Enforcing Confidentiality and Data Visibility Constraints: An OBDD Approach

Valentina Ciriani, Sabrina Capitani Di Vimercati, Sara Foresti, Giovanni Livraga, Pierangela Samarati

Enforcing Confidentiality and Data Visibility Constraints: An OBDD Approach

V. Ciriani, S. De Capitani di Vimercati, S. Foresti, G. Livraga, and P. Samarati

DTI - Università degli Studi di Milano, 26013 Crema, Italia
firstname.lastname@unimi.it

Abstract. The problem of enabling privacy-preserving data releases has become more and more important in the last years thanks to the increasing needs of sharing and disseminating information. In this paper we address the problem of computing data releases in the form of *fragments* (vertical views) over a relational table, which satisfy both confidentiality and visibility constraints, expressing needs for information protection and release, respectively. We propose a modeling of constraints and of the data fragmentation problem based on Boolean formulas and Ordered Binary Decision Diagrams (OBDDs). Exploiting OBDDs, we efficiently manipulate Boolean formulas, thus easily computing data fragments that satisfy the constraints.

Keywords: Privacy, fragmentation, confidentiality and visibility constraints, OBDDs.

1 Introduction

Information sharing and dissemination are typically selective processes. While on one side, there is a need - or demand - for making certain information available to others, there is on the other side an equally strong need to ensure proper protection of sensitive information. It is therefore important to provide data holders with means to express and enforce possible constraints over their data, modeling the need for information of the data recipients (visibility constraints) and the need for protecting confidential information from an improper disclosure (confidentiality constraints).

Recent proposals considering confidentiality and visibility constraints have put forward the idea of computing vertical fragments over the original data structure (typically a relation) in such a way that constraints are satisfied [1, 7, 8, 10]. While such proposals have been introduced as a way of departing from data encryption when relying on external storage services, data fragmentation can result appealing also in data publication scenarios. In fact, data fragments can be seen as different (vertical) views that a data holder can release to external parties to satisfy their demand for information while at the same time guaranteeing that confidential information is not disclosed. The problem of computing data views in a way that explicitly takes into consideration both privacy needs

and visibility requirements makes however the data fragmentation problem far from trivial. In particular, ensuring some meaningful form of minimality of the computed fragments (to the aim of avoiding unnecessary fragmentation), makes the problem NP-hard [10].

In this paper we propose a new modeling of the fragmentation problem that exploits the representation of confidentiality and visibility constraints as Boolean formulas, and of fragments as truth assignments over Boolean variables corresponding to attributes in the original relation. In this way, the computation of a fragmentation that satisfies the given constraints greatly relies on the efficiency with which Boolean formulas are manipulated and represented. Since the classical methods for operating on Boolean formulas are impractical for large-scale problems, we exploit reduced Ordered Binary Decision Diagrams (OBDDs). OBDDs are a canonical form for Boolean formulas that can be manipulated efficiently, thus being suitable for compactly representing large Boolean formulas [18]. The size of an OBDD does not directly depend on the size of the corresponding formula and therefore the complexity of the Boolean operators depends on the OBBD size only. Although the size of an OBDD could be, in the worst case, exponential in the number of variables appearing in the formula, the majority of Boolean formulas can be represented by very compact OBDDs. Our approach then consists in transforming all the inputs of the fragmentation problem into Boolean formulas, and in exploiting their representation through OBDDs to process different constraints simultaneously, and to easily check whether a fragmentation reflects the given confidentiality and visibility constraints.

The remainder of this paper is organized as follows. Section 2 introduces confidentiality and visibility constraints, and describes the fragmentation problem. Section 3 presents our modeling of the problem, defining OBDDs corresponding to constraints and truth assignments satisfying them, and illustrating how truth assignments can be composed for computing a solution to the problem. Section 4 illustrates an algorithm exploiting the OBDD-based modeling for determining a fragmentation. Section 5 discusses related work. Finally, Section 6 reports our conclusions.

2 Preliminary Concepts

We consider a scenario where, consistently with other proposals (e.g., [1, 8, 10, 19]), the data undergoing possible external release are represented with a single relation r over a relation schema $R(a_1, \ldots, a_n)$. We use standard notations of relational database theory and, when clear from the context, we will use R to denote either the relation schema R or the set $\{a_1, \ldots, a_n\}$ of attributes in R. We consider two kinds of constraints on data: *confidentiality constraints* that impose restrictions on the (joint) visibility of values of attributes in R, and *visibility constraints* expressing requirements on data views [8, 10].

Definition 1 (Confidentiality constraint). *Given a relation schema* $R(a_1, \ldots, a_n)$*, a* confidentiality constraint *c over R is a subset of* $\{a_1, \ldots, a_n\}$.

CensusData

SSN	Name	Birth	ZIP	Job	Employer
123-45-6789	Alice	56/12/07	94101	spy	special units
234-56-7654	Bob	79/03/01	94123	agent	FBI
345-67-8123	Carol	51/11/11	95173	sniper	army
456-78-9876	David	67/05/09	96234	undercover agent	FBI
567-89-0534	Emma	80/11/12	94143	scientist	army

(a)

$\mathcal{C}$

$c_1 = \{\texttt{SSN}\}$
$c_2 = \{\texttt{Name, Job}\}$
$c_3 = \{\texttt{Name, Employer}\}$
$c_4 = \{\texttt{Birth, ZIP, Job}\}$
$c_5 = \{\texttt{Birth, ZIP, Employer}\}$

(b)

$\mathcal{V}$

$v_1 = \texttt{ZIP} \vee \texttt{Employer}$
$v_2 = \texttt{SSN} \vee (\texttt{Birth} \wedge \texttt{ZIP})$
$v_3 = \texttt{Job} \wedge \texttt{Employer}$

(c)

Fig. 1. An example of relation (a), confidentiality (b) and visibility constraints (c)

Confidentiality constraints state that the values assumed by an attribute (*singleton constraint*) or the associations among the values of a given set of attributes (*association constraint*) are sensitive and should not be visible. More precisely, a singleton constraint $\{a\}$ states that the values of attribute a should not be visible. An association constraint $\{a_{i_1}, \ldots, a_{i_m}\}$ states that the values of attributes $a_{i_1}, \ldots, a_{i_m}$ should not be visible in association. For instance, Figure 1(b) illustrates one singleton (c_1) and four association (c_2,...,c_5) constraints for relation CensusData in Figure 1(a).

Visibility constraints are defined as follows.

