When is Naïve Evaluation Possible?

Amélie Gheerbrant
University of Edinburgh

Leonid Libkin
University of Edinburgh

Cristina Sirangelo
LSV, ENS-Cachan, INRIA & CNRS

ABSTRACT

The term naïve evaluation refers to evaluating queries over incomplete databases as if nulls were usual data values, i.e., to using the standard database query evaluation engine. Since the semantics of query answering over incomplete databases is that of certain answers, we would like to know when naïve evaluation computes them: i.e., when certain answers can be found without inventing new specialized algorithms. For relational databases it is well known that unions of conjunctive queries possess this desirable property, and results on preservation of formulae under homomorphisms tell us that within relational calculus, this class cannot be extended under the open-world assumption.

Our goal here is twofold. First, we develop a general framework that allows us to determine, for a given semantics of incompleteness, classes of queries for which naïve evaluation computes certain answers. Second, we apply this approach to a variety of semantics, showing that for many classes of queries beyond unions of conjunctive queries, naïve evaluation makes perfect sense under assumptions different from open-world. Our key observations are: (1) naïve evaluation is equivalent to monotonicity of queries with respect to a semantics-induced ordering, and (2) for most reasonable semantics, such monotonicity is captured by preservation under various types of homomorphisms. Using these results we find classes of queries for which naïve evaluation works, e.g., positive first-order formulae for the closed-world semantics. Even more, we introduce a general relation-based framework for defining semantics of incompleteness, show how it can be used to capture many known semantics and to introduce new ones, and describe classes of first-order queries for which naïve evaluation works under such semantics.

1. Introduction

Database applications need to handle incomplete data; this is especially true these days due to the proliferation of data obtained as the result of integrating or exchanging data sets, or data found on the Web. At the same time, there is a huge gap between our theoretical knowledge and the handling of incompleteness in practice:

- In SQL, the design of null-related features is one of the most criticized aspects of the language [13], due to the oversimplification of the model (which even leads to paradoxical behavior: it is consistent with SQL's semantics that $|X| > |Y|$ and $X - Y = \varnothing$, if the set Y contains nulls!)
- In theory, we understand that the proper way of evaluating queries on incomplete databases is to find *certain answers* to them. Unfortunately, for many classes of queries, even within first-order logic, this is an intractable problem [2], and even when it is tractable, there is no guarantee the algorithms can be easily implementable on top of commercial DBMSs [15].

Despite this seemingly enormous gap, there is one instance when theoretical approaches and functionalities of practical systems converge nicely. For some types of queries, evaluating them as if nulls were the usual data values does produce certain answers. This is usually referred to as *naïve evaluation* [1, 19]. To give an example, consider databases with *naïve nulls* (also called marked nulls), that appear most commonly in integration and exchange scenarios, and that can very easily be supported by commercial RDBMSs. Two such relations are shown below, with nulls indicated by the symbol $\perp$ with subscripts:

R:

A	B
1	$\perp_1$
$\perp_2$	$\perp_3$

S:

B	C
$\perp_1$	4
$\perp_3$	5

Suppose we have a conjunctive query $\pi_{AC}(R \bowtie S)$ or, equivalently, $\varphi(x, y) = \exists z \big(R(x, z) \wedge S(z, y)\big)$. Naïve evaluation says: proceed as if nulls were usual values; they are equal only if they are syntactically the same (for instance $\perp_1 = \perp_1$ but $\perp_1 \neq \perp_2$). Then evaluating

the above query results in two tuples: $(1, 4)$, and $(\perp_2, 5)$. Tuples with nulls cannot be certain answers, so we only keep the tuple $(1, 4)$.

There are two observations in order. First, one does not need any new functionalities of the DBMS to find the result of naïve evaluation (in fact most implementations of marked nulls are such that equality tests for them are really the syntactic equality). This is good, but the second point is that in general, naïve evaluation need not compute certain answers.

For the query above, the tuple $(1, 4)$ is however the certain answer. It is found by the naïve evaluation due to the fact [19] that if Q is a union of conjunctive queries, then naïve evaluation works for it (i.e., computes certain answers). This result is not so easy to extend: for instance, [24] showed that under the common open-world semantics (to be properly defined later), if naïve evaluation works for a Boolean first-order (FO) query Q, then Q must be equivalent to a union of conjunctive queries. That result crucially relied on a preservation theorem from mathematical logic [11], and in particular on its version over finite structures [30].

This observation suggests that the limits of naïve evaluation depend on the semantics of incompleteness, and that syntactic restrictions on queries admitting such evaluation may be obtained from preservation theorems in logic. This is the starting point of our investigation. In general we would like to understand how, for a given semantics of incompleteness, we can find the class of queries for which certain answers will be found naïvely.

In slightly more detail, we would like to answer the following three questions:

1. What are the most general conditions underlying naïve evaluation, under different semantics?
2. When can naïve evaluation be characterized by preservation results?
3. How can we find relevant classes of queries that admit naïve evaluation?

We answer these three questions, by clarifying the relationship between semantics, naïve evaluation, preservation, and syntactic classes. Roughly, our results can be seen as establishing the following equivalences:

Naïve evaluation works for a query Q
$\Vert$
Q is monotone wrt the semantic ordering
$\Vert$
Q is preserved under a class of homomorphisms

We now explain the key ideas behind the main equivalences and the terminology we use.

Naïve evaluation and monotonicity For the first group of results, we deal with a very abstract setting that can be applied to many data models (relational, XML, etc) under different semantics. We assume that incomplete database objects x come with a notion of semantics $[\![x]\!]$, which is the set of complete objects they describe. We define the semantic ordering in the standard way: $x \preceq y \Leftrightarrow [\![y]\!] \subseteq [\![x]\!]$ (that is, x is less informative if it describes more objects, i.e., has more incompleteness in it). In this setting we define queries, naïve evaluation, and certain answers and prove that naïve evaluation works for a query iff it is monotone with respect to the semantic ordering. This requires, as a precondition, the existence of what we call a representative set of instances. For many commonly considered semantics, the set of all instances is such and the precondition is not really needed. For other semantics considered previously in various closed-world scenarios [18, 25], the representative set happens to be the set of *cores* of instances (cf. [14, 17]).

Monotonicity and preservation We next connect monotonicity with preservation. To start, we analyze multiple semantics of incompleteness, and come up with a uniform scheme for generating them. The key observation is that each semantics is obtained in two steps. In step one, common to all interpretations, we substitute constant values for nulls. Step two, that essentially defines the semantics, is given by a relation R showing how the result of the substitution can be modified. For instance, under the open-world semantics, tuples can be added; under the strictest form of the closed-world semantics, nothing can be changed at all.

Having done that, we prove that under some very mild condition, monotonicity of a query Q corresponds to preservation under homomorphisms that respect relation R: that is, if Q is true in D (say, for a Boolean Q), and we have a homomorphism respecting R from D to D', then Q is true in D'. Instances of such homomorphisms are the usual homomorphisms, under the open-world semantics, or onto homomorphisms, under (a version of) the closed-world semantics.

Preservation and syntactic classes We have so far established that naïve evaluation is captured by preservation under a class of homomorphisms. Such preservation results are classical in mathematical logic [11], and thus we would like to use them to find syntactic classes of queries for which naïve evaluation works.

One immediate difficulty is that classical logic results are proved for infinite structures, and they tend to fail in the finite [4, 32], or are notoriously hard to establish (a well-known example is Rossman's theorem [30], which answered a question opened for many years). Thus, we are in general happy with good sufficient conditions for preservation, especially if they are given by nice syntactic classes corresponding to meaningful classes of database queries. The key ideas behind the classes we use are restrictions to positive formulae (admitting $\forall$ but disallowing $\neg$) or existential positive formulae (i.e., unions of conjunctive queries), and extending some of them with universally quantified *guarded* formulae.

This gives us a good understanding of what is required to make naïve evaluation work. In Sections 3–5 we carry out the program outlined above and obtain classes of FO queries for which naïve evaluation works under standard relational semantics. Also, to keep notations simple initially, in these early sections we deal with Boolean queries (all results extend to arbitrary queries easily, as we show in Section 9).

In Sections 6–8 we offer a more detailed study of other relational semantics of incompleteness. We look at Codd databases, that model SQL's null features, and discuss lifting semantic orderings from Codd databases to those with naïve nulls, resulting in new incompleteness semantics. We show how our methodology easily produces classes of FO queries which can be evaluated naïvely under those semantics. We look at *minimal* semantics that find their justification in the study of various forms of the closed world assumption. For them, the notion of a core of an instance [17] plays a crucial role: for example, naïve evaluation for previously considered classes of queries is only guaranteed if they are evaluated over cores.

Organization In Section 2, we give the main definitions. In Section 3, we explain the connection between naïve evaluation and monotonicity, and in Section 4 we relate monotonicity to preservation. In Section 5 we deal with Boolean FO queries and provide sufficient conditions for naïve evaluation. In Section 6, we study the Codd semantics of incompleteness, and in Section 7 we lift it to naïve databases, resulting in a new class of semantics, for which we study naïve evaluation. In Section 8, we carry out a similar program for minimal semantics. Section 9 shows how to lift all the results for Boolean queries to queries with free variables. Complete proofs of all the results are in the appendix.

2. Preliminaries

Incomplete databases We begin with some standard definitions. In incomplete databases there are two types of values: constants and nulls. The set of constants is denoted by Const and the set of nulls by Null. These are countably infinite sets. Nulls will normally be denoted by $\bot$, sometimes with sub- or superscripts.

A relational schema (vocabulary) is a set of relation names with associated arities. An incomplete relational instance D assigns to each k-ary relation symbol S from the vocabulary a k-ary relation over $\mathsf{Const} \cup \mathsf{Null}$, i.e., a finite subset of $(\mathsf{Const} \cup \mathsf{Null})^k$. Such incomplete relational instances are referred to as *naïve* databases [1, 19]; note that a null $\bot \in \mathsf{Null}$ can appear multiple times in such an instance. If each null $\bot \in \mathsf{Null}$ appears at most once, we speak of *Codd* databases. If we talk about single relations, it is common to refer to them as naïve tables and Codd tables.

We write $\mathsf{Const}(D)$ and $\mathsf{Null}(D)$ for the sets of constants and nulls that occur in a database D. The *active domain* of D is $\mathrm{adom}(D) = \mathsf{Const}(D) \cup \mathsf{Null}(D)$. A *complete* database D has no nulls, i.e., $\mathrm{adom}(D) \subseteq \mathsf{Const}$.

Homomorphisms They are crucial for us in two contexts: to define the semantics of incomplete databases, and to define the notion of preservation of logical formulae as a condition for naïve evaluation to work.

Given two relational structures D and D', a homomorphism $h : D \to D'$ is a map from the active domain of D to the active domain of D' so that for every relation symbol S, if a tuple $\bar{u}$ is in relation S in D, then the tuple $h(\bar{u})$ is in the relation S in D'.

In database literature, it is common to require that homomorphisms preserve elements of Const. That is, the map h is also required to satisfy $h(c) = c$ for every $c \in \mathsf{Const}$. Of course this can easily be cast as a special instance of the general notion, simply by extending the vocabulary with a constant symbol for each $c \in \mathsf{Const}$. To make clear what our assumptions are, whenever there is any ambiguity, we shall talk about *database homomorphisms* if they are the identity on Const.

Given a homomorphism h and a database D, by $h(D)$ we mean the image of D, i.e., the set of all tuples $S(h(\bar{u}))$ where $S(\bar{u})$ is in D. If $h : D \to D'$ is a homomorphism, then $h(D)$ is a subinstance of D'.

Semantics and valuations We shall see many possible semantics for incomplete information, but first we review two common ones: open-world and closed-world semantics. We need the notion of a *valuation*, which assigns a constant to each null. That is, a valuation is a database homomorphism whose values are in Const.

In general, the semantics $[\![D]\!]$ of an incomplete database is a set of *complete* databases D'. The semantics under the closed-world assumption (or CWA *semantics*) is defined as

$$[\![D]\!]_{\text{CWA}} \;=\; \{h(D) \mid h \text{ is a valuation}\}.$$

The semantics under the open-world assumption (or OWA *semantics*) is defined as

$$[\![D]\!]_{\text{OWA}} \;=\; \left\{ D' \;\middle|\; \begin{array}{l} D' \text{ is complete and} \\ \text{there is a valuation } h : D \to D' \end{array} \right\}.$$

Alternatively, $D' \in [\![D]\!]_{\text{OWA}}$ iff D' is complete and contains a database $D'' \in [\![D]\!]_{\text{CWA}}$.

As an example, consider $D_0 = \{(\bot, \bot'), (\bot', \bot)\}$. Then $[\![D_0]\!]_{\text{CWA}}$ contains all instances $\{(c, c'), (c', c)\}$ with $c, c' \in \mathsf{Const}$, and $[\![D_0]\!]_{\text{OWA}}$ has all instances containing $\{(c, c'), (c', c)\}$, for $c, c' \in \mathsf{Const}$.

Certain answers and naïve evaluation Given an incomplete database D, a semantics of incompleteness $[\![\,]\!]$, and a query Q, one normally computes *certain answers under the* $[\![\,]\!]$ *semantics*:

$$\mathsf{certain}(Q, D) \;= \bigcap \{Q(R) \mid R \in [\![D]\!]\},$$

i.e., answers that are true regardless of the interpretation of nulls. Even for first-order queries, the stan-

dard semantics are problematic in general: finding certain answers under the OWA semantics may be undecidable, and finding them under the CWA semantics may be coNP-hard [2].

Naïve evaluation of a query Q refers to a two-step procedure: first, evaluate Q as if nulls were values (i.e., equal iff they are syntactically the same: e.g., $\perp_1 = \perp_1$, $\perp_1 \neq \perp_2$, $\perp_1 \neq c$ for every $c \in \mathsf{Const}$), and then eliminate tuples with nulls from the result. Note that if Q is a Boolean query, the second step is unnecessary.

We say that *naïve evaluation works for* Q (under semantics $[\![\,]\!]$) if its result is exactly the certain answers under $[\![\,]\!]$, for every D.

Fact 1. (see [19, 24]) *Let Q be a union of conjunctive queries. Then naïve evaluation works for Q under both* OWA *and* CWA*. Moreover, if Q is a Boolean FO query and naïve evaluation works for Q under* OWA*, then Q is equivalent to a union of conjunctive queries.*

The last equivalence result only works under the OWA semantics. Consider again the instance D_0 and a query $\exists x, y\, D(x, y) \wedge D(y, x)$. The certain answer to this query is true under both OWA and CWA, and indeed it evaluates to true naïvely over D_0. On the other hand, a query Q given by $\forall x \exists y\ D(x, y)$ (not equivalent to a union of conjunctive queries) evaluated naïvely, returns true on D_0, but under OWA its certain answer is false. However, under CWA, its certain answer is true. This is not an isolated phenomenon: we will later see that Q belongs to a class, extending unions of conjunctive queries, for which naïve evaluation works under CWA on all databases.

3. Naïve evaluation and monotonicity

The goal of this section is twofold. First we present a very general setting for talking about incompleteness and its semantics, as well as orderings representing the notion of "having more information". We formulate the notion of naïve evaluation in this setting, and establish a result saying that it guarantees to compute certain answers for queries that are monotone.

Database domains, semantics, and ordering We now define a simple abstract setting for handling incompleteness. We operate with just four basic concepts: the set of instances, the set of complete instances, their isomorphism, and their semantics.

A *database domain* is a structure $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$, where $\mathcal{D}$ is a set and $\mathcal{C}$ is its subset, $[\![\cdot]\!]$ is a function from $\mathcal{D}$ to nonempty subsets of $\mathcal{C}$, and $\approx$ is an equivalence relation on $\mathcal{D}$. The interpretation is as follows:

- $\mathcal{D}$ is a set of database objects (e.g., incomplete relational databases over the same schema),
- $\mathcal{C}$ is the set of complete objects (e.g., databases without nulls);
- $[\![x]\!] \subseteq \mathcal{C}$ is the semantics of an incomplete database x, i.e., the set of all complete databases that x can represent; and
- $\approx$ is the structural equivalence relation, that we need to describe the notion of generic queries; for instance, for relational databases, $D \approx D'$ means that they are isomorphic as objects, i.e., $\pi(D) = D'$ for some 1-1 mapping on data values in D.

The semantic function of a database domain lets us describe the degree of incompleteness via an ordering defined as $x \preceq y$ iff $[\![y]\!] \subseteq [\![x]\!]$. Indeed, the less we know about an object, the more other objects it can potentially describe. This setting is reminiscent of the ideas in programming semantics, where partial functions are similarly ordered [16], and such orderings have been used to provide semantics of incompleteness in the past [9, 23, 24, 26, 31]. Note that $\preceq$ is a *preorder*, i.e., it is transitive and reflexive.

There is a natural duality between preorders and semantics: the semantics gives rise to an ordering, but conversely any preorder $\preccurlyeq$ on $\mathcal{D}$ gives a semantics $[\![x]\!]_{\preccurlyeq} = \{y \in \mathcal{C} \mid x \preccurlyeq y\}$. As the least requirement for database domains we need to impose that the semantics $[\![\,]\!]$ and its semantic ordering $\preceq$ agree: that is, the semantics that the ordering $\preceq$ gives rise to is the semantics $[\![\,]\!]$ itself. In that case, we call a database domain *admissible*.

Proposition 1. *A database domain $\mathbb{D}$ is admissible iff the following conditions hold:*

1. *$c \in [\![c]\!]$ for each $c \in \mathcal{C}$;*
2. *if $c \in [\![x]\!]$, then $[\![c]\!] \subseteq [\![x]\!]$.*

All the usual semantics – including those seen in the previous section – satisfy these conditions. The first one says that the semantics of a complete object should contain at least that object. The second says that by removing incompleteness from an object, we cannot get one that denotes more objects.

Certain answers For now we look at Boolean queries in the most abstract setting (we generalize them later). Given a database domain $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$, a *query* is a mapping $Q : \mathcal{D} \to \{0, 1\}$. We use 0 to represent *false* and 1 to represent *true*, as usual. A query is *generic* if $Q(x) = Q(y)$ whenever $x \approx y$.

For each $x \in \mathcal{D}$, the certain answer (under the semantics $[\![\,]\!]$) is

$$\mathsf{certain}(Q, x) = \bigwedge \{Q(c) \mid c \in [\![x]\!]\}$$

We say that *naïve evaluation works* for Q if $Q(x) = \mathsf{certain}(Q, x)$ for every x.

Representative sets We need this concept in order to characterize when naïve evaluation works for queries. Essentially, these sets of objects contain all complete ones and have a proper representation for each object x. Formally, a set $\mathcal{S} \subseteq \mathcal{D}$ is *representative* if

- $\mathcal{C} \subseteq \mathcal{S}$ (it contains all complete objects);
- for each $x \in \mathcal{S}$ there is $y \in [\![x]\!]$ such that $x \approx y$ (every object in $\mathcal{S}$ has a complete object in its semantics that is isomorphic to it); and
- there is a function $\chi_{\mathcal{S}} : \mathcal{D} \to \mathcal{S}$ such that $[\![x]\!] = [\![\chi_{\mathcal{S}}(x)]\!]$ for every $x \in \mathcal{D}$ (each object has a representation in $\mathcal{S}$ with the identical semantics).

In many examples, we shall have $\mathcal{S} = \mathcal{D}$, but there will be others (for instance, when the representative set contains all cores).

We say that a query Q is

- *monotone* if $x \preceq y$ implies $Q(x) \leqslant Q(y)$;
- *weakly monotone* if $y \in [\![x]\!]$ implies $Q(x) \leqslant Q(y)$.

Recall that $x \preceq y$ iff $[\![y]\!] \subseteq [\![x]\!]$. Of course $Q(x) \leqslant Q(y)$ says that if $Q(x)$ is true, then $Q(y)$ is true. Note that in an admissible domain, $y \in [\![x]\!]$ implies $x \preceq y$, so weak monotonicity is indeed weaker than monotonicity.

