

Using Deductive Databases Technology for Approximate Query Answering in Partially Complete Databases

Alvaro Cortés[◇] Marc Denecker[◇]
Ofer Arieli* Maurice Bruynooghe[◇]

◇: Katholieke Universiteit Leuven, Belgium

*: The Academic College of Tel-Aviv/Yaffo, Israel

Reiter's Closed World Assumption (1978)

In an arbitrary relational database, the Closed World Assumption (*CWA*) rules as *false* all those tuples that are not in the database.

Train Time Table

<i>Destination</i>	<i>Time</i>
Brussels Centraal	8:03
Antwerpen Centraal	8:05
Ghent St. Pieters	8:13
Brugge	8:22

Reiter's *CWA* allows us to conclude that there is no train to Hasselt at 8:04, for instance.

The *CWA* applies to stand-alone databases storing *complete* (and correct) information about the world.

Incomplete databases

What happens when the database is not complete?

Telephone		Department	
<i>Name</i>	<i>Telephone</i>	<i>Name</i>	<i>Department</i>
Leen Desmet	6531421	Bart Delvaux	Computer Sci.
Leen Desmet	09-23314	Leen Desmet	Philosophy
Bart Delvaux	5985625	David Finner	Computer Sci.

This database does not store *complete knowledge* about collaborators from the Philosophy department.

⇒ The *CWA* is not the correct approach in this context

An alternative view: Open-World Assumption

The other extreme: The **Open-World Assumption (OWA)**

- The world can be in any state in which all database atoms are true
- A common approach in data integration systems
- The **OWA** is often too incomplete and underestimates the knowledge in a database.

How to identify those parts of the database that *are* complete?

Different approaches were presented to specify that the database is partially complete.

- First approach: [Motro 88]
- We follow the approach of Local Closed-World Assumption of [Levy96] and [CDAB05]

A specification of the “areas” in the real world in which the tables of the database contain *all* (true) tuples.

Expressing Local Closed-World Assumptions

A *Local Closed-World Assumption* (\mathcal{LCWA}) is an expression [CDAB05]:

$$\mathcal{LCWA}(P(\bar{x}), \Psi[\bar{x}])$$

Where:

- $\Psi[\bar{x}]$: The window of expertise of the \mathcal{LCWA}
- $P(\bar{x})$: A database predicate, called the object of the \mathcal{LCWA}
- $\Psi[\bar{x}]$: The free variables in Ψ are a subset of \bar{x}

”For all \bar{x} such that Ψ holds in the *real world*, if $P(\bar{x})$ is true in the real world, then $P(\bar{x})$ appears in the database”

This extends Levy’s formalism of local completeness constraints, which allows only positive conjunctive queries as windows of expertise.

Example of the *LCWA*

Database knows about all the Telephone numbers of people in the CS (computer Science) department:

$LCWA(\text{Telephone}(x, y), \text{Dept}(x, \text{CS}))$

- Windows of expertise: $\text{Dept}(x, \text{CS})$
- Object of the *LCWA*: $\text{Telephone}(x, y)$

”For all persons x of the department of computer science, all true facts of the form $\text{Telephone}(x, y)$ appear in the database.”

Semantics of the LCWA

Let D a set of ground atoms (a database) and the LCWA expression

$$\theta = \text{LCWA}(P(\bar{x}), \Psi[\bar{x}]),$$

the meaning of θ given D is

$$\mathcal{M}_D(\theta) = \forall \bar{x} \left(\Psi[\bar{x}] \supset (P(\bar{x}) \equiv (P(\bar{x}) \in P^D)) \right)$$

Where “ $P(\bar{x}) \in P^D$ ” is a shorthand for

$$\bigvee_{\bar{a} \in P^D} (\bar{x} = \bar{a})$$

Semantics of a Locally Complete Database

A locally closed database \mathcal{D} over Σ is pair (D, \mathcal{L}) , where D is a database instance and $\mathcal{L} = \{\theta_1, \dots, \theta_m\}$ is a finite set of LCWAs.