Definition 2 (Visibility constraint). *Given a relation schema $R(a_1, \ldots, a_n)$, a* visibility constraint *v over R is a monotonic Boolean formula over attributes in R.*

Intuitively, a visibility constraint imposes the release of an attribute or the joint release of a set of attributes. Visibility constraint $v{=}a$ states that the values assumed by attribute a must be visible. Visibility constraint $v{=}v_i{\wedge}v_j$ states that v_i and v_j must be jointly visible (e.g., constraint v_3 in Figure 1(c) requires the joint release of attributes `Job` and `Employer` since the associations between their values must be visible). Visibility constraint $v{=}v_i{\vee}v_j$ states that at least one between v_i and v_j must be visible (e.g., constraint v_1 in Figure 1(c) requires that the values of attribute `ZIP` or the values of attribute `Employer` are released). Note that negations are not used in the definition of visibility constraints since they model requirements of non-visibility, which are already captured by confidentiality constraints.

Confidentiality and visibility constraints can be enforced by splitting (fragmenting) attributes in R in different sets (*fragments*). A fragmentation of relation R is a set of fragments, as formally defined in the following.

Definition 3 (Fragmentation). *Given a relation schema $R(a_1, \ldots, a_n)$, a* fragmentation *$\mathcal{F}$ of R is a set $\{F_1, \ldots, F_l\}$ of fragments, where each fragment F_i, $i = 1, \ldots, l$, is a subset of $\{a_1, \ldots, a_n\}$.*

Given a relation R, a set $\mathcal{C}$ of confidentiality constraints, and a set $\mathcal{V}$ of visibility constraints, a fragmentation $\mathcal{F}$ of R is *correct* if it satisfies all the confidentiality constraints in $\mathcal{C}$ and all the visibility constraints in $\mathcal{V}$. Formally, a correct fragmentation is defined as follows.

Definition 4 (Correctness). *Given a relation schema $R(a_1, \ldots, a_n)$, a set $\mathcal{C}$ of confidentiality constraints over R, and a set $\mathcal{V}$ of visibility constraints over R, a fragmentation $\mathcal{F}$ of R is* correct *with respect to $\mathcal{C}$ and $\mathcal{V}$ iff:*

F_1

Birth	ZIP
56/12/07	94101
79/03/01	94123
51/11/11	95173
67/05/09	96234
80/11/12	94143

F_2

Job	Employer
spy	special units
agent	FBI
sniper	army
undercover agent	FBI
scientist	army

Fig. 2. An example of correct fragmentation of relation CensusData in Figure 1(a)

1. $\forall c \in \mathcal{C}$, $\forall F \in \mathcal{F}$: $c \not\subseteq F$ (*confidentiality*);
2. $\forall v \in \mathcal{V}$, $\exists F \in \mathcal{F}$: F *satisfies* v (*visibility*);
3. $\forall F_i, F_j \in \mathcal{F}$, $i \neq j$: $F_i \cap F_j = \emptyset$ (*un-linkability*).

Condition 1 ensures that neither sensitive attributes nor sensitive associations are visible in a fragment. Condition 2 ensures that visibility constraints are satisfied. Condition 3 ensures that fragments do not have common attributes and therefore that association constraints cannot be violated by possibly joining fragments. We note that singleton constraints can be satisfied only by not releasing the corresponding sensitive attributes. Association constraints can be satisfied either by not releasing at least one of the attributes in the constraints, or by distributing the attributes among different (un-linkable) fragments. Visibility constraints are satisfied by ensuring that each constraint is satisfied by at least one fragment.

Given a set of confidentiality and visibility constraints, we are interested in a fragmentation that does not split attributes among fragments when it is not necessary for constraint satisfaction. The rationale is that maintaining a set of attributes in the same fragment releases their values and also their associations, thus maximizing the visibility over the data. Our goal is then to compute a *minimal* fragmentation, that is, a fragmentation that does not include fragments that can be merged without violating confidentiality constraints. The problem of computing a minimal fragmentation can be defined as follows.

*Problem 1 (*Min-Frag*).* Given a relation schema $R(a_1, \ldots, a_n)$, a set $\mathcal{C}$ of confidentiality constraints over R, and a set $\mathcal{V}$ of visibility constraints over R, determine (if it exists) a correct fragmentation $\mathcal{F}$ of R with respect to $\mathcal{C}$ and $\mathcal{V}$ such that there does not exist another correct fragmentation $\mathcal{F}'$ obtained by merging fragments in $\mathcal{F}$.

For instance, the fragmentation in Figure 2 is a minimal fragmentation since merging F_1 with F_2 would violate confidentiality constraints c_4 and c_5.

3 OBDD-based Modeling of the Fragmentation Problem

We model the fragmentation problem as the problem of managing a set of Boolean formulas that are conveniently represented through *reduced and ordered binary decision diagrams* (OBDDs) [3]. OBDDs allow us to efficiently manipulate confidentiality and visibility constraints, and to easily compute a minimal fragmentation (see Section 4).

$\mathcal{B}$	$\mathcal{C}$	$\mathcal{V}$
SSN	c_1 = SSN	v_1 = ZIP∨Employer
Name	c_2 = Name∧Job	v_2 = SSN∨(Birth∧ZIP)
Birth	c_3 = Name∧Employer	v_3 = Job∧Employer
ZIP	c_4 = Birth∧ZIP∧Job	
Job	c_5 = Birth∧ZIP∧Employer	
Employer		

Fig. 3. Boolean interpretation of the inputs of the MIN-FRAG problem in Figure 1

3.1 OBDD Representation of Constraints

In our modeling, attributes in R are interpreted as Boolean variables. Visibility constraints have already been defined as Boolean formulas (Definition 2). Each confidentiality constraint in $\mathcal{C}$ can be represented as the conjunction of the variables corresponding to the attributes in the constraint. For instance, Figure 3 represents the Boolean interpretation of the inputs of the MIN-FRAG problem in Figure 1, where $\mathcal{B}$ denotes the set of Boolean variables.

