A Bayesian Approach for on-Line Max and Min Auditing

Gerardo Canfora
RCOST - Department of Engineering
University of Sannio
gerardo.canfora@unisannio.it

Bice Cavallo
RCOST - Department of Engineering
University of Sannio
bice.cavallo@unisannio.it

ABSTRACT
In this paper we consider the on-line max and min query auditing problem: given a private association between fields in a data set, a sequence of max and min queries that have already been posed about the data, their corresponding answers and a new query, deny the answer if a private information is inferred or give the true answer otherwise. We give a probabilistic definition of privacy and demonstrate that max and min queries, without "no duplicates" assumption, can be audited by means of a Bayesian network. Moreover, we show how our auditing approach is able to manage user prior-knowledge.

Keywords
Bayesian network, Statistical database, Auditing

1. INTRODUCTION
The protection of privacy in statistical database (SDB) has been a problem of growing concern in recent years [5]. The goal is to provide statistical information about groups of individuals while protecting their privacy; only statistical information must be available and no sequence of queries should provide protected information about the individuals. If an user is able to infer protected information, then the SDB is compromised.
A number of disclosure control methods to protect a SDB have been proposed in the literature (see [1] for a survey); one of these is auditing [3], [4], [8], [9], [10], [14]. Two kinds of auditing have been studied:

1. *On-line auditing.* The queries are answered one by one in sequence and the auditor has to determine whether the SDB is compromised when answering a new query;
2. *Off-line auditing.* Given a set of queries $\{q_1, ..., q_t\}$, the auditor has to identify a maximum subset of queries that can be answered simultaneously without compromising the SDB.

In this paper we consider the first case, in particular the on-line max and min query auditing over real-valued data. Consider for example, a company database containing salaries of employees. A user may want to determine the max or a min salary of the employees in a subset of records in the database. She cannot, however, be allowed to glean the salary of any one employee in particular.
In an on-line max and min auditing problem, given a set of max and min queries $\{q_1, q_2, ...q_{t-1}\}$, the corresponding answers $\{m_1, m_2, ...m_{t-1}\}$ and the current query q_t, the auditor denies the answer to q_t if and only if a privacy breach occurs.
In [11] the authors deal with the max and min auditing, but the combinations of max and min queries in presence of duplicated sensitive values remains an open problem.
The original contribution of this paper is threefold:

1. to extend the classical notion of privacy to a probabilistic notion;
2. to show as to deal with the on-line max and min auditing, without "no duplicates" assumption, by means of a Bayesian network;
3. to manage user prior-knowledge in addition to the past history of user queries.

The rest of the paper is organized as follows: section 2 places our work in the context of previous research; section 3 presents a probabilistic approach for on-line max auditing; section 4 introduces the Bayesian network as a graphical model to deal with uncertainty and shows how a Bayesian network is able to deal with the on-line max and min auditing problem; section 5 outlines how our model deals with prior-knowledge; section 6 discusses conclusions and future research.

2. BACKGROUND
In this section we provide the notation that we will use in the sequel and place our work in the context of previous research.

2.1 Notation
We assume that:

- T is a table with n records;

- $K = \{1, 2, ..., n\}$;
- X and Y are two fields of T such that the elements of X represented by x_i are distinct among them (each x_i identifies uniquely a subject) and the elements of Y, represented by y_i, are real numbers;
- the sensitive field Y has r distinct values ($r \leq n$);
- the private information takes the form of an association, $(x_i, y_i) \subseteq X \times Y$, that is the pair of values in the same tuple;
- $S = \{s_1, ..., s_r\}$ is the set of distinct values of Y with $s_i < s_{i+1}$, $\forall i \in \{1, ..., r-1\}$;
- $S_0 = \{s_0, s_1, ..., s_r, s_{r+1}\}$, with $s_i \in S, \forall i \in \{1, ..., r\}$, s_0 a real number such that $s_0 < s_1$ and s_{r+1} a real number such that $s_{r+1} > s_r$;
- a *l-query* q is a subset of K, that is $q = \{i_1, ...i_l\} \subseteq K$. Let us assume that $i_j < i_{j+1} \quad \forall j \in \{1, ..., l-1\}$; it is not restrictive, for example $q_1 = \{1, 2\}$ and $q_2 = \{2, 1\}$ represent the same query. In this paper, we use the terms *query* and *l-query* interchangeably;
- the answer corresponding to a max query q is $m = max\{y_{i_j} | i_j \in q\}$;
- the answer corresponding to a min query q is $m = min\{y_{i_j} | i_j \in q\}$;
- $l = |q| > 1$, because if $q = \{j\}$, clearly, y_j is breached irrespective of the value of m and the association (x_j, m) is disclosed.

2.2 Related work

Given a table T, the private associations (x_j, y_j), a set of max and min queries $\{q_1, ..., q_t\}$, and the set of the corresponding answers $\{m_1, ..., m_t\}$, in [9] and in [11] the authors define, for each element y_j, the upper bound μ_j as follows:

Definition 1. *$\forall y_j$, $\mu_j = min\{m_k | j \in q_k$ with q_k a max query$\}$ is the minimum over the answers to the max queries containing j.*

In other words, μ_j is the best possible upper bound for y_j that can be obtained from the answers to the max queries.