Theorem 1. *Let $\mathbb{D}$ be an admissible database domain, and Q a generic Boolean query. Assume that $\mathbb{D}$ has a representative set $\mathcal{S}$. Then the following are equivalent:*

1. *Naïve evaluation works for Q;*
2. *Q is monotone;*
3. *Q is weakly monotone and $Q(x) = Q(\chi_{\mathcal{S}}(x))$ for every $x \in \mathcal{D}$.*

This establishes the first promised connection between monotonicity and naïve evaluation. We remark that the equivalence $1 \leftrightarrow 3$ does not actually require the admissibility of $\mathbb{D}$. Extension to non-Boolean queries will be given in Section 9.

4. Semantics, relations, and homomorphisms

We have seen that getting naïve evaluation to work (at least for Boolean queries), is equivalent to their monotonicity and, with an extra condition, even to weak monotonicity. Now we start applying this to concrete semantics. To do so, we need to understand how different semantics can be defined. We explain that most of them are obtained by composing two types of relations: one corresponds to applying valuations to nulls, and the other to specific semantic assumptions such as open or closed-world.

After that, we show a connection between naïve evaluation and preservation under a class of homomorphisms.

Semantics via relations

We have already seen two concrete relational semantics: the OWA semantics $[\![D]\!]_{\text{OWA}}$ and the CWA semantics $[\![D]\!]_{\text{CWA}}$. What is common to them is that they are all defined in two steps. First, valuations are applied to nulls (i.e., nulls are replaced by values). Second, the resulting database may be modified in some way (left as it was for CWA, or expanded arbitrarily for OWA). Our idea is then to capture this via two relations. We now define them in the setting of database domains and then show how they behave in concrete cases. Given a database domain $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$, we consider a pair $\mathcal{R} = (\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ of relations:

The *valuation* relation $\mathcal{R}_{\text{val}} \subseteq \mathcal{D} \times \mathcal{C}$ between arbitrary databases and complete databases that is total and, when restricted to $\mathcal{C}$, is the identity (i.e., $\mathcal{R}_{\text{val}} \cap (\mathcal{C} \times \mathcal{C}) = \{(c, c) \mid c \in \mathcal{C}\}$).

Intuitively, this corresponds to replacing nulls by constant values in databases. Since in the absence of nulls there is nothing to substitute them with, the restriction to $\mathcal{C}$ is the identity, and since for every object there must be some way to replace nulls by constants, the relation is total.

The *semantic* relation $\mathcal{R}_{\text{sem}}$ is a reflexive binary relation on $\mathcal{C}$ (i.e., $\mathcal{R}_{\text{sem}} \subseteq \mathcal{C} \times \mathcal{C}$).

Intuitively, this corresponds to the modification step such as extending complete relations by new tuples. Since, at the very least, one can do nothing with the result of the substitution of nulls by constants, such a relation must be reflexive.

We say that $[\![\,]\!]$ *is given by* $\mathcal{R}$ if $\mathcal{R}$ satisfies the above conditions, and $y \in [\![x]\!]$ iff $(x, y) \in \mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$.

Proposition 2. *Let $\mathbb{D}$ be a database domain whose semantics $[\![\,]\!]$ is given by a pair $\mathcal{R} = (\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$. Then $\mathbb{D}$ is admissible iff $\mathcal{R}_{\text{sem}}$ is transitive.*

When we move to relational databases, OWA and CWA semantics are given by pairs of relations, as follows:

Relation $\mathcal{R}_{\text{val}}$ is the same for both: $(D, D') \in \mathcal{R}_{\text{val}}$ iff $D' = v(D)$ for some valuation v. We denote this relation by $\mathcal{R}^{\text{all}}_{\text{val}}$.

Relation $\mathcal{R}_{\text{sem}}$ For CWA, it is the identity; for OWA, it is the subset relation.

Looking at semantics of incompleteness of relational databases given by pairs $\mathcal{R}$, where $\mathcal{R}_{\text{val}}$ is the relation $\mathcal{R}^{\text{all}}_{\text{val}}$ (e.g., OWA and CWA semantics), we can establish the following.

Proposition 3. *If a relational semantics is given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, then the set of all databases is a representative set.*
In particular, if Q is a generic Boolean query, then naïve evaluation works for Q iff Q is weakly monotone.

Naïve evaluation via homomorphism preservation

We shall now relate weak monotonicity and preservation under homomorphisms (at least for relational semantics). We now deal with relational databases over constants, and say that a mapping h defined on the active domain $\text{adom}(D)$ of D is an $\mathcal{R}_{\text{sem}}$-*homomorphism* from D to D' if $(h(D), D') \in \mathcal{R}_{\text{sem}}$. A query Q is *preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms* if for every database D and every $\mathcal{R}_{\text{sem}}$-homomorphism $h : D \to D'$, if Q is true in D, then Q is true in D'.

Proposition 4. *If a relational semantics is given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$ and Q is a generic Boolean query, then Q is weakly monotone iff it is preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms.*

Putting everything together, we have our first key result for naïve evaluation over incomplete databases.

Theorem 2. *Assume that a relational incompleteness semantics is given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, and let Q be a generic Boolean query. Then naïve evaluation works for Q iff Q is preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms.*

Relational semantics: OWA, CWA, and WCWA We now investigate what the notion of preservation under $\mathcal{R}_{\text{sem}}$-homomorphisms means for the standard OWA and CWA semantics. We also observe that another well studied preservation notion is closely related to a previously studied semantics of incompleteness [28], which we call WCWA, or *weak* closed-world assumption.

OWA **semantics** In this case $\mathcal{R}_{\text{sem}}$ is the subset relation, and $\mathcal{R}_{\text{sem}}$-homomorphisms are the usual homomorphisms.

CWA **semantics** In this case $\mathcal{R}_{\text{sem}}$ is the identity, and $\mathcal{R}_{\text{sem}}$-homomorphisms are the *strong onto* homomorphism, i.e., homomorphisms $D \to h(D)$.

One may ask what happens to the usual onto homomorphisms, and it turns out that they correspond to another semantics, considered in [27]. A *strong onto* homomorphism h is one mapping D to $h(D)$, while an *onto* homomorphism $h : D \to D'$ is one satisfying $h(\text{adom}(D)) = \text{adom}(D')$. In other words, no elements except those already present in $h(D)$ can appear in D'.

The semantics considered in [27] (defined in a slightly different, deductive-database way) boils down to the following: $[\![D]\!]_{\text{WCWA}}$ consists of all complete D' such that there is a valuation h satisfying $h(D) \subseteq D'$ and $\text{adom}(D') = \text{adom}(h(D))$. Here WCWA stands for *weak* CWA. This semantics is given by $\mathcal{R}^{\text{all}}_{\text{val}}$ and relation $\mathcal{R}_{\text{sem}}$ containing all pairs (D, D') so that $D \subseteq D'$ and $\text{adom}(D) = \text{adom}(D')$. In other words, D can be expanded only within its active domain.

For this relation $\mathcal{R}_{\text{sem}}$, the notion of preservation under $\mathcal{R}_{\text{sem}}$-homomorphisms is exactly the notion of preservation of under onto homomorphisms. Thus, the WCWA semantics, defined long time ago, also corresponds to a very natural logical notion of preservation.

Note that $[\![D]\!]_{\text{CWA}} \subseteq [\![D]\!]_{\text{WCWA}} \subseteq [\![D]\!]_{\text{OWA}}$, and in general inclusions can be strict. For instance, if $D = \{(\bot, \bot')\}$, then $\{(1,2)\}$ is in $[\![D]\!]_{\text{CWA}}$, while $\{(1,2),(2,1)\}$ is not in $[\![D]\!]_{\text{CWA}}$ but is in $[\![D]\!]_{\text{WCWA}}$, since it added a tuple $(2,1)$ that uses elements already presented $\{(1,2)\}$.

These three semantics lead to semantic orderings $\preceq_*$, where $*$ is OWA, CWA, or WCWA. They are characterized via database homomorphisms as follows (the first item was already shown in [24]).

Proposition 5. *$D \preceq_{\text{OWA}} D'$ (respectively $D \preceq_{\text{CWA}} D'$ or $D \preceq_{\text{WCWA}} D'$) iff there is a database homomorphism (respectively, strong onto, or onto database homomorphism) from D to D'.*

Naïve evaluation and relational semantics

We can finally state the equivalence of naïve evaluation and homomorphism preservation for three concrete relational semantics:

Corollary 1. *Let Q be a Boolean generic relational query. Then:*

- *Under* OWA*, naïve evaluation works for Q iff Q is preserved under homomorphisms.*
- *Under* CWA*, naïve evaluation works for Q iff Q is preserved under strong onto homomorphisms.*
- *Under* WCWA*, naïve evaluation works for Q iff Q is preserved under onto homomorphisms.*

5. Naïve evaluation and preservation for FO queries

Corollary 1 reduces the problem of checking whether naïve evaluation works to preservation under homomorphisms. Thus, for FO queries, we deal with a very well known notion in logic [11]. However, what we need is preservation on *finite* structures, and those notions are well known to behave differently from their infinite counterpart. In fact, it was only proved recently by Rossman that for FO sentences, preservation under arbitrary homomorphisms in the finite is equivalent to being an existential positive formula [30]. In database language, this means being a union of conjunctive queries, which led to an observation [24] that naïve evaluation works for a Boolean FO query Q iff Q is equivalent to a union of conjunctive queries.

The difficulty in establishing preservation results in the finite is due to losing access to classical logical tools such as compactness. Rossman's theorem, for instance, was a major open problem for many years. To make matters worse, even some existing infinite preservation results [21] have holes in their proofs.

Thus, it is unrealistic for a single paper to settle several very hard problems concerning preservation results in the finite (sometimes even without infinite analogs!). What we shall do instead is settle for classes of queries that *imply* preservation, and at the same time are easy to describe syntactically.

Positive and existential positive formulae Recall that the class Pos of *positive* formulae is defined inductively as follows:

- *true* and *false* are in Pos;
- every positive atomic formula (i.e., $R(\bar{x})$ or $x = y$) is in Pos;
- if $\varphi, \psi \in \mathsf{Pos}$, then $\varphi \vee \psi$ and $\varphi \wedge \psi$ are in Pos;
- if φ is in Pos, then $\exists x\varphi$ and $\forall x\psi$ are in Pos.

If only $\exists x\varphi$ remains in the class, we obtain the class $\exists\mathsf{Pos}$ of *existential positive formulae*. Safe formulae from $\exists\mathsf{Pos}$ are also known as unions of conjunctive queries.

Rossman's theorem [30] says that an FO sentence φ is preserved under homomorphisms over finite structures iff φ is equivalent to a sentence from $\exists\mathsf{Pos}$. Lyndon's theorem [11] says that an FO sentence φ is preserved under onto homomorphisms (over arbitrary structures) iff φ is equivalent to a sentence from Pos. Lyndon's theorem fails in the finite [4, 32] but the implication from being positive to preservation is still valid.

A characterization of preservation under strong onto homomorphisms was stated in [20, 21], but the syntactic class had a rather messy definition and was limited to a single binary relation. Even worse, we discovered a gap in one of the key lemmas in [21]. So instead we propose a simple extension of positive formulae that gives preservation under strong onto homomorphisms.

Extensions with universal guards The fragment $\mathsf{Pos} + \forall\mathsf{G}$, whose definition is inspired by [12], extends Pos with universal guards. It is defined as a fragment closed under all the formation rules for Pos and, in addition, the following rule:

- for a $\mathsf{Pos} + \forall\mathsf{G}$ formula φ, a tuple of distinct variables $\bar{x}$, and a relation symbol R, the formula $\forall\bar{x}\big(R(\bar{x}) \rightarrow \varphi\big)$ is in $\mathsf{Pos} + \forall\mathsf{G}$.

Clearly we have $\exists\mathsf{Pos} \subsetneq \mathsf{Pos} \subsetneq \mathsf{Pos} + \forall\mathsf{G}$.

Proposition 6. *Sentences in* $\mathsf{Pos} + \forall\mathsf{G}$ *are preserved under strong onto homomorphisms.*

We now combine all the previous implications (preservation → monotonicity → naïve evaluation) to show that naïve evaluation can work beyond unions of conjunctive queries under realistic semantic assumptions.

Theorem 3. *Let Q be a Boolean FO query. Then:*

- *If Q is in* $\exists\mathsf{Pos}$*, then naïve evaluation works for Q under* OWA.
- *If Q is in* Pos*, then naïve evaluation works for Q under* WCWA.
- *If Q is in* $\mathsf{Pos} + \forall\mathsf{G}$*, then naïve evaluation works for Q under* CWA.

Contrast this with the result of [24] saying that under OWA, the first statement is 'if and only if', i.e., one cannot go beyond $\exists\mathsf{Pos}$. Now we see that one can indeed go well beyond that class, essentially limiting only unrestricted negation, and still use naïve evaluation.

One immediate question is what happens with non-Boolean queries. There is a simple answer: *all results extend to non-Boolean queries.* We explain how this is done in Section 9, once we have looked at other semantics (as the lifting will apply to all of them).

6. Codd databases

We now look at a common way of interpreting nulls, essentially adopted by SQL. *Codd databases* are just like naïve databases, except that nulls cannot repeat. This model actually pre-dates SQL, which adopted essentially this approach, complementing it with the 3-valued logic for comparisons in queries. We now use our framework to describe Codd databases and semantic orderings for them.

Consider a function χ_{Codd} from naïve to Codd databases that "forgets" about equalities of nulls, e.g.,:

$$\chi_{\text{Codd}}\left(\begin{array}{|c|c|c|}\hline 1 & \bot & \bot' \\ \hline \bot & \bot' & 2 \\ \hline\end{array}\right) = \begin{array}{|c|c|c|}\hline 1 & \bot_1 & \bot_2 \\ \hline \bot_3 & \bot_4 & 2 \\ \hline\end{array}$$

Formally, for an instance D with n occurrences of nulls (not necessarily distinct), $\chi_{\text{Codd}}(D)$ replaces these occurrences with fresh nulls $\bot_1, \ldots, \bot_n$. It generates semantics $[\![D]\!]^{\text{Codd}}_{*} = [\![\chi_{\text{Codd}}(D)]\!]_{*}$, where $*$ is one of OWA, CWA, or WCWA. One can easily check that for all of them, the set of Codd databases is a representative set, with χ_{Codd} the witnessing function. Hence:

Corollary 2. *If Q is a Boolean generic relational query, then naïve evaluation works for it under the Codd interpretation and* OWA *(resp.,* CWA*, or* WCWA*) iff Q is preserved under homomorphisms (resp., strong onto, or onto homomorphisms) and $Q(D) = Q(\chi_{\text{Codd}}(D))$ for every database D.*

This explains a well known phenomenon that naïve evaluation under Codd semantics is problematic for instances with repeating nulls [1, 19]: the condition $Q(D) = Q(\chi_{\text{Codd}}(D))$ essentially forbids any equality comparisons in queries, ruling out operations such as join and selection.

Semantic orderings of Codd databases We now look at the orderings $\preceq_{\text{OWA}}$ and $\preceq_{\text{CWA}}$ (stating the existence of homomorphisms or strong onto homomorphisms) in the context of Codd databases. Note that

over Codd databases, these are the orderings given by the semantics $[\![\,]\!]^{\text{Codd}}_{\text{OWA}}$ and $[\![\,]\!]^{\text{Codd}}_{\text{CWA}}$.

For tuples $t = (a_1, \ldots, a_n)$ and $t' = (a'_1, \ldots, a'_n)$ over $\mathsf{Const} \cup \mathsf{Null}$, we write $t \sqsubseteq t'$ if $a_i \in \mathsf{Const}$ implies $a'_i = a_i$. There are two standard ways of lifting $\sqsubseteq$ to sets:

$$\begin{array}{lcl} D \sqsubseteq^{\text{H}} D' & \Leftrightarrow & \forall t \in D\ \exists t' \in D' : t \sqsubseteq t' \\ D \sqsubseteq^{\text{P}} D' & \Leftrightarrow & \forall t' \in D'\ \exists t \in D : t \sqsubseteq t' \ \text{ and } D \sqsubseteq^{\text{H}} D' \end{array}$$

Superscripts H and P stand for Hoare and Plotkin, who first studied these orderings in the context of the semantics of concurrent processes, cf. [16]. These had been previously accepted as the correct orderings to represent the OWA and the CWA semantics over Codd databases [9, 23, 26, 31]. This was partly justified in [24] which proved that over Codd databases,

- $\preceq_{\text{OWA}}$ and $\sqsubseteq^{\text{H}}$ coincide;
- $D \preceq_{\text{CWA}} D'$ iff $D \sqsubseteq^{\text{P}} D'$ and relation $\sqsubseteq$ has a perfect matching from D' to D.

So this leads to a question: is there is a "natural" semantic ordering over naïve databases that, when restricted to Codd databases, coincides precisely with $\sqsubseteq^{\text{P}}$? In the next section, we present such an ordering, and show that it gives rise to a whole new family of semantics of incompleteness.

7. Powerset semantics

Our search for the answer to the question at the end of the previous section leads us to consider a new class of semantics of incompleteness, in which not one, but several valuations can be applied to nulls. In other words, we produce several valuations (hence the name *powerset semantics*), and then combine them into a single one. Notationally, we distinguish them by using $(\!|\ |\!)$ brackets.

We start with a semantics defined as follows: $D' \in (\!|D|\!)_{\text{CWA}}$ iff there exists a set of valuations $h_1, \ldots, h_n$ on D so that $D' = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$. We call it the CWA powerset semantics, and denote the resulting semantic ordering by $\Subset_{\text{CWA}}$ (that is, $D \Subset_{\text{CWA}} D'$ iff $(\!|D'|\!)_{\text{CWA}} \subseteq (\!|D|\!)_{\text{CWA}}$). We explain the intuition behind this semantics after describing the ordering $\Subset_{\text{CWA}}$.

Proposition 7.
- *$D \Subset_{\text{CWA}} D'$ iff there exists a set of database homomorphisms $h_1, \ldots, h_n$ defined on D so that $D' = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$.*
- *Over Codd databases, $\Subset_{\text{CWA}}$ and $\sqsubseteq^{\text{P}}$ coincide.*

Update justification for $\preceq_{\text{OWA}}, \preceq_{\text{CWA}}$ and $\Subset_{\text{CWA}}$

To provide an intuition behind $(\!|\ |\!)_{\text{CWA}}$, we first use SQL's interpretation of nulls. Suppose we have two tuples $(1, 2)$ and $(2, 2)$, and somehow lose the value of the first attribute. SQL has a unique null value, so both tuples become $(\texttt{null}, 2)$, which thus must represent the instance $\{(1, 2), (2, 2)\}$ even under CWA, since no tuples were lost, only individual values. Alternatively, one can view this as an *allowed update*, under CWA, from $(\texttt{null}, 2)$, that produces a more informative instance $\{(1, 2), (2, 2)\}$. This idea of using updates to describe semantic orderings was developed for Codd databases in [23] to justify orderings $\sqsubseteq^{\text{H}}$ and $\sqsubseteq^{\text{P}}$. Now we show how to extend it to naïve databases.

We consider updates $D \rightarrowtail D'$ which are elementary steps that make D' more informative than D. We state which updates are intuitively allowed under different assumptions, and then prove that *sequences* of such updates precisely capture information orderings $\preceq_{\text{OWA}}, \preceq_{\text{CWA}}$ and $\Subset_{\text{CWA}}$.

Let $\bot$ be a null that occurs in D. By $D[v/\bot]$ we mean D in which $v \in \mathsf{Const} \cup \mathsf{Null}$ replaces $\bot$ everywhere. By $D \cup R(t)$ we mean D in which tuple t is added to relation R. Now we consider three types of updates.

- CWA update: $D \rightarrowtail_{\text{CWA}} D[v/\bot]$; this update simply replaces a null $\bot$ everywhere in D;
- copying CWA update: $D \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}} D[v/\bot] \cup D^{\text{fresh}}$, where D^{fresh} is a copy of D in which all nulls are replaced by fresh ones. This is a relaxation of CWA: we can add tuples in an update, but only in a very limited way, if they mimic the original database.
- OWA update: $D \rightarrowtail_{\text{OWA}} D \cup R(t)$, which allows, as is normal under OWA, to add a tuple to D.

