Cardinality-based Inference Control in Sum-only Data Cubes (Extended Version)*

Lingyu Wang, Duminda Wijesekera, and Sushil Jajodia

Center for Secure Information Systems
George Mason University,
Fairfax, VA 22030-4444, USA
{lwang3,dwijesek,jajodia}@gmu.edu

Abstract. This paper deals with the inference problems in data warehouses and decision support systems such as on-line analytical processing (OLAP) systems. Even though OLAP systems restrict user accesses to predefined aggregations, the possibility of inappropriate disclosure of sensitive attribute values still exists. Based on a definition of non-compromiseability to mean that any member of a set of variables satisfying a given set of their aggregates can have more than one value, we derive sufficient conditions for non-compromiseability in sum-only data cubes. Specifically, (1) the non-compromiseability of multi-dimensional aggregates can be reduced to that of one dimensional aggregates, (2) full or dense core cuboids are non-compromiseable, and (3) there is a tight lower bound for the cardinality of a core cuboid to remain non-compromiseable. Based on those conditions, and a three-tiered model for controlling inferences, we provide a divide-and-conquer algorithm that uniformly divides data sets into chunks and builds a data cube on each such chunk. The union of those data cubes are then used to provide users with inference-free OLAP queries.

1 Introduction

Decision support systems such as On-line Analytical Processing (OLAP) are becoming increasingly important in industry. These systems are designed to answer queries involving large amounts of data and their statistical averages in near real time. It is well known that access control alone is insufficient in eliminating all forms of disclosures, as information not released directly may be inferred indirectly from answers to legitimate queries. This is known as the *inference problem*. An OLAP query typically consists of multiple aggregations, and hence vulnerable to unwanted inferences. Providing inference-free answers to sum-only data cube style OLAP queries while not adversely impacting the performance or restricting the availability in an OLAP system is the subject matter of this paper.

The inference problem has been investigated since 70's and many inference control methods have been proposed for statistical databases. However, most of those methods become computationally infeasible if directly applied to OLAP systems. OLAP applications usually require short response time, and OLAP queries usually aggregate a large

* This work was partially supported by the National Science Foundation under grant CCR-0113515.

amounts of data [21, 16]. Because most existing inference control algorithms have run times proportional to the size of queries or data set, their impact upon performance renders them impractical for OLAP systems.

While arbitrary queries are common in statistical databases, OLAP queries usually comprise of well-structured operations such as group-by, cross-tab and sub-totals. Those operations can conveniently be integrated with data cube operator and various data cube operations, such as slicing-dicing, roll up and drill down [20]. We will show how the limited formats and predictable structures of the OLAP queries as well as the multi-dimensional hierarchical data model of OLAP systems can be exploited to simplify inference control.

Table 1 shows a small two-dimensional data set about monthly employee salaries. Individual salary should be hidden from users, and hence has been replaced with the symbol ?. The symbol N/a denotes null value for inapplicable combinations of month and employee, which is known to users. [1] Assume subtotals are allowed to be released to users through OLAP queries. Inference problem occurs if any of the values represented by symbol ? can be derived from the subtotals. No value in the first two quarters can be inferred, because infinitely many different values may fit in each ? symbol with the subtotals satisfied. For the third quarter, Mary's salary in September can be inferred from the subtotals of September salaries because she is the only employee with a valid salary for September. For the fourth quarter, by subtracting Bob's and Jim's fourth quarter salaries ($4300 and $3000 respectively) from the subtotals in October and November ($7100 and $4100 respectively) Alice's salary for October can be computed to be $3900.

Based on a definition of non-compromiseability to mean that any member of a set of variables satisfying a given set of their aggregates can have more than one value [2], we derive sufficient conditions for non-compromiseability in sum-only data cubes: (1) the non-compromiseability of multi-dimensional aggregates can be reduced to that of one dimensional aggregates, (2) full or dense core cuboids are non-compromiseable, and (3) there is a tight lower bound for the cardinality of a core cuboid to remain non-compromiseable. Based on our results, and a three-tiered model for controlling inferences, we provide a divide-and-conquer algorithm that uniformly divides data sets into chunks and builds a data cube on each such chunk. The union of those data cubes are then used to provide users with inference-free OLAP queries.

The rest of the paper is organized as follows. Section 2 reviews existing inference control methods proposed in statistical databases and OLAP systems. Section 3 formalizes sum-only data cube and proves sufficient conditions for its non-compromiseability. On the basis of a three-tiered model those conditions are integrated into an inference control algorithm in Section 4. Section 5 concludes the paper.

[1] In general, data values are known through various type of *external knowledge* (knowledge obtained through channels other than query.

[2] In the settings of this paper, each variable can have either one value or infinitely many values.

Quarter	Month	Alice	Bob	Jim	Mary	Sub Total
1	January	?	?	?	?	5500
	February	?	?	?	?	5500
	March	?	?	?	?	5500
	Sub Total	3000	3000	4500	6000	
2	April	?	?	?	?	6100
	May	?	N/a	?	?	6100
	June	?	?	?	?	4100
	Sub Total	4500	3300	4500	4000	
3	July	?	?	?	?	6100
	August	?	?	?	?	6100
	September	N/a	N/a	N/a	?	2000
	Sub Total	3500	2200	2500	6000	
4	October	?	?	?	N/a	7100
	November	N/a	?	?	N/a	4100
	December	?	N/a	N/a	?	4100
*	Bonus	?	N/a	N/a	?	6000
	Sub Total	7000	4300	3000	7000	

Table 1. An Example Data Cube

2 Related Work

Inference control has been extensively studied in statistical databases as surveyed in [14, 1, 15]. Inference control methods proposed in statistical databases are usually classified into two main categories; *restriction based* techniques and *perturbation based* techniques. Restriction based techniques [19] include restricting the size of a *query set* (i.e., the entities that satisfy a single query), overlap control [17] in query sets, auditing all queries in order to determine when inferences are possible [11, 8, 23, 25], suppressing sensitive data in a released statistical tables [12], partitioning data into mutually exclusive partition [9, 10], and restricting each query set to range over at most one partition. Perturbation based technique includes adding noise to source data [30], outputs [5, 26], database structure [28], or size of query sets (by sampling data to answer queries) [13]. Some variations of the inference problem have been studied lately, such as the inference caused by arithmetic constraints [7, 6], inferring approximate values instead of exact values [25] and inferring intervals enclosing exact values [24].

The inference control methods proposed for statistical databases generally sacrifice efficiency for the ability of controlling the inference caused by arbitrary queries, which is essential to general databases. However, this sacrifice is not desirable for OLAP systems, because in OLAP systems near real time response takes priority over the generality of answerable queries. Hence most of those methods are computationally infeasible in OLAP systems. As an example, Audit Expert [11] models inference problem with a linear system $Ax = b$ and detects the occurrence of inference by transforming the m by n matrix A (corresponding to m queries on n attribute values) to its reduced row echelon form. The transformation has a well-known complexity of $O(m^2n)$, which is prohibitive in the context of data warehouses and OLAP systems since m and n can be as large as a million.