Similarly, the lower bound λ_j is defined as follows:

Definition 2. *$\forall y_j$, $\lambda_j = max\{m_k | j \in q_k$ with q_k a min query$\}$ is the maximum over the answers to the min queries containing j.*

Moreover, if q_k is a max query, then an *extreme element* for q_k is define as follows:

Definition 3. *j is an extreme element for the query q_k if $j \in q_k$ and $\mu_j = m_k$.*

Instead, if q_k is a min query, then an *extreme element* for q_k is define as follows:

Definition 4. *j is an extreme element for the query q_k if $j \in q_k$ and $\lambda_j = m_k$.*

In [9], the authors demonstrate, for the on-line max auditing, the following theorem:

Theorem 1. *A value y_j is uniquely determined if and only if there exists a query q_k for which j is the only extreme element.*

This means that the private association (x_j, y_j) is disclosed; this theorem is valid also for the on-line min auditing.

Example 1. *Let $q_1 = \{2, 3, 4\}$ a max query. If the user submits the max query $q_2 = \{1, 2, 3, 4, 5\}$ and $m_1 < m_2$, then the auditor can answer, because q_1 has 3 extreme elements, that are 2, 3, 4 and q_2 has 2 extreme elements, that are 1 and 5.*

Example 2. *Let $q_1 = \{1, 2, 3, 4\}$ a max query. If the user submits the max query $q_2 = \{1, 2, 3, 4, 5\}$ and $m_1 < m_2$, then the auditor must deny the answer to q_2, because q_1 has 4 extreme elements, that are 1, 2, 3, 4 and q_2 has only an extreme element, that is 5.*

Example 3. *Let $q_1 = \{1, 2, 3\}$ a min query. If the user submits the min query $q_2 = \{1, 4\}$ and $m_1 > m_2$, then the auditor must deny the answer to q_2, because q_1 has 3 extreme elements, that are 1, 2, 3 and q_2 has only an extreme element, that is 4.*

Focusing on max auditing, reference [8] proposes an enhanced approach that considers the implicit delivery of information that derives from denying the answer to a query.

In [11] the authors deal with on-line max and min auditing with "no duplicates"assumption. They provide the following theorem:

Theorem 2. *Given a set of queries, $\{q_1, ..., q_t\}$ and the corresponding answers $\{m_1, ..., m_t\}$, the database is secure if and only if every max query or min query has more than one extreme element and does not exist any max query, q_i, and min query q_j such that $m_i = m_j$.*

Example 4. *Let $q_1 = \{2, 3\}$ a max query. If the user submits the min query $q_2 = \{1, 3\}$ and $m = m_1 = m_2$, then the auditor must deny the answer to q_2, because the user discloses the association (x_3, m).*

The question of auditing combination of max and min queries in presence of duplicated remains, for the authors, an open problem, because, in their approach, the duplicates make the problem harder. To see why duplicated make the problem harder, we consider the following example:

Example 5. *Let $q_1 = \{3, 5\}$ a max query and $m_1 = 98$, $q_2 = \{4, 6\}$ a max query and $m_2 = 98$. Now, if the auditor gives the answer to min query $q_3 = \{5, 6\}$ with $m_3 = 96$, then in addition to the previous information, the user also knows that $max\{y_3, y_4\}$ must be 98 since one of y_5 or y_6 has to be 96. Thus, in addition to examining extreme elements for the query that have actually been posed, the auditor needs to examine extreme elements for such inferred queries as well, and there can be a blow up in the number of queries that need to be maintained. This does not happen in the absence of duplicates, since the first two queries could never have the same answer.*

2.3 Problem statement

We consider the following definition of probabilistic compromise:

Definition 5. *A privacy breach occurs if and only if a private association is disclosed with probability greater than a given tolerance probability tol. If a private association is disclosed with $tol = 1$, then the SDB is full compromised.*

In order to show how the probabilistic definition is more secure than the classical one, we give the following example:

Example 6. *Given $y_1 = 100$ and q_i, $i \in \{1, ..., 6\}$ a max query, with $q_1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $m_1 = 100$, $q_2 = \{2, 3, 4\}$ and $m_2 = 99$, $q_3 = \{5, 6, 7, 13\}$ and $m_3 = 96$, $q_4 = \{1, 10\}$ and $m_4 = 100$, $q_5 = \{1, 11\}$ and $m_5 = 100$, $q_6 = \{7, 8\}$ and $m_6 = 94$. If the auditor answers the sequence of queries $\{q_1, q_2, q_3, q_4, q_5, q_6\}$, then the user knows that $y_1 = 100$ with probability equal to 0.8889. Even if in classical definition of privacy, a breach does not occur, because each query has at least two extreme elements, if we select $tol = 0.87$ then the auditor must deny the answer m_6 because the private association $(x_1, 100)$ is disclosed with probability equal to 0.8889. In section 4.2, we will show as our model is able to compute this probability.*

With our model, we will be able to answer the following questions:

- *What is the probability that an element y_j is equal to μ_j after a sequence of queries?*
- *What is the probability that an element y_j is equal to λ_j after a sequence of queries?*
- *What are the probabilistic dependencies among the sensitive values after a sequence of queries?*
- *How we can deal with the blow up in the number of queries that need to be maintained in an on-line max and min auditing? For example, how to represent that $max\{y_3, y_4\}$ must be 98 in the example 5, without maintaining the additional query?*
- *How we can model the user knowledge after a sequence of queries?*
- *How we can add user prior-knowledge? For example, what happens, in example 6, if the user knows that, with probability equal to 0.8, $y_1 \geq 100$?*

3. A PROBABILISTIC APPROACH

In this section we deal with max queries; the min case is analogous.
Let $q = \{i_1, ..., i_l\}$ a l-query and $m = max\{y_{i_1}, ..., y_{i_l}\}$ the corresponding answer, if the auditor gives the answer m then the user knows that:

$$y_{i_j} \leq m \quad \forall i_j \in q \tag{1}$$

$$Pr(OR(y_{i_1}, ..., y_{i_l}) = m) = 1 \tag{2}$$

The relation in (2) ensures that $\exists j \in \{1, ..., l\}$ such that $y_{i_j} = m$.

