Keeping Secrets in Possibilistic Knowledge Bases with Necessity-Valued Privacy Policies

Lena Wiese

Technische Universität Dortmund, Germany
wiese@ls6.cs.tu-dortmund.de
http://ls6-www.cs.tu-dortmund.de

Abstract. Controlled Query Evaluation (CQE) is a logical framework for the protection of secrets in databases. In this article, we extend the CQE framework to possibilistic logic: knowledge base, a priori knowledge and privacy policy are expressed with necessity-valued formulas that represent several degrees of certainty. We present a formal security definition and analyze an appropriate controlled evaluation algorithm for this possibilistic case.

1 Introduction

A major security goal for databases is keeping secret entries in a database instance confidential. Two general mechanisms for the enforcement of confidentiality are *modification* of data (like perturbation, noise generation, cover stories, or "lied" answers) and *restriction* (refusal, denial of access, blocking, cell suppression, generalization, data upgrading etc).

The framework of Controlled Query Evaluation (CQE; see for example [1, 2]) uses both mechanisms (in the form of lying and refusal) in a logic-based framework. In this article, we extend the CQE framework to possibilistic logic such that degrees of certainty can be specified and attached to logical formulae. We consider a client-server architecture where a user issues queries to a knowledge base system. The knowledge base *kb* contains public as well as private data; to achieve confidentiality of the private data, the system maintains a personalized privacy policy *policy* and a user history *log* for each particular user registered at the system. Based on *policy* and *log*, queries are then evaluated in the knowledge base in a controlled way by the evaluation function *cqe* that makes use of a subroutine called *censor*. The censor module takes care of the modifications and restrictions of data that are necessary to keep entries of the privacy policy confidential.

As the syntactical basis we consider a propositional logic $\mathscr{L}$ that involves a finite alphabet $\mathcal{P}$ of propositional variables and the connectives negation $\neg$, conjunction $\wedge$, and disjunction $\vee$; on occasion, material implication $\rightarrow$ is used as an abbreviation (for a negation and a disjunction).

If to a formula p of $\mathscr{L}$ a weight $\alpha \in (0, 1]$ is attached, we get the "standard possibilistic logic" (SPL; see [3]). As usual, we write possibilistic formulas as

(p, α). The weight α denote a *lower bound* for a necessity degree N of p. A necessity degree specifies the certainty of formulas: it "evaluates to what extent p is entailed by the available knowledge" [4]. In other words, with (p, α) we express that $N(p) \geq \alpha$: the certainty of p is at least α. A high certainty for p denotes that a countermodel of p (that is a model of $\neg p$) is relatively impossible to be the "real" world; we will give a precise definition below. In this reading, it is natural that 0 is excluded as a weight: $N(p) \geq 0$ (denoting that the certainty of p is somewhere between 0 and 1) always holds for any formula p.

We now recall how a necessity measure can be induced by a possibilistic knowledge base. A possibilistic knowledge base can be defined as a set of possibilistic formulas. In this article, we let *kb* denote such a knowledge base. The formulas in *kb* have as possible interpretations all $2^{card(\mathcal{P})}$ classical interpretations ("worlds") of the propositional variables $\mathcal{P}$. On these possible worlds, *kb* induces a **possibility distribution** π_{kb}. This possibility distribution assigns to each world u a value from the interval $[0, 1]$. This value specifies how possible it is for u to be the real world – that is, the right one of all the possible interpretations for *kb*. $\pi_{kb}(u) = 0$ means that it is totally impossible for u to be the real world. Hence, worlds that violate formulas in *kb* have a lower value than worlds that propositionally satisfy all formulas in *kb*. The worlds with possibility value 1 are "models" (denoted $\models$) of the propositional formulas in *kb*:

$$\pi_{kb}(u) := \begin{cases} 1 & \text{iff } u \models p_1 \wedge \ldots \wedge p_n \\ 1 - \max\{\alpha_i \mid (p_i, \alpha_i) \in kb \text{ and } u \models \neg p_i\} & \text{otherwise} \end{cases}$$

There need not exist a world with possibility 1 (that is, not all formulas p_i can be satisfied at the same time); in this case, π_{kb} is called *subnormalized.* It is called *normal* if there is at least one world with possibility 1. However, there may also exist more than one world with possibility 1 for a given *kb*. Knowledge bases with a subnormalized possibility distribution have an **inconsistency level** $Inc(kb)$ above 0; it is defined as follows:

$$Inc(kb) := 1 - \max_u\{\pi_{kb}(u)\}$$

Based on the possibility distribution, we can compute the **possibility degree** for any formula p':

$$\Pi_{kb}(p') := \max\{\pi_{kb}(u) \mid u \models p'\}$$

That is, the possibility degree identifies the world with maximal possibility that satisfies p: it "it evaluates to what extent p is consistent with the available knowledge" [4].