Our work shares similar motivation with that of [17], i.e., to efficiently control inference with the cardinality of data and queries, which can be easily obtained, stored and maintained. Dobkin et al. gave sufficient conditions for the non-compromiseability of arbitrary sum only queries [17]. The conditions are based on the smallest number of queries that suffices to compromise the individual data. Our work deals with multi-dimensional data cube queries. The fact that data cube queries are a special case of arbitrary queries implies better results.

To the best of our knowledge, inference control for OLAP systems and data warehouses are limited to [3, 2, 18, 27]. They all attempt to perturb sensitive values while preserving the data distribution model. Hence the classification or association rules obtained before or after the perturbation remains the same. Those works are application-specific, that is, the sole purpose of data analysis is limited to classification (association rule mining). We do not have this restriction. Moreover, we do not use perturbation in this paper.

3 Cardinality-based Non-compromiseability Criteria for Data Cubes

This section defines our model for sum-only data cubes and formalizes compromiseability. We then derive cardinality-based sufficient conditions for non-compromiseability in two cases.

3.1 Problem Formulation

In our model, a k-dimensional *data cube* consists of one *core cuboid* and several *aggregation cuboids*. In addition, we use an *aggregation matrix* to abstract the relationship between them. Each *dimension* is modeled as a closed integer interval. The Cartesian product of the k dimensions is called *full core cuboid*. Each integer vector in the full core cuboid is a *tuple*. A *core cuboid* is any subset of the full core cuboid that includes at least one tuple for each value chosen from each dimension. This allow us to uniquely identify the size of each dimension for a given core cuboid. Definition 1 formalizes these concepts.

Definition 1 (Core Cuboids and Slices).

Given a set of k integers D_i satisfying $D_i > 1$ for all $1 \leq i \leq k$. A k-dimensional core cuboid is any subset S of $\Pi_{i=1}^{k}[1, D_i]$ satisfying the property that for any $x_i \in [1, D_i]$ there exist ($k-1$) integers $x_j \in [1, D_j]$ for all $1 \leq j \leq k$ and $j \neq i$, such that $(x_1, \dots x_{i-1}, x_i, x_{i+1}, \dots x_k) \in S$. C_c denotes a core cuboid. Each vector $t \in C_c$ is referred to as a tuple. Further, the i^{th} element of vector $t \in C_c$, denoted by $t[i]$, is referred to as the i^{th} dimension of t. We say that $\Pi_{i=1}^{k}[1, D_i]$ is the full core cuboid denoted by C_f. We say a tuple t is missing from the core cuboid C_c if $t \in C_f \setminus C_c$. The subset of C_c defined by $\{t \mid t \in C_c, t[i] = j\}$ for each $j \in [1, D_i]$ is said to be the j^{th} slice of C_c on the i^{th} dimension, denoted by $P_i(C_c, j)$. If $P_i(C_c, j) = \{t \mid t \in C_f, t[i] = j, j \in [1, D_i]\}$, we say that $P_i(C_c, j)$ is a full slice.

As an illustration, the fourth quarter data given in Table 1 is modeled in Table 2. It has two dimensions: month (dimension 1) and employee name (dimension 2). Both have four different values that are mapped to the integer interval $[1, 4]$. The full core cuboid C_f is $[1, 4] \times [1, 4]$. The core cuboid C_c contains totally nine tuples and seven tuples are missing from C_c (shown as N/a in C_c).

To define aggregates of a data cube, we follow [20] to augment each dimension with a special value *ALL*, for which we use symbol *. Each *aggregation vector* is similar to a tuple except that it is formed with the augmented dimensions. An aggregation vector selects a set of tuples in core cuboids with its * values, which form its *aggregation set*. All aggregation vectors having * value in the same dimensions form an *aggregation cuboid*. The concepts of aggregation vector, aggregation cuboid and aggregation set are formalized in Definition 2.

Definition 2 (j-* Aggregation Vectors, Cuboids and Data Cubes).
A j- aggregation vector t is a k dimensional vector satisfying $t \in \Pi_{i=1}^{k}([1, D_i] \cup \{*\})$ and $\mid \{i : t[i] = * \quad for \quad 1 \leq i \leq k\} \mid= j$. If $t[i] = *$, then we say that the i^{th} element is a *-elements, and others are called non *-elements. A j-* aggregation cuboid is a set of aggregation vectors C such that for any $t, t' \in C$, $\{i : t[i] = *\} = \{i : t'[i] = *\}$ and $\mid \{i : t[i] = *\} \mid= j$. The aggregation set of an j-* aggregation vector t is defined as $\{t' \ : t' \in C_c$ such that $t'[i] = t[i], \forall i \, t[i] \neq *\}$. We use the notation $Qset(t)$ for the aggregation set of t. The aggregation set of a set of aggregation vectors S is defined as the union of $Qset(t)$ for all $t \in S$. We use notation $Qset(S)$ for the aggregation set of S.*

A data cube is defined as a pair $< C_c, S_{all} >$, where C_c is a core cuboid, and S_{all} is the set of all j- aggregation cuboids, for all $1 \leq j \leq k$.*

As an illustration, the subtotals of fourth quarter data given in Table 1 is modeled in Table 2. Each subtotal is represented as an aggregation vector with * value. For example, $(1, *)$ represents the subtotal in October. The aggregation set of $(1, *)$ is $\{(1, 1), (1, 2), (1, 3)\}$. The set of four aggregation vectors $\{(1, *), (2, *), (3, *), (4, *)\}$ form an aggregation cuboid since they all have * value in the second dimension.

	1 (Al)	2 (Bob)	3 (Jim)	4 (Ma)	5 (SubT)
1 (Oct)	(1,1)	(1,2)	(1,3)	N/a	(1,*)
2 (Nov)	N/a	(2,2)	(2,3)	N/a	(2,*)
3 (Dec)	(3,1)	N/a	N/a	(3,4)	(3,*)
4 (Bonus)	(4,1)	N/a	N/a	(4,4)	(4,*)
5 (SubT)	(*,1)	(*,2)	(*,3)	(*,4)	(*,*)

Table 2. Illustration of Data Cube

To abstract the relationship between the core cuboid and aggregation cuboids in a given data cube, we define *aggregation matrix*. Each element of aggregation matrix is associated with a tuple and an aggregation vector. An element of one means the tuple is in the aggregation set of the aggregation vector, zero otherwise. We assign the tuples in C_f and C in dictionary order, the aggregation cuboids in S_{all} in ascending order on the number of *-elements and descending order on the index of the *-element. This assignment enables us to refer to the i^{th} tuple in C_f as $C_f[i]$ (similarly for C_c, S_{all} or their subsets). We use $M[i,j]$ for the $(i,j)^{th}$ element of matrix M. The concept of aggregation matrix is formalized in Definition 3.

Definition 3 (Aggregation Matrix).

The aggregation matrix of the aggregation cuboid C on the core cuboid C_c is defined as the following $(m \times n)$ matrix $M_{C_c,C}$ (or simply M when C_c and C are clear from context).

$$M_{C_c,C}[i,j] = \begin{cases} 1, \textit{ if } C_f[j] \in Qset(C[i]); \\ 0, \textit{ otherwise.} \end{cases}$$

We define the aggregation matrix of S on C_c as the row block matrix with the i^{th} row block as the aggregation matrix of the i^{th} aggregation cuboid in S.