Example 7. *Let $q = \{1, 3\}$ and $m = max\{y_1, y_3\} = 8$, then the user knows that it is verified one of the following cases: $y_1 = 8$ and $y_3 = 8$; $y_1 < 8$ and $y_3 = 8$; $y_1 = 8$ and $y_3 < 8$. Since it is likely that the user has not prior knowledge on the domain of the sensitive field Y, we assume that each of three cases is equally probable.*
Then

$$Pr(y_i = 8|m = 8) = \frac{2}{3}, \quad \forall i \in \{1, 3\}$$

$$Pr(y_i < 8|m = 8) = \frac{1}{3}, \quad \forall i \in \{1, 3\}$$

Moreover

$$Pr(y_1 = 8|m = 8, y_3 = 8) = \frac{1}{2}$$
$$Pr(y_1 < 8|m = 8, y_3 = 8) = \frac{1}{2}$$
$$Pr(y_3 = 8|m = 8, y_1 = 8) = \frac{1}{2}$$
$$Pr(y_3 < 8|m = 8, y_1 = 8) = \frac{1}{2}$$

and

$$Pr(y_1 = 8|m = 8, y_3 < 8) = 1$$
$$Pr(y_3 = 8|m = 8, y_1 < 8) = 1$$

Given a l-query q and the corresponding answer m, the following propositions compute the probability that y_j is equal to m, for each $j \in q$, the probabilistic dependencies among the sensitive values in q and consider user prior-knowledge.

Proposition 1. *Let $q = \{i_1, ..., i_l\} \subseteq K$, then, $\forall j \in q$:*

$$Pr(y_j = m|max\{y_{i_1}, ..., y_{i_l}\} = m) = \frac{2^{l-1}}{2^l - 1} \tag{3}$$

$$Pr(y_j < m|max\{y_{i_1}, ..., y_{i_l}\} = m) = \frac{2^{l-1} - 1}{2^l - 1} \tag{4}$$

and

$$\lim_{l \to \infty} Pr(y_j = m|max\{y_{i_1}, ..., y_{i_l}\} = m) = \frac{1}{2} \tag{5}$$

$$\lim_{l \to \infty} Pr(y_j < m|max\{y_{i_1}, ..., y_{i_l}\} = m) = \frac{1}{2} \tag{6}$$

Proposition 2. *Given $q = \{i_1, ..., i_l\}$ such that $m_q = max\{y_{i_1}, ..., y_{i_l}\} = m$, given $q' \subset q$, with $l' = |q'| > 0$ such* *that $m_{q'} = max\{y_s | s \in q'\} = m$, then, $\forall j \in q \setminus q'$:*

$$Pr(y_j = m | m_q = m, m_{q'} = m) = \frac{1}{2}$$
$$Pr(y_j < m | m_q = m, m_{q'} = m) = \frac{1}{2}$$

EXAMPLE 8. *Let $q = \{1,2,3,4,5\}$, let $m_q = max\{y_1, ..., y_5\}$ $=m$, then if the user knows that $m_{q'} = max\{y_4, y_5\} = m$ then $Pr(y_j = m | m_q = m,\ m_{q'} = m) = \frac{1}{2}$, for $j = 1,2,3$.*

PROPOSITION 3. *Given $q = \{i_1, ..., i_l\}$ such that $m_q = max\{y_{i_1}, ..., y_{i_l}\} = m$, given $q'' \subset q$, with $l'' = |q''| > 0$ such that $m_{q''} = max\{y_s | s \in q''\} < m$, then, $\forall j \in q \setminus q''$:*

$$Pr(y_j = m | m_q = m, m_{q''} < m) = \frac{2^{l-(l''+1)}}{2^{l-l''} - 1}$$
$$Pr(y_j < m | m_q = m, m_{q''} < m) = \frac{2^{l-(l''+1)} - 1}{2^{l-l''} - 1}$$

EXAMPLE 9. *Let $q = \{1,2,3,4,5\}$, if the user knows that $m_q = max\{y_1, ..., y_5\} = m$ and $m_{q''} = max\{y_2, y_3, y_4\} < m$, then $Pr(y_i = m | m_q = m,\ m_{q''} < m) = \frac{2}{3}$, for $i = 1, 5$. Instead, if $m_{q''} = max\{y_1, y_2, y_3, y_4\} < m$ then $Pr(y_5 = m | m_q = m,\ m_{q''} < m) = 1$*

REMARK 1. *In propositions 2 and 3 $max\{y_s | s \in q'\} = m$ and respectively $max\{y_s | s \in q''\} < m$ represent user prior-knowledge. In particular if we consider $q = q_1$ and $q' = q_2$ (resp. $q = q_1$ and $q'' = q_2$) the equations in proposition 2 (resp. in proposition 3) represent the probabilities after the sequence of queries $\{q_1, q_2\}$.*

4. A BAYESIAN APPROACH

In this section we present the Bayesian network as a graphical model to deal with uncertainty and show how a Bayesian network is able to deal with the on-line max auditing problem and the on-line max and min auditing problem, also in presence of duplicated sensitive values.