Each type of updates defines a binary relation on instances. Sequences of such updates give us the semantic orderings. We now state the result and explain its meaning. We use the superscript * to denote the transitive closure of a relation.

Theorem 4.
- $\rightarrowtail^*_{\text{CWA}} = \preceq_{\text{CWA}}$;
- $(\rightarrowtail_{\text{CWA}} \cup \rightarrowtail_{\text{OWA}})^* = \preceq_{\text{OWA}}$;
- $(\rightarrowtail_{\text{CWA}} \cup \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}})^* = \Subset_{\text{CWA}}$.

In other words, D is less informative than D' iff D' is obtained from D by a sequence of

- CWA updates, under CWA;
- CWA and OWA updates, under OWA;
- CWA updates (both usual and copying), under the $(\!|\ |\!)_{\text{CWA}}$ semantics.

Abstract framework for powerset semantics

We now cast the powerset semantics in our general relation-based framework, which enables us to establish when naïve evaluation works for it. For a set $\mathcal{D}$ of databases and a set $\mathcal{C}$ of complete databases, we have a pair $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ of relations with $\boldsymbol{\mathcal{R}}_{\text{val}} \subseteq \mathcal{D} \times 2^{\mathcal{C}}$ and $\boldsymbol{\mathcal{R}}_{\text{sem}} \subseteq 2^{\mathcal{C}} \times \mathcal{C}$. The first relation corresponds to applying multiple valuations (e.g., relating D with sets

$\{h_1(D), \ldots, h_n(D)\}$). The second relation, in our example, is $\mathcal{R}_\cup = \{(\mathcal{X}, X) \mid X = \bigcup \mathcal{X}\}$.

The basic conditions on these relations are essentially the same as we used before for non-powerset semantics except that we need to deal with relations between $\mathcal{C}$ and $2^{\mathcal{C}}$. Let $\mathsf{id}_\ell \subseteq \mathcal{C} \times 2^{\mathcal{C}}$ contain precisely all pairs $(c, \{c\})$ and $\mathsf{id}_r \subseteq 2^{\mathcal{C}} \times \mathcal{C}$ contain precisely all pairs $(\{c\}, c)$ for $c \in \mathcal{C}$. We say that a semantics $[\![\,]\!]_{\mathcal{R}}$ is given by $\mathcal{R}$ if both relations are total, relation $\mathcal{R}_{\text{val}}$ equals id_ℓ when restricted to $\mathcal{C}$, relation $\mathcal{R}_{\text{sem}}$ contains id_r, and $D' \in [\![D]\!]_{\mathcal{R}}$ iff $D(\mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}})D'$. Previously we just used identity instead of id_ℓ and id_r.

We say that $\mathcal{R}_{\text{sem}}$ is transitive if $\mathcal{R}_{\text{sem}} \circ \mathsf{id}_\ell \circ \mathcal{R}_{\text{sem}} \subseteq \mathcal{R}_{\text{sem}}$. Note that $\mathcal{R}_\cup$ is transitive. Now we have an analog of Proposition 2.

Proposition 8. *A pair $\mathcal{R} = (\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ gives rise to an admissible database domain if $\mathcal{R}_{\text{sem}}$ is transitive.*

Preservation for powerset semantics

We now apply general-framework results to easily derive a condition for naïve evaluation to work under $(\!|\,|\!)_{\text{CWA}}$ and related semantics. We look at complete databases of vocabulary σ, and a relation $\mathcal{R}_{\text{sem}}$ between a set of such databases and one such database. An $\mathcal{R}_{\text{sem}}$-*homomorphism* between D and D' is a set $\{h_1, \ldots, h_n\}$ of mappings defined on $\text{adom}(D)$ so that $\{h_1(D), \ldots, h_n(D)\}\mathcal{R}_{\text{sem}}D'$ (note that if $n = 1$, this is exactly the notion of $\mathcal{R}_{\text{sem}}$-homomorphisms seen earlier). It is very easy to show that any semantics based on $\mathcal{R}^{\text{all}}_{\text{val}} = \{(D, \{v_1(D), \ldots, v_n(D)\}) \mid v_i\text{'s are valuations}\}$ has the set of all databases as a representative set. Thus:

Proposition 9. *For every powerset semantics based on $\mathcal{R}^{\text{all}}_{\text{val}}$, naïve evaluation works for a generic Boolean query Q iff Q is preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms.*

When the relation $\mathcal{R}_{\text{sem}}$ is $\mathcal{R}_\cup$, the notion of preservation under $\mathcal{R}_{\text{sem}}$-homomorphisms becomes preservation *under union of strong onto homomorphisms*: if Q is true in D, and $h_1, \ldots, h_n$ are homomorphisms defined on D, then Q is true in $h_1(D) \cup \ldots \cup h_n(D)$.

For previous preservation results among FO queries, we looked at classes Pos and $\exists\mathsf{Pos}$ of positive and existential positive queries, and the class $\mathsf{Pos} + \forall\mathsf{G}$ of positive queries with universal guards. Now let $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$ be the class of existential positive queries extended with *Boolean universal guards*, i.e., universally guarded formulae which are sentences. More precisely, if $\bar{x}$ is a tuple of distinct variables, $\varphi(\bar{y})$ is a formula in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$, where all $\bar{y}$ variables are contained in $\bar{x}$, and R is a relation symbol, then $\forall\bar{x}\ (R(\bar{x}) \rightarrow \varphi(\bar{y}))$ is in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$.

Lemma 1. *Sentences in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$ are preserved under unions of strong onto homomorphisms.*

Combining, we get the following result.

Corollary 3. *If Q is a Boolean query from the class $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$, then naïve evaluation works for Q under the $(\!|\,|\!)_{\text{CWA}}$ semantics.*

Semantics similar to $(\!|\,|\!)_{\text{CWA}}$ did appear in the literature. In fact, the closest comes from the study of CWA in the context of data exchange [5]. It was presented in [18] (and based in turn on a semantics from [25]), and essentially boils down to the $(\!|\,|\!)_{\text{CWA}}$ semantics, but based on a restricted notion of valuations, namely minimal valuations. We study those in the next section.

8. Minimal valuations and their semantics

So far all the semantics that we saw allowed arbitrary valuations to be applied to instances with nulls. These are not the only possible semantics. In fact [18], based on earlier work in the area of logic programming [25], proposed a semantics that is based on *minimal* valuations. We now introduce it in our context (as [18] defined it in the context of data exchange).

For now we deal with database homomorphisms, i.e., $h(c) = c$ for each $c \in \mathsf{Const}$. We say that a homomorphism h defined on an instance D is D-*minimal* if no proper subinstance of $h(D)$ is a homomorphic image of D; equivalently, there is no other homomorphism h' so that $h'(D) \subsetneq h(D)$. If h is a valuation, then we talk about a D-minimal valuation.

The semantics of [18], denoted by $(\!|D|\!)^{\text{min}}_{\text{CWA}}$, is defined as

$$\left\{ D' \;\middle|\; \begin{array}{l} D' = \bigcup\{h(D) \mid h \in \mathcal{H}\}, \\ \mathcal{H} \text{ is a nonempty set of } D\text{-minimal valuations.} \end{array} \right\}$$

This is a powerset-based semantics given by the pair $\mathcal{R} = (\mathcal{R}^{\text{min}}_{\text{val}}, \mathcal{R}_\cup)$ where $\mathcal{R}^{\text{min}}_{\text{val}}$ has all the pairs $(D, \{h(D) \mid h \in \mathcal{H}\})$, with $\mathcal{H}$ ranging over nonempty sets of D-minimal valuations.

One can define a non-powerset analog of such a semantics with valuation relation $\mathcal{R}^{\text{min}}_{\text{val}} = \{(D, h(D)) \mid h \text{ is a } D\text{-minimal valuation}\}$. For the identity relation, playing the role of $\mathcal{R}_{\text{sem}}$ for CWA, this gives us

$$[\![D]\!]^{\text{min}}_{\text{CWA}} = \{h(D) \mid h \text{ is a } D\text{-minimal valuation}\}.$$

Combining $\mathcal{R}^{\text{min}}_{\text{val}}$ with the subset relation (playing the role of $\mathcal{R}_{\text{sem}}$ for OWA) gives us the usual OWA semantics.

Thus, we study the $[\![\,]\!]^{\text{min}}_{\text{CWA}}$ and $(\!|\,|\!)^{\text{min}}_{\text{CWA}}$ semantics. We start by looking at the connection between minimal homomorphisms and the closely related notion of cores.

Minimal homomorphisms and cores Recall that a *core* of a structure D (in our case, a relational database of vocabulary σ) is a substructure $D' \subseteq D$ such that D' is a homomorphic image of D but no proper subinstance of D' is. In other words, there is a homomorphism $h : D \rightarrow D'$ but there is no homomorphism $g : D \rightarrow D''$ for $D'' \subsetneq D'$. It is known that a core is unique up

to isomorphism, so we can talk of *the* core of D, and denote it by $\text{core}(D)$. A structure is called a core if $D = \text{core}(D)$. The cores are commonly used over graphs [17] but here we shall use them with the database notion of homomorphism that preserves constants (in which case all results about cores remain true [14]).

Even if minimal homomorphisms are related to cores, their images cannot be described precisely in terms of cores. We now strengthen results given in several examples in [18] (where they used constants in an essential way).

Proposition 10. *If h is D-minimal, then $h(D)$ is a core and $h(D) = h(\text{core}(D))$. However, there exists a core D and a homomorphism h defined on it so that $h(D)$ is a core, but h is not D-minimal. In fact this is true if both D and $h(D)$ contain only nulls, and if D is a graph.*

This result also shows that $[\![D]\!]^{\min}_{\text{CWA}}$ need not be the same as $[\![\text{core}(D)]\!]_{\text{CWA}}$. Nevertheless, cores do play a crucial role in the study of minimal semantics.

Theorem 5. *For the semantics $[\![\,]\!]^{\min}_{\text{CWA}}$ and $(\!|\,|\!)^{\min}_{\text{CWA}}$, the set of cores is a representative set.*
This remains true for every semantics given by pairs $\mathcal{R} = (\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$ or $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$.

Immediately from here we obtain:

Corollary 4. *Let Q be a generic Boolean relational query. Then naïve evaluation works for Q under the $[\![\,]\!]^{\min}_{\text{CWA}}$ or the $(\!|\,|\!)^{\min}_{\text{CWA}}$ semantics iff Q is weakly monotone (under the corresponding semantics), and $Q(D) = Q(\text{core}(D))$ for every D.*

Hence, the crucial new condition for minimal semantics is that Q cannot distinguish a database from its core.

Preservation and naïve evaluation We now relate weak monotonicity to homomorphism preservation. For this, we consider minimality for instances D over Const. For such an instance, and a homomorphism h, we let $\text{fix}(h, D) = \{c \in \mathsf{Const}(D) \mid h(c) = c\}$. Then h is called D-minimal if there is no homomorphism g with $\text{fix}(h, D) \subseteq \text{fix}(g, D)$ and $g(D) \subsetneq h(D)$. Note that database homomorphisms fix precisely the set of constants in D, so the first condition was not necessary.

Given a Boolean query Q, we say that it is *preserved under minimal homomorphisms* if, whenever D is a database over Const and h is a D-minimal homomorphism, then $Q(D) = 1$ implies $Q(h(D)) = 1$. Likewise, Q is preserved under unions of minimal homomorphisms, if for any nonempty set $\mathcal{H}$ of D-minimal homomorphisms such that $\text{fix}(h, D) = \text{fix}(g, D)$ whenever $f, g \in \mathcal{H}$, we have that $Q(D) = 1$ implies $Q(\bigcup\{h(D) \mid h \in \mathcal{H}\}) = 1$.

Proposition 11. *Let Q be a Boolean generic query. Then it is weakly monotone under $[\![\,]\!]^{\min}_{\text{CWA}}$ (respectively, under $(\!|\,|\!)^{\min}_{\text{CWA}}$) iff it is preserved under minimal homomorphisms (respectively, their unions).*

Combining this with Corollary 4 and results in Section 5, we obtain:

Corollary 5. *Let Q be a Boolean FO query such that $Q(D) = Q(\text{core}(D))$ for all D.*

- *If Q is in $\mathsf{Pos} + \forall\mathsf{G}$, then naïve evaluation works for Q under the $[\![\,]\!]^{\min}_{\text{CWA}}$ semantics.*
- *If Q is in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$, then naïve evaluation works for Q under the $(\!|\,|\!)^{\min}_{\text{CWA}}$ semantics.*

The precondition $Q(D) = Q(\text{core}(D))$ is essential for the result to work. To see this, consider an incomplete instance $D = \{(\bot, \bot), (\bot, \bot')\}$. Every D-minimal valuation h must satisfy $h(\bot) = h(\bot')$, i.e., their images are precisely the instances $\{(c, c)\}$ for $c \in \mathsf{Const}$. Hence, under $[\![\,]\!]^{\min}_{\text{CWA}}$, the certain answer to $\forall x\, D(x, x)$ is true, while evaluating this formula on D produces false. The reason naïve evaluation does not return certain answers is that $Q(D) \neq Q(\text{core}(D))$, since $\text{core}(D) = \{(\bot, \bot)\}$.

Thus, the extra condition is essential, but it is not fully satisfactory, as we do not know how to check for this condition in relevant FO fragments. We present two ways to deal with this issue.

First, a corollary of our results is that if we only need to compute queries on cores, then the condition is not necessary. More precisely, we say that naïve evaluation works for Q over a class $\mathcal{K}$ of instances, under a given semantics, if for each $D \in \mathcal{K}$, certain answer to Q over D is the same as $Q(D)$. Then

Corollary 6. *Let Q be a Boolean FO query.*

- *If Q is in $\mathsf{Pos} + \forall\mathsf{G}$, then naïve evaluation works for Q over cores under the $[\![\,]\!]^{\min}_{\text{CWA}}$ semantics.*
- *If Q is in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$, then naïve evaluation works for Q over cores under the $(\!|\,|\!)^{\min}_{\text{CWA}}$ semantics.*

A second corollary states that for the above classes of queries, even without the extra condition we can conclude that if naïve evaluation returns true, then so will the certain answer. In other words, we can run Q naïvely on D, not on $\text{core}(D)$. If the result is true, then the certain answer is true; but if the result is false, we cannot conclude anything. That is, naïve evaluation provides an *approximation* of certain answers.

Corollary 7. *Let Q be a Boolean FO query. If Q is in $\mathsf{Pos} + \forall\mathsf{G}$ (or in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$), and $Q(D) = 1$, then the certain answer to Q over D under the $[\![\,]\!]^{\min}_{\text{CWA}}$ (respectively $(\!|\,|\!)^{\min}_{\text{CWA}}$) semantics is true.*

The table in Figure 1 summarizes results on naïve evaluation for fragments of FO queries. The first line of

Semantics	symbol	Naïve evaluation works for
open world	$[\![\,]\!]_{\text{OWA}}$	$\exists\mathsf{Pos}$ = unions of CQs
weak closed-world	$[\![\,]\!]_{\text{WCWA}}$	Pos
closed world:	$[\![\,]\!]_{\text{CWA}}$	$\mathsf{Pos} + \forall\mathsf{G}$
powerset closed-world	$(\!|\,|\!)_{\text{CWA}}$	$\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$
minimal closed-world	$[\![\,]\!]^{\text{min}}_{\text{CWA}}$	$\mathsf{Pos} + \forall\mathsf{G}$, over cores; result always contained in certain answers
minimal, powerset closed-world	$(\!|\,|\!)^{\text{min}}_{\text{CWA}}$	$\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$, over cores; result always contained in certain answers

Figure 1: Summary of naïve evaluation results for FO queries

course is the classical result of [19], proved to be optimal in [24]. Other results were shown using the methodology established here, that reduced naïve evaluation to monotonicity and preservation under homomorphisms.

9. Lifting results to non-Boolean queries

So far our results dealt with Boolean queries. Now we show how to lift them to the setting of arbitrary k-ary relational queries. The basic idea is to consider database domains where objects are pairs consisting of a database and a k-tuple of constants. This turns queries into Boolean, and we apply our results. This requires more technical development than seems to be implied by the simple idea, but it can be carried out for all the semantics. We sketch now how the extension works.

A k-ary query Q maps a database D to a subset of $\text{adom}(D)^k$. It is *generic* if, for each one-to-one map f : $\text{adom}(D) \to \mathsf{Const} \cup \mathsf{Null}$, we have $Q(f(D)) = f(Q(D))$.

Given a semantics $[\![\,]\!]$, certain answers to Q are defined as $\mathsf{certain}(Q, D) = \bigcap\{Q(D') \mid D' \in [\![D]\!]\}$. Naïve evaluation *works* for Q if $\mathsf{certain}(Q, D)$ is precisely the set of tuples in $Q(D)$ that do not have nulls. We refer to this set (i.e., $Q(D) \cap \mathsf{Const}^k$) as $Q^{\mathsf{C}}(D)$.

As before, Q is monotone if $D \preceq D'$ implies $Q^{\mathsf{C}}(D) \subseteq Q^{\mathsf{C}}(D')$ for the semantic ordering $\preceq$, and Q is weakly monotone if the above is true whenever $D' \in [\![D]\!]$.

A representative set $\mathcal{S}$ is called *strong* if for each finite set $C \subset \mathsf{Const}$ and each instance D there is an isomorphic instance $D' \in [\![D]\!]$ such that both the isomorphism $D \to D'$ and its inverse are the identity on C. If we deal with semantics given by pairs $\mathcal{R} = (\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$, we say that a k-ary query is *weakly preserved* under a class of $\mathcal{R}_{\text{sem}}$-homomorphisms if for every database D, a k-tuple t of constants, and an $\mathcal{R}_{\text{sem}}$-homomorphism $h : D \to D'$ from the class that is the identity on t, the condition $t \in Q(D)$ implies $t \in Q(D')$. Note that for Boolean queries this is the same as preservation under $\mathcal{R}_{\text{sem}}$-homomorphisms.

Then the main connections continue to hold.

Theorem 6. *Let $\mathbb{D}$ be an admissible relational database domain, and Q a k-ary generic query. If $\mathbb{D}$ has a strong representative set, then the following are equivalent:*

1. *Naïve evaluation works for Q;*
2. *Q is monotone;*
3. *Q is weakly monotone and $Q^{\mathsf{C}}(x) = Q^{\mathsf{C}}(\chi_{\mathcal{S}}(x))$ for every $x \in \mathcal{D}$.*

Furthermore, for semantics given by $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, naïve evaluation works for Q iff Q is weakly preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms, and for semantics given by $(\mathcal{R}^{\text{min}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, naïve evaluation works for Q iff Q is weakly preserved under minimal $\mathcal{R}_{\text{sem}}$-homomorphisms and $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(\text{core}(D))$ for each D.

One can then check that for all the classes of FO formulae considered here, preservation results hold when extended to formulae with free variables. In addition, one can develop similar transfer technique for powerset semantics (this is done in the appendix) and conclude that all the results remain true for non-Boolean queries.

Theorem 7. *All the results from Theorem 3 and Corollaries 3 and 5 remain true for k-ary FO queries.*

10. Conclusions

We now present the main directions in which we would like to extend this work.

Other data models So far we looked at either a very general setting, which can subsume practically every data model, or at relational databases. We would like to extend our results to XML. At this time, we have a good understanding of the semantics of incomplete XML documents and the complexity of answering queries over them [3, 8, 15]. That will be the starting point of our investigation of classes of XML queries which admit naïve evaluation under various semantics.

Other languages When we dealt with relations, we studied FO as the main query language. However, our structural results are in no way limited to FO. In fact it is known that naïve evaluation works for datalog (without negation). Given the toolkit of this paper, we would like to consider queries in languages that go beyond FO and admit naïve evaluation.