We use S_1 to represent the set of all 1- aggregation cuboids for a given C_c, and M_1 the aggregation matrix of S_1 on C_c (that is M_{C_c,S_1}), referred to as the 1-* aggregation matrix.*

The concept of aggregation matrix and compromiseability is illustrated in Table 3. By representing individual salary with variable x_i we get linear system $M_{C_c,S_1} \cdot \overrightarrow{X} = \overrightarrow{B}$. It has at least one solution since $\overrightarrow{B}$ are calculated from the ”real” salary values, which must satisfy the linear system. By linear algebra theory [22], each x_i can have either a unique value or infinitely many different values among all the solutions to $M_{C_c,S_1} \cdot \overrightarrow{X} = \overrightarrow{B}$. This depends on M_{C_c,S_1} but not on $\overrightarrow{B}$ (this is not valid if additional knowledge about $\overrightarrow{X}$ is learned by users, for example, salaries are non-negative [24, 25, 23]). If an x_i has a unique value among all the solutions then clearly the sensitive value represented by x_i was compromised. In this example x_1 has the value of 3900 in any solution so Alice's salary for October is compromised. In Definition 4 we formalize the definition of compromiseability. We distinct two cases of compromiseability, that is, the trivial case illustrated by the third quarter data of Table 1, and the complementary cases of the fourth quarter data.

Definition 4 (Compromiseability).

Given a data cube $< C_c, S_{all} >$ and a set of aggregation cuboids $S \subseteq S_{all}$, B is arbitrarily chosen such that $M_{C_c,S}.\overrightarrow{X} = \overrightarrow{B}$ has at least one solution. S is said to compromise C_c, if at least one component x_i of $\overrightarrow{X}$ has the same value among all the solutions to $M_{C_c,S}.\overrightarrow{X} = \overrightarrow{B}$.

1. *C_c is trivially compromised by S if there is an integer $i \in [1,m]$ such that the i^{th} row of $M_{C_c,S}$ is e_j. Here $1 \leq j \leq n$.*
2. *C_c is non-trivially compromised by S if C_c is not trivially compromised by S.*

It is well-known that C_c is compromised by S if and only if there exists at least one unit row vector e_i (where $e_i[i] = 1$ and $e_i[j] = 0$ for $j \neq i$) in any reduced row echelon form of $M_{C_c,S}$ [22]. This yields an alternative definition of compromiseability which we shall use in the rest of this paper.

	1 (Alice)	2 (Bob)	3 (Jim)	4 (Mary)	5 (Sub Total)
1 (Oct)	x_1	x_2	x_3	N/a	7100
2 (Nov)	N/a	x_4	x_5	N/a	4100
3 (Dec)	x_6	N/a	N/a	x_7	4100
4 (Bonus)	x_8	N/a	N/a	x_9	6000
5 (Sub Total)	7000	4300	3000	7000	-

$$\begin{pmatrix} 1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1 \\ 1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0 \\ 0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 0 \\ 0 \\ x_4 \\ x_5 \\ 0 \\ x_6 \\ 0 \\ 0 \\ x_7 \\ x_8 \\ 0 \\ 0 \\ x_9 \end{pmatrix} = \begin{pmatrix} 7100 \\ 4100 \\ 4100 \\ 6000 \\ 7000 \\ 4300 \\ 3000 \\ 7000 \end{pmatrix}$$

Table 3. Equations Formulating the Disclosure of the Core Cuboid Given in Table 2

3.2 Trivial Compromises

In this section, we derive cardinality-based criteria of non-compromiseability in the trivial case. We have two results. Firstly, full core cuboids cannot be trivially compromised. The second is an upper bound on the cardinality of the core cuboid such that it is trivially compromised by the set of all 1-* aggregation cuboids. They are stated and proved in Theorem 1.

Theorem 1. *1. A full core cuboid C_f cannot be trivially compromised by any set of aggregation cuboids S.*
2. C_c is trivially compromised by S_1 if $|C_c| < 2^{k-1} \cdot max(D_1, D_2, \ldots, D_k)$ for $k \geq 2$

Proof: See the Appendix.

Theorem 1 provides cardinality-based criteria for the two extreme cases, i.e., the core cuboid is either full or sparse. However, cardinality-based criteria is ineffective for the case in between. As an example, consider the third quarter data in Table 1, which is trivially compromised. Without changing the cardinality, evenly distributing the three “N/a” in three months makes the core cuboid free of trivial compromise. This invalidates any cardinality based criteria because trivial compromiseability varies for core cuboids with exactly the same cardinality.

3.3 Non-trivial Compromiseability

In this section, we derive cardinality-based criteria to determine the compromiseability in the non-trivial case. We have two results. The first is that full core cuboids cannot be non-trivially compromised. The second is a lower bound on the cardinality of the core cuboid such that it remains safe from non-trivial compromise. First we have Lemma 1.

Lemma 1. *1. C_c can not be non-trivially compromised by any single cuboid.*
2. If C_c cannot be compromised by S_1, then it cannot be compromised by S_{all}.
3. For any integers k and $D_1, D_2, \ldots, D_k$ that satisfy $D_i \geq 4$ for $1 \leq i \leq k$, there is a k-dimensional data cube $< C_c, S_{all} >$, with integer boundaries D_i, such that C_c is non-trivially compromised by S_1.

Proof: See the Appendix.

Because of the second claim of Lemma 1, it is sufficient to safeguard the core cuboid from 1-* aggregation cuboids. The last condition in Lemma 1 shows that it is impossible to obtain a criteria for preventing non-trivial compromiseability by only looking at the dimension cardinalities.

Theorem 2 (Non-trivial Compromiseability).

1. *C_f cannot be non-trivially compromised by S_1.*
2. *For any integers k and D_i, there exists a k-dimensional data cube $< C_c, S_{all} >$ satisfying $|C_f - C_c| = 2D_l + 2D_m - 9$ such that C_c is non-trivially compromised by S_1, where D_l and D_m are the least two among D_i.*
3. *If $|C_f - C_c| < 2D_l + 2D_m - 9$, then C_c cannot be non-trivially compromised.*

Proof: See the Appendix.

The first claim in Theorem 2 guarantees the non-trivial compromiseability of full core cuboid. The second and third claims give a tight lower bound on cardinality for a core c uboid to remain free of non-trivial compromise. The second claim also implies that no cardinality based criteria can be derived for sparse core cuboids (a core cuboid is sparse if its cardinality goes under the lower bound).

Corollary 1 (Non-trivial Compromiseability).
If for any $i \in [1, k]$, there exists $j \in [1, D_i]$ such that $|P_i(C_f, j) - P_i(C_c, j)| = 0$, C_c cannot be non-trivially compromised.

Proof: Follows from the proof of Theorem 2. □

Corollary 1 says full slices on every dimension suffices the non-compromiseability in the non-trivial case.