4.1 Bayesian network

A Bayesian network (BN) is a probabilistic graphical model that represents a set of variables and their probabilistic dependencies [13]. A BN, also called belief net, is a directed acyclic graph (DAG) which consists of nodes to represent variables and arcs to represent dependencies between variables. Arcs or links also represent causal influences among the variables. The strength of an influence between variables is represented by the conditional probabilities which are summarized in a conditional probability table (CPT). If there is an arc from node A to another node B, A is called a parent of B, and B is a child of A. The set of parent nodes of a node X_i is denoted $parents(X_i)$. The size of the CPT of a node X_i depends on the number s of its states, the number n of $parents(X_i)$, and the number s_j of parent states, in the following way:

$$size(CPT) = s \cdot \prod_{j=1}^{n} s_j \qquad (7)$$

For every possible combination of parent states, there is an entry listed in the CPT. Notice that for a large number of parents the CPT will expand drastically. A directed acyclic graph is a BN relative to a set of variables if the joint distribution of the node values can be written as the product of the local distributions of each node and its parents:

$$Pr(X_1, ...X_n) = \prod_{i=1}^{n} Pr(X_i | parents(X_i))$$

If node X_i has no parents, its local probability distribution is said to be *unconditional*, otherwise it is *conditional*. If the value of a node is observed, then the node is said to be an *evidence* node.

4.2 Max auditing

In this section we build a BN able to deal with on-line max auditing problem. We represent the variable y_i, for $i \in K$, and a max query, by means of a node in the BN. The model is an optimized version of the BN shown in [2].
Let $K = \{1, 2, ..., n\}$ then $|P(K)| = 2^n$. Because the number of l-query from the set K is the binomial coefficient $C_n^l = \left(\begin{array}{c} n \\ l \end{array}\right) = \frac{n!}{l!(n-l)!}$ and because we consider only l-query with $l > 1$ then the total number of queries on K is:

$$\left(\begin{array}{c} n \\ 2 \end{array}\right) + ... + \left(\begin{array}{c} n \\ n-1 \end{array}\right) + \left(\begin{array}{c} n \\ n \end{array}\right) = 2^n - (1+n) \qquad (8)$$

Therefore, because the number of nodes grows in exponential way when n increases (if we encode all max queries and variables y_i, then it needs $2^n - 1$ nodes), we incrementally build the BN at run-time.
Given a l-query $q = \{i_1, ...i_l\}$, if we build the node encoding q with l parents $\{y_{i_1}, ...y_{i_l}\}$, then, by equation (7), CPT size grows exponentially, thus we decompose the l-ary max operator into a set of binary max operators and we build the CPTs with a minimum number of states.

We have implemented our BN model using the Hugin engine and its Java API [7].
Given a new query q_t, in order to represent the user knowledge, we consider the following phases:

1. if the node encoding the max query q_t does not exist then we realize the temporal transformation, described in section 4.2.1, optimizing CPT size, as discussed in section 4.2.2;
2. we insert evidence on node encoding the max query;
3. we check whether a probabilistic compromise occurs or not.

In this section we assume that:

- $Z = \{z_1, ..., z_n\}$ is a permutation of $Y = \{y_1, ..., y_n\}$ such that $z_j \geq z_{j+1}$, $\forall j \in \{1, ..., n-1\}$;
- given $S_0 = \{s_0, s_1, ..., s_r, s_{r+1}\}$ defined in section 2.1, and a l-query $q = \{i_1, ..., i_l\}$, with $i_j < i_{j+1} \quad \forall j \in \{1, ..., l-1\}$, then $\exists k \in \{1, ..., r\}$ such that m= max= $\{z_{i_1}, ..., z_{i_l}\}$=$z_{i_1} = s_k$;

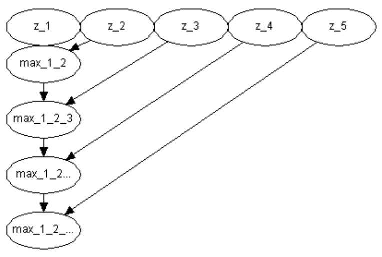

Figure 1: Temporal transformation for a max query.

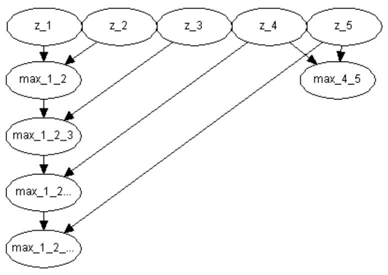

Figure 2: Sequence of temporal transformations.

- because each node encoding a l-query is a binary max operator, p_1 and p_2 are its parents.

4.2.1 Decomposition of a max query

The size of a conditional distribution that encodes the max operator can be reduced when the l-ary max operator is decomposed into a set of binary max operators. Two well known approaches to the decomposition are: parent divorcing [12] and temporal transformation [6].
Parent divorcing constructs a binary tree in which each node encodes the binary operator. Temporal transformation constructs a linear decomposition tree in which each node encodes the binary operator. In order to optimize our BN we use temporal transformation. For example, the BN resulting from temporal decomposition for the query $q_1 = \{1, 2, 3, 4, 5\}$ is shown in figure 1. Each max node in the transformation has size equal to r^3, where r is the domain size, in our context, r is the number of distinct values of sensitive date. In the general case of a l-query, *l-1* max nodes are needed. Inserting evidence on node encoding the l-query, we can compute the probability of the corresponding l nodes in the first level.
If the auditor answers the query $\{4, 5\}$, then the sequence of temporal transformations is shown in figure 2.
Therefore, in the first level of BN there are the nodes encoding the variables z_j, and in the level l, with $l > 1$, there are the nodes encoding the l-queries. We build the BN at run-time: given the current query q_t, we execute a temporal transformation if and only if the privacy is not breached and the answer to q_t gives new information. If no query is answered then the BN has not nodes.