Then we are interested in the **necessity degree** of a formula p'' induced by Π_{kb} by subtracting from 1 the possibility degree of $\neg p''$ (that is, the maximal possibility degree of a countermodel of p''):

$$N_{kb}(p'') := 1 - \Pi_{kb}(\neg p'')$$

Note that we could also skip computation of the possibility degree $(\Pi_{kb}(\neg p''))$ by letting $N_{kb}(p'') = \min\{1 - \pi_{kb}(u) \mid u \text{ is model of } \neg p''\}$.

Lastly, **implication** in SPL is defined. We say that kb implies p with maximal necessity degree α (written $kb \models_{SPL} (p, \alpha)$) if $N_{kb}(p) = \alpha$.[1] It has been shown in [3] that this implication can (soundly and completely) be implemented with a set of syntactic inference rules (where $\vdash$ denotes syntactic entailment). These include the following:

- **resolution:** $(\neg p \vee q, \alpha); (p \vee r, \beta) \vdash (q \vee r, \min\{\alpha, \beta\})$
- **weight fusion:** $(p, \alpha); (p, \beta) \vdash (p, \max\{\alpha, \beta\})$
- **weight weakening:** for $\beta \leq \alpha$, $(p, \alpha) \vdash (p, \beta)$

With such rules, the implication $kb \models_{SPL} (p, \alpha)$ can be decided by the refutation $kb \cup \{(\neg p, 1)\} \vdash (\bot, \alpha)$. Alternatively, entailment of (p, α) can also be expressed as the inconsistency level $\alpha = \mathit{Inc}(kb \cup \{(\neg p, 1)\})$. Moreover, algorithmically entailment takes only a bounded number propositional satisfiability checks: the bound is the logarithm of the number of certainty degrees occurring in the knowledge base. We refer to [3–5] for further details.

The remainder of this article is organized as follows: Section 2 describes the system settings. Section 3 formally defines "confidentiality-preservation" and Section 4 presents a CQE algorithm for possibilistic knowledge bases. The article concludes in Section 5 with a discussion of our approach and related work.

2 System Components

In this article, we transfer the CQE framework to a possibilistic setting where the knowledge base consists of formulas at differing degrees of certainty and also the privacy policy and the user history are maintained in possibilistic logic.

The possibilistic **knowledge base** kb is a finite set of possibilistic formulas (p_i, α_i) for $i = 1 \dots n$; hence $p_i \in \mathcal{P}$ is a propositional formula and each $\alpha_i \in (0, 1]$ is the necessity degree of p_i. For the time being, we assume here that kb is a consistent set of formulas – although possibilistic logic has the ability to cope with inconsistencies in the knowledge base. That is, we assume that the set of the *propositional* formulas occurring in kb (denoted Prop_{kb}) form a propositionally consistent set and as such do not lead to a contradiction. In other words, the *inconsistency level* of kb is 0. To illustrate the settings, we give a small example of a medical knowledge base that contains information about some medical treatment (`med`) and some diagnoses (`aids` and `cancer`):

$$kb = \{(\texttt{med}, 0.9), (\texttt{aids}, 0.8), (\texttt{cancer}, 0.7)\}$$

The knowledge base is able to answer queries based on the necessity degree induced by kb, and hence on implication $\models_{SPL}$ in SPL. That is, the input is a propositional query formula p and **evaluation** of this query outputs the possibilistic formula (p, α) such that α is $N_{kb}(p)$ (that is, the maximal degree of necessity for p in the knowledge base kb) and hence $kb \models_{SPL} (p, \alpha)$:

$$\mathit{eval}_{kb}(p) := (p, \alpha) \text{ where } \alpha = N_{kb}(p)$$

[1] We say "maximal necessity degree" because it also holds that for all $\alpha' \in (0, \alpha]$ that $kb \models_{SPL} (p, \alpha')$.