4 A Cardinality-based Inference Control Algorithm for Data Cubes

This section describes an algorithm to control inferences in data cube style OLAP queries, using the results on compromiseability developed in Section 3. Our algorithm is based on a three-tired model consisting of core data, pre-computed aggregates and answerable queries.

4.1 Three-Tiered Inference Control Model

Our three-tiered model consists of three basic components, and two abstract relations in between, as given below, and illustrated in Figure 1. In addition we enforce three properties on the model.

1. **Three Tiers:**
   (a) A set of data items D.
   (b) A set of aggregations A.
   (c) A set of queries Q.
2. **Relations Between Tiers:**
   (a) $R_{AD} \subseteq A \times D$.
   (b) $R_{QA} \subseteq Q \times A$.
3. **Properties:**
   (a) $|A| << |Q|$.
   (b) There exist partitions P_D on D and P_A on A, such that for any $(a, d) \in R_{AD}$ and $(a', d') \in R_{AD}$, d and d' are in the same chunk of P_D if and only if a and a' are in the same chunk of P_A.
   (c) D is not compromised by A.

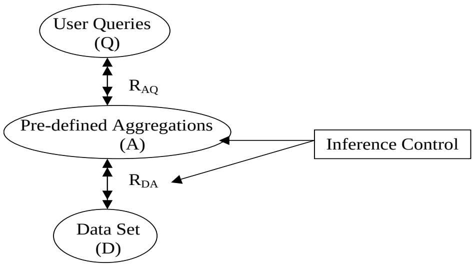

Fig. 1. Three-Tiered Model for Controlling Inferences

Three-tired inference control model simplifies inference control problem in several ways. Firstly, since all queries in Q are derived from aggregations in A, it suffices to ensure the non-compromiseability A instead of Q. This reduces the complexity of inference control due to the first characteristic of A. Secondly, the second characteristic of A allows us to adopt a divide-and-conquer approach to further reduce the complexity of inference control. Thirdly, inference control is embedded in the off-line design of A and R_{AD}, so the overhead of on-line inference control is eliminated or reduced. Although the restriction of Q to be derived from A reduces the total number of answerable queries, A can be designed in such a way that it contains most semantics required by the application, hence the restricted queries are mostly arbitrary and meaningless with respect to application requirements.

```
Algorithm Ctrl_Inf_Cube
Input: Data Cube $< C_c, S_{all} >$, integers $D_i^j \in [1, D_i]$ for $1 \le i \le k$ and $0 \le j \le m_i$, where
       $m_i$ is fixed for each $i$ and $D_i^j$ satisfy $D_i^0 = 1$, $D_i^{m_i} = D_i$, and $D_i^{j-1} < D_i^j$ for all
       $1 \le j \le m_i$.
Output: a set of aggregations $S_A$ that do not compromise $C_c$
Method:
    1. Let $S_A = \phi$;
    2. For each $k$ dimensional vector $v \in \prod_{i=1}^{k} [1, m_i]$
        Let $C_{tmp} = \{t : t \in C_c, \forall i \in [1, k]\ \ t[i] \in [D_i^{v[i]-1}, D_i^{v[i]}]\}$;
        Let $C_p = \{t : \exists\, t' \in C_{tmp}, \forall i \in [1, k]\ \ t[i] = t'[i] - D_i^{v[i]-1} + 1\}$;
        Let $S_A = S_A \cup$ Ctrul_Inf_Chunk( $C_p, [1, D_i^{v[i]} - D_i^{v[i]-1} + 1]$);
    3. Return $S_A$.

Subroutine Ctrl_Inf_Chunk
Input: $k$ dimensional core cuboid $C'_c$ and the $k$ dimension domains $[1, D'_i], i = 1, 2, \ldots, k$
Output: $S'_{all}$ if it does not compromise $C'_c$ according to the cardinality-based criteria,
$\phi$ otherwise, where $S'_{all}$ is the set of all aggregation cuboids defined on $C'_c$
Method:
    1. If $|C'_c| = 0$ or $|C'_c| = |C'_f|$, where $C'_f$ is the full core cuboid of $C'_c$
        Return $S'_{all}$;
    2. If $C'_c$ is trivially compromised by $S'_{all}$
        Return $\phi$;
    3. Let $D_l$, $D_m$ be the two smallest among $D'_i$;
    4. If $|\ C'_f - C'_c\ | < 2D_l + 2D_m - 9$
        Return $S'_{all}$;
    5. If for all $i \in [1, k]$ there exists $j \in [1, D'_i]$ such that $|\ P_i(C'_f, j) - P_i(C'_c, j)\ | = 0$
        Return $S'_{all}$;
    6. Return $\phi$.
```

Fig. 2. The algorithm of inference control in data cube

4.2 Inference Control Algorithm

The inference control algorithm shown in Figure 2 applies the results given in Section 3 on the basis of our three-tiered model. The algorithm first partitions the core cuboid into disjointed chunks, which are then passed to the subroutine *Ctrl_Inf_Chunk*. The subroutine checks the non-compromiseability of the *sub-data cube* defined on this chunk of data, using the cardinality based criteria. If it is compromised the subroutine returns an empty set, indicating no aggregation is allowed on the data. Otherwise, the subroutine returns all the aggregation cuboids of the sub-data cube. The final outcome is the union of all the sub-data cubes returned by the subroutine. This set of aggregations can then be used to answer data cube style OLAP queries without inference problem.

Correctness The correctness of the algorithm, that is, the non-compromiseability of the final result is straight-forward. The subroutine Ctrl_Inf_Chunk guarantees the non-compromiseability of each sub-data cube respectively. In addition, the sub-data cubes are disjointed, making the non-compromiseability of each of them independent of others.

Runtime Analysis: The main routine of the algorithm partitions C_c by evaluating the k dimensions of each tuple. Let $n = |C_c|$, so the runtime of the main routine is $O(nk)$=$O(n)$ (suppose k is constant compared to n). The subroutine *Ctrl_Inf_Chunk* is called for each of the $N = \prod_{i=1}^{k} m_i$ chunks (m_i are defined in the algorithm). It evaluates the cardinality of each input chunk C'_c, which has the same complexity as establishing its 1-* aggregation matrix M'_1.

Let $n' = \prod_{i=1}^{k} D'_i$ be the number of columns in M'_1 (D'_i are defined in the algorithm), then $m' = n' \sum_{i=1}^{k} \frac{1}{D'_i}$ is the number of rows. Let D_i^{max} be the maximum value among D'_i. Out of the $(m'n')$ elements, $O(m' \cdot D_i^{max})$ elements must be considered to compute M'_1. Suppose $(\sum_{i=1}^{k} \frac{1}{D'_i})D_i^{max} = O(k)$. Then the runtime of the subroutine is $O(k \cdot \prod_{i=1}^{k} D'_i)$. It is called N times so the total runtime is $O(k \cdot \prod_{i=1}^{k} m_i \cdot \prod_{i=1}^{k} D'_i) = O(k \cdot \prod_{i=1}^{k} m_i \cdot \prod_{i=1}^{k} \frac{D_i}{m_i})$, which is $O(k \cdot \prod_{i=1}^{k} D_i) = O(n)$. We note that by definition, determining non-compromiseability has a complexity of $O(n^3)$ and the maximum non-compromiseable subset of aggregations cannot be found in polynomial time [11].