We use OBDDs as an effective and efficient solution to represent and manipulate Boolean formulas. An OBDD represents a Boolean formula as a rooted directed acyclic graph with two leaf nodes labeled 1 (true) and 0 (false), respectively, corresponding to the truth values of the formula. Each internal node in the graph represents a Boolean variable in the formula and has two outgoing edges, labeled 1 and 0, representing the assignment of values 1 and 0, respectively, to the variable. The variables occur in the same order on all the paths of the graph. Also, to guarantee a compact representation of the Boolean formula, the subgraphs rooted at the two direct descendants of each internal node in the graph are disjoint, and any possible pair of subgraphs rooted at two different nodes are not isomorphic. Figure 4 and Figure 5 illustrate the OBDDs of the Boolean formulas in Figure 3 that model the confidentiality and visibility constraints, respectively, in Figure 1. Here and in the following, edges labeled 1 are represented by solid lines, and edges labeled 0 are represented by dashed lines. A truth assignment to the Boolean variables in a formula corresponds to a path from the root to one of the leaf nodes of the OBDD of the formula. The outgoing edge of a node in the path is the value assigned to the variable represented by the node. For instance, with respect to the OBDD of v_1 in Figure 5, path ⟨ZIP, Employer, 1⟩ represents truth assignment [ZIP=0, Employer=1] since the edge in the path outgoing from node ZIP is labeled 0, and the edge in the path outgoing from node Employer is labeled 1. We call *one-paths* (*zero-paths*, respectively) all the paths of an OBDD that reach leaf node 1 (0, respectively), which correspond to the assignments that satisfy (do not satisfy, respectively) the formula. For instance, with respect to the OBDD of v_1 in Figure 5, path ⟨ZIP, Employer, 1⟩ is a one-path of the OBDD. Variables in the formula that do not occur in a path from the root to a leaf node are called *don't care* variables, that is, variables whose values do not influence the truth value of the formula. For instance, with respect to the one-path ⟨ZIP, 1⟩ of the OBDD of v_1 in Figure 5, Employer is a don't care variable. If there is at least a don't care variable along a path, the

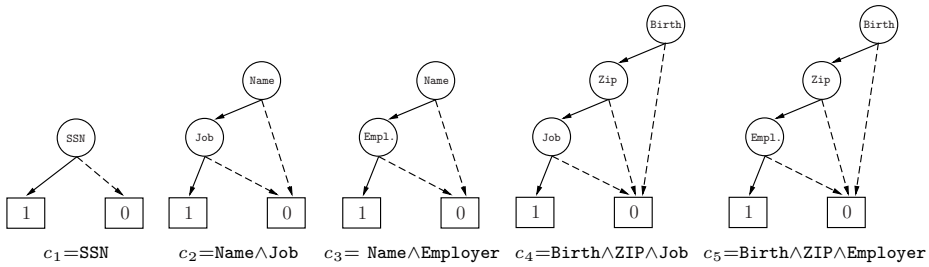

Fig. 4. OBDDs representing the confidentiality constraints in Figure 3

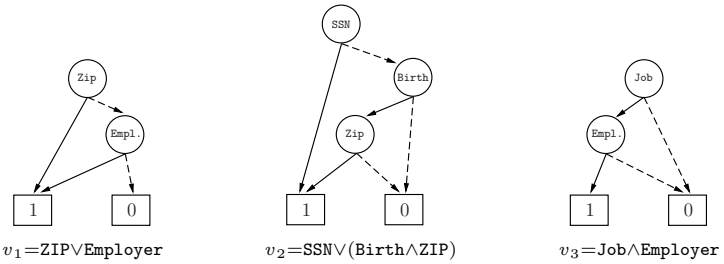

Fig. 5. OBDDs representing the visibility constraints in Figure 3

corresponding truth assignment is *partial* (in contrast to *complete*), since only a subset of the variables in the formula is assigned a value. We note that a partial truth assignment with k don't care variables is a compact representation of a set of 2^k complete truth assignments, obtained by assigning to the don't care variables value 1 or 0. For instance, the OBDD of v_1 in Figure 5 has two one-paths, corresponding to truth assignments [ZIP=1] and [ZIP=0, Employer=1]. Partial truth assignment [ZIP=1] is a shorthand for [ZIP=1, Employer=1] and [ZIP=1, Employer=0], where don't care variable Employer has value 1 and 0, respectively.

3.2 Truth Assignments

In the Boolean modeling of the fragmentation problem, a fragment $F \in \mathcal{F}$ can be interpreted as a complete truth assignment, denoted I_F, over the set $\mathcal{B}$ of Boolean variables. Function I_F assigns value 1 to each variable corresponding to an attribute in F, and value 0 to all the other variables. A fragmentation is then represented by a set of complete truth assignments, which is formally defined as follows.

Definition 5 (Set of truth assignments). *Given a set $\mathcal{B}$ of Boolean variables, a* set $\mathcal{I}$ of truth assignments *is a set $\{I_1, \ldots, I_l\}$ of functions, such that each I_i in $\mathcal{I}$, $i = 1, \ldots, l$, is defined as $I_i : \mathcal{B} \rightarrow \{0,1\}$.*

With a slight abuse of notation, we use I to denote also the list of truth values assigned by I to variables in $\mathcal{B}$. For instance, fragmentation $\mathcal{F}$ in Figure 2 corresponds to the set $\mathcal{I}$=$\{I_{F_1}$,$I_{F_2}\}$ of truth assignments, where I_{F_1} = [SSN=0, Name=0, Birth=1, ZIP=1, Job=0, Employer=0] and I_{F_2} = [SSN=0, Name=0, Birth=0, ZIP=0, Job=1, Employer=1]. Given a Boolean formula f, defined over Boolean variables $\mathcal{B}$, and a truth assignment I, $I(f)$ denotes the result of the evaluation of f with respect to truth assignment I. A set $\mathcal{I}$ of truth assignments corresponds to a correct fragmentation if it satisfies all confidentiality and visibility constraints and each Boolean variable is set to true by at most one truth assignment, as formally defined in the following.

Definition 6 (Correct set of truth assignments). *Given a set $\mathcal{B}$ of Boolean variables, a set $\mathcal{C}$ of confidentiality constraints over $\mathcal{B}$, and a set $\mathcal{V}$ of visibility constraints over $\mathcal{B}$, a set $\mathcal{I}$ of truth assignments is* correct *with respect to $\mathcal{C}$ and $\mathcal{V}$ iff:*

1. $\forall c \in \mathcal{C}$, $\forall I \in \mathcal{I}$: $I(c) = 0$ (*confidentiality*);
2. $\forall v \in \mathcal{V}$, $\exists I \in \mathcal{I}$: $I(v) = 1$ (*visibility*);
3. $\forall I_i, I_j \in \mathcal{I}, i \neq j$, $\forall a \in \mathcal{B}$ *s.t.* $I_i(a) = 1$: $I_j(a) = 0$ (*un-linkability*).