For example, the necessity degree for query $\texttt{aids}\vee\texttt{cancer}$ is $N_{kb}(\texttt{aids}\vee\texttt{cancer}) = 1 - \Pi_{kb}(\neg\texttt{aids} \wedge \neg\texttt{cancer}) = 0.8$ and hence $eval_{kb}(\texttt{aids} \vee \texttt{cancer}) = (\texttt{aids} \vee \texttt{cancer}, 0.8)$. For the query $\texttt{aids} \wedge \texttt{cancer}$ we have $N_{kb}(\texttt{aids} \wedge \texttt{cancer}) = 1 - \Pi_{kb}(\neg\texttt{aids} \vee \neg\texttt{cancer}) = 0.7$ and hence $eval_{kb}(\texttt{aids} \wedge \texttt{cancer}) = (\texttt{aids} \wedge \texttt{cancer}, 0.7)$.

The **privacy policy** *policy* is a finite set of possibilistic formulas. Semantically, a policy entry (q, β) specifies that the user is never allowed to know that q is certain in *kb* at a necessity degree above β. He may however learn that q is certain at least with degree β. For example the following policy states that `aids` may be known with a lower bound of certainty of 0.3 (that is, $N_{kb}(\texttt{aids}) \geq 0.3$) and `cancer` with a lower bound of certainty of 0.2 (that is, $N_{kb}(\texttt{cancer}) \geq 0.2$); he must however never learn greater lower bound values for $N_{kb}(\texttt{aids})$ and $N_{kb}(\texttt{cancer})$:

$$policy = \{(\texttt{aids}, 0.3), (\texttt{cancer}, 0.2)\}$$

As an exceptional value, we explicitly allow entries with necessity degree 0 in *policy*: an entry $(q, 0) \in policy$ denotes that we do not want to reveal any information on the state of p in *kb*, that is, we do not give the user any certainty about p.

The **user history** *log* records all answers (as possibilistic formulas) that were given by the knowledge base to a sequence of user queries $Q = \langle q_1, q_2, \ldots q_m\rangle$ for propositional formulas q_j; that is, we have a sequence of history logs where log_j denotes the state of the history after the j-th answer was given. In particular log_0 may contain additional **a priori knowledge** that the user has before starting the query sequence. For example the a priori knowledge may state that a treatment with some medicine implies both diagnoses but at different levels of certainty:

$$log_0 = \{(\texttt{med} \rightarrow \texttt{aids}, 0.6), (\texttt{med} \rightarrow \texttt{cancer}, 0.5)\}$$

It may occur, that some answer with necessity degree 0 is returned to the user – either because the necessity degree in *kb* is 0 indeed or because the privacy policy prohibits any more specific return value; yet, in this case while the answer $(q_j, 0)$ is given to the user to acknowledge his query, it is not added to the user history log_j because from a reasoning point of view it has no effect.

In the following we will devise a **controlled query evaluation** function $cqe(Q, kb, log_0, policy)$ that shields the $eval_{kb}$-function from direct access by the user. The *cqe*-function will – whenever necessary – modify the $eval_{kb}$-answers; that is, the *cqe*-function will answer the query sequence Q in such a way that the sequence of history files log_j will reveal an entry of the privacy policy at most at the level of certainty specified in the policy. For example, for the query sequence $Q = \langle\texttt{aids}, \texttt{cancer}\rangle$ (and *kb*, log_0 and *policy* as in the examples above) we will have the answer sequence $A = cqe(Q, kb, log_0, policy) = \langle(\texttt{aids}, 0.3), (\texttt{cancer}, 0.2)\rangle$; hence the only knowledge that the user receives is that $N_{kb}(\texttt{aids}) \geq 0.3$ and $N_{kb}(\texttt{cancer}) \geq 0.2$. Without controlling the evaluation, the normal evaluation would be $eval_{kb}(\texttt{aids}) = (\texttt{aids}, 0.8)$ (revealing $N_{kb}(\texttt{aids}) \geq 0.8$) and $eval_{kb}(\texttt{cancer}) = (\texttt{cancer}, 0.7)$ (revealing $N_{kb}(\texttt{cancer}) \geq 0.7$); and hence both truthful answers would violate *policy*.

3 A Formal Security Definition

In this section, we adapt the formal definition of confidentiality preservation of a controlled query evaluation function to the possibilistic case. Appropriate definitions were already established for complete databases [2] and incomplete databases with policies in modal logic [1,6].