Preservation results There are open questions related to preservation results in both finite and infinite model theory. We already mentioned that the results

of [21] about preservation under strong onto homomorphisms are limited to a simple vocabulary, and even then appear to be problematic. We would like to establish a precise characterization in the infinite case, and see whether it holds or fails in the finite. We also want to look at preservation on restricted classes of structures, following [7] which looked at bounded treewidth. We note in passing that [7] does not apply directly to the study of XML since models of documents with data generate relational structures of arbitrary treewidth.

The impact of constraints It is well known that constraints (e.g., keys and foreign keys) have a huge impact on the complexity of finding certain answers [10, 33]. It is thus natural to ask how they affect good classes we described in this paper. Constraints appear in another model of incompleteness – conditional tables [19] – that in general have higher complexity of query evaluation [2] but are nonetheless useful in several applications [6].

Applications In applications such as data integration and exchange, finding certain answers is the standard query answering semantics [5, 22]. In fact one of our semantics came from data exchange literature [18]. We would like to see whether our techniques help find classes of queries for which query answering becomes easy in exchange and integration scenarios.

Bringing back the infinite We have used a number of results from infinite model theory to get our syntactic classes. Another way of appealing to logic over infinite structures to handle incompleteness was advocated by Reiter [27, 29] three decades ago. In that approach, an incomplete database D is viewed as a logical theory T_D, and finding certain answers to Q amounts to checking whether T_D entails Q. This is in general an undecidable problem, and entailment in the finite is known to be more problematic than unrestricted one. This is reminiscent of the situation with homomorphism preservation results, but we saw that we can use infinite results to obtain useful sufficient conditions. Motivated by this, we would like to revisit Reiter's proof-theoretic approach and connect it with our semantic approach.

Acknowledgments Work partly supported by EPSRC grants G049165 and J015377.

11. References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] S. Abiteboul, P Kanellakis, and G. Grahne. On the representation and querying of sets of possible worlds. *TCS*, 78(1):158–187, 1991.

[3] S. Abiteboul, L. Segoufin, and V. Vianu. Representing and querying XML with incomplete information. *ACM TODS*, 31(1):208–254, 2006.

[4] M. Ajtai, Y. Gurevich. Monotone versus positive. *J. ACM* 34(4):1004-1015 (1987).

[5] M. Arenas, P. Barceló, L. Libkin, F. Murlak. *Relational and XML Data Exchange*. Morgan & Claypool, 2010.

[6] M. Arenas, J. Pérez, J. Reutter. Data exchange beyond complete data. In *PODS'11*, pages 83–94.

[7] A. Atserias, A. Dawar, P. Kolaitis. On preservation under homomorphisms and unions of conjunctive queries. *J. ACM* 53(2):208-237 (2006).

[8] P. Barceló, L. Libkin, A. Poggi, and C. Sirangelo. XML with incomplete information. *J. ACM* 58(1): 1–62 (2010).

[9] P. Buneman, A. Jung, A. Ohori. Using powerdomains to generalize relational databases. *TCS* 91 (1991), 23–55.

[10] A. Calì, D. Lembo, and R. Rosati. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *PODS'03*, pages 260–271.

[11] C.C. Chang and H.J. Keisler. *Model Theory*. North Holland, 1990.

[12] K. Compton. Some useful preservation theorems. *J. Symb. Logic* 48 (1983), 427–440.

[13] C. Date and H. Darwin. *A Guide to the SQL Standard*. Addison-Wesley, 1996.

[14] R. Fagin, P. Kolaitis, L. Popa. Data exchange: getting to the core. *ACM TODS* 30(1):174-210 (2005).

[15] A. Gheerbrant, L. Libkin, T. Tan. On the complexity of query answering over incomplete XML documents. *ICDT'12*, pages 169-181.

[16] C. Gunter. *Semantics of Programming Languages*. The MIT Press, 1992.

[17] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[18] A. Hernich. Answering non-monotonic queries in relational data exchange. *LMCS* 7(3) (2011).

[19] T. Imieliński and W. Lipski. Incomplete information in relational databases. *J. ACM*, 31(4):761–791, 1984.

[20] H. J. Keisler. Finite approximations of infinitely long formulas. *Symp. Theory of Models*, 1965, pages 158–169.

[21] H. J. Keisler. Some applications of infinitely long formulas. *Journal of Symbolic Logic*, 1965, pages 339–349.

[22] M. Lenzerini. Data integration: a theoretical perspective. In *PODS'02*, pages 233–246.

[23] L. Libkin. A semantics-based approach to design of query languages for partial information. In *Semantics in Databases*, LNCS 1358, 1998, pages 170–208.

[24] L. Libkin. Incomplete information and certain answers in general data models. *PODS'11*, pages 59–70.

[25] J. Minker. On indefinite databases and the closed world assumption. In *CADE'82*, pages 292–308.

[26] A. Ohori. Semantics of types for database objects. *Theoretical Computer Science* 76 (1990), 53–91.

[27] R. Reiter. On closed world data bases. In *Logic and Data Bases*, 1977, pages 55–76.

[28] R. Reiter. Equality and domain closure in first-order databases. *J. ACM* 27(2): 235-249 (1980).

[29] R. Reiter. Towards a logical reconstruction of relational database theory. In *On Conceptual Modelling*, 1982, pages 191–233.

[30] B. Rossman. Homomorphism preservation theorems. *J. ACM* 55(3): (2008).

[31] B. Rounds. Situation-theoretic aspects of databases. In *Situation Theory and Appl.*, CSLI vol. 26, 1991, pages 229-256.

[32] A. Stolboushkin. Finitely monotone properties. In *LICS'95*, pages 324–330.

[33] M. Vardi. On the integrity of databases with incomplete information. In *PODS'86*, pages 252–266.

APPENDIX

Proofs

We shall use the following notations throughout the appendix.

If $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ is a database domain, $Q : \mathcal{D} \to \{0, 1\}$ is a query, and $\mathcal{R} \subseteq \mathcal{D} \times \mathcal{D}$, we say that Q is *preserved under* $\mathcal{R}$ if $Q(x) = 1$ implies $Q(y) = 1$ whenever $(x, y) \in \mathcal{R}$. Notice that monotonicity corresponds to preservation under the semantic ordering $\preceq$, and weak monotonicity corresponds to preservation under the graph of $[\![\ ]\!]$. If $\mathcal{D}'$ is a subset of $\mathcal{D}$, with $\mathcal{C} \subseteq \mathcal{D}'$, we say that Q is monotone (weakly monotone, preserved under $\mathcal{R}$, respectively) *over* $\mathcal{D}'$ if Q is monotone (weakly monotone, preserved under $\mathcal{R}$, respectively) when restricted to $\mathcal{D}'$.

We say that $\mathbb{D}$ is a *relational database domain* if $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$, where $\mathcal{D}$ is the set of (possibly incomplete) relational instances, $\mathcal{C}$ is the set of complete relational instances and $\approx$ is the isomorphism relation between instances (i.e. $D \approx D'$ iff there exists an injective mapping π on $\mathrm{adom}(D)$ such that $\pi(D) = D'$).

If D is a relational instance, h is a mapping $h : \mathrm{adom}(D) \to \mathsf{Const} \cup \mathsf{Null}$ and $K \subseteq \mathsf{Const}$, we say that h is the *identity on* K if $h(c) = c$ for all $c \in \mathrm{adom}(D) \cap K$.

Proofs for Section 3

Proof of Proposition 1

Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be a database domain and let $\preceq$ be the preorder obtained from it. Recall that by definition $x \preceq y$ iff $[\![y]\!] \subseteq [\![x]\!]$ and so $[\![x]\!]_{\preceq} = \{c \in \mathcal{C} \mid x \preceq c\}$, i.e., $[\![x]\!]_{\preceq} = \{c \in \mathcal{C} \mid [\![c]\!] \subseteq [\![x]\!]\}$. Now we want to show that $\mathbb{D}$ is admissible, i.e., for all x, $[\![x]\!] = [\![x]\!]_{\preceq}$.

So let $x \in \mathcal{D}$, $c \in \mathcal{C}$ such that $c \in [\![x]\!]$. By condition 2, $[\![c]\!] \subseteq [\![x]\!]$. But then $c \in \{c \in \mathcal{C} \mid [\![c]\!] \preceq [\![x]\!]\}$, i.e., $c \in [\![x]\!]_{\preceq}$ and so for all x, $[\![x]\!] \subseteq [\![x]\!]_{\preceq}$.

Now let $x \in \mathcal{D}$, $c \in \mathcal{C}$ such that $c \in [\![x]\!]_{\preceq}$. So $[\![c]\!] \preceq [\![x]\!]$. By condition 1, $c \in [\![c]\!]$, which yields $c \in [\![x]\!]$ and so for all x, $[\![x]\!]_{\preceq} \subseteq [\![x]\!]$.

Proof of Theorem 1

Theorem 1 follows immediately from the following two Lemmas

Lemma 2. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be a database domain, and Q a generic Boolean query. Assume that $\mathbb{D}$ has a representative set $\mathcal{S}$, and let $\mathcal{D}'$ be a set $\mathcal{S} \subseteq \mathcal{D}' \subseteq \mathcal{D}$. Then naïve evaluation works for Q over $\mathcal{D}'$ iff Q is weakly monotone over $\mathcal{D}'$ and $Q(x) = Q(\chi_{\mathcal{S}}(x))$ for every $x \in \mathcal{D}'$.*
In particular if $\mathcal{S} = \mathcal{D}$ then naïve evaluation works for Q iff Q is weakly monotone.

Proof. Let Q be a Boolean generic query. Assume that naïve evaluation works for Q over $\mathcal{D}'$; then weak monotonicity of Q over $\mathcal{D}'$ immediately follows.

For all $x \in \mathcal{D}'$, we have $[\![x]\!] = [\![\chi_{\mathcal{S}}(x)]\!]$; moreover naïve evaluation works for Q on both x and $\chi_{\mathcal{S}}(x)$ (because $\mathcal{D}' \supseteq \mathcal{S}$). Then we have $Q(x) = \mathsf{certain}(Q, x) = \mathsf{certain}(Q, \chi_{\mathcal{S}}(x)) = Q(\chi_{\mathcal{S}}(x))$.

Conversely assume that Q is weakly monotone over $\mathcal{D}'$ and $Q(x) = Q(\chi_{\mathcal{S}}(x))$ for all $x \in \mathcal{D}'$. Let $x \in \mathcal{D}'$. By weak monotonicity over $\mathcal{D}'$ (and because $\mathcal{D}' \supseteq \mathcal{S} \supseteq \mathcal{C}$) we have $Q(x) \leqslant \mathsf{certain}(Q, x)$. To prove the converse, assume $\mathsf{certain}(Q, x) = 1$. Recall that $[\![x]\!] = [\![\chi_{\mathcal{S}}(x)]\!]$ and $\chi_{\mathcal{S}}(x) \in \mathcal{S}$. Therefore there exists $c \in [\![x]\!]$ such that $c \approx \chi_{\mathcal{S}}(x)$. We know $Q(c) = 1$; then by genericity $Q(\chi_{\mathcal{S}}(x)) = 1 = Q(x)$. Hence $\mathsf{certain}(Q, x) = Q(x)$ for all $x \in \mathcal{D}'$.

We have thus proved that naïve evaluation works for Q over $\mathcal{D}'$ if and only if Q is weakly monotone over $\mathcal{D}'$ and $Q(x) = Q(\chi_{\mathcal{S}}(x))$ for all $x \in \mathcal{D}'$. Now if in particular $\mathcal{S} = \mathcal{D}$ we can always assume $\chi_{\mathcal{S}}$ to be the identity mapping $\mathcal{D} \to \mathcal{D}$. In this case then naïve evaluation works for Q if and only if Q is weakly monotone. □

Lemma 3. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be an admissible database domain, and Q a generic query. Assume that $\mathbb{D}$ has a representative set $\mathcal{S}$, and let $\mathcal{D}'$ be a set $\mathcal{S} \subseteq \mathcal{D}' \subseteq \mathcal{D}$. Then naïve evaluation works for Q over $\mathcal{D}'$ iff Q is monotone over $\mathcal{D}'$.*

PROOF. Assume that naïve evaluation works Q over $\mathcal{D}'$, and consider objects $x, y \in \mathcal{D}'$ such that $x \preceq y$ and $Q(x) = 1$. We prove $Q(y) = 1$. We have $Q(x) = \mathsf{certain}(Q, x) = 1$ and $[\![y]\!] \subseteq [\![x]\!]$, therefore $\mathsf{certain}(Q, y) = Q(y) = 1$.

Conversely assume that Q is monotone over $\mathcal{D}'$. Let x be in $\mathcal{D}'$, we prove that $Q(x) = \mathsf{certain}(Q, x)$. Let $c \in [\![x]\!]$. Recall that $\mathcal{D}' \supseteq \mathcal{S} \supseteq \mathcal{C}$ therefore $c \in \mathcal{D}'$. Since the database domain is admissible, $x \preceq c$. Then the monotonicity of Q implies $Q(x) \leqslant Q(c)$, and therefore $Q(x) \leqslant \mathsf{certain}(Q, x)$.

For the converse implication assume $\mathsf{certain}(Q, x) = 1$. Recall that $[\![x]\!] = [\![\chi_{\mathcal{S}}(x)]\!]$, therefore $\mathsf{certain}(Q, \chi_{\mathcal{S}}(x)) = 1$. But $\chi_{\mathcal{S}}(x) \in \mathcal{S}$, then there exists $c' \in [\![\chi_{\mathcal{S}}(x)]\!]$ such that $c' \approx \chi_{\mathcal{S}}(x)$. We know $Q(c') = 1$, then by genericity, $Q(\chi_{\mathcal{S}}(x)) = 1$. Now since $\chi_{\mathcal{S}}$ preserves the semantics, $\chi_{\mathcal{S}}(x) \preceq x$. Observe also that $\chi_{\mathcal{S}}(x) \in \mathcal{D}'$ (because $\mathcal{D}' \supseteq \mathcal{S}$), then we can use monotonicity of Q over $\mathcal{D}'$, and conclude that $Q(x) = 1$. This shows $Q(x) = \mathsf{certain}(Q, x)$ – i. e. naïve evaluation works for Q over $\mathcal{D}'$ – and concludes the proof of the lemma. □

Proofs for Section 4

Proof of Proposition 2

Assume first that $\mathcal{R}_{\text{sem}}$ is transitive, and take arbitrary $x \in \mathcal{D}$ and $c \in \mathcal{C}$. We have

1. $c \in [\![c]\!]$.
   Indeed we know $(c, c) \in \mathcal{R}_{\text{val}}$ and $(c, c) \in \mathcal{R}_{\text{sem}}$, then $c \in [\![c]\!]$.
2. $c \in [\![x]\!]$ implies $[\![c]\!] \subseteq [\![x]\!]$.
   Indeed if $c \in [\![x]\!]$ then there exists $y \in \mathcal{C}$ such that $(x, y) \in \mathcal{R}_{\text{val}}$ and $(y, c) \in \mathcal{R}_{\text{sem}}$. Moreover if $c' \in [\![c]\!]$ then $(c, c') \in \mathcal{R}_{\text{sem}}$ (because $\mathcal{R}_{\text{val}}$ is the identity when restricted to $\mathcal{C}$). By transitivity of $\mathcal{R}_{\text{sem}}$ we then have $(y, c') \in \mathcal{R}_{\text{sem}}$. This implies $(x, c') \in \mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$, and therefore $c' \in [\![x]\!]$.

By Proposition 1, $\mathbb{D}$ is admissible.

Conversely assume that $\mathbb{D}$ is admissible, and assume there exist (c, d) and (d, e) in $\mathcal{R}_{\text{sem}}$. Now recall that (c, c) and (d, d) are in $\mathcal{R}_{\text{val}}$, thus (c, d) and (d, e) are in $\mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$, i.e. $d \in [\![c]\!]$ and $e \in [\![d]\!]$. By item 2. of Proposition 1 $e \in [\![c]\!]$. Then $(c, e) \in \mathcal{R}_{\text{sem}}$. This proves that $\mathcal{R}_{\text{sem}}$ is transitive.

Strong representative sets

We define a stronger notion of representative set over relational database domains. If $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ is a relational database domain, $\mathcal{S} \subseteq \mathcal{D}$ is a *strong representative set* if the following three conditions are satisfied:

1. $\mathcal{C} \subseteq \mathcal{S}$
2. for each $D \in \mathcal{S}$, and for each finite set of constants $K \subseteq \mathsf{Const}$ there exists $D' \in [\![D]\!]$ and a bijection $i : \text{adom}(D) \to \text{adom}(D')$ such that both i and i^- are the identity on K and $i(D) = D'$.
3. There exists a function $\chi_{\mathcal{S}} : \mathcal{D} \to \mathcal{S}$ such that $[\![D]\!] = [\![\chi_{\mathcal{S}}(D)]\!]$ for every $D \in \mathcal{D}$.

Lemma 4. *If a relational semantics is given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, the set of all relational instances is a strong representative set.*

PROOF. Clearly the set of all relational instances contains the complete ones. Moreover given a finite set of constants K and a (possibly incomplete) relational instance D, let v be an injective valuation $v : \text{adom}(D) \to \mathsf{Const} \backslash K$. It is easy to check that v is a bijection $\text{adom}(D) \to \text{adom}(v(D))$ such that both v and v^- are the identity on K. Clearly $(D, v(D)) \in \mathcal{R}^{\text{all}}_{\text{val}}$. Moreover since $\mathcal{R}_{\text{sem}}$ is reflexive, $(v(D), v(D)) \in \mathcal{R}_{\text{sem}}$. As a consequence $v(D) \in [\![D]\!]$.

The function $\chi_{\mathcal{D}}$ is the identity. □

Proof of Proposition 3

Proposition 3 immediately follows from Lemma 4 and Lemma 2, since any strong representative set is also a representative set.

Weak monotonicity and $\mathcal{R}_{\text{sem}}$-homomorphisms

If $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ is a database domain, $\mathcal{R}$ and $\mathcal{R}'$ are subsets of $\mathcal{D} \times \mathcal{C}$, we say that $\mathcal{R}'$ is $\approx$-*equivalent* to $\mathcal{R}$ if the following two conditions are satisfied:

1. if $(x, c) \in \mathcal{R}$ then there exists $x' \in \mathcal{D}$ such that $x' \approx x$ and $(x', c) \in \mathcal{R}'$;
2. if $(x, c) \in \mathcal{R}'$ then there exists $x' \in \mathcal{D}$ such that $x' \approx x$ and $(x', c) \in \mathcal{R}$.

We say that $\mathcal{R}'$ is *strongly* $\approx$-*equivalent* to $\mathcal{R}$ if moreover x' in the definition of $\approx$-equivalence only depends on x (an not on c).

