N.R., et al.: Security-Control Methods for Statistical Databases: A Comparative Study. ACM Computing Surveys 21(4), 515–556 (1989)
2. Agarwal, S., et al.: On the Computation of Multidimensional Aggregates. In: VLDB, pp. 506–521 (1996)
3. Agrawal, R., et al.: Privacy-Preserving OLAP. In: ACM SIGMOD, pp. 251–262 (2005)
4. Chin, F.Y., et al.: Auditing and Inference Control in Statistical Databases. IEEE Trans. on Software Engineering 8(6), 574–582 (1982)
5. Cuzzocrea, A.: Overcoming Limitations of Approximate Query Answering in OLAP. In: IEEE IDEAS, pp. 200–209 (2005)
6. Cuzzocrea, A.: Accuracy Control in Compressed Multidimensional Data Cubes for Quality of Answer-based OLAP Tools. In: IEEE SSDBM, pp. 301–310 (2006)
7. Cuzzocrea, A.: Improving Range-Sum Query Evaluation on Data Cubes via Polynomial Approximation. Data & Knowledge Engineering 56(2), 85–121 (2006)

8. Denning, D.E., et al.: Inference Controls for Statistical Databases. IEEE Computer 16(7), 69–82 (1983)
9. UCI KDD Archive, The Forest CoverType Data Set, `http://kdd.ics.uci.edu/databases/covertype/covertype.html`
10. Gray, J., et al.: Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals. Data Mining and Knowledge Discovery 1(1), 29–54 (1997)
11. Han, J., et al.: Efficient Computation of Iceberg Cubes with Complex Measures. In: ACM SIGMOD, pp. 1–12 (2001)
12. Hua, M., Zhang, S., Wang, W., Zhou, H., Shi, B.-L.: FMC: An Approach for Privacy Preserving OLAP. In: Tjoa, A.M., Trujillo, J. (eds.) DaWaK 2005. LNCS, vol. 3589, pp. 408–417. Springer, Heidelberg (2005)
13. Machanavajjhala, A., et al.: *L*-diversity: Privacy beyond *k*-Anonymity. ACM Trans. on Knowledge Discovery from Data 1(1), art. no. 3 (2007)
14. Malvestuto, F.M., et al.: Auditing Sum-Queries to Make a Statistical Database Secure. ACM Trans. on Information and System Security 9(1), 31–60 (2006)
15. Schlorer, J.: Security of Statistical Databases: Multidimensional Transformation. ACM Trans. on Database Systems 6(1), 95–112 (1981)
16. Stuart, A., et al.: Kendall's Advanced Theory of Statistics: Distribution Theory, 6th edn., vol. 1. Oxford University Press, New York City (1998)
17. Sung, S.Y., et al.: Privacy Preservation for Data Cubes. Knowledge and Information Systems 9(1), 38–61 (2006)
18. Sweeney, L.: *k*-Anonymity: A Model for Protecting Privacy. International Journal on Uncertainty Fuzziness and Knowledge-based Systems 10(5), 557–570 (2002)
19. Transaction Processing Council, TPC Benchmark H, `http://www.tpc.org/tpch/`
20. Wang, L., et al.: Securing OLAP Data Cubes against Privacy Breaches. In: IEEE SSP, pp. 161–175 (2004)
21. Wang, L., et al.: Cardinality-based Inference Control in Data Cubes. Journal of Computer Security 12(5), 655–692 (2004)
22. Zhang, N., et al.: Cardinality-based Inference Control in OLAP Systems: An Information Theoretic Approach. In: ACM DOLAP, pp. 59–64 (2004)
23. Cuzzocrea, A., Saccà, D.: A Constraint-Based Framework for Computing Privacy Preserving OLAP Aggregations on Data Cubes. In: ADBIS CEUR Workshop Proceedings, vol. 789, pp. 95–106 (2011)