Condition 1 ensures that the evaluation of any confidentiality constraint with respect to any truth assignment (fragment) is false (i.e., no confidentiality constraint is violated). Condition 2 ensures that, for each visibility constraint, there is at least one truth assignment (fragment) that makes the visibility constraint true (i.e., all visibility constraints are satisfied). Condition 3 ensures that there is at most one truth assignment (fragment) that sets a variable to true (i.e., different fragments do not have common attributes). It is immediate to see that a set of truth assignments is correct with respect to $\mathcal{C}$ and $\mathcal{V}$ iff the corresponding fragmentation is correct with respect to $\mathcal{C}$ and $\mathcal{V}$ (i.e., Definition 6 is equivalent to Definition 4). The correctness of a set $\mathcal{I}$ of truth assignments can be efficiently verified by using the OBDDs representing the confidentiality and visibility constraints: *i)* each assignment I must correspond to a zero-path in all the OBDDs of the confidentiality constraints; and *ii)* for each visibility constraint, at least one assignment I must correspond to a one-path in the OBDD of the constraint. For instance, consider the OBDDs of confidentiality and visibility constraints in Figure 4 and Figure 5 and the set $\mathcal{I} = \{I_{F_1}, I_{F_2}\}$ of truth assignments representing the fragmentation in Figure 2. $\mathcal{I}$ is correct, since: *1)* I_{F_1} and I_{F_2} correspond to zero-paths of the OBDDs of the confidentiality constraints (confidentiality); *2)* I_{F_2} corresponds to a one-path of the OBDDs of v_1 and v_3, and I_{F_1} corresponds to a one-path of the OBDD of v_2 (visibility); and *3)* each variable in $\mathcal{B}$ is set to 1 by at most one between I_{F_1} and I_{F_2} (un-linkability).

Note that given two fragments F_i and F_j and the corresponding truth assignments I_{F_i} and I_{F_j}, the truth assignment representing merged fragment F_{ij}=$F_i \cup F_j$ is $I_{F_{ij}}$=$I_{F_i} \vee I_{F_j}$. The Min-Frag problem can now be reformulated as follows.

*Problem 2 (*Min-Truth*).* Given a set $\mathcal{B}$ of Boolean variables, a set $\mathcal{C}$ of confidentiality constraints over $\mathcal{B}$, and a set $\mathcal{V}$ of visibility constraints over $\mathcal{B}$, determine

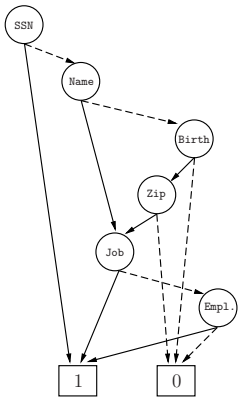

Fig. 6. OBDD representing the disjunction $(c_1 \vee c_2 \vee c_3 \vee c_4 \vee c_5)$

(if it exists) a correct set $\mathcal{I}$ of truth assignments such that there does not exist another correct set $\mathcal{I}'$ of truth assignments obtained by combining two truth assignments in $\mathcal{I}$ through the OR operator.

Our approach to solve the MIN-TRUTH problem exploits properties of the OBDDs to efficiently check if a set of truth assignments is correct. In principle, a set of truth assignments should be checked for correctness against each confidentiality and each visibility constraint. We can cut down on such controls by noting that if a truth assignment I does not make true any confidentiality constraint, Boolean formula $c_1 \vee \ldots \vee c_m$ evaluates to false with respect to I. Also, if truth assignment I makes true at least one of the confidentiality constraints in $\mathcal{C}$, Boolean formula $c_1 \vee \ldots \vee c_m$ evaluates to true with respect to I. In other words, we can check all the confidentiality constraints together in a single step. Formally, this observation is expressed as follows.

Observation 1 *Given a set $\mathcal{B} = \{a_1, \ldots, a_n\}$ of Boolean variables, a set $\mathcal{C} = \{c_1, \ldots, c_m\}$ of confidentiality constraints over $\mathcal{B}$, and a truth assignment I:*

$$\forall c \in \mathcal{C},\ I(c) = 0 \iff I(c_1 \vee \ldots \vee c_m) = 0.$$

To verify whether a truth assignment I satisfies the confidentiality constraints, we can then simply check if I characterizes a zero-path of the OBDD representing the disjunction of confidentiality constraints. For instance, consider the confidentiality constraints in Figure 3, the OBDD representing their disjunction in Figure 6, and truth assignment I_{F_2} = [`SSN`=0, `Name`=0, `Birth`=0, `ZIP`=0, `Job`=1, `Employer`=1], representing fragment F_2 in Figure 2. I_{F_2} corresponds to a zero-path in the OBDD in Figure 6, implying that I_{F_2} does not violate the confidentiality constraints.

For each visibility constraint v, a correct set of truth assignments must include at least a truth assignment I satisfying v, while not violating confidentiality constraints (i.e., $I(v)$=1 and $I(c_1 \vee \ldots \vee c_m)$=0). This is equivalent to say

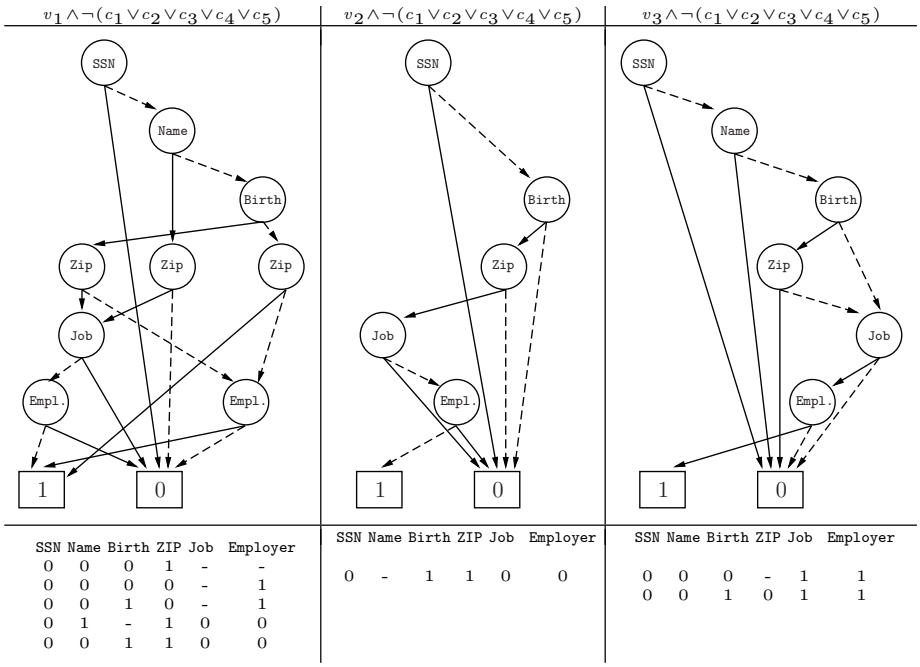

Fig. 7. OBDDs representing the composition of each visibility constraint in Figure 5 with the negated disjunction of the confidentiality constraints in Figure 4, and their one-paths

that the evaluation of Boolean formula $v \wedge \neg(c_1 \vee \ldots \vee c_m)$ with respect to truth assignment I is true, as formally observed in the following.