Enhancing the Algorithm: The algorithm demonstrates a simple application of the cardinality based criteria in Section 3, which can be improved in many aspects. The dimension hierarchies inherent to most OLAP datasets can be exploited to increase the semantics preserved by the algorithm. For example, assume the time dimension have the hierarchy composed of day, week, month and year. Instead of partitioning the dataset arbitrarily, each chunk can be defined on week. Hence queries about weeks, months and years can be answered with aggregations in algorithm output.

Notice that the key to cardinality-based non-compromiseability is that each chunk in the partition of core cube must be either empty or dense (full). The row shuffling [4] technique proposed by Barbara et al. increases the subspaces density of data sets by shuffling rows in those categorical, unordered dimensions. Row shuffling can be integrated into the inference control algorithm as a pre-processing step prior to partitioning.

Data Cube Operations: We briefly describe how our algorithm may address common data cube operations such as slicing, dicing, rolling up, drilling down and range queries. Slicing, dicing and range query require aggregations to be defined on a subspace formed by intervals in dimension domains. Our algorithm also partitions the data set into small chunks. Therefore, in order to enhance our algorithm to address these operations, the subspace required by these data cube operations should be formed as the union of multiple chunks. Rolling up and drilling down require aggregations to be defined at different granularities than those in the original data cube. Rolling up does not directly create inference threat because with coarser granulated queries include less information about individual data. Our ongoing work is addressing these details.

Although update operations are uncommon in decision support systems, data stored in data warehouses need to be updated over time. Our algorithm is suitable for update operations in two aspects. Firstly, the change of values has no effect on the cardinality, which determines non-compromiseability in our algorithm. Secondly, because we have *localized* protection by partitioning data set into small disjointed chunks, the effect of an insertion or deletion is restricted to only the chunks containing updated tuples.

5 Conclusions

Based on a definition of non-compromiseability to mean that there are more than one choices for any of the unknown individual value to fit a given set of their aggregates, we have derived sufficient conditions for non-compromiseability in sum-only data cubes. Compromiseability of arbitrary aggregates can be reduced to those of one dimensional aggregates. Full or dense core cuboids are free from inferences, and that there is a tight lower bound on the cardinality of a core cuboid for it to remain non-compromiseable. To apply our results for inference control of data cube style OLAP queries, we have shown a *divide and conquer* algorithm based on a three-tiered model. Future work includes enhancing our results and algorithm to include data cube operations and consider other variations of OLAP queries.

References

1. N.R. Adam and J.C. Wortmann. Security-control methods for statistical databases: a comparative study. *ACM Computing Surveys*, 21(4):515–556, 1989.
2. D. Agrawal and C.C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 247–255, 2001.
3. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 439–450, 2000.
4. D. Barbar and X. Wu. Using approximations to scale exploratory data analysis in datacubes. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 382–386, 1999.
5. L.L. Beck. A security mechanism for statistical databases. *ACM Trans. on Database Systems*, 5(3):316–338, 1980.
6. A. Brodsky, C. Farkas, and S. Jajodia. Secure databases: Constraints, inference channels, and monitoring disclosures. *IEEE Trans. Knowledge and Data Engineering*, 12(6):900–919, 2000.
7. A. Brodsky, C. Farkas, D. Wijesekera, and X.S. Wang. Constraints, inference channels and secure databases. In *the 6th International Conference on Principles and Practice of Constraint Programming*, pages 98–113, 2000.
8. F.Y. Chin, P. Kossowski, and S.C. Loh. Efficient inference control for range sum queries. *Theoretical Computer Science*, 32:77–86, 1984.
9. F.Y. Chin and G. Özsoyoglu. Security in partitioned dynamic statistical databases. In *Proc. of IEEE COMPSAC*, pages 594–601, 1979.
10. F.Y. Chin and G. Özsoyoglu. Statistical database design. *ACM Trans. on Database Systems*, 6(1):113–139, 1981.
11. F.Y. Chin and G. Özsoyoglu. Auditing and inference control in statistical databases. *IEEE Trans. on Software Engineering*, 8(6):574–582, 1982.

12. L.H. Cox. Suppression methodology and statistical disclosure control. *Journal of American Statistic Association*, 75(370):377–385, 1980.
13. D.E. Denning. Secure statistical databases with random sample queries. *ACM Trans. on Database Systems*, 5(3):291–315, 1980.
14. D.E. Denning and P.J. Denning. Data security. *ACM computing surveys*, 11(3):227–249, 1979.
15. D.E. Denning and J. Schlörer. Inference controls for statistical databases. *IEEE Computer*, 16(7):69–82, 1983.
16. P.M. Deshpande, K. Ramasamy, A. Shukla, and J.F. Naughton. Caching multidimensional queries using chunks. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 259–270, 1998.
17. D. Dobkin, A.K. Jones, and R.J. Lipton. Secure databases: protection against user influence. *ACM Trans. on Database Systems*, 4(1):97–106, 1979.
18. A. Evfimievski, R. Srikant, , R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proceedings of the 8th Conference on Knowledge Discovery and Data Mining (KDD'02)*, 2002.
19. L.P. Fellegi. On the qestion of statistical confidentiality. *Journal of American Statistic Association*, 67(337):7–18, 1972.
20. J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A relational operator generalizing group-by, crosstab and sub-totals. In *Proceedings of the 12th International Conference on Data Engineering*, pages 152–159, 1996.
21. V. Harinarayan, A. Rajaraman, and J.D. Ullman. Implementing data cubes efficiently. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 205–227, 1996.
22. K. Hoffman. *Linear Algebra*. Prentice-Hall, 1961.
23. J. Kleinberg, C. Papadimitriou, and P. Raghavan. Auditing boolean attributes. In *Proc. of the 9th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 86–91, 2000.
24. Y. Li, L. Wang, X.S. Wang, and S. Jajodia. Auditing interval-based inference. In *Proceedings of the 14th Conference on Advanced Information Systems Engineering (CAiSE'02)*, 2001.
25. F.M. Malvestuto and M. Moscarini. Computational issues connected with the protection of sensetive statistics by auditing sum-queries. In *Proc. of IEEE Scientific and Statistical Database Management*, pages 134–144, 1998.
26. J.M. Mateo-Sanz and J. Domingo-Ferrer. A method for data-oriented multivariate microaggregation. In *Proceedings of the Conference on Statistical Data Protection'98*, pages 89–99, 1998.
27. S. Rizvi and J.R. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th Conference on Very Large Data Base (VLDB'02)*, 2002.
28. J. Schlörer. Security of statistical databases: multidimensional transformation. *ACM Trans. on Database Systems*, 6(1):95–112, 1981.
29. R.P. Tewarson. *Sparse Matrices*. Academic Press, 1973.
30. J.F. Traub, Y. Yemini, and H. Woźniakowski. The statistical security of a statistical database. *ACM Trans. on Database Systems*, 9(4):672–679, 1984.