Lemma 5. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be an arbitrary database domain and let $\mathcal{R}' \subseteq \mathcal{D} \times \mathcal{C}$ be $\approx$-equivalent to the graph of $[\![\,]\!]$. Then a generic Boolean query over $\mathbb{D}$ is weakly monotone iff it is preserved under $\mathcal{R}'$.*

Proof. Assume that Q is a generic Boolean query over $\mathbb{D}$, and Q is weakly monotone. Consider a pair $(x, c) \in \mathcal{R}'$ and assume that $Q(x) = 1$. By the fact that $\mathcal{R}'$ is $\approx$-equivalent to the graph of $[\![\,]\!]$, there exists $y \in \mathcal{D}$, such that $y \approx x$ and $c \in [\![y]\!]$. Since Q is generic $Q(y) = 1$, and since Q is weakly monotone $Q(c) = 1$. This proves that Q is preserved under $\mathcal{R}'$. The converse is proved symmetrically. □

When the semantics is given by a pair $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$, we have:

Lemma 6. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be a database domain whose semantics $[\![\,]\!]$ is given by a pair $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ and let $\mathcal{R}' \subseteq \mathcal{D} \times \mathcal{C}$ be $\approx$-equivalent to $\mathcal{R}_{\text{val}}$, then $\mathcal{R}' \circ \mathcal{R}_{\text{sem}}$ is $\approx$-equivalent to the graph of $[\![\,]\!]$ (i.e. to $\mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$). In particular a generic Boolean query over $\mathbb{D}$ is weakly monotone iff it preserved under $\mathcal{R}' \circ \mathcal{R}_{\text{sem}}$*

Proof. Assume that $(x, c) \in \mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$. Then there exists $e \in \mathcal{C}$ such that $(x, e) \in \mathcal{R}_{\text{val}}$ and $(e, c) \in \mathcal{R}_{\text{sem}}$. We know that there exists $x' \in \mathcal{D}$ such that $x' \approx x$ and $(x', e) \in \mathcal{R}'$. Then $(x', c) \in \mathcal{R}' \circ \mathcal{R}_{\text{sem}}$. Symmetrically we prove that for all $(x', c) \in \mathcal{R}' \circ \mathcal{R}_{\text{sem}}$ there exists $x \in \mathcal{D}$ such that $x' \approx x$ and such that $(x, c) \in \mathcal{R}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$. We conclude by Lemma 5. □

If $\mathcal{M}$ is a function associating to each complete relational instance D a class of mappings $\text{adom}(D) \to \mathsf{Const}$, we say that $\mathcal{M}$ is a *mapping type*.
If $\mathcal{M}$ is a mapping type, we denote by $\mathcal{R}_{\mathcal{M}}$ the set of pairs $\{(D, h(D)) \mid D$ is a complete relational instance and $h \in \mathcal{M}(D)\}$. Given two complete relational instances D and D', an $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-*homomorphism* from D to D' is an $\mathcal{R}_{\text{sem}}$-homomorphism h from D to D' such that $h \in \mathcal{M}(D)$.

The following claim follows directly from definitions:

Claim 1. *If $\mathcal{M}$ is a mapping type then $(D, D') \in \mathcal{R}_{\mathcal{M}} \circ \mathcal{R}_{\text{sem}}$ iff there exists an $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphism from D to D'*

By combining the above claim with Lemma 6 we have:

Corollary 8. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be a relational database domain whose semantics $[\![\,]\!]$ is given by a pair $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ and let $\mathcal{M}$ be a mapping type. Assume that $\mathcal{R}_{\mathcal{M}}$ is $\approx$-equivalent to $\mathcal{R}_{\text{val}}$. Then a generic Boolean query is weakly monotone iff it preserved under $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms.*

Proof of Proposition 4

We consider the mapping type $\mathcal{M} = \text{all}$ which associates to each complete relational instance D the set of all mappings $\text{adom}(D) \to \mathsf{Const}$, and we prove the following lemma:

Lemma 7. *If $\mathcal{M} = \text{all}$ and $\approx$ is relational isomorphism, then $\mathcal{R}_{\mathcal{M}}$ is strongly $\approx$-equivalent to $\mathcal{R}^{\text{all}}_{\text{val}}$.*

Proof. Let D be a (possibly incomplete) relational instance. We prove that there exists a complete relational instance E such that 1) $D \approx E$ and 2) $(D, D') \in \mathcal{R}^{\text{all}}_{\text{val}}$ implies $(E, D') \in \mathcal{R}_{\mathcal{M}}$.

The instance E is obtained from D by replacing nulls of D with new distinct constants not occurring in $\mathsf{Const}(D)$. Clearly there exists an isomorphism $i : E \to D$, thus $E \approx D$. Now let $(D, D') \in \mathcal{R}^{\text{all}}_{\text{val}}$. Then $D' = v(D)$ for some valuation v. Let $h = v \circ i$; then $h(E) = v(D) = D'$ and hence $(E, D') \in \mathcal{R}_{\mathcal{M}}$ (because $\mathcal{M} = \text{all}$). This prove 1) and 2) above.

Conversely let E be a complete relational instance. We prove that there exists a relational instance D such that 1) $D \approx E$ and 2) $(E, D') \in \mathcal{R}_{\mathcal{M}}$ implies $(D, D') \in \mathcal{R}^{\text{all}}_{\text{val}}$.

The instance D is obtained from E by replacing each element of $\text{adom}(E)$ with a new distinct null. Clearly this replacement defines an isomorphism $i : D \to E$ and therefore $E \approx D$. Now let $(E, D') \in \mathcal{R}_{\mathcal{M}}$. We know that $D' = h(E)$ where h is an arbitrary mapping $\text{adom}(E) \to \mathsf{Const}$. Let $v = h \circ i$. Then v is a valuation on D (because $\text{adom}(D)$ contains no constants, and D' is complete) and hence $(D, D') \in \mathcal{R}^{\text{all}}_{\text{val}}$. □

Now remark that with $\mathcal{M} = \text{all}$, $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms coincide with $\mathcal{R}_{\text{sem}}$-homomorphisms. Then the proposition follows immediately from Corollary 8 with $\mathcal{M} = \text{all}$.

Proof of Proposition 5

Let $\mathcal{R}_{\text{sem}}$ belong to one of the following semantic relations

1. OWA: $\{(D, D') \mid D \text{ is a complete relational instance and } D \subseteq D'\}$;
2. CWA: $\{(D, D) \mid D \text{ is a complete relational instance}\}$;
3. WCWA: $\{(D, D') \mid D \text{ is a complete relational instance, } D \subseteq D' \text{ and } \text{adom}(D) = \text{adom}(D')\}$.

Let $[\![\,]\!]$ be the semantics given by the pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$ (this semantics is OWA, CWA and WCWA, respectively), and let $\preceq_{[\![\,]\!]}$ be the ordering arising from $[\![\,]\!]$.

From Lemma 4 we know that, for any finite set of constants K, there exists $D' \in [\![D]\!]$ and a bijection $i : \text{adom}(D') \to \text{adom}(D)$ which is the identity on K and such that $i(D') = D$.

Assume D and D' are two relational instances and $D \preceq_{[\![\,]\!]} D'$. Let $E \in [\![D']\!]$ be an instance having a bijection $i : \text{adom}(E) \to \text{adom}(D')$ which is the identity on $\mathsf{Const}(D)$ and such that $i(E) = D'$. We know $E \in [\![D]\!]$ therefore $(E, D) \in \mathcal{R}^{\text{all}}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$, or in other words there exists a valuation $h : \text{adom}(D) \to \mathsf{Const}$ such that $(h(D), E) \in \mathcal{R}_{\text{sem}}$. Let $h' = i \circ h$, we prove that $h'(D)$ and D' are in the same relationship as $h(D)$ and E

1. Under OWA: $h(D) \subseteq E$ therefore $h'(D) = i(h(D)) \subseteq i(E) = D'$
2. Under CWA: $h(D) = E$ therefore $h'(D) = i(h(D)) = i(E) = D'$
3. Under WCWA: $h(D) \subseteq E$ and $\text{adom}(h(D)) = \text{adom}(E)$, therefore $h'(D) = i(h(D)) \subseteq i(E) = D'$ and $\text{adom}(h'(D)) = i(\text{adom}(h(D))) = i(\text{adom}(E)) = \text{adom}(D')$.

Moreover h' is the identity on $\mathsf{Const}(D)$, because both h and i are, and $h'(D)$ and D' are related according to $\mathcal{R}_{\text{sem}}$.

This implies that:

1. Under OWA h' is a database homomorphism $D \to D'$
2. Under CWA h' is a database strong onto homomorphism $D \to D'$
3. Under WCWA h' is a database onto homomorphism $D \to D'$.

Conversely assume that there exists a database $*$-homomorphism $D \to D'$ where $*$ is

1. "arbitrary", if $[\![\,]\!] = \text{OWA}$,
2. "strong onto" $[\![\,]\!] = \text{CWA}$,
3. "onto" if $[\![\,]\!] = \text{WCWA}$

Remark that the existence of a database $*$-homomorphism is a transitive relation, i.e. if there exists a database $*$-homomorphism from D to D' and a database $*$-homomorphism from D' to D'', then there exists a database $*$-homomorphism from D to D''.

Remark also that $[\![D']\!]$ is precisely the set of complete relational instance E such that there exists a database $*$-homomorphism from D' to E.

Then, by transitivity, there exists a database $*$-homomorphism from D to each $E \in [\![D']\!]$. Hence $E \in [\![D]\!]$ for all $E \in [\![D']\!]$. In other words, $[\![D']\!] \subseteq [\![D]\!]$, and therefore $D \preceq_{[\![\,]\!]} D'$.

Proofs for Section 5

We prove that sentences in given FO fragments are preserved under (strong onto) homomorphisms by structural induction. To this end we need first to define what it means for a formula with free variables to be preserved under (strong onto) homomorphisms.

If Q is a k-ary relational query over complete instances (i.e. a mapping associating to each complete relational instance D a k-ary relation over $\mathrm{adom}(D)$), we say that Q is preserved under (strong onto) homomorphisms if whenever h is a (strong onto) homomorphism from an instance D to an instance D', and $\bar{a} \in Q(D)$ then $h(\bar{a}) \in Q(D')$.

Proof of Proposition 6

Proposition 6 immediately follows from the lemma below:

Lemma 8. *Formulae in* $\mathsf{Pos} + \forall\mathsf{G}$ *are preserved under strong onto homomorphisms.*

PROOF. We proceed by structural induction on the formula φ.

If $\varphi = \mathit{false}$ or $\varphi = \mathit{true}$, it is clearly preserved under strong onto homomorphisms.

Assume now that $\varphi(\bar{x})$ is a positive atom $R(\bar{y})$ (including the case of an equality atom), where variables occurring in $\bar{y}$ are precisely $\bar{x}$. It follows from the definition of homomorphism that if an instance $D \models \varphi(\bar{a})$ then $h(D) \models \varphi(h(\bar{a}))$, for every homomorphism h.

It is also easy to verify that if φ_1 and φ_2 are preserved under strong onto homomorphisms, so are $\varphi_1 \wedge \varphi_2$ and $\varphi_1 \vee \varphi_2$.

Now assume $\varphi(\bar{x}) = \exists y \varphi'(y, \bar{x})$, where φ' is preserved under strong onto homomorphisms. Assume that an instance $D \models \varphi(\bar{a})$, and that h is a strong onto homomorphism from D to $D' = h(D)$. Then $D \models \varphi'(b, \bar{a})$ for some value $b \in \mathrm{adom}(D)$. Since φ' is preserved under strong onto homomorphisms, $D' \models \varphi'(h(b), h(\bar{a}))$. Thus $D' \models \exists y \varphi'(y, h(\bar{a}))$, i.e. $D' \models \varphi(h(\bar{a}))$.

Assume now that $\varphi(\bar{x}) = \forall y \varphi'(y, \bar{x})$. Assume that an instance $D \models \varphi(\bar{a})$ and D has a strong onto homomorphism h to D'. We prove $D' \models \varphi(h(\bar{a}))$. Let $b \in \mathrm{adom}(D')$, we have to prove $D' \models \varphi'(b, h(\bar{a}))$. Since $D' = h(D)$, there exists $a \in \mathrm{adom}(D)$ such that $h(a) = b$; moreover $D \models \varphi'(a, \bar{a})$. Now, by the induction hypothesis $\varphi'(y, \bar{x})$ is preserved under strong onto homomorphism, therefore $D' \models \varphi'(h(a), h(\bar{a})) = \varphi'(b, h(\bar{a}))$.

Now consider the case that $\varphi = \forall \bar{x}(R(\bar{x}) \to \varphi')$. Let $\bar{y}$ be the free variables of φ' not occurring in $\bar{x}$. So we write $\varphi' = \varphi'(\bar{x}', \bar{y})$ and $\varphi = \varphi(\bar{y})$, where variables in $\bar{x}'$ are from $\bar{x}$. Assume that an instance $D \models \varphi(\bar{a})$, and $D' = h(D)$, for some homomorphism h; we prove $D' \models \varphi(h(\bar{a}))$. Assume $D' \models R(\bar{b})$ for some tuple $\bar{b}$ over $\mathrm{adom}(D')$; we have to prove $D' \models \varphi'(\bar{b}', h(\bar{a}))$, where $\bar{b}'$ is the restriction of $\bar{b}$ to $\bar{x}'$. By the fact that h is a strong onto homomorphism from D to D', we know that there exists a tuple $\bar{c}$ such that $R(\bar{c})$ holds in D and $h(\bar{c}) = \bar{b}$. Thus, since variables $\bar{x}$ are pairwise distinct, $D \models R(\bar{x})$ in $\bar{x} = \bar{c}$, and therefore $D \models \varphi'(\bar{c}', \bar{a})$, where $\bar{c}'$ is the restriction of $\bar{c}$ to $\bar{x}'$.

We now use the induction hypothesis on φ' to conclude that $D' \models \varphi'(\bar{b}', h(\bar{a}))$. □

Proofs for Section 6

For an arbitrary relational semantics $[\![\,]\!]$, the semantics $[\![\,]\!]^{\mathrm{Codd}}$ for an instance D is defined as $[\![D]\!]^{\mathrm{Codd}} = [\![\chi_{\mathrm{Codd}}(D)]\!]$.

We first prove the following auxiliary lemma:

Lemma 9. *Let* $[\![\,]\!]$ *be a relational semantics given by a pair* $(\mathcal{R}^{\mathrm{all}}_{\mathrm{val}}, \mathcal{R}_{\mathrm{sem}})$*, and let* Q *be a Boolean relational query such that* $Q(D) = Q(\chi_{\mathrm{Codd}}(D))$ *for every relational instance* D*. Then* Q *is weakly monotone under* $[\![\,]\!]^{\mathrm{Codd}}$ *iff* Q *is weakly monotone under* $[\![\,]\!]$*.*

PROOF. We first show that $[\![D]\!]^{\mathrm{Codd}} \supseteq [\![D]\!]$, for every relational instance D. In fact there exists a natural database homomorphism h_{Codd} from $\chi_{\mathrm{Codd}}(D)$ to D such that $h_{\mathrm{Codd}}(\chi_{\mathrm{Codd}}(D)) = D$. Now assume $D' \in [\![D]\!]$, then $(D, D') \in \mathcal{R}^{\mathrm{all}}_{\mathrm{val}} \circ \mathcal{R}_{\mathrm{sem}}$. Then there exists a valuation h on D such that $(h(D), D') \in \mathcal{R}_{\mathrm{sem}}$. This mapping composed with h_{Codd} defines a valuation h' on $\chi_{\mathrm{Codd}}(D)$ such that $(h'(\chi_{\mathrm{Codd}}(D)), D') \in \mathcal{R}_{\mathrm{sem}}$. Hence $D' \in [\![\chi_{\mathrm{Codd}}(D)]\!] = [\![D]\!]^{\mathrm{Codd}}$.

Assume that Q is weakly monotone under $[\![\,]\!]^{\text{Codd}}$. Since $[\![D]\!]^{\text{Codd}} \supseteq [\![D]\!]$ for all D, we have that Q is also weakly monotone under $[\![\,]\!]$.

Now assume that Q is weakly monotone under $[\![\,]\!]$. Then $Q(\chi_{\text{Codd}}(D)) \preccurlyeq Q(D')$ for all relational instances D and all $D' \in [\![\chi_{\text{Codd}}(D)]\!]$. Thus we have $Q(D) = Q(\chi_{\text{Codd}}(D)) \preccurlyeq Q(D')$ for all relational instances D and all $D' \in [\![D]\!]^{\text{Codd}}$. Hence Q is weakly monotone under $[\![\,]\!]^{\text{Codd}}$. □

Proof of Corollary 2

We prove a more general statement:

Proposition 12. *Let $[\![\,]\!]$ be a relational semantics given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$ and let Q be a Boolean generic query. Then naïve evaluation works for Q under $[\![\,]\!]^{\text{Codd}}$ iff Q is preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms and $Q(D) = Q(\chi_{\text{Codd}}(D))$ for every relational instance D.*

Proof. First notice that $[\![\,]\!]^{\text{Codd}}$ is given by the pair $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ where $(D, D') \in \mathcal{R}_{\text{val}}$ iff $(\chi_{\text{Codd}}(D), D') \in \mathcal{R}^{\text{all}}_{\text{val}}$. Then the relational database domain having semantics $[\![\,]\!]^{\text{Codd}}$ has a representative set consisting of all *Codd* databases.

Then, by Theorem 1, one has that naïve evaluation works for Q under $[\![\,]\!]^{\text{Codd}}$ iff Q is weakly monotone (under $[\![\,]\!]^{\text{Codd}}$) and $Q(D) = Q(\chi_{\text{Codd}}(D))$ for all relational instances D.

By Lemma 9, Q is weakly monotone under $[\![\,]\!]^{\text{Codd}}$ and $Q(D) = Q(\chi_{\text{Codd}}(D))$ for all D iff Q is weakly monotone under $[\![\,]\!]$ and $Q(D) = Q(\chi_{\text{Codd}}(D))$ for all D.

Moreover by Proposition 4, Q is weakly monotone under $[\![\,]\!]$ iff it is preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms. This concludes the proof of the proposition. □

Proofs for Section 7

Proof of Proposition 7

Let D and D' be two Codd databases. Assume $D \mathbb{E}_{\text{CWA}} D'$, i.e., there exists a set of homomorphisms $h_1, \ldots, h_n$ from D so that $D' = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$. So for every tuple $(a_1, \ldots, a_m) \in D$, there is some $1 \leqslant i \leqslant n$ such that $(h_i(a_1), \ldots, h_i(a_m)) \in D'$, i.e., $(a_1, \ldots, a_m) \sqsubseteq (h_i(a_1), \ldots, h_i(a_m))$. It follows that $D \sqsubseteq^{\text{H}} D'$. Similarly for every tuple $(b_1, \ldots, b_m) \in D'$, there exists i such that $(b_1, \ldots, b_m) \in h_i(D)$, which entails that there is $(a_1, \ldots, a_m) \in D$ such that for every $1 \leqslant j \leqslant m$, $h_i(a_j) = b_j$ and so $(a_1, \ldots, a_m) \sqsubseteq (b_1, \ldots, b_m)$. It follows that $D \sqsubseteq^{\text{P}} D'$.

Conversely, assume $D \sqsubseteq^{\text{P}} D'$. For every tuple $t \in D$, consider the set $\{t' \in D' \mid t \sqsubseteq t'\}$ and observe that it is both finite and non empty. Now for every tuple $t \in D$, let $H_t = t'_1, \ldots, t'_k$ be a finite arbitrarily ordered sequence of tuples such that for every $1 \leqslant i \leqslant k$:

$$t'_i \in H_t \text{ iff } t'_i \in \{t' \in D' \mid t \sqsubseteq t'\}$$

Note that nothing prevents tuples to be repeated in the H_t's. So without loss of generality we can assume that there is some m big enough so that for every $t \in D$, $H_t = t'_1, \ldots, t'_m$ for some $t'_1, \ldots, t'_m \in D'$. For every $1 \leqslant i \leqslant m$, we can now put:

$$D'_i = \{t' \in D' \mid \exists t \in D \text{ such that } H_t = t'_1, \ldots, t'_i, \ldots, t'_m \text{ and } t' = t'_i\}$$

Observe that by $D \sqsubseteq^{\text{P}} D'$, $\bigcup_{1 \leqslant i \leqslant m} D'_i = D'$. Now for every $1 \leqslant i \leqslant m$ let $h_i : D \to D_i$ be as follows. For every $x \in \mathsf{Null} \cup \mathsf{Const}$ occurring as the j^{th} component in a tuple $t \in D$, we define $h_i(x)$ as the j^{th} component of the i^{th} tuple in H_t. As nulls are repeated neither in D nor in D' and by $D \sqsubseteq^{\text{P}} D'$, h_i is a homomorphism and moreover $h_i(D) = D_i$. It follows that $D \mathbb{E}_{\text{CWA}} D'$.

Proof of Theorem 4

We first show $(\rightarrowtail_{\text{CWA}} \cup \twoheadrightarrow_{\text{CWA}})^* = \mathbb{E}_{\text{CWA}}$.