Observation 2 *Given a set $\mathcal{B} = \{a_1, \ldots, a_n\}$ of Boolean variables, a set $\mathcal{C} = \{c_1, \ldots, c_m\}$ of confidentiality constraints over $\mathcal{B}$, a visibility constraint v over $\mathcal{B}$, and a truth assignment I:*

$$I(v) = 1 \text{ and } I(c_1 \vee \ldots \vee c_m) = 0 \iff I(v \wedge \neg(c_1 \vee \ldots \vee c_m)) = 1.$$

In other words, the one-paths of the OBDD, denoted O_i, of Boolean formula $v_i \wedge \neg(c_1 \vee \ldots \vee c_m)$, represent in a compact way all and only the truth assignments that satisfy v_i and that do not violate any confidentiality constraint. Note that all variables in $\mathcal{B}$ not appearing in the formula are considered as don't care variables. For instance, consider the confidentiality and visibility constraints in Figure 4 and in Figure 5. Figure 7 illustrates the OBDDs of formulas $v_i \wedge \neg(c_1 \vee \ldots \vee c_5)$, $i = 1, \ldots, 3$, along with their one-paths. In the figure and in the remainder of the paper, we use '-' as value for the don't care variables. For instance, attribute `Name` does not appear in the OBDD representing $v_2 \wedge \neg(c_1 \vee \ldots \vee c_5)$ and therefore it appears as a don't care variable in the one-path of O_2 (i.e., [`SSN`=0, `Name`=-, `Birth`=1, `ZIP`=1, `Job`=0, `Employer`=0]). To satisfy Condition 1 (confidentiality) and Condition 2 (visibility) in Definition 6,

a set $\mathcal{I}$ of truth assignments must include, for each $v_i \in \mathcal{V}$, one truth assignment in the set of one-paths of O_i. However, not all the sets of truth assignments that include one of the one-paths of O_i for each $v_i \in \mathcal{V}$ are correct, since they may violate Condition 3 in Definition 6 (un-linkability). In the following, we discuss how to combine truth assignments representing one-paths of $O_1, \ldots, O_k$ to incrementally compute a correct set $\mathcal{I}$ of truth assignments. We note that one-paths of O_i may represent partial truth assignments, while a correct set of truth assignments is composed of complete assignments only (Definition 5). As a consequence, don't care variables must be set either to 0 or 1 before inserting one-paths of $O_1, \ldots, O_k$ into $\mathcal{I}$.

3.3 Comparison of Assignments

Goal of our approach is to incrementally create a correct set of truth assignments that solves the MIN-TRUTH problem and that corresponds to a correct and minimal fragmentation. To this purpose, we first introduce the concepts of *linkable* and *mergeable* truth assignments.

Definition 7 (Linkable truth assignments). *Given two assignments I_i and I_j over Boolean variables $\mathcal{B}$, we say that I_i and I_j are* linkable *iff* $\exists a \in \mathcal{B} : I_i(a) = I_j(a) = 1$.

According to Definition 7, two assignments are linkable iff there is a Boolean variable in $\mathcal{B}$ such that the truth value of the variable is 1 with respect to the given assignments. Intuitively, this implies that the fragments corresponding to them have an attribute in common. For instance, the two assignments [`SSN`=0, `Name`=0, `Birth`=0, `ZIP`=-, `Job`=1, `Employer`=1] and [`SSN`=0, `Name`=0, `Birth`=0, `ZIP`=-, `Job`=1, `Employer`=-] are linkable since they both assign 1 to variable `Job`.

Definition 8 (Mergeable truth assignments). *Given two assignments I_i and I_j over Boolean variables $\mathcal{B}$, we say that I_i and I_j are* mergeable *iff $\forall a \in \mathcal{B}$ s.t. $I_i(a)$=1, $I_j(a)$=1 or $I_j(a)$=- and vice versa.*

According to Definition 8, two truth assignments are mergeable iff for each variable a in $\mathcal{B}$ the truth values of the variable in the two assignments are not in contrast, where being in contrast for variable a means that a is assigned 1 by one assignment while being assigned 0 by the other one. Intuitively, two mergeable assignments define the truth value of variables in a way that they can be represented through a single assignment. As an example, consider the two assignments [`SSN`=0, `Name`=-, `Birth`=0, `ZIP`=-, `Job`=-, `Employer`=1] and [`SSN`=0, `Name`=0, `Birth`=-, `ZIP`=1, `Job`=-, `Employer`=-]. For each variable set to 1 in one of these assignments, the correspondent truth value in the other assignment is either 1 or -, and therefore the two assignments are mergeable. Assignments [`SSN`=0, `Name`=0, `Birth`=1, `ZIP`=0, `Job`=-, `Employer`=1] and [`SSN`=0, `Name`=0, `Birth`=1, `ZIP`=1, `Job`=0, `Employer`=-] are linkable (`Birth` is set to 1 by both assignments) but not mergeable since there is a conflict on variable `ZIP`. Note

$\odot$	0	1	-
0	0	n.a.	0
1	n.a.	1	1
-	0	1	-

Fig. 8. Assignment composition operator

that the presence of don't care variables does not influence the linkability or mergeability of two truth assignments.

Mergeable assignments can be composed according to the composition operator $\odot$ in Figure 8. The composition of two mergeable truth assignments I_i and I_j results in a new truth assignment, where the truth value of a variable coincides with its truth value in the assignment in which it does not appear as a don't care variable. If a variable appears as a don't care variable in both I_i and I_j, then its value in the new assignment remains don't care. For instance, assignments [`SSN`=0, `Name`=-, `Birth`=0, `ZIP`=-, `Job`=-, `Employer`=1] and [`SSN`=0, `Name`=0, `Birth`=-, `ZIP`=1, `Job`=-, `Employer`=-] are mergeable and the result of their composition is assignment [`SSN`=0, `Name`=0, `Birth`=0, `ZIP`=1, `Job`=-, `Employer`=1].

4 Computing a Minimal Set of Truth Assignments

Figure 9 illustrates our heuristic algorithm for computing a solution to the MIN-TRUTH problem (Problem 2). The algorithm takes as input a set $\mathcal{B}$ of Boolean variables, a set $\mathcal{C} = \{c_1, \ldots, c_m\}$ of confidentiality constraints, and a set $\mathcal{V} = \{v_1, \ldots, v_k\}$ of visibility constraints. It incrementally builds a correct set of truth assignments by inserting, for each v in $\mathcal{V}$, a truth assignment satisfying v while not violating confidentiality constraints. A truth assignment can be inserted in an existing set either as a new truth assignment (if it is not linkable with any assignment in the set) or by composing it with an existing assignment (if it is linkable and mergeable with an assignment in the set). It returns a correct and minimal set $\mathcal{I}_{sol}$ of truth assignments, if such a set exists; it returns NULL, otherwise.