Confidentiality preservation of a controlled query evaluation function *cqe* is ensured by the following Definition 1. It demands that there exists an alternative knowledge base that is compatible with the a priori knowledge log_0 and for which the *cqe*-function returns the same answers (Item *1*); that is, from the observable behaviour (via the *cqe*-function) *kb* and kb' are indistinguishable. However the alternative knowledge base does not violate the privacy policy when queries are evaluated without control (Item *2*).

Definition 1 (Confidentiality preservation). *A controlled query evaluation function cqe is* confidentiality-preserving *iff for all admissible inputs Q, kb,* log_0 *and policy there is an alternative knowledge base* kb' *such that* $kb' \cup log_0$ *is consistent, and the following two properties hold:*

1. [**same controlled answers**]
$cqe(Q, kb, log_0, policy) = cqe(Q, kb', log_0, policy)$
2. [**alternative knowledge base is secure**]
there is no policy entry $(q, \beta) \in$ *policy such that* $eval_{kb'}(q) = (q, \beta')$ *with* $\beta' > \beta$*.*

Preconditions for Q, *kb*, log_0 and *policy* can be defined to specify what inputs are “admissible” in Definition 1. More precisely, we allow inputs with the following properties:

1. As already mentioned in Section 2, *kb* is assumed to be a consistent possibilistic knowledge base; that is, $Inc(kb) = 0$.
2. The a priori knowledge log_0 is consistent: $Inc(log_0) = 0$.
3. Moreover, *kb* and log_0 must be compatible; that is, when combined they are also consistent: $Inc(kb \cup log_0) = 0$.
4. Lastly, the user does not know a policy entry a priori and hence log_0 must be compatible with *policy*: there is no policy entry $(q, \beta) \in policy$ such that $log_0 \models_{SPL} (q, \beta')$ with $\beta' > \beta$.

4 A Censor for Possibilistic Knowledge Bases

As a subroutine of a controlled query evaluation function *cqe*, the *censor* is responsible to decide whether a modification or restriction of a database answer is necessary and if so, compute the modified or restricted answer. We list a censor that is appropriate for controlled query evaluation in possibilistic knowledge bases in Figure 1. This censor proceeds as follows: given the current query q_j, it checks whether there are any violated policy entries when adding the correct evaluation $eval_{kb'}(q_j)$ to the current user history log_{j-1}. The set of the necessity degrees of all those violated entries is determined. If there are no violated entries, this set is empty and the correct evaluation can be returned without modification. If however there are violated entries, the minimal necessity degree γ is determined and as the modified answer the query with necessity degree γ is returned.

```
$censor(log_{j-1}, q_j)$:

    $S := \{\beta \mid (q,\beta) \in policy \text{ such that } log_{j-1} \cup eval_{kb}(q_j) \models_{SPL} (q,\beta') \text{ with } \beta' > \beta\}$
    if $S = \emptyset$
        return $eval_{kb}(q_j)$
    else
        $\gamma := \min S$
        return $(q_j, \gamma)$
```

Fig. 1. Possibilistic censor

Note that if the policy is ordered in ascending order of the necessity degrees, the violation check could start with policy entries at the least degree and move on to greater degrees until a violation is encountered. In this manner, the minimum of S can easily be determined without checking all policy entries in the optimal case.

A complete implementation of the *cqe*-function can be made by calling the censor for every query q_j in the query sequence, constructing and returning the answer sequence A and updating the user history log_j; see Figure 2. When updating the user history, answers with necessity degree 0 are filtered out: the fact that $N_{kb}(q_j) \geq 0$ does not carry any information and need not be recorded.

```
$cqe(Q, kb, log_0, policy)$:

    $A = \langle a_1, \ldots, a_m \rangle$
    for $j = 1 \ldots m$
        $a_j := censor(log_{j-1}, q_j)$
        if $a_j = (q_j, \gamma)$ with $\gamma > 0$
            $log_j := log_{j-1} \cup \{(q_j, \gamma)\}$
        else
            $log_j := log_{j-1}$
    return $A$
```