Appendix

Proof(Theorem 1):

1. We show that for any $t \in S$, we have $|Qset(t)| > 1$. Let $t \in S$ be any j-* aggregation vector. Without loss of generality, let t be $(*, *, \dots, *, x_{j+1}, x_{j+2}, \dots, x_k)$. From Definition 2, we have that $Qset(t) = \{t' : \ t' \in C_f, \ t'[j+1] = x_{j+1}, \ t'[j+2] = x_{j+2}, \dots, t'[k] = x_k\}$. Because $C_f = \Pi_{i=1}^{k}[1, D_i]$ we have that $|Qset(t)| = \prod_{i=1}^{j} D_i$. With the assumption of $min(D_1, D_2, \dots, D_k) > 1$ we have $|Qset(t)| > 1$.
2. Suppose that C_c is not trivially compromised by S_1. We show $|C_c| \geq 2^{k-1} \cdot max(D_1, D_2, \dots, D_k)$ for $k \geq 2$. Without loss of generality, we assume $D_k = max(D_1, D_2, \dots, D_k)$. Notice that there are totally D_k slices of C_c on the k^{th} dimension. Without loss of generality it suffices to show that $|P_k(C_c, 1)| \geq 2^{k-1}$. We do so by mathematical induction as given below.
   Inductive Hypothesis: For every $i \leq k$, there is a subset $S_i \subseteq P_k(C_c, 1)$ such that $|S_i| = 2^{i-1}$ satisfying the condition that for any $t_1, t_2 \in S_i$, $t_1[j] = t_2[j]$ for all $j \geq i$.
   Base Case: By Definition 1, there exists $t \in C_c$ such that $t[k] = 1$. Let S_1 be $\{t\}$. Then we have that $S_1 \subseteq P_k(C_c, 1)$ and $|S_1| = 1$, validating the base case of our inductive hypothesis.
   Inductive Case: Suppose we have $S_i \subseteq P_k(C_c, 1)$ for $1 \leq i < k$ such that $|S_i| = 2^{i-1}$ and for any $t_1, t_2 \in S_i$, $t_1[j] = t_2[j]$ for all $j \geq i$. We show that there exists $S_{i+1} \subseteq P_k(C_c, 1)$ such that $|S_{i+1}| = 2^i$, satisfying the condition that for any $t_1, t_2 \in S_i$, $t_1[j] = t_2[j]$ for all $j \geq i + 1$.
   For any $t_1 \in S_i$, let $t'_1[i] = *$ and $t'_1[j] = t[j]$ for all $j \neq i$. We have $t'_1 \in S_1$ and since $t_1 \in Qset(t'_1)$ we have $|Qset(t'_1)| \geq 1$. Since C_c is not trivially compromised by S_1, according to Definition 4, we have $|Qset(t'_1)| > 1$. Hence, there exists $t''_1 \in Qset(t'_1) \subseteq C_c$ such that $t''_1[i] \neq t_1[i]$ and $t''_1[j] = t_1[j]$ for all $j \neq i$; which implies $t''_1 \notin S_i$.
   Now we show that for any $t_2 \in S_i$ such that $t_1 \neq t_2$, we have $t''_1 \neq t''_2$, where $t''_2 \in C_c$, $t''_2[i] \neq t_2[i]$ and $t''_2[j] = t_2[j]$ for all $j \neq i$. Since $t_1[j] = t_2[j]$ for all $j \geq i$ there must be $l < i$ such that $t_1[l] \neq t_2[l]$. We know that $t''_1[j] = t'_1[j] = t_1[j]$ and $t''_2[j] = t'_2[j] = t_2[j]$ for all $j < i$. Hence we have that $t''_1[l] \neq t''_2[l]$; that is, $t''_1 \neq t''_2$.
   Hence there exists $S'_i \subset C_c$ satisfying: $|S'_i| = |S_i|$, and for any $t \in S_i$, there exists one and only one $t' \in S'_i$ such that $t[i] \neq t'[i]$ and $t[j] = t'[j]$ for all $j \neq i$. Define S_{i+1} as $S_i \cup S'_i$. Since $|S_i| = 2^{i-1}$ we have $|S_{i+1}| = 2^i$.
   This proves the inductive case of our induction, from which the claim $|P_k(C_c, 1)| \geq 2^{k-1}$ follows.

□

Proof(Lemma 1):

1. Let $C \in S_{all}$. We show that C_c cannot be nontrivially compromised by C. For any $t \in C_c$ there exists one and only one $t' \in C$ such that $t \in Qset(t')$. Hence in M each non-zero column is a unit column vector, which implies that M could be transformed into its reduced row echelon form by merely permuting the columns. Furthermore, each row of M must contain at least two 1's since no trivial compromise is assumed. Hence no unit row vector is in the reduced row echelon form of M, that is, C_c cannot be nontrivially compromised by C. This concludes our proof.

2. Without loss of generality, let t be a j-* $(j > 1)$ aggregation vector satisying that $t[i] = *$ for any $i \in [1, j]$. Let C be the set of 1-* aggregation vectors defined as: $\{t' : \ t'[1] = *, t'[i] \in [1, D_i] \forall i \in [2, j], t'[i] = t[i] \forall i \in [j+1, k]\}$. We have that $Qset(t) = Qset(C)$. Hence in the aggregation matrix $M_{m \times n}$ of S_{all} on C_c, any row corresponding to a j-* $(j > 1)$ aggregation vector can be represented as the linear combination of the rows corresponding to the 1-* aggregation vectors. The rest of the proof follows from linear algebra.
3. First we show that the Lemma for $k > 2$ can be reduced to the Lemma for $k = 2$. For $k > 2$, let S_1' be $\{t : \ t \in S_1, t[j] = 1 \vee t[j] = * \forall j > 2\}$. Then $Qset(S_1') = \{t : \ t \in C_c, t[j] = 1 \forall j > 2\}$. The pair $< Qset(S_1'), S_1' >$ can be regarded as a special two dimensional data cube. Hence, we can build $Qset(S_1')$ in such a way that it is nontrivially compromised by S_1', as shown in the succeeding case of $k = 2$. By Lemma 1, if a tuple is nontrivially compromised by S_1' in the data cube $< Qset(S_1'), S_1' >$, then it is also nontrivially compromised by S_1 in the data cube $< C_c, S_{all} >$. This reduces proof for $k > 2$ to that of $k = 2$, which we prove now.
   For the proof of $k = 2$, without loss of generality, we use mathematical induction on D_1, for an arbitrary, but fixed value of $D_2 \geq 4$.
   Inductive Hypothesis: For any $D_1, D_2 \geq 4$, we can build a two dimensional data cube C_c with integer boundaries D_1, D_2 such that C_c is nontrivially compromised by S_1.
   Base Case: When $D_1 = D_2 = 4$, consider the core cuboid C_c corresponding to the fourth quarter data in Table 1. It validates the base case of our inductive hypothesis.
   Inductive Case: Assuming that there is nontrivially compromiseable two dimensional core cuboid with integer boundaries $\{D_1, D_2\}$, we show how to obtain a nontrivially compromiseable two dimensional core cuboid with integer boundaries $\{D_1 + 1, D_2\}$.
   Suppose we are given a core cuboid C_c with boundary $D_1 = j \geq 4$. Further suppose without loss of generality that the tuple $(1, 1)$ is nontrivially compromised in $< C_c, S_{all} >$. Then, there is a row vector a such that $a \cdot M_1 = e_1$. Now we show how to build a core cuboid C_c' for $D_1 = j + 1$ such that the tuple $(1, 1)$ is nontrivially compromised in $< C_c', S_{all} >$ also.
   First define a set of tuples C as:
   - for any $t \in C$, $t[1] = j + 1$
   - for any $l \in [1, D_2]$, $(j + 1, l) \in C$ if and only if $(j, l) \in P_1(C_c, j)$