⇒ Let $D \mathbb{E}_{\text{CWA}} D'$, i.e., there exists a set of homomorphisms $h_1, \ldots, h_n$ from D so that $D' = \bigcup_{1 \leqslant j \leqslant n} h_j(D)$. We take $m = \lceil \log n \rceil$ (i.e., the smallest integer m such that $n \leqslant 2^m$) and we let $D_0 \rightarrowtail_{\text{CWA}} \ldots \twoheadrightarrow_{\text{CWA}} D_m$ be a sequence

of $\rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}}$-updates of length m, where $D_0 = D$ and for every $1 \leqslant i \leqslant m$, we have $D_i = D_{i-1}[\bot/\bot] \cup D_{i-1}^{\text{fresh}}$, where $\bot \in \mathsf{Null}$ is some arbitrary null. Observe that D_m consists of 2^m isomorphic copies of D. Now let $\{\bot_1, \ldots, \bot_k\}$ be the set of nulls occurring in D. Then D_m is of the form

$$D_m = \bigcup_{1 \leqslant j \leqslant 2^m} D[\bot_1^j/\bot_1, \ldots, \bot_k^j/\bot_k],$$

where $\bot_i^j$ are distinct null values.

We now merge nulls in D_m in order to shrink it to a database D_{m+s} containing n distinct isomorphic copies of D as follows. We let $D_m \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}} \ldots \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}} D_{m+s}$ be a finite sequence of $\rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}}$-updates such that

$$D_{m+s} = \bigcup_{1 \leqslant j \leqslant n} D[\bot_1^j/\bot_1, \ldots, \bot_k^j/\bot_k]$$

where for every $m < i \leqslant m + s$

$$D_i = D_{i-1}[\bot_l^n/\bot_l^j] \text{ for some } j > n \text{ and some } 1 \leqslant l \leqslant k.$$

Finally we define one last finite sequence of $\rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}}$-updates $D_{m+s} \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}} \ldots \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}} D_{m+s+r}$ such that

$$D_{m+s+r} = \bigcup_{1 \leqslant i \leqslant n} h_i(D)$$

where for every $m + s < i \leqslant m + s + r$

$$D_i = D_{i-1}[h_j(\bot_l)/\bot_l^j] \text{ for some } 1 \leqslant j \leqslant n \text{ and some } 1 \leqslant l \leqslant k.$$

This proves that $\mathbb{E}_{\text{CWA}}$ is contained in $(\rightarrowtail_{\text{CWA}} \cup \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}})^*$.

$\Leftarrow$ Assume $D(\rightarrowtail_{\text{CWA}} \cup \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}})^* D'$. Then there is a sequence of nulls $\bot_1, \ldots, \bot_k$, a sequence $v_1, \ldots, v_k$ over $\mathsf{Const} \cup \mathsf{Null}$, a sequence of uniform renamings of nulls denoted by $\text{fresh}_1, \ldots, \text{fresh}_k$ (i.e., functions from current nulls to fresh nulls), and a sequence $D_0 \rightarrowtail D_1 \rightarrowtail \ldots \rightarrowtail D_k$ of updates length k such that

- $D_0 = D$,
- $D_k = D$,
- each $\rightarrowtail$ update is either $\rightarrowtail_{\text{CWA}}$ or $\rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}}$;
- $D_i = D_{i-1}[v_i/\bot_i]$ whenever $D_{i-1} \rightarrowtail_{\text{CWA}} D_i$, for $i \leqslant k$,
- $D_i = D_{i-1}[v_i/\bot_i] \cup D_{i-1}^{\text{fresh}_i}$ whenever $D_{i-1} \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}} D_i$, for $i \leqslant k$. Here, slightly abusing notation, we let $D_{i-1}^{\text{fresh}_i}$ stand for D_i^{fresh} in which nulls were replaced according to fresh_i.

Out of this sequence of $\rightarrowtail_{\text{CWA}}$ and $\rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}}$-updates of length k, we will now construct for every $0 \leqslant i \leqslant k$ and for every $1 \leqslant j \leqslant 2^i$ a family of homomorphisms h_j^i's from D to D_i so that $D_i = \bigcup_{1 \leqslant j \leqslant 2^i} h_j^i(D)$, which will entail in particular that $D' = \bigcup_{1 \leqslant j \leqslant 2^k} h_j^k(D)$, i.e., $D \; \mathbb{E}_{\text{CWA}} \; D'$ and will achieve the proof of $(\rightarrowtail_{\text{CWA}} \cup \rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}})^* = \mathbb{E}_{\text{CWA}}$.

We construct the h_j^i's by induction on i, first ordering them in a binary tree of depth k to ease the construction. We start by defining h_1^0 and use it to label the root of the tree. We then label the rest of the nodes so that each homomorphism h_j^i lies at depth i and labels the j^{th} node according to the left to right ordering in the tree. This will conveniently allow us to define each h_j^i as function of some previously defined homomorphism lying at depth $i - 1$. Now for each h_j^i with $i \neq 0$, observe that there is a unique s such that h_j^i is the child of h_s^{i-1}. We can now proceed to define the h_j^i's. We let h_1^0 be the identity and for all $i \neq 0$ we define h_j^i as follows:

- whenever h_j^i is the first child of its parent h_s^{i-1}, then h_j^i is exactly as h_s^{i-1}, except that it assigns the value v_i to all the preimages of $\bot_i$ by h_s^{i-1},
- whenever h_j^i is the second child of its parent h_s^{i-1} and the i^{th} update is a $\rightarrowtail_{\text{CWA}}$-update, then h_j^i is exactly as h_s^{i-1}, except that it assigns the value v_i to all the preimages of $\bot_i$ by h_s^{i-1},
- whenever h_j^i is the second child of its parent h_s^{i-1} and the i^{th} update is a $\rightarrowtail\!\!\!\!\!\rightarrow_{\text{CWA}}$-update, then for every $\bot \in \mathsf{Null}$ occurring in D_{i-1}, we let h_j^i assign the null $\text{fresh}_i(\bot)$ to all the preimages by h_s^{i-1} of $\bot$.

We now show the correctness of the construction. Assume as inductive hypothesis that for all $i < k$, the following property holds:

- for every $1 \leqslant j \leqslant 2^i$, the function $h_j^i : D \to D_i$ is a homomorphism and moreover $D_i = \bigcup_{1 \leqslant j \leqslant 2^i} h_j^i(D)$.

(Notice in particular that the property holds trivially for $i = 0$.) We now derive that it also holds for $i = k$. For each $1 \leqslant j \leqslant 2^k$, the fact that $h_j^k : D \to D_k$ is a homomorphism follows from the fact that its parent $h_s^{k-1} : D \to D_{k-1}$ is a homomorphism. Indeed, either h_j^k is exactly as h_s^{k-1} except that it assigns the value v_k to all the preimages of $\bot_k$ by h_s^{k-1}. But then $h_j^k(D) = D_{k-1}[v_k/\bot_k]$, which by assumption is a subinstance of D_k. Or $h_j^k(D) = D_{k-1}^{\text{fresh}_k}$, which by assumption is also a subinstance of D_k. But given that $D_{k-1} = \bigcup_{1 \leqslant j \leqslant 2^{k-1}} h_j^{k-1}(D)$, this also implies that $D_k = \bigcup_{1 \leqslant j \leqslant 2^k} h_j^k(D)$.

Observe now that a $\rightarrowtail^*_{\text{CWA}}$-update is just a special case of $(\rightarrowtail_{\text{CWA}} \cup \twoheadrightarrow_{\text{CWA}})^*$-update and so the proof of $(\rightarrowtail_{\text{CWA}} \cup \twoheadrightarrow_{\text{CWA}})^* = \Subset_{\text{CWA}}$ adapts immediately to a proof of $\rightarrowtail^*_{\text{CWA}} = \preceq_{\text{CWA}}$. Showing $\rightarrowtail^*_{\text{CWA}} \supseteq \preceq_{\text{CWA}}$ amounts to restricting in the first direction of the proof to the last type of updates described (which are all $\rightarrowtail_{\text{CWA}}$-updates), while showing $\rightarrowtail^*_{\text{CWA}} \subseteq \preceq_{\text{CWA}}$ amounts to restricting in the second direction to the special case where every two nodes lying at the same depth in the tree actually correspond to one and the same homomorphism.

The fact that $(\rightarrowtail_{\text{OWA}} \cup \rightarrowtail_{\text{CWA}})^* = \preceq_{\text{OWA}}$ now follows from $\rightarrowtail^*_{\text{CWA}} = \preceq_{\text{CWA}}$. Consider indeed $D \preceq_{\text{OWA}} D'$. So there is a homomorphism h such that $h(D)$ is a subinstance of D' and $D \rightarrowtail^*_{\text{CWA}} h(D)$. But then there is also a sequence of OWA updates $h(D) \rightarrowtail_{\text{OWA}} \ldots \rightarrowtail_{\text{OWA}} D'$ and so $D(\rightarrowtail_{\text{OWA}} \cup \rightarrowtail_{\text{CWA}})^* D'$. Conversely let $D(\rightarrowtail_{\text{OWA}} \cup \rightarrowtail_{\text{CWA}})^* D'$. So there is a sequence $D = D_0 \rightarrowtail D_1 \rightarrowtail \ldots \rightarrowtail D_k = D'$, where each $\rightarrowtail_i$ is either $\rightarrowtail_{\text{OWA}}$ or $\rightarrowtail_{\text{CWA}}$. In both cases, we trivially have a homomorphism from D_i to D_{i+1}, for $i < k$; in other words, $D_i \preceq_{\text{OWA}} D_{i+1}$. Since $\preceq_{\text{OWA}}$ is transitive, and contains both $\rightarrowtail_{\text{OWA}}$ and $\rightarrowtail_{\text{CWA}}$, we conclude that it contains $(\rightarrowtail_{\text{OWA}} \cup \rightarrowtail_{\text{CWA}})^*$, finishing the proof.

Proof of Proposition 8

We prove a necessary and sufficient condition for admissibility:

Lemma 10. *A powerset semantics given by $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ gives rise to an admissible database domain iff $\boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \circ \mathsf{id}_\ell \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \subseteq \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$. In particular if $\boldsymbol{\mathcal{R}}_{\text{sem}}$ is transitive then the database domain is admissible.*

PROOF. Assume first that $\boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \circ \mathsf{id}_\ell \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \subseteq \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$, and take an arbitrary $x \in \mathcal{D}$ and $c \in \mathcal{C}$. We have

1. $c \in [\![c]\!]_{\boldsymbol{\mathcal{R}}}$.
   Indeed we know $(c, \{c\}) \in \boldsymbol{\mathcal{R}}_{\text{val}}$ and $(\{c\}, c) \in \boldsymbol{\mathcal{R}}_{\text{sem}}$, then $c \in [\![c]\!]_{\boldsymbol{\mathcal{R}}}$.
2. $c \in [\![x]\!]_{\boldsymbol{\mathcal{R}}}$ implies $[\![c]\!]_{\boldsymbol{\mathcal{R}}} \subseteq [\![x]\!]_{\boldsymbol{\mathcal{R}}}$.
   Indeed if $c \in [\![x]\!]_{\boldsymbol{\mathcal{R}}}$ there exists $y \subseteq \mathcal{C}$ such that $(x, y) \in \boldsymbol{\mathcal{R}}_{\text{val}}$ and $(y, c) \in \boldsymbol{\mathcal{R}}_{\text{sem}}$. Moreover if $c' \in [\![c]\!]_{\boldsymbol{\mathcal{R}}}$ then $(c, c') \in \mathsf{id}_\ell \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$ (because $\boldsymbol{\mathcal{R}}_{\text{val}}$ is id_ℓ when restricted to $\mathcal{C}$). Hence $(x, c') \in \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \circ \mathsf{id}_\ell \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$. This implies $(x, c') \in \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$, and therefore $c' \in [\![x]\!]_{\boldsymbol{\mathcal{R}}}$.

By Proposition 1, the database domain is admissible.

Conversely assume that the database domain is admissible, and $(x, c) \in \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \circ \mathsf{id}_\ell \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$, then there exist c' such that $(x, c') \in \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$ and $(c', c) \in \mathsf{id}_\ell \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$. Then $c' \in [\![x]\!]_{\boldsymbol{\mathcal{R}}}$ and $c \in [\![c']\!]_{\boldsymbol{\mathcal{R}}}$ (because $\boldsymbol{\mathcal{R}}_{\text{val}}$ coincides with id_ℓ over $\mathcal{C}$). Then by admissibility, $c \in [\![x]\!]_{\boldsymbol{\mathcal{R}}}$, and hence $(x, c) \in \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$. □

Additional properties of powerset semantics

We prove some auxiliary results about a special form of powerset semantics.

Given a database domain $\mathbb{D}$ with set of objects $\mathcal{D}$ and complete objects $\mathcal{C}$, a relation $\mathcal{R} \subseteq \mathcal{D} \times \mathcal{C}$, and a relation $\boldsymbol{\mathcal{R}} \subseteq \mathcal{D} \times 2^{\mathcal{C}}$, we say that $\boldsymbol{\mathcal{R}} = \mathcal{P}(\mathcal{R})$ if $\boldsymbol{\mathcal{R}}$ consists of precisely the pairs $(x, \mathcal{X})$ such that $\mathcal{X} \neq \varnothing$ and $(x, y) \in \mathcal{R}$ for all $y \in \mathcal{X}$.

Remark that $\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}} = \mathcal{P}(\mathcal{R}^{\text{all}}_{\text{val}})$, and $\boldsymbol{\mathcal{R}}^{\text{min}}_{\text{val}} = \mathcal{P}(\mathcal{R}^{\text{min}}_{\text{val}})$.

Claim 2. *If a powerset semantics $[\![\,]\!]_{\boldsymbol{\mathcal{R}}}$ is given by a pair $(\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ where $\boldsymbol{\mathcal{R}}_{\text{val}} = \mathcal{P}(\mathcal{R}_{\text{val}})$. The following holds:*

- *For all $x, x' \in \mathcal{D}$, if x and x' are related by $\mathcal{R}_{\text{val}}$ to the same set of instances (i.e $(x, c) \in \mathcal{R}_{\text{val}}$ iff $(x', c) \in \mathcal{R}_{\text{val}}$ for all $c \in \mathcal{C}$) then $[\![x]\!]_{\boldsymbol{\mathcal{R}}} = [\![x']\!]_{\boldsymbol{\mathcal{R}}}$.*
- $\boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}} \supseteq \mathcal{R}_{\text{val}}$

PROOF. The first item immediately follows from the fact that $\boldsymbol{\mathcal{R}}_{\text{val}} = \mathcal{P}(\mathcal{R}_{\text{val}})$.

As for the second item, assume $(x, c) \in \mathcal{R}_{\text{val}}$ then $(x, \{c\}) \in \boldsymbol{\mathcal{R}}_{\text{val}}$. Since $\boldsymbol{\mathcal{R}}_{\text{sem}}$ contains id_r, we have that $(\{c\}, c) \in \boldsymbol{\mathcal{R}}_{\text{sem}}$. Hence $(x, c) \in \boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$. □

The following lemma easily follows:

Lemma 11. *Let* id *be the identity relation over complete relational instances. Assume that* $\mathcal{S}$ *is a strong representative set under a relational semantics given by a pair* $(\mathcal{R}_{\text{val}}, \mathsf{id})$*. Then* $\mathcal{S}$ *is a strong representative set also under any powerset semantics given by* $(\mathcal{P}(\mathcal{R}_{\text{val}}), \boldsymbol{\mathcal{R}}_{\text{sem}})$*.*

PROOF. Because $\mathcal{S}$ is a strong representative set under a semantics, $\mathcal{S} \supseteq \mathcal{C}$. Moreover there exists a function $\chi_{\mathcal{S}} : \mathcal{D} \to \mathcal{S}$, such that D and $\chi_{\mathcal{S}}(D)$ are related by $\mathcal{R}_{\text{val}}$ to precisely the same instances. Then by Claim 2, D and $\chi_{\mathcal{S}}(D)$ have the same semantics under $(\mathcal{P}(\mathcal{R}_{\text{val}}), \boldsymbol{\mathcal{R}}_{\text{sem}})$.

We further know that for all $D \in \mathcal{S}$ and for all $K \subseteq \mathsf{Const}$ there exists D' with $(D, D') \in \mathcal{R}_{\text{val}}$ and a bijection $i : \text{adom}(D) \to \text{adom}(D')$ with $i(D) = D'$ such that i and i^- are the identity on K. Again by Claim 2, $(D, D') \in \mathcal{P}(\mathcal{R}_{\text{val}}) \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$.

This proves that $\mathcal{S}$ is a strong representative set under the semantics $(\mathcal{P}(\mathcal{R}_{\text{val}}), \boldsymbol{\mathcal{R}}_{\text{sem}})$. □

Weak monotonicity and $\mathcal{R}_{\text{sem}}$-homomorphisms for powerset semantics

We define a notion of $\approx$-equivalence for powerset semantics.

If $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ is a database domain, $\boldsymbol{\mathcal{R}}$ and $\boldsymbol{\mathcal{R}}'$ are subsets of $\mathcal{D} \times 2^{\mathcal{C}}$, we say that $\boldsymbol{\mathcal{R}}'$ is $\approx$*-equivalent* to $\boldsymbol{\mathcal{R}}$ if the following two conditions are satisfied:

1. if $(x, \mathcal{X}) \in \boldsymbol{\mathcal{R}}$ then there exists $x' \in \mathcal{D}$ such that $x' \approx x$ and $(x', \mathcal{X}) \in \boldsymbol{\mathcal{R}}'$;
2. if $(x, \mathcal{X}) \in \boldsymbol{\mathcal{R}}'$ then there exists $x' \in \mathcal{D}$ such that $x' \approx x$ and $(x', \mathcal{X}) \in \boldsymbol{\mathcal{R}}$.

When the semantics is given by a pair $(\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$, we have the exact analog of Lemma 6:

Lemma 12. *Let* $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ *be a database domain whose semantics* $[\![\,]\!]$ *is given by a pair* $(\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ *and let* $\boldsymbol{\mathcal{R}}' \subseteq \mathcal{D} \times 2^{\mathcal{C}}$ *be* $\approx$*-equivalent to* $\boldsymbol{\mathcal{R}}_{\text{val}}$*, then* $\boldsymbol{\mathcal{R}}' \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$ *is* $\approx$*-equivalent to the graph of* $[\![\,]\!]$ *(i.e. to* $\boldsymbol{\mathcal{R}}_{\text{val}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$*). In particular a generic Boolean query over* $\mathbb{D}$ *is weakly monotone iff it is preserved under* $\boldsymbol{\mathcal{R}}' \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$*.*

A *powerset mapping type* $\boldsymbol{\mathcal{M}}$ is a function which associates to each complete relational instance D a class $\{\mathcal{H}_1, \dots \mathcal{H}_n, \dots\}$, where each $\mathcal{H}_i$ is a finite non-empty set of mappings $\text{adom}(D) \to \mathsf{Const}$.

If $\boldsymbol{\mathcal{M}}$ is a powerset mapping type, we denote by $\boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$ the set of pairs $(D, \{h_1(D), \dots h_k(D)\})$ such that D is a complete instance and $\{h_1, \dots h_k\} \in \boldsymbol{\mathcal{M}}(D)$. Given two complete relational instances D and D', an $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-*homomorphism* from D to D' is an $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphism $\{h_1, \dots h_k\}$ from D to D' which belongs to $\boldsymbol{\mathcal{M}}(D)$.

The following claim follows directly from definitions:

Claim 3. *If* $\boldsymbol{\mathcal{M}}$ *is a powerset mapping type then* $(D, D') \in \boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}} \circ \boldsymbol{\mathcal{R}}_{\text{sem}}$ *iff there exists an* $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$*-homomorphism from* D *to* D'

By combining the above claim with Lemma 12 we have:

Corollary 9. *Let* $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ *be a relational database domain whose semantics* $[\![\,]\!]$ *is given by a pair* $(\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ *and let* $\boldsymbol{\mathcal{M}}$ *be a powerset mapping type. Assume that* $\boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$ *is* $\approx$*-equivalent to* $\boldsymbol{\mathcal{R}}_{\text{val}}$*. Then a generic Boolean query is weakly monotone iff it preserved under* $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$*-homomorphisms.*

We now consider a special case when $\boldsymbol{\mathcal{R}}_{\text{val}} = \mathcal{P}(\mathcal{R}_{\text{val}})$. If $\mathcal{M}$ is a mapping type, we denote as $\mathcal{P}(\mathcal{M})$ the powerset mapping type associating to each instance D the set consisting of all possible finite non-empty $\mathcal{H} \subseteq \mathcal{M}(D)$.