4.2.2 Optimizing CPT size

In this section we optimize the model by building CPT with minimum number of states.
Given $\{q_1, ..., q_{t-1}\}$ a sequence of queries already posed, $\{m_1, ..., m_{t-1}\}$ the corresponding answer and a new query q_t with answer m_t, for each j in q_t, one of the two cases is verified:

1. $\exists\ i \in \{1, ..., t-1\} \mid j \in q_i$ and $\mu_j = m_i$ (j is extreme element for q_i),

2. $\nexists$ a max query q_i with $i \in \{1, ..., t-1\}, \mid j \in q_i$.

In the case 1, if μ_j, related to $\{q_1, ..., q_{t-1}\}$, is such that $\mu_j > m_t$, and in the case 2, we set $\mu_j = m_t$.
Therefore, for each z_j in the BN, we have to represent the following probabilities:

$$Pr(z_j < \mu_j) \tag{9}$$
$$Pr(z_j = \mu_j). \tag{10}$$

We build the nodes in the BN in the following way:

NODE ENCODING z_j. The node encoding z_j has two states zs_1 and zs_2 with $zs_1 < zs_2 = \mu_j$. Therefore, unconditional distribution table has size equal to 2. Since we assume that the user has not knowledge about the domain of the sensitive field, the prior distribution is $(\frac{1}{2}, \frac{1}{2})$. In our implementation we can set $zs_1 = s_0$. Obviously, zs_2 is updated after a query q_t if μ_j, related to $\{q_1, ..., q_{t-1}\}$, is such that $\mu_j > m_t$.

NODE ENCODING A MAX QUERY. Given $q_t = \{i_1, ..., i_l\}$ the current query, m_t=*max* $\{z_{i_1}, ..., z_{i_l}\} = z_{i_1} = s_k$, *max Node* the node encoding the query, then *maxNode* has three states ms_1, ms_2, and ms_3, with $ms_1 < ms_2 = s_k < ms_3$.
Because $z_j \geq z_{j+1}\, \forall j \in \{1, ..., n-1\}$, also each node, in the corresponding temporal transformation, has the same states as $maxNode$.
Because equation (7), we have that:

- the conditional distribution table of a node encoding a 2-query has size equal to $3(2 \cdot 2) = 12$;

- the conditional distribution table of a node encoding a l-query, with $l > 2$, has size equal to $3(3 \cdot 2) = 18$.

In our implementation, we set $ms_1 = s_0$, $ms_3 = s_{r+1}$, and define the max expression as follows:
$if(and(p_1 < s_k, p_2 < s_k))\ Distribution(1, 0, 0);$
$else\,if(or(p_1 \leq s_k, p_2 \leq s_k))\ Distribution(0, 1, 0);$
$else\ Distribution(0, 0, 1);$
Inserting evidence on $maxNode$ in the following way:

$$Pr(maxNode = s_k) = 1 \tag{11}$$

the BN is able to compute the probability in (9) and (10), for each z_j in the BN.

EXAMPLE 10. *Given the table 1, we set $s_0 = 0$, if the auditor answers $q_1 = \{1, 2, 3, 4, 5\}$ and $q_2 = \{4, 5\}$ then the BN is shown in figure 3, and for $j \in \{1, 2, 3\}$, according to proposition 3,*

$$Pr(z_j = s_0) = Pr(z_j < 9) = \frac{3}{7}$$
$$Pr(z_j = 9) = \frac{4}{7}$$

Table 1: n=5, r=4.

X	Z
x_1	9
x_2	8
x_3	8
x_4	5
x_5	4

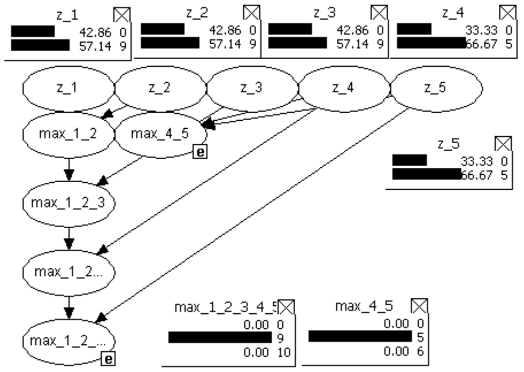

Figure 3: $q_1 = \{1, 2, 3, 4, 5\}$ and $q_2 = \{4, 5\}$.

and, according to proposition 1, for $j \in \{4, 5\}$,

$$Pr(z_j = s_0) = Pr(z_j < 5) = \frac{1}{3}$$
$$Pr(z_j = 5) = \frac{2}{3}$$

REMARK 2. *$\forall z_j$ in the BN, let $A = Pr(z_j < \mu_j)$ and $B = Pr(z_j = \mu_j)$, then we can compute the entropy of z_j as:*

$$H = A \cdot log_2 \frac{1}{A} + B \cdot log_2 \frac{1}{B}. \quad (12)$$

REMARK 3. *With our model, we are able to compute the probability of the private association $(x_1, 100)$ in the example 6 of the section 2.3. See Figure 4.*

4.3 Max and Min auditing

Starting from BN model shown in section 4.2, in this section, we build a BN able to deal with the on-line max and min auditing also in presence of duplicated sensitive values; our model resolves the problem of the blow up of the queries that need to be maintained shown in section 2.2.
In this section we assume that:

- $Z = \{z_1, ..., z_n\}$ is a permutation of $Y = \{y_1, ..., y_n\}$ such that $z_j \geq z_{j+1}$, $\forall j \in \{1, ..., n-1\}$;
- given $S_0 = \{s_0, s_1, ..., s_r, s_{r+1}\}$ defined in section 2.1, and a max l-query $q = \{i_1, ..., i_l\}$, with $i_j < i_{j+1} \quad \forall j \in \{1, ..., l-1\}$, then $\exists k \in \{1, ..., r\}$ such that m= *max* $\{z_{i_1}, ..., z_{i_l}\}$=$z_{i_1} = s_k$;