   Then, we define $C_c' = C_c \cup C$. Consequently, we have $P_1(C_c', j + 1) = C$. Let M_1' be the 1-* aggregation matrix of S_1 on C_c'. Hence, $M_1' = (M_1 | M_c)$, where the non-zero columns in M_c correspond to the tuples in C. From the definition of C we further have $M_1 = (M_1'' | M_c)$, where the non-zero columns of M_c correspond to the tuples in $P_1(C_c, j)$. Thus, $M_1' = (M_1'' | M_c | M_c)$. Since $a \cdot M_1 = (a \cdot M_1'' | a \cdot M_c) = e_1$, $a \cdot M_1' = (a \cdot M_1'' | a \cdot M_c | a \cdot M_c) = (e_1 | 0)$. Hence, C_c' is nontrivially compromised by S_1, validating the inductive case of our inductive hypothesis.

□

Proof(Theorem 2):

1. Due to Theorem 1, we only need to show the case of nontrivial compromise. In this respect, without loss of generality, we show that $t_0 = (1, 1, \ldots, 1)$ cannot be

nontrivially compromised by S_1. Let $C'_f = \{t : \forall i \in [1,k], t[i] = 1 \vee t[i] = 2\}$. Since $(D_1, D_2, \ldots, D_k) \geq 2$, we have that $C'_f \subseteq C_f$ and $|C'_f| = 2^k$. Because of Lemma 1, we only need to prove that t_0 cannot be compromised by S_1 in the data cube $< C'_f, S_{all} >$. Let M'_1 be the 1-* aggregation matrix of S_1 on C'_f. According to Definition 3, there are 2^k non-zero column vectors in M'_1, corresponding to the 2^k tuples in C'_f. In the rest of the proof we formally show that each of the 2^k non-zero column vectors can be represented by the linear combination of the left $2^k - 1$ column vectors. Then, it follows from linear algebra that t_0 cannot be compromised by S_1 in data cube $< C'_f, S_{all} >$.

In order to prove our informally stated claim, we define the *sign assignment vector* as an n dimensional column vector t_{sign} where n is $|C_f|$, as follows:

- $t_{sign}[1] = 1$
- $t_{sign}[2^i + j] = -t_{sign}[j]$ for all $0 \leq i \leq k-1$ and $1 \leq j \leq 2^i$
- $t_{sign}[j] = 0$ for all $j > 2^k$

Claim: $M'_1 \cdot t_{sign} = 0$, where 0 is the n dimensional zero column vector.
Justification:

Let $t = S_1[i]$, $t[l] = *$ for $l \in [1,k]$.
Let v be $M'_1[i,-]$.
Suppose $t[j] = 1$ or $t[j] = 2$ for all $j \neq l$.
Then $|Qset(t)| = 2$, and as a consequence we get $Qset(t) = \{t_1, t_2\}$
where $t_1, t_2 \in C_f$, $t_1[l] = 1$,$t_1[l] = 2$
and $t_1[j] = t_2[j] = t[j]$ for all $j \neq l$
Hence, there are two integers $j_1, j_2 \in [1,n]$ satisfying
$v[j_1] = v[j_2] = 1$ and $v[j] = 0$ for any $j \neq j_1, j_2$.
By Definition 3, $M'_1[-,j_1]$ and $M'_1[-,j_2]$ correspond to t_1 and t_2 respectively.
Because C'_f is formed in dictionary order, we get $j_2 = j_1 + 2^{l-1}$.
Hence, we have $v \cdot t_{sign} = 0$.
Otherwise, $|Qset(t)| = 0$; and hence $Qset(t) = \phi$.
Hence, $v = 0$, and hence, $0 \cdot t_{sign} = 0$.
This justifies our claim.

Hence, as stated earlier, the justification of our claim concludes the main proof.

2. Without loss of generality we assume D_1, D_2 are the least two among D_i's. For an arbitrary but fixed value of D_2, we show by induction on D_1 that C_c as constructed in the proof of Lemma 1 satisfies $|C_f - C_c| = 2D_1 + 2D_2 - 9$.
Inductive Hypothesis: C_c as constructed in the proof of Lemma 1 satisfies:
   - $|C_f - C_c| = 2j + 2D_2 - 9$ for any $j \geq 4$.
   - $\mid P_1(C_f, j) - P_1(C_c, j) \mid= 2$ for any $j \in [1, D_1]$.

   Base Case: In the base case of the proof of Lemma 1, the core cuboid C_c satisfies $|C_f - C_c| = 2D_1 + 2D_2 - 9$. Notice that the core cuboid, $D_1 = 4$, and $\mid P_1(C_f, j) - P_1(C_c, j) \mid= 2$. This validates the base case of our inductive hypothesis.
   Inductive Case: Suppose for $D_1 = j$ we have $|C_f - C_c| = 2j + 2D_2 - 9$ and $\mid P_1(C_f, j) - P_1(C_c, j) \mid= 2$. Let C'_f be the full core cuboid corresponding to

C'_c for $D_1 = j + 1$. By the definition of C in the proof of Lemma 1, we have $|C| = |P_1(C_c, j)|$ and as a consequence $|C'_f - C'_c| = |C_f - C_c| + 2 = 2(j+1) + 2D_2 - 9$. Since $P_1(C'_c, j+1) = C$. Hence, $|\ P_1(C'_f, j) - P_1(C'_c, j)\ | = 2$. This validates the inductive case of our inductive argument and consequently concludes our proof of the tightness of the cardinality lower bound for avoiding nontrivial compromiseability.

Lower Bound: We show that if C_c is nontrivially compromised then we have $|C_f - C_c| \geq 2D_1 + 2D_2 - 9$. First we make following assumptions.

(a) The tuple $t = (1, 1, \ldots, 1) \in C_c$ is nontrivially compromised by S_1
(b) No tuple in C_c is trivially compromised
(c) There exists $S \subseteq S_1$ such that for any $C \in S$, t cannot be nontrivially compromised by $S \setminus C$
(d) For any $t' \in C_f \setminus C_c$, t cannot be nontrivially compromised by S_1 in data cube $< C_c \cup \{t'\}, S_{all} >$. That is, $|C_f - C_c|$ has reached its the lower bound.