Fig. 2. *cqe* implementation

We continue our example with the query sequence $Q = \langle \texttt{aids}, \texttt{cancer}, \texttt{med}\rangle$. Obviously, with the correct answer $eval_{kb}(\texttt{aids}) = (\texttt{aids}, 0.8)$ the first policy entry $(\texttt{aids}, 0.3)$ is violated; the censor thus modifies the first answer to $(\texttt{aids}, 0.3)$. Equivalently, the second answer is modified to $(\texttt{cancer}, 0.2)$. As for the third query, $log_2 = log_0 \cup \{(\texttt{aids}, 0.3), (\texttt{cancer}, 0.2)\}$ combined with $eval_{kb}(\texttt{med}) = (\texttt{med}, 0.9)$ violates both policy entries, because $log_2 \cup \{(\texttt{med}, 0.9)\} \models_{SPL} (\texttt{aids}, 0.6)$ and $log_2 \cup \{(\texttt{med}, 0.9)\} \models_{SPL} (\texttt{cancer}, 0.5)$. That is, we have $S = \{0.3, 0.2\}$ (due to the policy entries). We take its minimum and return $(\texttt{med}, 0.2)$. The complete answer sequence is thus $A = \langle(\texttt{aids}, 0.3), (\texttt{cancer}, 0.2), (\texttt{med}, 0.2)\rangle$. The resulting user history is $log_m = log_0 \cup \{(\texttt{aids}, 0.3), (\texttt{cancer}, 0.2), (\texttt{med}, 0.2)\}$.

On our way to show that the above *cqe*-function is compliant with Definition 1, we need the following two lemmas and then move on to the main theorem:

Lemma 1 (User history is consistent). *For $j = 1 \dots m$ the user history log_j is consistent; that is, $Inc(log_j) = 0$.*

Proof. By assumption, *kb* is consistent in itself and with the a priori knowledge log_0 (see the preconditions at the end of Section 3). But then also the set of database answers $eval_{kb}(q_k)$ with necessity degree above 0 for $k = 1 \dots j$ is consistent. Reducing the correct necessity degree of q_j to a lower value (but still above 0) with the *censor*-function does not influence consistency. As all answers with necessity degree 0 are left out of the user history and the a priori knowledge log_0 is consistent by assumption, each log_j is consistent.

Lemma 2 (User history is secure). *For each user history log_j it holds that there is no policy entry $(q, \beta) \in policy$ such that $log_j \models_{SPL} (q, \beta')$ with $\beta' > \beta$.*

Proof. By assumption, the security property holds for log_0. Inductively, we argue that if log_{j-1} is secure, then also log_j is. In the *censor*-function there are two cases: if upon adding the correct answer to the history the policy is not violated ($S = \emptyset$), log_j is obviously secure. However, when adding the correct answer violates some policy entries ($S \neq \emptyset$), taking the minimal necessity degree $\min S$ avoids the violation. This is due to the fact that in possibilistic logic a logical consequence is only supported up to the necessity degree of the "weakest link" in its proof chain for the entailment; see [3] for details. For example, the possibilistic resolution rule in Section 1 also takes the minimum of the degrees of the input formulas. In other words, because log_{j-1} is secure, addition of the current answer enables the entailment of a violation in a proof chain; we thus weaken the necessity degree of the answer such that no harmful inference is possible anymore.

Theorem 1 (Possibilistic *cqe* preserves confidentiality). *The cqe-function presented in Figure 2 is confidentiality-preserving.*

Proof. We have to identify an alternative knowledge base kb', such that $kb' \cup log_0$ is consistent and the two properties of Definition 1 hold. Let $kb' := log_m$. Clearly, $log_0 \subseteq log_m$ and kb' is consistent by Lemma 1.
Indistinguishability of *kb* and kb' (Item *1*) can be established by induction on the

query sequence and the user history. Base case: Both *cqe*-answer sequences, for *kb* and *kb'*, start with the same log_0 by definition of the *cqe*-function. Inductive case: Assume that calling the *cqe*-function on *kb* and *kb'* led to the same log_{j-1}. We show that for query q_j, the same answer is generated (for *kb* and *kb'*) and hence both *cqe* answer sequences lead to the same log_j. Assume to the contrary that q_j is answered differently: *cqe* on *kb* returns (q_j, β_j) and *cqe* on *kb'* returns (q_j, β'_j) with $\beta_j \neq \beta'_j$. We consider two cases:

- Case 1 ($\beta_j > \beta'_j$): Then, $(q_j, \beta_j) \in log_j$ (because it is returned by *cqe* on *kb* and $\beta_j > 0$). By definition of *kb'*, $log_j \subseteq kb'$; by Lemma 2 and the rule of weight fusion (see Section 1) $kb' \models_{SPL} (q_j, \beta_j)$ and (q_j, β_j) will also be returned as the answer of *kb'*. Hence the assumption that $\beta_j > \beta'_j$ leads to a contradiction.
- Case 2 ($\beta_j < \beta'_j$): It holds that $(q_j, \beta_j) \in log_j$ if $\beta_j > 0$; otherwise $\beta_j = 0$ and $log_j = log_{j-1}$. To deduce (q_j, β'_j) in *kb'*, all formulas in the proof chain for (q_j, β'_j) must have necessity degree β'_j or above (see Proposition 9 in [3]). But such formulas cannot exist in *kb'* because formulas in *kb'* have same or lower degrees than formulas in *kb* and log_{j-1}; indeed, for every (r, γ) such that $kb \cup log_{j-1} \models_{SPL} (r, \gamma)$ it holds that $kb' \models_{SPL} (r, \gamma')$ with $\gamma \leq \gamma'$ due to weight minimization in the *cqe*-function. Hence again we have a contradiction.

We conclude that $\beta_j = \beta'_j$ and thus the same answer and history sequence is generated both from *cqe* calls on *kb* as well as *kb'*. Security of *kb'* (Item *2*) follows directly from Lemma 2, because $eval_{kb}$ is based on implication $\models_{SPL}$.

Lastly, we argue that the runtime complexity of the *cqe*-function is dominated by the complexity of solving the satisfiability (SAT) problem for propositional formulas. In particular for fixed sizes of the query sequence Q and privacy policy *policy*, a number of SAT checks that is bounded by the logarithm of the number of necessity degrees occurring in the inputs *kb* and log_0 and *policy* suffices.

Theorem 2 (Complexity of possibilistic ***cqe*****).** *For fixed-sized Q and policy, the number of SAT checks used in the cqe-function is logarithmically bounded by the number of necessity degrees occurring in the inputs kb, log_0 and policy.*

Proof. For one single query q_j, the *censor*-function determines $eval_{kb}(q_j)$. This can be done with $\lceil \log d_{kb} \rceil$ SAT checks (with $Prop_{kb}$ and q_j as inputs to the SAT solver) where d_{kb} is the number of necessity degrees occurring in *kb*; see [3]. Next, for each of the *card*(*policy*) many policy entries, the censor determines $log_{j-1} \cup eval_{kb}(q_j) \models_{SPL} (q, \beta')$; this takes accordingly $\lceil \log(d_{log_{j-1}} + 1) \rceil$ SAT checks (with $Prop_{log_{j-1}}$, q_j and q as inputs) where $d_{log_{j-1}}$ is the number of necessity degrees occurring in log_{j-1}. In the worst case, in log_{j-1} all necessity degrees mentioned in *kb* and log_0 and *policy* occur. Hence let d be the number of necessity degrees mentioned in *kb* and log_0 and *policy*. Then the combined runtime of the *cqe*-function for the whole query sequence $Q = \langle q_1, \ldots, q_m \rangle$ is bounded by

$$m \cdot [\lceil \log d \rceil \cdot SAT + (card(policy) \cdot \lceil \log(d+1) \rceil \cdot SAT)]$$

By taking m and $card(policy)$ as constants, the result follows.

Note that although the propositional SAT problem is the classical NP-complete problem, several highly efficient SAT solving programs exist. In the context of CQE, such SAT solvers have been used to preprocess a secure ("inference-proof") view of an input database (see [7]). Hence, it appears to be the case that also possibilistic CQE is efficiently implementable.

5 Discussion and Related Work

In summary, we presented a security definition and a Controlled Query Evaluation function that avoids harmful inferences which would disclose confidential information in a possibilistic database. It used weakening of necessity degrees to achieve compliance with a personalized privacy policy. This can be seen as a form of data restriction: the query formulas are not modified, instead less specific answers are returned to the user where as a last resort answers with necessity degree 0 are the most general (and least informative) answers that can be given.