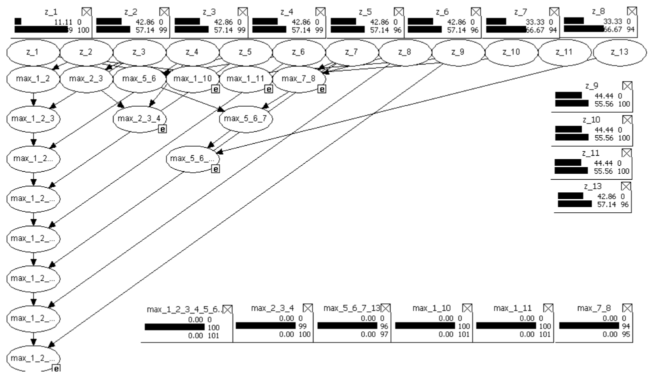

Figure 4: Sequence of max queries

Table 2: n=7, r=5.

X	Z
x_1	100
x_2	99
x_3	98
x_4	98
x_5	96
x_6	96
x_7	94

- given a min l-query $q = \{i_1, ..., i_l\}$, with $i_j < i_{j+1} \quad \forall j \in \{1, ..., l-1\}$, then $\exists k \in \{1, ..., r\}$ such that m= *min* $\{z_{i_1}, ..., z_{i_l}\}$=$z_{i_l} = s_k$;
- because each node encoding a l-query is a binary max or a min operator, p_1 and p_2 are its parents.

Given $\{q_1, ..., q_{t-1}\}$ a sequence of max and min queries already posed, $\{m_1, ..., m_{t-1}\}$ the corresponding answer and a new max or min query q_t with answer m_t, for each j in q_t, one of the four cases is verified:

1. $\exists\ i \in \{1, ..., t-1\}$ and q_i a max query $\mid j \in q_i$ and $\mu_j = m_i$,
   $\nexists$ a min query q_k with $k \in \{1, ..., t-1\}$, $\mid j \in q_k$;
2. $\nexists$ a max query q_i with $i \in \{1, ..., t-1\}$, $\mid j \in q_i$,
   $\exists\ k \in \{1, ..., t-1\}$ and q_k a min query $\mid j \in q_k$ and $\lambda_j = m_k$;
3. $\exists\ i \in \{1, ..., t-1\}$ and q_i a max query $\mid j \in q_i$ and $\mu_j = m_i$,
   $\exists\ k \in \{1, ..., t-1\}$ and q_k a min query $\mid j \in q_k$ and $\lambda_j = m_k$;
4. $\nexists$ a max query $q_i \mid j \in q_i$,
   $\nexists$ a min query $q_k \mid j \in q_k$.

We build the nodes in the BN in the following way:

NODE ENCODING z_j. We suppose that q_t is a max query; if it is a min query the reasoning is analogous. We build the node dynamically.
In the case 1 and 4, we build the node encoding z_j with only two states zs_1 and zs_2 in the same way as the on-line max

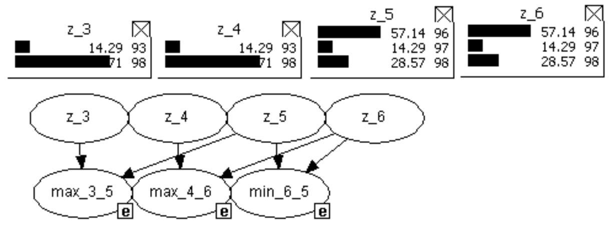

Figure 5: Not blow up of queries.

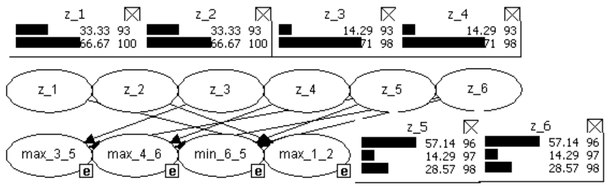

Figure 6: Add max query $\{1, 2\}$.

auditing problem.
In the case 2 and 3, we build the node encoding z_j with three states zs_1, zs_2 and zs_3, with $\lambda_j = zs_1 < zs_2 < zs_3 = \mu_j$, where λ_j and μ_j are the lower bound and the upper bound after the sequence of query $\{q_1, ..., q_t\}$. Moreover, $P(zs_2) = Pr(\lambda_j < z_j < \mu_j)$. Since we assume that the user has not knowledge about the domain of the sensitive field, the prior distribution of this node is $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. In our implementation we can set $zs_2 = \frac{zs_1+zs_3}{2}$.

NODE ENCODING A MAX QUERY. It is equal to a max node in on-line max auditing, but its CPT size is equal to $3 \cdot (size_{p_1} \cdot size_{p_2})$, where $size_{p_1}$ and $size_{p_2}$ are the number of states of the parents and they can be 2 or 3.

NODE ENCODING A MIN QUERY. We build the node in analogous way to a node encoding a max query; we define the min expression as follows:
$if(and(p_1 > s_k, p_2 > s_k))$ $Distribution(0, 0, 1)$;
$else\,if(or(p_1 \geq s_k, p_2 \geq s_k))$ $Distribution(0, 1, 0)$;
$else$ $Distribution(1, 0, 0)$; In the following example, we consider the table 2, the queries in example 5 of section 2.2 and we see as our model deals with "duplicated values" in an efficient way.