It is easy to check that if $\boldsymbol{\mathcal{M}} = \mathcal{P}(\mathcal{M})$ then $\boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}} = \mathcal{P}(\mathcal{R}_{\mathcal{M}})$.

Lemma 13. *On an arbitrary database domain, assume* $\mathcal{R} \subseteq \mathcal{D} \times \mathcal{C}$ *and* $\boldsymbol{\mathcal{R}} = \mathcal{P}(\mathcal{R})$*. If* $\mathcal{R}' \subseteq \mathcal{D} \times \mathcal{C}$ *is strongly* $\approx$*-equivalent to* $\mathcal{R}$*, then* $\mathcal{P}(\mathcal{R}')$ *is* $\approx$*-equivalent to* $\boldsymbol{\mathcal{R}}$*.*

If a powerset relational semantics $[\![\ ]\!]$ is based on $\boldsymbol{\mathcal{R}}_{\text{val}} = \mathcal{P}(\mathcal{R}_{\text{val}})$ and $\mathcal{R}_{\mathcal{M}}$ is strongly $\approx$-equivalent to $\mathcal{R}_{\text{val}}$, for some mapping type $\mathcal{M}$, then a generic Boolean query is weakly monotone iff it is preserved under $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms, where $\boldsymbol{\mathcal{M}} = \mathcal{P}(\mathcal{M})$.

PROOF. Assume $\mathcal{R}'$ is strongly $\approx$-equivalent to $\mathcal{R}$. Let $(x, \mathcal{X})$ be in $\boldsymbol{\mathcal{R}}$. Note that $(x, c) \in \mathcal{R}$ for all $c \in \mathcal{X}$. Since $\mathcal{R}'$ is strongly $\approx$-equivalent to $\mathcal{R}$, there exists $y \approx x$ such that $(y, c) \in \mathcal{R}'$ for all $c \in \mathcal{X}$. Thus $(y, \mathcal{X}) \in \mathcal{P}(\mathcal{R}')$. Symmetrically we prove that if $(y, \mathcal{X})$ is in $\mathcal{P}(\mathcal{R}')$ then there exists $x \approx y$ such that $(x, \mathcal{X}) \in \boldsymbol{\mathcal{R}}$. This proves that $\mathcal{P}(\mathcal{R}')$ is $\approx$-equivalent to $\boldsymbol{\mathcal{R}}$.

Now assume a powerset relational semantics is based on $\boldsymbol{\mathcal{R}}_{\text{val}} = \mathcal{P}(\mathcal{R}_{\text{val}})$, and $\mathcal{R}_{\mathcal{M}}$ is strongly $\approx$-equivalent to $\mathcal{R}_{\text{val}}$. The $\mathcal{P}(\mathcal{R}_{\mathcal{M}})$ is $\approx$-equivalent to $\boldsymbol{\mathcal{R}}_{\text{val}}$. But $\mathcal{P}(\mathcal{R}_{\mathcal{M}}) = \boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$ for $\boldsymbol{\mathcal{M}} = \mathcal{P}(\mathcal{M})$. Then by Corollary 9, a generic Boolean query is weakly monotone iff it is preserved under $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms. □

Proof of Proposition 9

Recall that $\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}} = \mathcal{P}(\mathcal{R}^{\text{all}}_{\text{val}})$. Moreover by Lemma 7, if $\mathcal{M} = \text{all}$, then $\mathcal{R}_{\mathcal{M}}$ is strongly $\approx$-equivalent to $\mathcal{R}^{\text{all}}_{\text{val}}$. Remark also that for $\mathcal{M} = \text{all}$, $\mathcal{P}(\mathcal{M})$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms are precisely $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms. It follows then from Lemma 13 that a generic Boolean query is weakly monotone iff it is preserved under $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms. Now note that, by Lemma 11, for all semantics based on $\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}}$, the set of all instances is a representative set. Then the statement immediately follows from Lemma 2.

Preservation under unions of strong onto homomorphisms

We first define the notion of preservation under unions of strong onto homomorphisms for non-Boolean queries.

If Q is a k-ary query over complete relational instances (i.e. Q associates to each complete relational instance D a k-ary relation over $\text{adom}(D)$), we say that Q is preserved under unions of strong onto homomorphisms if, whenever there exists a union of strong onto homomorphisms $\{h_1 \dots h_n\}$ from an instance D to an instance D', and $\bar{a} \in Q(D)$, then $h_i(\bar{a}) \in Q(D')$, for all $i \in 1..k$.

Proof of Lemma 1

Lemma 1 immediately follows from the lemma below:

Lemma 14. *Formulae in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$ are preserved under unions of strong onto homomorphisms.*

PROOF. We proceed by structural induction on the formula φ.

If $\varphi =$*false* or $\varphi =$*true*, it is clearly preserved under unions of strong onto homomorphisms.

Assume now that $\varphi(\bar{x})$ is a positive atom $R(\bar{y})$ (including the case of an equality atom), where variables occurring in $\bar{y}$ are precisely $\bar{x}$. It follows from the definition of homomorphism that if an instance $D \models \varphi(\bar{a})$ then $h(D) \models \varphi(h(\bar{a}))$, for every homomorphism h. Then if $D' = h_1(D) \cup \dots \cup h_k(D)$ one has that $D' \models \varphi(h_i(\bar{a}))$ for all $i = 1..k$.

It is also easy to verify that if φ_1 and φ_2 are preserved under unions of strong onto homomorphisms, so are $\varphi_1 \wedge \varphi_2$ and $\varphi_1 \vee \varphi_2$.

Now assume $\varphi(\bar{x}) = \exists y \varphi'(y, \bar{x})$, where φ' is preserved under unions of strong onto homomorphisms. Assume that an instance $D \models \varphi(\bar{a})$, and that $D' = h_1(D) \cup \dots \cup h_k(D)$. Then $D \models \varphi'(b, \bar{a})$ for some value $b \in \text{adom}(D)$. Since φ' is preserved under unions of strong onto homomorphisms, $D' \models \varphi'(h_i(b), h_i(\bar{a}))$ for each $i \in 1..k$. Thus $D' \models \exists y \varphi'(y, h_i(\bar{a}))$, i.e. $D' \models \varphi(h_i(\bar{a}))$, for each $i \in 1..k$.

Now assume that φ is a sentence of the form $\forall \bar{x}(R(\bar{x}) \to \varphi'(\bar{y}))$ where variables $\bar{x}$ are pairwise distinct and variables $\bar{y}$ are contained in $\bar{x}$. Assume that an instance $D \models \varphi$ and that $D' = h_1(D) \cup \dots h_k(D)$. We prove $D' \models \varphi$. Assume that $D' \models R(\bar{b})$ for some tuple $\bar{b}$; then $h_i(D) \models R(\bar{b})$ for some $i \in 1..k$. Thus there exists a tuple $\bar{a}$ over $\text{adom}(D)$ such that $D \models R(\bar{a})$ and $h_i(\bar{a}) = \bar{b}$. Since $D \models \varphi$ one has that $D \models \varphi'(\bar{a}')$, where $\bar{a}'$ is the restriction of $\bar{a}$ to $\bar{y}$. Now, by the induction hypothesis, $\varphi'(\bar{x})$ is preserved under union of strong onto homomorphisms, therefore $D' \models \varphi'(h_i(\bar{a}')) = \varphi'(\bar{b}')$, where $\bar{b}'$ is the restriction of $\bar{b}$ to $\bar{y}$. Since this holds for all $\bar{b}$ such that $D' \models R(\bar{b})$, we have that $D' \models \varphi$. □

Proofs for Section 8

Proof of the claim that OWA non-powerset minimal semantics coincides with usual OWA semantics

We show here that as informally claimed in Section 8, combining $\mathcal{R}^{\min}_{\text{val}}$ with the subset relation (playing the role of $\mathcal{R}_{\text{sem}}$ for OWA) gives us the usual OWA semantics. Recall that the non-powerset minimal valuation semantics under the open-world assumption is defined as

$$[\![D]\!]^{\min}_{\text{OWA}} = \left\{ D' \;\middle|\; \begin{array}{l} D' \text{ is complete and} \\ \text{there is a } D\text{-minimal valuation } h : D \to D' \end{array} \right\}.$$

Proposition 13. $[\![\,]\!]^{\min}_{\text{OWA}} = [\![\,]\!]_{\text{OWA}}$.

Proof. $[\![\,]\!]^{\min}_{\text{OWA}} \subseteq [\![\,]\!]_{\text{OWA}}$ is immediate, so let $D' \in [\![D]\!]_{\text{OWA}}$. So there is a homomorphism $h : D \to D'$ such that $h(D)$ is a subinstance of D'. Now let $Hom = \{f(D) \mid f : D \to h(D) \text{ with } f \text{ a homomorphism}\}$. Observe that $(Hom, \subseteq)$ is a finite preorder. It follows that there is $f(D) \in Hom$ such that $f(D) \subseteq h(D)$ and there exists no $f'(D) \in Hom$ such that $f'(D) \subset f(D)$, i.e., there exists a D-minimal homomorphism h' such that $h'(D) \subseteq h(D)$. It follows that $D' \in [\![D]\!]^{\min}_{\text{OWA}}$. □

Minimal mappings

We now extend the D-minimality notion to arbitrary mappings. For an arbitrary mapping $h : \text{adom}(D) \to \mathsf{Const} \cup \mathsf{Null}$ we define, $\text{fix}(h, D) = \{c \in \mathsf{Const}(D) \mid h(c) = c\}$.

Given a relational instance D and a mapping $h : \text{adom}(D) \to \mathsf{Const} \cup \mathsf{Null}$ we say that h is D-minimal if there is no mapping $g : \text{adom}(D) \to \mathsf{Const} \cup \mathsf{Null}$ with $\text{fix}(h, D) \subseteq \text{fix}(g, D)$ and $g(D) \subsetneq h(D)$.

Notice that a D-minimal database homomorphism (D-minimal homomorphism, D-minimal valuation, resp.) is a database homomorphism (homomorphism, valuation, resp.) which is also a D-minimal mapping.

We first prove a technical lemma about minimal mappings.

Lemma 15. *Let D and D' be relational instances and assume there exists a D-minimal mapping $h : adom(D) \to \mathsf{Const} \cup \mathsf{Null}$ with $D' = h(D)$. Let E and E' be relational instances with isomorphisms $\mu : E \to D$ and $\mu' : D' \to E'$, such that μ, μ' and their inverses are the identity on* $\text{fix}(h, D)$. *Then the mapping $\mu' \circ h \circ \mu$ is E-minimal.*

Proof. Let $h' = \mu' \circ h \circ \mu$. First notice that h' is a mapping over $\text{adom}(E)$ such that $h'(E) = E'$, and h' is the identity on $\text{fix}(h, D)$.

Now assume by contradiction that there exists a mapping $g : \text{adom}(E) \to \mathsf{Const} \cup \mathsf{Null}$ such that $\text{fix}(h', E) \subseteq \text{fix}(g, E)$ and $g(E) \subsetneq h'(E)$. Then $g(E) \subsetneq E'$ and g is the identity on $\text{fix}(h, D)$. Let $g' = \mu'^{-} \circ g \circ \mu^{-}$; clearly g' is a mapping over $\text{adom}(D)$ and is the identity on $\text{fix}(h, D)$; therefore $\text{fix}(h, D) \subseteq \text{fix}(g', D)$. We now show that $g'(D) \subsetneq h(D)$. In fact $g'(D) = \mu'^{-}(g(E))$. Recall that $g(E) \subsetneq E'$, therefore $\mu'^{-}(g(E)) \subsetneq \mu'^{-}(E') = D' = h(D)$

This contradicts the assumption that h is D-minimal. □

Proof of Proposition 10

Let D be a relational instance and let h be a D-minimal database homomorphism. Assume by contradiction that $h(D)$ is not a core. Then there exists a database homomorphism h' on $h(D)$ such that $h'(h(D)) \subsetneq h(D)$. Clearly $h' \circ h$ is a database homomorphism on D, then this contradicts the D-minimality of h.

Now assume by contradiction that $h(core(D)) \subsetneq h(D)$, and let h_{core} be the database homomorphism from D onto $\text{core}(D)$. Clearly $h \circ h_{\text{core}}$ is a database homomorphism on D and $h_{\text{core}}(h(D)) = h(\text{core}(D)) \subsetneq h(D)$. Again this contradicts the D-minimality of h.

We now prove that there exists a core D and a database homomorphism $h : \text{adom}(D) \to \mathsf{Null}$ such that $h(D)$ is a core, but h is not D-minimal.

Fix a schema with a single 4-ary relation, and consider instances

D

$\perp_1$	$\perp_1$	$\perp_2$	$\perp_3$
$\perp_4$	$\perp_5$	$\perp_2$	$\perp_2$

$h(D)$

$\perp_6$	$\perp_6$	$\perp_7$	$\perp_7$
$\perp_6$	$\perp_7$	$\perp_7$	$\perp_7$

where $h:\ \perp_1 \to \perp_6,\ \perp_2 \to \perp_7,\ \perp_3 \to \perp_7,\ \perp_4 \to \perp_6,\ \perp_5 \to \perp_7$

It is easy to check that both D and and $h(D)$ are cores. However h is not D-minimal. In fact there exists a mapping $h':\ \perp_1 \to \perp_6,\ \perp_2 \to \perp_7,\ \perp_3 \to \perp_7,\ \perp_4 \to \perp_6,\ \perp_5 \to \perp_6$ such that $h'(D) \subsetneq h(D)$.

In fact one can produce a pure graph example (below, we shall assume that the nodes in graphs are distinct nulls, so we use the standard graph homomorphisms).

Let C_n be the directed cycle on n vertices. Let $G = C_4 + C_6$, where + stands for disjoint union. Note that each C_n is a core. Moreover, G is a core, since there is no homomorphism from C_6 to C_4. Let $H = C_3 + C_2$. Likewise, it is a core, and there is a strong onto homomorphism $h : G \to H$ that sends C_4 to C_2 and C_6 to C_3 (as in general we have $C_{2n} \to C_n$). Hence, H, G are cores, but h is not G-minimal since $G \to C_2$, as G is 2-colorable.

This also provides an example of D such that $[\![D]\!]^{\min}_{\text{CWA}} \neq [\![\text{core}(D)]\!]_{\text{CWA}}$ (which was stated right after Proposition 10). Indeed, take D to be $C_6 + C_4$ consisting of all nulls; note that $\text{core}(D) = D$. Let C_n^{C} be the cycle C_n whose nodes are distinct constants. Then $C_3^{\mathsf{C}} + C_2^{\mathsf{C}}$ is in $[\![D]\!]_{\text{CWA}}$. However, it is not in $[\![D]\!]^{\min}_{\text{CWA}}$. Indeed, if it were, there would be an onto homomorphism $h : C_6 + C_4 \to C_3^{\mathsf{C}} + C_2^{\mathsf{C}}$. Since we have no homomorphism $C_4 \to C_3$, then C_4 ought to be mapped by h to C_2^{C}, and hence C_6 will be mapped by h to C_3^{C} as h is onto. But we already saw that such a homomorphism cannot be minimal, since we have a homomorphism $g : C_6 + C_4 \to C_2^{\mathsf{C}}$. Thus, $C_3^{\mathsf{C}} + C_2^{\mathsf{C}} \notin [\![D]\!]^{\min}_{\text{CWA}}$.

Proof of Theorem 5

The theorem is straightforward from the following proposition:

Proposition 14. *If a relational semantics is given by a pair $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$ or $(\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$, the set $\mathcal{S}$ of cores is a strong representative set, and $\chi_{\mathcal{S}}(D) = core(D)$ for every instance D.*

Proof. It suffices to prove that for a semantics $[\![\ ]\!]$ given by $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$ the set of cores is a strong representative set. This will imply by Lemma 11, that the set of cores is a strong representative set also under any powerset semantics given by $(\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ (because $\boldsymbol{\mathcal{R}}^{\min}_{\text{val}} = \mathcal{P}(\mathcal{R}^{\min}_{\text{val}})$).

Clearly the set of cores contains all complete instances (recall that cores are defined w.r.t database homomorphisms here).

We now prove that if D is a core and $K \subseteq \mathsf{Const}$, there exists a D-minimal valuation v such that D and $v(D)$ are isomorphic in the way required by the definition of strong representative set. We observe that this property indeed follows from [18] (Proposition 6.11 (1) and (2)), but we prove it here directly for completeness.

If D is a core and $K \subseteq \mathsf{Const}$, let v be an arbitrary injective valuation $\text{adom}(D) \to \mathsf{Const}\backslash K$. Clearly v is an isomorphism between D and $v(D)$ and both v and v^- are the identity on $\mathsf{Const}(D) \cup K$, and therefore the identity on K, as required by the definition of strong representative set. We need to prove that $v(D) \in [\![D]\!]$. Now notice that the identity mapping over $\text{adom}(D)$ is D-minimal, because D is a core. Moreover v and v^- are the identity on $\mathsf{Const}(D)$, which is precisely the set of constants fixed by the identity mapping on D. Then we can apply Lemma 15 with $D = D' = E$ and conclude that v is D-minimal.

Thus $(D, v(D)) \in \mathcal{R}^{\min}_{\text{val}}$ and, since $\mathcal{R}_{\text{sem}}$ contains the identity, $v(D) \in [\![D]\!]$.

It remains to prove that $[\![D]\!] = [\![core(D)]\!]$. This will show that one can define $\chi_{\mathcal{S}}(D) = \text{core}(D)$ for every instance D. Let D be a relational instance. We prove that D and $\text{core}(D)$ have the same minimal images, i.e. $(D, D') \in \mathcal{R}^{\min}_{\text{val}}$ iff $(\text{core}(D), D') \in \mathcal{R}^{\min}_{\text{val}}$, for all D'. This will imply $[\![D]\!] = [\![core(D)]\!]$. We observe that this property has been proved in [18] (Lemma 6.9) restricted to the canonical solution in data exchange and its core.

Assume first that $(D, D') \in \mathcal{R}^{\min}_{\text{val}}$, then there exists a D-minimal valuation h such that $D' = h(D)$. We know by Proposition 10 that $h(\text{core}(D)) = h(D) = D'$. Moreover h has to be a $\text{core}(D)$-minimal valuation. In fact assume by contradiction that there exists a valuation h' on $\text{core}(D)$ such that $h'(core(D)) \subsetneq h(\text{core}(D)) = D'$. Let h_{core} be the database homomorphism from D to $\text{core}(D)$. Then $h' \circ h_{\text{core}}$ is a valuation on D and $h' \circ h_{\text{core}}(D) = h'(\text{core}(D)) \subsetneq D'$. This contradicts the assumption that h is D-minimal. Then h is a $\text{core}(D)$-minimal valuation and thus $(core(D), D') \in \mathcal{R}^{\min}_{\text{val}}$.

Conversely assume that $(\text{core}(D), D') \in \mathcal{R}^{\min}_{\text{val}}$, then there exists a core(D)-minimal valuation h such that $h(\text{core}(D)) = D'$. Therefore $h \circ h_{\text{core}}$ is a valuation and $h(h_{\text{core}}(D)) = h(\text{core}(D)) = D'$. We prove that $h \circ h_{\text{core}}$ is D-minimal. Assume by contradiction that there exists a valuation h' on D such that $h'(D) \subsetneq D'$. Then, since $\text{core}(D) \subseteq D'$ we have $h'(\text{core}(D)) \subseteq h'(D) \subsetneq D'$,contradicting the fact that h is core(D)-minimal.