In this sense the presented possibilistic *cqe*-function is akin to generalization techniques for k-anonymity [8] that in the extreme case have one and the same value for an attribute in all tuples. Another approach is minimal upgrading of attributes in multilevel secure databases: [9] devise a set of algorithms to satisfy lower bound (and upper bound) constraints for attribute classifications. In [10] disclosure of secret information is detected retroactively after all answers have been truthfully given; their model is either based on sets of possible worlds (the "possibilistic" case) or on a probability distribution on all possible worlds (the "probabilistic" case). Yet, in contrast to all the above, we apply weakening of necessity degrees in a interactive setting with respect to query sequences. Interestingly, [10] consider gaining confidence in secret information harmful (but not losing confidence); this is similar to our security definition (Definition 1). [11] analyze secrecy in multi-agent systems in the "runs and systems" framework and provide several formal secrecy definitions including a setting with plausibility measures to represent uncertainty. Previous approaches for CQE in incomplete databases (see [1, 6]) and approaches that detect inferences in ontological knowledge bases [12] handle the case that a query can have one of the three values *true*, *false* or *undefined*. In comparison to these, weakening of necessity degrees offers a fine-grained way to protect secret information while still returning useful answers. The approach in [13] models a priori knowledge in a fuzzy relation while the database content is assumed to be exact.

There are several open research questions for the possibilistic CQE setting that can be pursued further; we just list a few:

- Can we apply a measure for the loss of utility of the weakened answers?

- Can heuristics be applied to the weakening process to avoid a high loss of utility? For example, if $\alpha > \alpha'$ prefer weakening (p', α') to weakening (p, α).
- Can other preferences on the possible worlds (like Φ- or Δ-based preferences; see [4]) be included?
- Can the *cqe*-approach be extended to other base logics (like fragments of first-order logic)?
- Can the *cqe*-approach be extended to inconsistent knowledge bases or knowledge bases with updates?
- Can the possibilistic setting also be used in a preprocessing approach that computes a secure ("inference-proof"; see [7]) view of the knowledge base?

References

1. Biskup, J., Weibert, T.: Keeping secrets in incomplete databases. International Journal of Information Security **7**(3) (2008) 199–217
2. Biskup, J., Bonatti, P.: Controlled query evaluation for enforcing confidentiality in complete information systems. International Journal of Information Security **3** (2004) 14–27
3. Lang, J.: Possibilistic logic: complexity and algorithms. In: Handbook of Defeasible Reasoning and Uncertainty Management Systems. Volume 5. Kluwer Academic Publishers (2000) 179–200
4. Benferhat, S., Dubois, D., Prade, H.: Towards a possibilistic logic handling of preferences. Applied Intelligence **14**(3) (2001) 303–317
5. Dubois, D., Prade, H.: Possibilistic logic: a retrospective and prospective view. Fuzzy Sets and Systems **144**(1) (2004) 3–23
6. Biskup, J., Tadros, C., Wiese, L.: Towards controlled query evaluation for incomplete first-order databases. In: Foundations of Information and Knowledge Systems (FoIKS2010). Lecture Notes in Computer Science (2010) To appear.
7. Tadros, C., Wiese, L.: Using SAT-solvers to compute inference-proof database instances. In: 4th International Workshop on Data Privacy Management (DPM09). Volume 5939 of Lecture Notes in Computer Science. (2010)
8. Ciriani, V., di Vimercati, S.D.C., Foresti, S., Samarati, P.: *k*-anonymity. In: Secure Data Management in Decentralized Systems. Volume 33 of Advances in Information Security. Springer (2007) 323–353
9. Dawson, S., di Vimercati, S.D.C., Lincoln, P., Samarati, P.: Minimal data upgrading to prevent inference and association. In: Symposium on Principles of Database Systems (PODS1999), ACM Press (1999) 114–125
10. Evfimievski, A.V., Fagin, R., Woodruff, D.P.: Epistemic privacy. In: Symposium on Principles of Database Systems (PODS2008), ACM (2008) 171–180
11. Halpern, J.Y., O'Neill, K.R.: Secrecy in multiagent systems. ACM Transactions on Information and System Security (TISSEC) **12**(1) (2008)
12. Stouppa, P., Studer, T.: Data privacy for knowledge bases. In: Logical Foundations of Computer Science (LFCS). Volume 5407 of Lecture Notes in Computer Science., Springer (2009) 409–421
13. Hale, J., Shenoi, S.: Analyzing fd inference in relational databases. Data & Knowledge Engineering **18**(2) (1996) 167–183