EXAMPLE 11. *Set $s_0 = s_1 - 1 = 93$ and $s_{r+1} = s_r + 1 = 101$, given the max queries $q_1 = \{3, 5\}$ and $q_2 = \{4, 6\}$, and the min query $q_3 = \{5, 6\}$, then the corresponding BN is shown in figure 5. For $j \in \{3, 4\}$, because $zs_1 = s_0 = 93$, it means that the user knows that $Pr(z_j < 98) = 0.1429$ and $Pr(z_j = 98) = 0.8571$. For $j \in \{5, 6\}$, the nodes have three states because the variables z_5 and z_6 are present in min and max queries. The user knows that $Pr(z_j = 96) = 0.5714$, $Pr(96 < z_j < 98) = 0.1429$ and $Pr(z_j = 98) = 0.2857$. Our model is able to infer that $max\{z_3, z_4\}$ is equal to 98; in fact we can see that $z_3 \leq 98$, $z_4 \leq 98$ and if we insert evidence on first state of z_3, that is $z_3 < 98$ (resp. $z_4 < 98$), then the probability of the second state of z_4, that is $z_4 = 98$ (resp. $z_3 = 98$), is equal to 1.*
Moreover, if the user submits the max query $q_4 = \{3, 4\}$ then the corresponding max node has entropy equal to 0; thus the node is not added to BN.
Moreover, if the tolerance tol is such that $tol < 0.8571$, then the auditor must deny the answer to q_3.

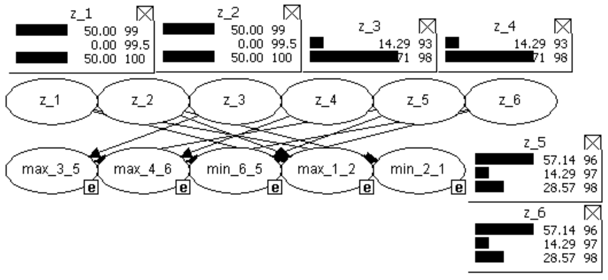

Figure 7: Add min query $\{1, 2\}$.

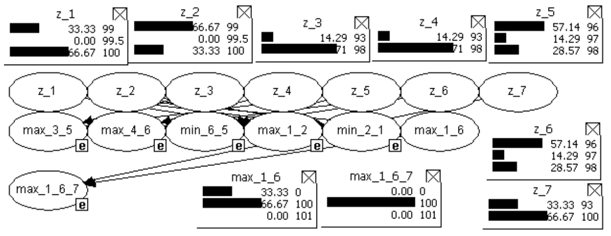

Figure 8: Add max query $\{1, 6, 7\}$.

REMARK 4. *By example 11, we can see that our model is able to manage efficiently inferred queries, in addition to posed queries, without maintaining additional nodes. Moreover, if the information is already present in the model, then the corresponding max or min node has entropy equal to zero and the probabilities of the nodes z_j not change; therefore we do not add the query node.*

REMARK 5. *Similarly to max auditing, we can compute the entropy for each node z_j in the BN.*

In order to better understand the combined on-line max and min auditing, we continue the example 11.

EXAMPLE 12. *If the auditor gives the answer to max query $\{1, 2\}$ then the BN is update as in figure 6; $Pr(z_i < 100) = \frac{1}{3}$, $Pr(z_i = 100) = \frac{2}{3}$, $\forall i \in \{1, 2\}$.*
If the auditor gives the answer to min query $\{1, 2\}$ then the BN is update as in figure 7; $Pr(z_i = 99) = \frac{1}{2}$, $Pr(99 < z_i < 100) = 0$ and $Pr(z_i = 100) = \frac{1}{2}$, $\forall i \in \{1, 2\}$.
If the user submits the max query $\{2, 3\}$, which answer is equal to 99, then the auditor must deny the answer, because it is disclosed that $z_1 = 100$ and $z_2 = 99$ (the node encoding z_3 does not change because $\mu_3 = 98 < 99$, the node encoding z_2 has only a state equal to 99 and the node encoding z_1 is such that $Pr(z_1 = 100) = 1$); therefore the BN remains as figure 7. If the user submits max query $\{1, 6, 7\}$, which answer is equal to 100, then we add the nodes encoding the variable z_7, max queries $\{1, 6\}$ and $\{1, 6, 7\}$, and insert evidence only in the node $\{1, 6, 7\}$(see figure 8). The node z_6 does not

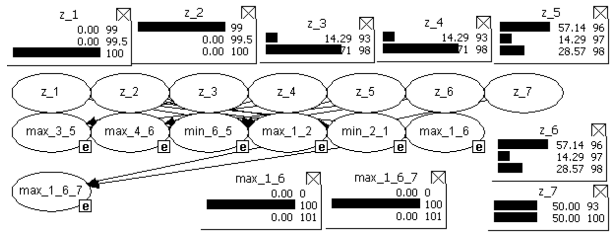

Figure 9: Inserting evidence on node encoding $max\{z_1, z_6\}$ the privacy is breached.

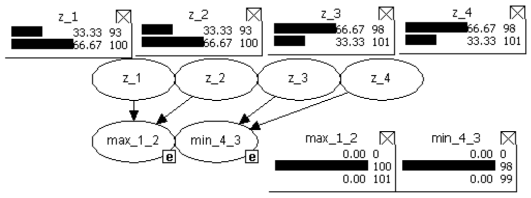

Figure 10: BN after a sequence of queries.

change, because $\mu_6 = 98 < 100$; node encoding z_1 is such that $Pr(z_i = 99) = 0.3333$, $Pr(z_i = 100) = 0.6666$; node encoding z_7 is such that $Pr(z_7 < 100) = 0.3333$, $Pr(z_7 = 100) = 0.6666$. We can see that, also in the node encoding z_2 the probabilities change. If the user submits max query $\{1, 6\}$, the corresponding node already exists, then we insert only evidence on the node (see figure 9), but because the privacy is breached, we can not update the BN, it remains as in figure 8 and the auditor must deny the answer.