The *semantics* of \mathcal{D} is:

$$\mathcal{M}(\mathcal{D}) = \mathcal{A} \wedge \bigwedge_{j=1}^m \mathcal{M}_D(\theta_j) \wedge \text{UNA}(\Sigma) \wedge \text{DCA}(\Sigma).$$

Where

- \mathcal{A} : The conjunction of atoms in D .
- $\text{UNA}(\Sigma)$: Unique Names Axioms.
- $\text{DCA}(\Sigma)$: Domain Closure Axioms.

Query Answering

We are interested in evaluating queries Q with respect to $\mathcal{M}(\mathcal{D})$:

- \bar{t} is a *certain answer* for $Q[\bar{x}]$ in $\mathcal{M}(\mathcal{D})$, if

$$\mathcal{M}(\mathcal{D}) \models Q[\bar{t}/\bar{x}].$$

Set of certain answers: $Cert_{\mathcal{D}}(Q[\bar{x}])$.

- \bar{t} is a *possible answer* for $Q[\bar{x}]$ in $\mathcal{M}(\mathcal{D})$, if

$$\mathcal{M}(\mathcal{D}) \cup Q[\bar{t}/\bar{x}] \text{ is satisfiable.}$$

Set of possible answers: $Poss_{\mathcal{D}}(Q[\bar{x}])$.

Complexity result [CDAB05]:

- Computing $Cert_{\mathcal{D}}(Q[\bar{x}])$ is in co-NP-complete.
- Computing $Poss_{\mathcal{D}}(Q[\bar{x}])$ is in NP-complete.

\Rightarrow Checking whether $\mathcal{M}(\mathcal{D}) \models Q[\bar{d}]$ is an expensive task.

Query Answering II

We present a tractable method for *approximate query answering*

- Approximate answers to queries:
 - Under approximation of **certain answers**.
 - Over approximation of **possible answers**.

What kind of technology? Deductive databases

This strategy allows:

- An **efficient** and **sound** method to compute certain and possible answers.
- To generalize previous approaches that imposed restrictions on the structure of the database [CDAB-AAAI07].

The compromise:

- The answers are incomplete (but in important cases, complete)

The Approach Using Using Deductive Databases

- Translate \mathcal{D} into a **deductive database under the Well-Founded semantics**
- Certain or possible queries are translated into the deductive database query
- Translated queries are then evaluated using **standard deductive databases inference systems**

Constructing the Deductive Database

Given:

- Locally closed database $\mathcal{D} = (D, \mathcal{L})$ based on vocabulary Σ .

Define the *program vocabulary* Σ_{Π} of a logic program Π based on Σ as follows:

- Variables and constant symbols are copied from Σ to Σ_{Π} ;
- for every predicate P in Σ introduce symbols in Σ_{Π} .
 - P_+^c, P_-^c : Represent what is certainly true/false
 - P_+^p, P_-^p : Represent what is possibly true/false
 - P_{db} : Represents the atoms of the database

The Deductive Database for Certain Answers

Let $\mathfrak{D} = (D, \mathcal{L})$ be a locally closed database based on vocabulary Σ .
 The *approximation program* $\Pi_{\mathfrak{D}}$ based on Σ_{Π} is defined as follows:

$$\Pi_{\mathfrak{D}} : \left\{ \begin{array}{l} P_{db}(\bar{a}) \text{ for every atom } P(\bar{a}) \text{ in } D. \\ P_{+}^c(\bar{x}) \leftarrow P_{db}(\bar{x}). \\ P_{-}^c(\bar{x}) \leftarrow \text{not } P_{db}(\bar{x}), \Psi_{+}^c(\bar{x}). \end{array} \right\}$$

$\Psi_{+}^c(\bar{x})$ is derived from $\Psi_P(\bar{x})$, the window of expertise of predicate P in \mathcal{L} , by:

- replacing positive literals $Q(\bar{x})$ by $Q_{+}^c(\bar{x})$ and
- negative literals $\neg Q(\bar{x})$ by $Q_{-}^c(\bar{x})$.

The Deductive Database for Possible Answers

For possible answers, extend the approximation database $\Pi_{\mathcal{D}}$ with the rules

$$\left\{ \begin{array}{l} P_+^p(\bar{x}) \leftarrow \text{not } P_-^c(\bar{x}). \\ P_-^p(\bar{x}) \leftarrow \text{not } P_+^c(\bar{x}). \end{array} \right\}$$

and perform a similar transformation on the query where positive occurrences of an atom $P(\bar{x})$ are replaced by $P_+^p(\bar{x})$ and negative occurrences by $P_-^p(\bar{x})$.