The algorithm first defines, for each $v_i \in \mathcal{V}$, the OBDD representing Boolean formula $v_i \wedge \neg(c_1 \vee \ldots \vee c_m)$, extracts the set $\mathcal{I}_{v_i}$ of one-paths, and orders them by decreasing number of don't care variables (lines 1-4). The reason for this ordering is that truth assignments with a high number of don't care variables impose less constraints on subsequent choices, and therefore are less likely to be in contrast with them. Also, $\mathcal{I}_{v_1}, \ldots, \mathcal{I}_{v_k}$ are ordered by increasing number of truth assignments (line 5). The reason for such ordering is to consider first sets for which fewer truth assignments are possible.

The algorithm calls function **DefineAssignments** (line 6), which receives as input a set $\mathcal{I}_{sol}$ of truth assignments and an integer number i, $1 \leq i \leq k$, indicating that $\mathcal{I}_{sol}$ has been obtained by combining one truth assignment from

INPUT
$\mathcal{B} = \{a_1,\ldots,a_n\}$ /* Boolean variables */
$\mathcal{C} = \{c_1,\ldots,c_m\}$ /* Boolean interpretation of confidentiality constraints */
$\mathcal{V} = \{v_1,\ldots,v_k\}$ /* visibility constraints */

OUTPUT
$\mathcal{I}_{sol} = \{I_1,\ldots,I_l\}$ /* correct and minimal set of truth assignments */

MAIN
1: **for each** $v_i \in \mathcal{V}$ **do** /* define the OBDDs representing the constraints */
2: let O_i be the OBDD representing $v_i \wedge \neg(c_1 \vee \ldots \vee c_m)$
3: let $\mathcal{I}_{v_i}$ be the set of one-paths of O_i
4: order $\mathcal{I}_{v_i}$ by decreasing number of -
5: let $[\mathcal{I}_1,\ldots,\mathcal{I}_k]$ be the list obtained ordering $\{\mathcal{I}_{v_1},\ldots,\mathcal{I}_{v_k}\}$ by increasing number of one-paths
6: $\mathcal{I}_{sol}$:= **DefineAssignments**($\emptyset$,1) /* compute a correct set of truth assignments */
7: **if** $\mathcal{I}_{sol} \neq$ NULL **then** /* a correct set of truth assignments exists */
8: **for** i:=1,...,$(|\mathcal{I}_{sol}| - 1)$ **do** /* compose truth assignments to make $\mathcal{I}_{sol}$ minimal */
9: **for** j:=$(i+1)$,...,$|\mathcal{I}_{sol}|$ **do**
10: **if** MERGEABLE($\mathcal{I}_{sol}[i]$,$\mathcal{I}_{sol}[j]$) **then**
11: $\mathcal{I}_{sol}[i] := \mathcal{I}_{sol}[i] \odot \mathcal{I}_{sol}[j]$
12: remove $\mathcal{I}_{sol}[j]$ from $\mathcal{I}_{sol}$
13: **for each** $I \in \mathcal{I}_{sol}$ **do** assign 0 to don't care variables in I
14: **return**($\mathcal{I}_{sol}$)

DEFINEASSIGNMENTS($\mathcal{I}_{sol}$,i)
15: **for** j:=1,...,$|\mathcal{I}_i|$ **do**
16: *satisfied* := TRUE /* true if $\mathcal{I}'_{sol}$ includes a truth assignment from $\mathcal{I}_i$ */
17: *LinkableAssignments* := $\{I \in \mathcal{I}_{sol}$:LINKABLE($\mathcal{I}_i[j]$,$I$)$\}$ /* assignments linkable with $\mathcal{I}_i[j]$ */
18: $\mathcal{I}'_{sol} := \mathcal{I}_{sol} \setminus$ *LinkableAssignments* /* remove assignments linkable with $\mathcal{I}_i[j]$ */
19: $I_{new} := \mathcal{I}_i[j]$
20: **while**(*satisfied* AND *LinkableAssignments* $\neq \emptyset$) **do**
21: I:= **ExtractAssignment**(*LinkableAssignments*)
22: **if** MERGEABLE(I_{new},I) **then** $I_{new} := I_{new} \odot I$ /* compose truth assignments */
23: **else** *satisfied* := FALSE /* I is linkable but not mergeable with I_{new} */
24: **if** *satisfied* **then**
25: $\mathcal{I}'_{sol} := \mathcal{I}_{sol} \cup \{I_{new}\}$
26: **if** $i=k$ **then return**($\mathcal{I}'_{sol}$) /* $\mathcal{I}'_{sol}$ is correct */
27: $\mathcal{I}'_{sol}$:= **DefineAssignments**($\mathcal{I}'_{sol}$,$i+1$) /* recursive call */
28: **if** $\mathcal{I}'_{sol} \neq$ NULL **then return**($\mathcal{I}'_{sol}$) /* $\mathcal{I}'_{sol}$ is correct */
29: **return**(NULL)

Fig. 9. Algorithm that computes a correct and minimal set of truth assignments

each $\mathcal{I}_j$, $j = 1,\ldots,(i-1)$. Function **DefineAssignments** tries to insert into $\mathcal{I}_{sol}$ a truth assignment that belongs to $\mathcal{I}_i$, possibly composing it, through the $\odot$ operator, with a truth assignment in $\mathcal{I}_{sol}$ if they are linkable and mergeable. For the j-th truth assignment $\mathcal{I}_i[j]$ in $\mathcal{I}_i$ ($j = 1,\ldots,|\mathcal{I}_i|$), the function first identifies the set *LinkableAssignments* of truth assignments in $\mathcal{I}_{sol}$ linkable with $\mathcal{I}_i[j]$ (line 17) and iteratively composes them with $\mathcal{I}_i[j]$, obtaining truth assignment I_{new} (lines 19-23). We note that mergeable assignments that are not linkable are kept separate, even if they could be composed without violating any confidentiality constraint. In fact, by composing a pair of not linkable truth assignments, the algorithm would discard, without evaluation, all the correct solutions where the two truth assignments are kept separate. If $\mathcal{I}_i[j]$ and *LinkableAssignments* are not mergeable, $\mathcal{I}_i[j]$ can be inserted into $\mathcal{I}_{sol}$ neither as an un-linkable assignment nor by composing it with existing assignments (variable *satisfied*=FALSE). Otherwise, $\mathcal{I}'_{sol}$ is obtained removing *LinkableAssignments* and including I_{new} into $\mathcal{I}_{sol}$ (line 25). If i=k, $\mathcal{I}'_{sol}$ represents a correct fragmentation and is returned (line 26); **DefineAssignments** is recursively called over $\mathcal{I}'_{sol}$ and $i+1$,

otherwise (line 27). If the set $\mathcal{I}'_{sol}$ resulting from the recursive call is not NULL, it is correct and is returned (line 28). If no assignment in $\mathcal{I}_i$ can be inserted into $\mathcal{I}_{sol}$, the function returns NULL (line 29).