5. DEALING WITH PRIOR-KNOWLEDGE

An user can learn something about the state of the BN not only with a sequence of queries, but also by prior and background knowledge. In this section we show as our model is able to capture user prior-knowledge. We consider two kind of user prior-knowledge:

1. prior-knowledge represented by likelihood on one or more nodes in the model;
2. prior-knowledge that derives from integration with an other knowledge domain.

5.1 Likelihood

Adding likelihood is what we do when the user learns something about the state of the BN which can be entered into a node. The simplest form is the evidence, that is, the probability that a state is 1 while the probability of each other states is 0. In general adding likelihood we can choose a value in $[0, 1]$ and take this value as probability of a state. Obviously, the sum of all probabilities is necessarily 1.

EXAMPLE 13. *We suppose that the BN is in the state of figure 10, therefore the user knows that $max\{z_1, z_2\} = 100$. Moreover we suppose that z_1 represents Alice's salary, z_2 Bob's salary and that the user is a Bob's friend. If the user knows that it is very unlikely that Bob's salary is equal to 100, exactly that $Pr(z_2 = 100) = 0.1$ and $Pr(z_2 < 100) =$*

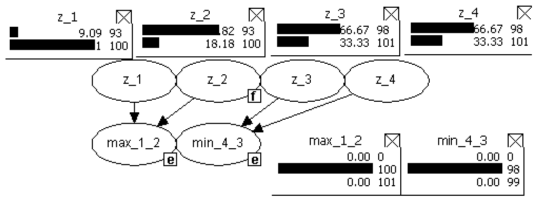

Figure 11: BN after a sequence of queries with prior-knowledge $Pr(z_2 < 100) = 0.9$.

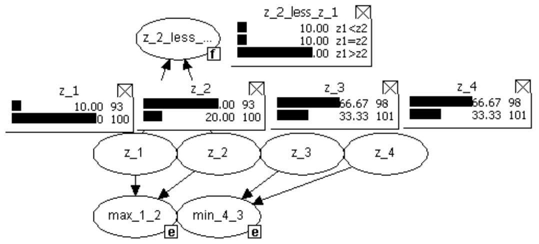

Figure 12: BN after a sequence of queries with prior-knowledge $Pr(z_1 > z_2) = 0.8$.

0.9, then we insert this likelihood on node z_2. Thus, the probabilities in BN change as in figure 11; in particular $Pr(z_1 = 100) = 0.9091$ and if $tol = 0.9$ then the privacy is breached.

5.2 Integration with other domains

The BNs are able to manage uncertainty between the different knowledge components, including the interactions among the various domains of uncertainty. We consider the following example:

EXAMPLE 14. *If the BN is in the state of figure 10 and the user knows that it is very likely that Alice's Salary is greater than Bob's Salary, more exactly that $Pr(z_1 > z_2) = 0.8$, then the probabilities in BN change as in figure 12. If $tol < 0.9$ then the privacy is breached.*

6. CONCLUSIONS AND FUTURE WORK

We have introduced a novel definition of privacy that extends the classical notion and we have demonstrate that max and min queries can be audited, without "no duplicates" assumption, by means of a BN. We have shown as background knowledge may compromise a SDB and how our model is able to manage user prior-knowledge in addition to past history of user queries. The goal of our future work is threefold:

- to consider the case in which the probability distribution of the sensitive field is public, and in particular is known to the user, therefore we will integrate this domain in our model;
- to use a BN as a unifying framework including the interactions among the various domains of uncertainty;
- to include combinations of different statistical queries.

7. REFERENCES

[1] N. R. Adam and J. C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Computing Surveys (CSUR)*, 21(4), December 1989.

[2] G. Canfora and B. Cavallo. A bayesian approach for on-line max auditing. ARES, March 2008, to appear.

[3] F. Y. Chin. Security problems on inference control for sum, max, and min queries. *Journal of the ACM*, 33(3):451–464, July 1986.

[4] F. Y. Chin and G. Ozsoyoglu. Auditing and inference control in statistical databases. *IEEE Transaction on Software Engineering*, SE-8(6):574–582, November 1982.

[5] J. Domingo-Ferrer. *Inference Control in Statistical Databases: From Theory to Practice*. Springer-Verlag, Berlin Heidelberg, 2002.

[6] D. Heckerman. Causal independence for knowledge acquisition and inference. pages 122–127, 1993.

[7] Hugin. *www.hugin.com*.

[8] K. Kenthapadi, N. Mishra, and K. Nissim. Simulatable auditing. pages 118–127. PODS, June 2005.

[9] J. Kleinberg, C. Papadimitriou, and P. Raghavan. Auditing boolean attributes. *Journal of Computer and System Sciences*, 66(1):244–253, February 2003.

[10] F. M. Malvestuto, M. Mezzini, and M. Moscarini. Auditing sum-queries to make a statistical database secure. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):31–60, February 2006.

[11] S. U. Nabar, B. Marthi, K. Kenthapadi, N. Mishra, and R. Motwani. Towards robustness in query auditing. pages 151–162. VLDB, September 2006.

[12] K. G. Olesen, U. Kjaerulff, F. Jensen, F. V. Jensen, B. Falck, S. Andreassen, and S. K. Andersen. A munin network for the median nerve - a case study in loops. *Applied Artificial Intelligence*, 3(2-3):385–403, 1989.

[13] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, San Francisco, CA, USA, 1998.

[14] S. P. Reiss. Security in databases: A combinatorial study. *Journal of the ACM*, 26(1):45–57, January 1979.