Assumption 2 holds by Definition 4. Assumption 3 is reasonable, as by Lemma 1 S must contain at least two 1-* aggregation cuboids. Assumption 4 is reasonable, because by Theorem 2, $|C_f - C_c|$ has a lower bound if C_c is nontrivially compromiseable.

Claim: Suppose Assumption 1,2,3, and 4 hold. Furthermore assume that there is a $C \in S$ where $t \in C$ satisfies $t[i] = *$. Then $|P_i(C_f, 1) - P_i(C_c, 1)| \geq 1$, and $|P_i(C_f, j) - P_i(C_c, j)| \geq 2$ holds for any $j \in [2, D_i]$.

Justification: The proof is by contradiction. Without loss of generality, we only justify the claim for $i = k$ and $j = 2$. That is, given a $C \in S$ satisfying $t[k] = *$ for any $t \in C$, we prove that $|P_k(C_f, 2) - P_k(C_c, 2)| \geq 2$.

First we transform the aggregation matrix of S on C_c by row permutation into a singly bordered block diagonal form (SBBDF) [29], denoted by $M_{m \times n}$. The i^{th} diagonal block of M corresponds to $P_k(C_c, i)$ and $\{t\ :\ t \in S \setminus C, t[k] = i\}$, and the border of M denotes the aggregation cuboid C. We call the columns of M corresponding to the i^{th} diagonal block as the i^{th} *slice of* M.

Due to Assumption 1, there exists a row vector a satisfying $a \cdot M = e_1$. Let r_i be $M[i, -]$ then we get $e_1 = \sum_{i=1}^{m} a[i] \cdot r_i$. Suppose each diagonal block of M has size $m' \times n'$. Use r_i^j, for $1 \leq j \leq D_k$ to represent the row vector composed of the elements of r_i that falls into the j^{th} slice of M. Notice that there are n' elements in r_i^j. We also use e'_1 and $0'$ to represent the n' dimensional unit row vector and n' dimensional zero row vector, respectively. Then the following are true:

i. $e'_1 = \sum_{i=1}^{m'} a[i] r_i^1 + \sum_{i=m-m'+1}^{m} a[i] r_i^1$
ii. $0' = \sum_{i=m'+1}^{2m'} a[i] r_i^2 + \sum_{i=m-m'+1}^{m} a[i] r_i^2$

First we suppose $|P_k(C_f, 2) - P_k(C_c, 2)| = 0$, that is, the second slice of M contains no zero column. We then derive contradictions to our assumptions.

Since $|P_k(C_f, 2) - P_k(C_c, 2)| = 0$ the first slice of M contains no more non-zero columns than the second slice of M does. Intuitively if the latter is transformed into a zero vector then applying the same transformation on the former leads to a zero vector, too. This is formally represented as:

iii. $0' = \sum_{i=1}^{m'} a[m' + i] r_i^1 + \sum_{i=m-m'+1}^{m} a[i] r_i^1$.

Subtracting (iii) from (i) gives $e'_1 = \sum_{i=1}^{m'}(a[i] - a[m' + i])r_i^1$. That implies C_c is nontrivially compromised by $S \setminus \{C_k\}$, contradicting Assumption 3. Thus $P_k(C_f, 2) - P_k(C_c, 2)| \neq 0$.

Next we assume $|P_k(C_f, 2) - P_k(C_c, 2)| = 1$ and derive a contradiction to our assumptions.

First the row vector r_i^3 satisfies the following condition:

iv. $0' = \sum_{i=2m'+1}^{3m'} a[i]r_i^3 + \sum_{i=m-m'+1}^{m} a[i]r_i^3$.

Let $t' \in P_k(C_f, 2) \setminus P_k(C_c, 2)$. Notice that (i), (ii) still hold. Suppose t' corresponds to $M[-, y] = 0$. Now assume we add t' to $P_k(C_c, 2)$, consequently we have $M[-, y] \neq 0$. Due to Assumption 4, we have that the left side of (ii) becomes e'_1, that is, $a \cdot M[-, y] = 1$. There is also an extra 1-element $M[x, y]$ in the border of M.

Now let t'' be the tuple corresponding to $M[-, y + n']$ in the third slice of M. Suppose $t'' \in P_k(C_c, 3)$ and consequently $M[-, y + n'] \neq 0$. We have that $M[-, y + n'] = M[-, y]$ and consequently $a \cdot M[-, y + n'] = 1$.

By removing t' from $P_k(C_c, 2)$ we return to the original state that all our assumption hold. Now we show by contradiction that $t'' \in P_k(C_c, 3)$ cannot hold any longer. Intuitively, since t' is the only missing tuple in the second slice of M, the third slice of M contains no more non-zero vectors than the second slice of M does, except t''. Because $a \cdot M[-, y + n'] = 1$, elements of a transform the second slice of M to a zero vector, as shown by (ii), also transform the third slice of M to a unit vector. This is formally represented in (v):

v. $e'' = \sum_{i=2m'+1}^{3m'} a[i - m']r_i^3 + \sum_{i=m-m'+1}^{m} a[i]r_i^3$

Subtracting (iv) from (v) we get that $e'' = \sum_{i=2m'+1}^{3m'}(a[i - m'] - a[i])r_i^3$; implying C_c is compromised by $S \setminus \{C_i\}$. Hence, Assumption 3 is false. Consequently, $t'' \notin C_c$.

Similar proof exists for the i^{th} slice of C_c, where $i = 4, 5, \ldots, D_k$. However, $M[x, -] \neq 0$ because if so, we can let a_x be zero and then decrease the number of missing tuples in C_c, contradicting Assumption 4. Hence $M[x, -]$ is a unit vector with the 1-element in the first slice of M. However, this further contradicts Assumption 2, that no trivial compromise is possible. Hence we have that $|P_k(C_f, 2) - P_k(C_c, 2)| = 1$ is false.

Now consider $|P_k(C_f, 1) - P_k(C_c, 1)|$. Suppose all the assumptions hold, and $|P_k(C_f, 1) - P_k(C_c, 1)| = 0$. Let $t_1, t_2 \in P_k(C_f, 2) \setminus P_k(C_c, 2)$. Now define $C'_c = C_c \setminus \{t\} \cup \{t_1\}$ and M' be the aggregation matrix of S on C'_c. From $a \cdot M = e_1$, and Assumption 4 we get $a \cdot M' = e_i$, where $M[-, i]$ corresponds to t_1. This implies the nontrivially compromise of t_1 in $< C'_c, S_{all} >$, with $|P_k(C_f, 1) - P_k(C'_c, 1)| = 1$, which contradicts what we have already proved. Hence, we get $|P_k(C_f, 1) - P_k(C_c, 1)| \geq 1$. This concludes the justification of our claim.

The claim implies that the number of missing tuples in C_c increases monotonically with the following:

- The number of aggregation cuboids in S.
- D_i, provided there is $C \in S$ satisfying $t[i] = *$ for any $t \in C$.

Hence $|C_f - C_c|$ reaches its lower bound when $S = \{C_1, C_2\}$, which is equal to $2D_1 + 2D_2 - 9$, as shown in the first part of the current proof - concluding the proof of Theorem 2. □