The set $\mathcal{I}_{sol}$ computed by function **DefineAssignments** may not be minimal, since it may include mergeable truth assignments that are not linkable. The algorithm therefore possibly composes each truth assignment $\mathcal{I}_{sol}[i]$ with each $\mathcal{I}_{sol}[j]$, $j > i$ (lines 7-12). We note that it is not necessary to check the truth assignment resulting from the composition with assignments $\mathcal{I}_{sol}[l]$, $l < i$, since if $\mathcal{I}_{sol}[l]$ and $\mathcal{I}_{sol}[i]$ are not mergeable, then also $\mathcal{I}_{sol}[l]$ and $\mathcal{I}_{sol}[i] \odot \mathcal{I}_{sol}[j]$ are not mergeable. The algorithm finally assigns 0 to don't care variables in $\mathcal{I}_{sol}$ and returns $\mathcal{I}_{sol}$ (lines 13-14).

Example 1. Consider relation CensusData and the confidentiality and visibility constraints over it in Figure 1. First, the algorithm builds O_1, O_2, and O_3 in Figure 7, representing the conjunction of each visibility constraint (v_1, v_2, and v_3) with the disjunction ($c_1 \vee \ldots \vee c_5$) of confidentiality constraints. It then extracts their one-paths, orders the one-paths of each $\mathcal{I}_v$ by decreasing number of -, and orders the set of $\mathcal{I}_v$ by increasing number of one-paths. The ordered list [$\mathcal{I}_1$,$\mathcal{I}_2$,$\mathcal{I}_3$] of sets of truth assignments is illustrated in Figure 10, where $\mathcal{I}_1 = \mathcal{I}_{v_2}$, $\mathcal{I}_2 = \mathcal{I}_{v_3}$, and $\mathcal{I}_3 = \mathcal{I}_{v_1}$. Figure 10 presents the recursive calls to function **DefineAssignments** illustrating for each execution: the value of input parameters $\mathcal{I}_{sol}$ and i; the candidate truth assignment $\mathcal{I}_i[j]$ in $\mathcal{I}_i$; the set *LinkableAssignments* of assignments in $\mathcal{I}_{sol}$ that are linkable with $\mathcal{I}_i[j]$; the iterative composition of $\mathcal{I}_i[j]$ with the assignments in *LinkableAssignments* and the resulting truth assignment $\mathcal{I}_{new}$; and the computed set $\mathcal{I}'_{sol}$. In the figure, for simplicity, we do not report attribute names in truth assignments and we assume that truth values are assigned, in the order, to `SSN`, `Name`, `Birth`, `ZIP`, `Job`, `Employer`. The fragmentation corresponding to the set of truth assignments returned by the algorithm is illustrated in Figure 2.

The correctness and complexity of the algorithm in Figure 9 are stated by the following theorems. The proofs of the theorems are omitted for space constraints.

Theorem 1 (Correctness). *Given a set $\mathcal{B}$ of Boolean variables, a set $\mathcal{C}$ of confidentiality constraints over $\mathcal{B}$, and a set $\mathcal{V}$ of visibility constraints over $\mathcal{B}$, the algorithm in Figure 9 terminates and computes, if it exists, a correct and minimal set of truth assignments with respect to $\mathcal{C}$ and $\mathcal{V}$.*

Theorem 2 (Complexity). *Given a set $\mathcal{B}$ of Boolean variables, a set $\mathcal{C}$ of confidentiality constraints over $\mathcal{B}$, and a set $\mathcal{V}$ of visibility constraints over $\mathcal{B}$, the complexity of the algorithm in Figure 9 is $O(\prod_{v \in \mathcal{V}} |\mathcal{I}_v| \cdot |\mathcal{B}| + (|\mathcal{V}| + |\mathcal{C}|)2^{|\mathcal{B}|})$ in time, where $\mathcal{I}_v$ is the set of one-paths of the OBDD representing $v \wedge \neg(c_1 \vee \ldots \vee c_m)$.*

The computational cost of the algorithm is obtained as the sum of the cost of building the OBDDs, which is $O((|\mathcal{V}| + |\mathcal{C}|)2^{|\mathcal{B}|})$, and the cost of determining $\mathcal{I}_{sol}$ through recursive function **DefineAssignments**, which is $O(\prod_{v \in \mathcal{V}} |\mathcal{I}_v| \cdot$

$\mathcal{I}_1$:=([0,-,1,1,0,0])
$\mathcal{I}_2$:=([0,0,0,-,1,1],[0,0,1,0,1,1])
$\mathcal{I}_3$:=([0,0,0,1,-,-],[0,0,0,0,-,1],[0,0,1,0,-,1],[0,1,-,1,0,0],[0,0,1,1,0,0])
DefineAssignments(∅,1)
  $\mathcal{I}_1$[1]=[0,-,1,1,0,0]
  LinkableAssignments:=∅
  I_{new}:=[0,-,1,1,0,0]
  $\mathcal{I}'_{sol}$:={[0,-,1,1,0,0]}
  DefineAssignments({[0,-,1,1,0,0]},2)
    $\mathcal{I}_2$[1]=[0,0,0,-,1,1]
    LinkableAssignments:=∅
    I_{new}:=[0,0,0,-,1,1]
    $\mathcal{I}'_{sol}$:{[0,-,1,1,0,0],[0,0,0,-,1,1]}
    DefineAssignments({[0,-,1,1,0,0],[0,0,0,-,1,1]},3)
      $\mathcal{I}_3$[1]=[0,0,0,1,-,-]
      LinkableAssignments:={[0,-,1,1,0,0]}
      I_{new}:=[0,0,0,1,-,-]
      MERGEABLE([0,0,0,1,-,-],[0,-,1,1,0,0])=FALSE
      $\mathcal{I}_3$[2]=[0,0,0,0,-,1]
      LinkableAssignments:={[0,0,0,-,1,1]}
      I_{new}:=[0,0,0,0,-,1]
      MERGEABLE([0,0,0,0,-,1],[0,0,0,-,1,1])=TRUE
      I_{new}:=[0,0,0,0,-,1] ⊙ [0,0,0,-,1,1]:=[0,0,0,0,1,1]
      return({[0,-,1,1,0,0],[0,0,0,0,1,1]})

Fig. 10. Example of the execution of the algorithm in Figure 9 with the inputs in Figure 3

$|\mathcal{B}|$). We note that the computational cost of the construction of the OBDDs is exponential in the worst case, but in the majority of real-world applications OBBD-based approaches are computationally efficient [3, 16].