Query Answering Using Deductive Databases

Consider: $\mathcal{L} = \{\mathcal{LCWA}(P(x), Q(x)), \mathcal{LCWA}(Q(x), x=c)\}$,
 $D = \{P(a), Q(c)\}$, and the query $Q[x] = P(x)$.

$$\Pi_{\mathfrak{D}} : \left\{ \begin{array}{ll} P_{db}(\bar{a}). & Q_{db}(\bar{c}). \\ P_+^c(\bar{x}) \leftarrow P_{db}(\bar{x}). & Q_+^c(\bar{x}) \leftarrow Q_{db}(\bar{x}). \\ P_-^c(\bar{x}) \leftarrow \text{not } P_{db}(\bar{x}), Q_+^c(\bar{x}). & Q_-^c(\bar{x}) \leftarrow \text{not } Q_{db}(\bar{x}), x = c. \\ P_+^p(\bar{x}) \leftarrow \text{not } P_-^c(\bar{x}). & P_-^p(\bar{x}) \leftarrow \text{not } P_+^c(\bar{x}). \\ Q_+^p(\bar{x}) \leftarrow \text{not } Q_-^c(\bar{x}). & Q_-^p(\bar{x}) \leftarrow \text{not } Q_+^c(\bar{x}). \end{array} \right\}$$

$Cert_{\Pi_{\mathfrak{D}}}(Q[x]) = \{a\}$, $Poss_{\Pi_{\mathfrak{D}}}(Q[x]) = Dom(\mathfrak{D}) - \{c\}$.

$Poss_{\Pi_{\mathfrak{D}}}(Q[x])$ is not domain independent!

Soundness & Completeness

Let $\mathfrak{D} = (D, \mathcal{L})$ be a locally closed database.

(Soundness) For a query Q :

$$Cert_{\Pi_{\mathfrak{D}}}(Q[\bar{x}]) \subseteq Cert_{\mathfrak{D}}(Q[\bar{x}]) \subseteq Poss_{\mathfrak{D}}(Q[\bar{x}]) \subseteq Poss_{\Pi_{\mathfrak{D}}}(Q[\bar{x}])$$

(Completeness) If every window of expertise in \mathcal{L} is a conjunction of literals:

If $Q[\bar{x}]$ is a conjunction of literals, then

$$Cert_{\Pi_{\mathfrak{D}}}(Q[\bar{x}]) = Cert_{\mathfrak{D}}(Q[\bar{x}]).$$

If $Q[\bar{x}]$ is a disjunction of literals, then

$$Poss_{\Pi_{\mathfrak{D}}}(Q[\bar{x}]) = Poss_{\mathfrak{D}}(Q[\bar{x}]).$$

Computational Complexity

The computation time of $Cert_{\Pi_{\mathcal{D}}}(\mathcal{Q}[\bar{x}])$ and $Poss_{\Pi_{\mathcal{D}}}(\mathcal{Q}[\bar{x}])$ by $\Pi_{\mathcal{D}}$ is **polynomial** in $|D|$.

Domain Independent Queries

⇒ Queries that can be answered with the information in the database **regardless the underlying domain**.

- $Poss_{\Pi_{\mathcal{D}}}(Q[x])$ was **not** domain independent.
 - $Poss_{\Pi_{\mathcal{D}}}(Q[x]) = Dom(\mathcal{D}) - \{c\}$
- Translation of queries may introduce or delete domain independence. Translation may turn a safe query into an unsafe one or vice versa.
- In locally closed databases, domain independence of queries depend on the form of queries and the form of the LCWAs.
 - Deciding whether a query is domain independent is **undecidable**.
 - **Syntactic restrictions** in queries and LCWAS can be imposed to ensure domain independence.

Conclusions and Ongoing Work

Tractable methods for approximate querying in locally closed databases, based on standard deductive database techniques

Deductive databases approach **generalizes** algorithm based on rewriting techniques

Extensions for future work:

- Integrity constraints.
- **Null** values.
- Views