5 Related Work

Data fragmentation has been studied as a solution to enforce confidentiality constraints while ensuring an efficient query execution in outsourcing scenarios, where data are stored and managed at external honest-but-curious servers [9, 14, 20]. In particular, the proposals based on fragmentation can be classified as solutions that: *1)* combine fragmentation and encryption and split data between two fragments stored on two non-communicating servers [1], or among multiple fragments [8], possibly stored on a single server, in such a way to minimize query execution costs [6]; *2)* depart from encryption [7, 21] and satisfy confidentiality constraints by splitting the data over two fragments, one of which is stored at the data owner. Although our approach shares with these proposals the use of fragmentation for properly protecting sensitive data and/or associations, we

take into consideration a different scenario and address a different problem. In fact, our proposal considers a data publishing scenario, in contrast to data outsourcing, and aims at satisfying also visibility constraints, which have been introduced in [10] where the authors exploit SAT solvers to compute a correct fragmentation.

The work presented in this paper has some affinity with the proposals that introduce a policy based classification of the data to protect their confidentiality (e.g., [2]). Such solutions however do not use fragmentation and are concerned with returning to users query results that do not contain combinations of values that are sensitive or that can be exploited for inferring sensitive information.

Other related work is represented by proposals that introduce OBDD-based approaches for solving constraint satisfaction problems (or CSPs, e.g. [13, 15, 17]). These approaches aim at computing a truth assignment for a set of variables that satisfies a set of constraints among the variable values. The solution described in this paper differs from the techniques proposed for general constraint satisfaction problems, since our approach takes advantage of the monotonicity of confidentiality and visibility constrains and therefore fully exploits the implicit representation of sets of truth assignments provided by OBDDs. These peculiarities of the minimal fragmentation problem permit to limit the computational effort required to compute an optimal solution.

6 Conclusions

We presented a novel OBDD-based approach for computing a fragmentation that fulfills both the need of properly protecting sensitive data and the need of guaranteeing visibility requirements when a dataset is publicly released. Our modeling of the fragmentation problem relies on the interpretation of both confidentiality and visibility constraints as Boolean formulas and of fragments as truth assignments to variables. OBDDs allow us to compactly represent multiple constraints and to simply check whether a fragmentation satisfies them.

Acknowledgments

This work was supported in part by the EU within the 7FP project "PrimeLife" under grant agreement 216483, by the Italian Ministry of Research within the PRIN 2008 project "PEPPER" (2008SY2PH4), and by the Università degli Studi di Milano within the "UNIMI per il Futuro - 5 per Mille" project "PREVIOUS".

References

1. Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Motwani, R., Srivastava, U., Thomas, D., Xu, Y.: Two can keep a secret: A distributed architecture for secure database services. In: Proc. of CIDR 2005. Asilomar, CA, USA (January 2005)

2. Biskup, J., Wiese, L.: Combining consistency and confidentiality requirements in first-order databases. In: Proc. of ISC 2009. Pisa, Italy (September 2009)
3. Bryant, R.: Graph-based algorithms for Boolean function manipulation. IEEE TC 35(8), 677–691 (August 1986)
4. Cao, N., Wang, C., Li, M., Ren, K., Lou, W.: Privacy-preserving multi-keyword ranked search over encrypted cloud data. In: Proc. of INFOCOM 2011. Shanghai, China (April 2011)
5. Cimato, S., Gamassi, M., Piuri, V., Sassi, R., Scotti, F.: Privacy-aware biometrics: Design and implementation of a multimodal verification system. In: Proc. of ACSAC 2008. Anaheim, CA, USA (December 2008)
6. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Fragmentation design for efficient query execution over sensitive distributed databases. In: Proc. of ICDCS 2009. Montreal, Canada (June 2009)
7. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Keep a few: Outsourcing data while maintaining confidentiality. In: Proc. of ESORICS 2009. Saint Malo, France (September 2009)
8. Ciriani, V., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Combining fragmentation and encryption to protect privacy in data storage. ACM TISSEC 13(3), 22:1–22:33 (July 2010)
9. Damiani, E., De Capitani di Vimercati, S., Jajodia, S., Paraboschi, S., Samarati, P.: Balancing confidentiality and efficiency in untrusted relational DBMSs. In: Proc. of CCS 2003. Washington, DC, USA (October 2003)
10. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Fragments and loose associations: Respecting privacy in data publishing. Proc. of the VLDB Endowment 3(1), 1370–1381 (September 2010)
11. Gamassi, M., Lazzaroni, M., Misino, M., Piuri, V., Sana, D., Scotti, F.: Accuracy and performance of biometric systems. In: Proc. of IMTC 2004. Como, Italy (May 2004)
12. Gamassi, M., Piuri, V., Sana, D., Scotti, F.: Robust fingerprint detection for access control. In: Proc. of RoboCare Workshop 2005. Rome, Italy (May 2005)
13. Gange, G., Stuckey, P.J., Lagoon, V.: Fast set bounds propagation using a bdd-sat hybrid. J. Artif. Int. Res. 38, 307–338 (May 2010)
14. Hacigümüs, H., Iyer, B., Mehrotra, S.: Providing database as a service. In: Proc. of ICDE 2002. San Jose, CA, USA (February 2002)
15. Hadzic, T., Hansen, E.R., O'Sullivan, B.: On automata, MDDs and BDDs in constraint satisfaction. In: Proc. of ECAI 2008. Patras, Greece (July 2008)
16. Knuth, D.E.: The Art of Computer Programming, Volume 4, Fascicle 1: Bitwise Tricks & Techniques; Binary Decision Diagrams. Addison-Wesley Professional (2009)
17. Kurihara, M., Kondo, H.: Efficient BDD encodings for partial order constraints with application to expert systems in software verification. In: Orchard, B., Yang, C., Ali, M. (eds.) Innovations in Applied Artificial Intelligence. Springer (2004)
18. Meinel, C., Theobald, T.: Algorithms and Data Structures in VLSI Design. Springer-Verlag (1998)
19. Samarati, P.: Protecting respondents' identities in microdata release. IEEE TKDE 13(6), 1010–1027 (November/December 2001)
20. Samarati, P., De Capitani di Vimercati, S.: Data protection in outsourcing scenarios: Issues and directions. In: Proc. of ASIACCS 2010. Beijing, China (April 2010)
21. Wiese, L.: Horizontal fragmentation for data outsourcing with formula-based confidentiality constraints. In: Proc. of IWSEC 2010. Kobe, Japan (November 2010)