We have then shown that the set $\mathcal{S}$ of cores is a representative set and $\chi_{\mathcal{S}}(D) = core(D)$ for all relational instances D. This concludes the proof of the proposition. □

Weak monotonicity and minimal homomorphisms

From Proposition 14 and Lemma 2 we derive the relationship between naïve evaluation and weak monotonicity for general semantics based on $\mathcal{R}^{\min}_{\text{val}}$ and $\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$:

Corollary 10. *If a relational semantics is given by a pair $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$ or $(\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ and Q is a generic Boolean query, then naïve evaluation works for Q iff Q is weakly monotone (under the corresponding semantics), and $Q(D) = Q(\text{core}(D))$ for every D.*

We now characterize weak monotonicity. We consider the mapping type $\mathcal{M} = \min$ which associates to each complete relational instance D the set of all D-minimal mappings $\text{adom}(D) \to \mathsf{Const}$.

We prove the following lemma:

Lemma 16. *If $\mathcal{M} = \min$ and $\approx$ is relational isomorphism, then $\mathcal{R}_{\mathcal{M}}$ is $\approx$-equivalent to $\mathcal{R}^{\min}_{\text{val}}$.*

Proof. Let $(D, v(D)) \in \mathcal{R}^{\min}_{\text{val}}$, where v is a D-minimal valuation; we prove that there exists a complete relational instance $E \approx D$ such that $(E, v(D)) \in \mathcal{R}_{\mathcal{M}}$.

The instance E is obtained from D by replacing nulls of D with new distinct constants not occurring in $\mathsf{Const}(D)$. Clearly there exists an isomorphism $i : E \to D$, thus $E \approx D$. Note that both i and i^- are the identity on $\mathsf{Const}(D)$. Let $h = v \circ i$, then $h(E) = v(D)$. Note that i and i^- are the identity on $\text{fix}(v, D) = \mathsf{Const}(D)$. Hence by Lemma 15 h is an E-minimal mapping. As a consequence $(E, v(D)) \in \mathcal{R}_{\mathcal{M}}$ (because $\mathcal{M} = \min$). This proves one direction.

Conversely assume $(E, h(E)) \in \mathcal{R}_{\mathcal{M}}$, where h is an E-minimal mapping; we prove that there exists a relational instance $D \approx E$ such that $(D, h(E)) \in \mathcal{R}^{\min}_{\text{val}}$.

The instance D is obtained from E by replacing each element of $\text{adom}(E) \backslash \text{fix}(h, E)$ with a new distinct null. Clearly this replacement defines an isomorphism $i : D \to E$ and therefore $E \approx D$. Note that both i and i^- are the identity on $\text{fix}(h, E)$. Then the mapping $v = h \circ i$ is also the identity on $\text{fix}(h, E)$; moreover $v(D) = h(E)$. But $\mathsf{Const}(D) = \text{fix}(h, E)$, then v is a valuation on D. Moreover by Lemma 15, v is D-minimal, and hence $(D, h(E)) \in \mathcal{R}^{\min}_{\text{val}}$. □

We say that a set $\mathcal{H} = \{h_1, \ldots h_k\}$ of mappings over $\text{adom}(D)$ is D-minimal if each h_i is D-minimal and $\text{fix}(h_i, D) = \text{fix}(h_j, D)$ for all $i, j \in 1..k$. We now consider the powerset mapping type $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ which associates to each D the class consisting of all non-empty finite D-minimal sets of mappings $\text{adom}(D) \to \mathsf{Const}$

Lemma 17. *If $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ and $\approx$ is relational isomorphism, then $\boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$ is $\approx$-equivalent to $\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$*

Proof. Let $(D, \mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$; we prove that there exists a complete relational instance $E \approx D$ such that $(E, \mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$. Let $\mathsf{Const}(\mathcal{X})$ be the union of $\mathsf{Const}(D')$, for all $D' \in \mathcal{X}$. The instance E is obtained from D by replacing nulls of D with new distinct constants not occurring in $\mathsf{Const}(D) \cup \mathsf{Const}(\mathcal{X})$. Clearly there exists an isomorphism $i : E \to D$, thus $E \approx D$. Note that both i and i^- are the identity on $\mathsf{Const}(D) \cup \mathsf{Const}(\mathcal{X})$.

For each $D' \in \mathcal{X}$ there exists a D-minimal valuation v such that $v(D) = D'$. Let $h = v \circ i$, then $h(E) = D'$ and, by Lemma 15 h is E-minimal. Note also that $\text{fix}(h, E) = \mathsf{Const}(D)$. Since such an h exists for all $D' \in \mathcal{X}$, we have $(E, \mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$. This proves one direction.

Conversely assume $(E, \mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$, then $\mathcal{X} = \{h_1(E), \ldots h_k(E)\}$ where where $\{h_1, \ldots h_k\}$ is E-minimal; we prove that there exists a relational instance $D \approx E$ such that $(D, \mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$. Let $K = \text{fix}(h_i, E)$ (which is the same for all $i \in 1..k$).

The instance D is obtained from E by replacing each element of $\text{adom}(E) \backslash K$ with a new distinct null. Clearly this replacement defines an isomorphism $i : D \to E$ and therefore $E \approx D$. Note that both i and i^- are the identity on K. Then the mappings $v_j = h_j \circ i$, for $j \in 1..k$ are all D-minimal, by Lemma 15. Moreover notice that $\mathsf{Const}(D) = K$, then v_j is a D-minimal valuation on D, and $v_j(D) = h_j(E)$, for all $j = 1..k$. It follows that $(D, \mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$. □

$\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms with $\mathcal{M} = \text{min}$ will be also referred to as *minimal $\mathcal{R}_{\text{sem}}$-homomorphisms*. Similarly $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms with $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ will be also referred to as *minimal $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms*.

Using Corollary 8 with $\mathcal{M} = \text{min}$, and Corollary 9 with $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ we then have:

Corollary 11. *If a relational semantics is given by a pair $(\mathcal{R}^{\text{min}}_{\text{val}}, \mathcal{R}_{\text{sem}})$ (or $(\boldsymbol{\mathcal{R}}^{\text{min}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$, respectively) and Q is a generic Boolean relational query, then Q is weakly monotone (under the corresponding semantics) iff it is preserved under minimal $\mathcal{R}_{\text{sem}}$-homomorphisms (minimal $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms respectively).*
Moreover naïve evaluation works for Q iff Q is preserved under minimal $\mathcal{R}_{\text{sem}}$-homomorphisms (minimal $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms respectively), and $Q(D) = Q(\text{core}(D))$ for every D.

Proof of Proposition 11

Notice that minimal homomorphisms are precisely minimal $\mathcal{R}_{\text{sem}}$-homomorphisms where $\mathcal{R}_{\text{sem}}$ is the identity. Similarly unions of minimal homomorphisms are precisely minimal $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms where $\boldsymbol{\mathcal{R}}_{\text{sem}} = \boldsymbol{\mathcal{R}}_{\cup}$. Then the proposition is a special case of Corollary 11.

Proof of Corollary 6

As a corollary of Lemma 2, we have that for a database domain $\mathbb{D}$ which has a representative set $\mathcal{S}$ and a generic query Q, naïve evaluation works for Q over $\mathcal{S}$ iff Q is weakly monotone over $\mathcal{S}$. By applying this to relational database domains, under the $[\![\,]\!]^{\text{min}}_{\text{CWA}}$ (respectively the $(\!|\,|\!)^{\text{min}}_{\text{CWA}}$ semantics) we have that naïve evaluation works for a generic query Q over cores iff Q is weakly monotone (under the corresponding semantics) over cores. By Proposition 11, if Q is preserved under minimal homomorphisms (their unions, respectively) then it is weakly monotone over cores. The corollary then follows from Proposition 6 and Lemma 1.

Proofs for Section 9

In order to be able to deal with constants in queries, we use the notion of C-genericity (instead of the stronger notion of genericity). If $C \subseteq \mathsf{Const}$, a relational k-ary query is *C-generic* if for all relational instances D and all one-to-one mappings $i : \text{adom}(D) \cup C \to \mathsf{Const} \cup \mathsf{Null}$ which are the identity on C, one has $Q(i(D)) = i(Q(D))$.

Clearly if $C = \varnothing$ the notion of C-genericity coincides with the usual notion of genericity for k-ary queries.

We now relate the notions of naïve evaluation and monotonicity for k-ary queries. To this end, for each relational database domain $\mathbb{D}$ and k-ary query Q over $\mathbb{D}$, we define a new database domain $\mathbb{D}^*$ and a Boolean query Q^* over $\mathbb{D}^*$. These are defined so that the "Boolean" notions of certain answers , naïve evaluation and monotonicity for Q^* over $\mathbb{D}^*$ are precisely equivalent to the corresponding notions for Q over $\mathbb{D}$. We then apply results from the Boolean case to Q^* over $\mathbb{D}^*$, and so derive corresponding results for Q over $\mathbb{D}$.

In what follows, if t is tuple over Const, with a little abuse of notation, we denote as t also the set of constants occurring in the tuple t

Given a relational database domain $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$, and a C-generic k-ary query Q over $\mathbb{D}$, we define $\mathbb{D}^* = \langle \mathcal{D}^*, \mathcal{C}^*, [\![\cdot]\!]^*, \approx^* \rangle$, and Q^* over $\mathbb{D}^*$ as follows:

- $\mathcal{D}^*$ is the set of pairs (D, t) where $D \in \mathcal{D}$ and t is a k-tuple over Const;
- $\mathcal{C}^*$ is the set of pairs of $\mathcal{D}^*$ where the instance D is in $\mathcal{C}$
- for all pairs $(D, t) \in \mathcal{D}^*$ the semantics $[\![(D, t)]\!]^*$ is defined as the set of pairs (D', t) such that $D' \in [\![D]\!]$.
- $(D, t) \approx^* (D', t')$ iff there exists a bijection $i : \text{adom}(D) \cup t \to \text{adom}(D') \cup t'$ such that $D' = i(D)$ and $t' = i(t)$ (as tuples), and both i and i^- are the identity on C.
- $Q^*(D, t) = 1$ iff $t \in Q(D)$.

Note that $\mathbb{D}^*$ and Q^* depend on D and Q.

The following claim easily follows from definitions:

Claim 4. *1) If $\mathbb{D}$ is admissible, $\mathbb{D}^*$ is also admissible;*
2) Q^* is generic (i.e. it does not distinguish $\approx^*$-equivalent objects);

3) $\mathsf{certain}(Q^*, D, t) = 1$ *iff* $t \in \mathsf{certain}(Q, D)$*, for every* $(D, t) \in \mathcal{D}^*$ *;*

4) Naïve-evaluation works for Q^* *iff naïve-evaluation works for* Q *;*

5) Q^* *is monotone iff* Q *is monotone;*

6) Q^* *is weakly monotone iff* Q *is weakly monotone;*

7) $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(D')$ *iff for every* k*-tuple* t *over* Const *one has* $Q^*(D, t) = Q^*(D', t)$*;*

8) If $\mathbb{D}$ *has a strong representative set* $\mathcal{S}$*, then* $\mathbb{D}^*$ *has a representative set* $\mathcal{S}^*$ *with* $\chi_{\mathcal{S}^*}(D, t) = (\chi_{\mathcal{S}}(D), t)$*.*

PROOF. 1) Assume $\mathbb{D}$ is admissible and consider $(D, t) \in \mathcal{C}^*$. Since $\mathbb{D}$ is admissible $D \in [\![D]\!]$. The $(D, t) \in [\![(D, t)]\!]^*$. Assume now that $(D, t) \in [\![(D', t)]\!]^*$. Then $D \in [\![D']\!]$. We also have $[\![(D, t)]\!]^* = \{(E, t) \mid E \in [\![D]\!]\}$, and since $\mathbb{D}$ is admissible $[\![D]\!] \subseteq [\![D']\!]$. Thus $[\![(D, t)]\!]^* \subseteq \{(E, t) \mid E \in [\![D']\!]\} = [\![(D', t)]\!]^*$.
By Proposition 1, $\mathbb{D}^*$ is admissible.

2) We know Q is C-generic. Consider two objects $(D, t), (D', t') \in \mathcal{D}^*$ such that $(D, t) \approx^* (D', t')$. We prove $Q^*(D, t) = Q^*(D', t')$, i.e. $t' \in Q(D')$ iff $t \in Q(D)$.
We know there exists an bijection $i : \mathrm{adom}(D) \cup t \to \mathrm{adom}(D') \cup t'$ such that $i(D) = D'$, $i(t) = t'$, and both i and i^- are the identity C. Note that i can be extended to a bijection $f : \mathrm{adom}(D) \cup t \cup C \to \mathrm{adom}(D') \cup t' \cup C$ which is the identity on C and such that $f(D) = D'$ and $f(t) = t'$.
Since f is injective on $\mathrm{adom}(D) \cup C$, it is the identity on C, and Q is C-generic, $Q(D') = f(Q(D))$. Thus $t' \in Q(D')$ iff $t' \in f(Q(D))$. Since f is injective over $\mathrm{adom}(D) \cup t$ and $f(t) = t'$, we have that $t' \in f(Q(D))$ iff $t \in Q(D)$. Then $t' \in Q(D')$ iff $t \in Q(D)$.

3) Consider $D \in \mathcal{D}^*$ and a k-tuple t over Const. We have $\mathsf{certain}(Q^*, D, t) = 1$ iff $Q^*(D', t) = 1$ for all $(D', t) \in [\![(D, t)]\!]^*$. This is equivalent to say that $t \in Q(D')$ for all $D' \in [\![D]\!]$, i.e that $t \in \mathsf{certain}(Q, D)$.

4) We recall that naïve evaluation works for Q^* iff $\mathsf{certain}(Q^*, D, t) = Q^*(D, t)$ for every $D \in \mathcal{D}$ and every k-tuple t over Const. By using the previous item, $\mathsf{certain}(Q^*, D, t) = Q^*(D, t)$ is equivalent to say that $t \in \mathsf{certain}(Q, D)$ iff $t \in Q(D)$. In other words naïve evaluation works for Q^* iff $\mathsf{certain}(Q, D) = Q^{\mathsf{C}}(D)$, for every $D \in \mathcal{D}$ iff naïve evaluation works for Q.

5) Assume that Q^* is monotone and consider $D, D' \in \mathcal{D}$ such that $[\![D']\!] \subseteq [\![D]\!]$. We prove that $Q^{\mathsf{C}}(D) \subseteq Q^{\mathsf{C}}(D')$.
By definition of $[\![\,]\!]^*$ we know that $[\![(D', t)]\!]^* \subseteq [\![(D, t)]\!]^*$, for all k-tuples t over Const. Since Q^* is monotone, $Q^*(D, t) \leqslant Q^*(D', t)$, i.e. $t \in Q(D)$ implies $t \in Q(D')$ for all k-tuples t over Const. Then $Q^{\mathsf{C}}(D) \subseteq Q^{\mathsf{C}}(D')$.
Assume now that Q is monotone and consider (D, t) and (D', t') in $\mathcal{D}^*$ such that $[\![(D', t')]\!]^* \subseteq [\![(D, t)]\!]^*$. Then $t' = t$ and $[\![D']\!] \subseteq [\![D]\!]$. Since Q is monotone, $Q^{\mathsf{C}}(D) \subseteq Q^{\mathsf{C}}(D')$; then $Q^*(D, t) \leqslant Q^*(D', t) = Q^*(D', t')$.

6) It is proved in the same way as 5).

7) It immediately follows from the definition of Q^*.

8) Assume $\mathbb{D}$ has a strong representative set $\mathcal{S}$, and take $\mathcal{S}^* = \{(D, t) | D \in S$ and t is a k-tuple over Const$\}$. We prove that $\mathcal{S}^*$ is representative for $\mathbb{D}^*$.
Notice that for all $(D, t) \in \mathcal{C}^*$ we have that $D \in \mathcal{C}$, therefore $D \in S$. Thus $(D, t) \in \mathcal{S}^*$.
Now consider $(D, t) \in \mathcal{S}^*$, then $D \in S$; therefore for $K = C \cup t$ there exists $D' \in [\![D]\!]$ and a bijection $i : \mathrm{adom}(D) \to \mathrm{adom}(D')$ such that $i(D) = D'$ and both i and i^- are the identity on K. Then $(D', t) \in [\![(D, t)]\!]^*$. We let i' be the mapping obtained by extending i with the identity mapping on t. It is easy to see that i' is a bijection $\mathrm{adom}(D) \cup t \to \mathrm{adom}(D') \cup t$, such that $i'(E) = D$ and $i'(t) = t$. Moreover both i' and i'^- are the identity on C. Therefore $(D, t) \approx^* (D', t)$.
Now we define $\chi_{\mathcal{S}^*}(D, t) = (\chi_{\mathcal{S}}(D), t)$, for all $(D, t) \in \mathcal{D}^*$. Clearly $[\![\chi_{\mathcal{S}^*}(D, t)]\!]^* = \{(D', t) \mid D' \in [\![\chi_{\mathcal{S}}(D)]\!]\}$, for all $(D, t) \in \mathcal{D}^*$. Therefore $[\![\chi_{\mathcal{S}^*}(D, t)]\!]^* = \{(D', t) \mid D' \in [\![D]\!]\} = [\![(D, t)]\!]^*$.

□

Using this claim in addition to the known relationship between naïve evaluation, monotonicity and weak monotonicity over $\mathbb{D}^*$ and Q^*, we immediately get the following corollaries.

From Lemma 2 on $\mathbb{D}^*$ and Q^*:

Corollary 12. *Let* $\mathbb{D}$ *be a relational database domain that has a strong representative set* $\mathcal{S}$*. and let* Q *be a* C*-generic* k*-ary query. Then naïve evaluation works* Q *if and only if*

1. Q *is weakly monotone and*

2. $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(\chi_{\mathcal{S}}(D))$ *for all* $D \in \mathcal{D}$

In particular if the whole set $\mathcal{D}$ *is strong representative, then naïve evaluation works for* Q *if and only if* Q *is weakly monotone.*

Similarly from Lemma 3:

Corollary 13. *Let* $\mathbb{D}$ *be an admissible relational database domain, and* Q *be a* C*-generic* k*-ary query. Assume that* $\mathbb{D}$ *has a strong representative set. Then naïve evaluation works for* Q *if and only if* Q *is monotone.*

Now recall that for relational semantics based on $\mathcal{R}^{\text{all}}_{\text{val}}$ as well as on $\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}}$ the whose set $\mathcal{D}$ is strong representative (see Lemma 4 and Lemma 11). Then we have:

Corollary 14. *If a relational semantics is given by a pair* $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$ *or a pair* $(\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$*, and* Q *is a* C*-generic* k*-ary query, naïve evaluation works for* Q *iff* Q *is weakly monotone (under the corresponding semantics).*

As for semantics based on minimal valuations, using Proposition 14 we have:

Corollary 15. *If a relational semantics is given by a pair* $(\mathcal{R}^{\text{min}}_{\text{val}}, \mathcal{R}_{\text{sem}})$ *or by a pair* $(\boldsymbol{\mathcal{R}}^{\text{min}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$*, and* Q *is a* C*-generic* k*-ary query, naïve evaluation works for* Q *iff* Q *is weakly monotone and* $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(\text{core}(D))$*, for every relational instance* D*.*

Weak monotonicity and $\mathcal{R}_{\text{sem}}$-homomorphisms for arbitrary queries

We now characterize weak monotonicity in terms of preservation for C-generic k-ary queries. In the sequel we use the following additional notation.

If $\mathcal{H} = \{h_1, \ldots h_n\}$ is a set of mappings over $\text{adom}(D)$, we say that $\mathcal{H}$ is the identity on a set of constants K if h_i is the identity on K for all $i \in 1..n$. Moreover we let $\mathcal{H}(D)$ denote the set $\{h_1(D), \ldots h_n(D)\}$.

If t is a tuple over Const and $\mathcal{X} = \{D_1 \ldots D_n\}$ is a set of instances we let $(\mathcal{X}, t)$ denote the set $\{(D_1, t), \ldots (D_n, t)\}$.

Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\cdot]\!], \approx \rangle$ be a relational database domain and where $[\![\,]\!]$ is given by a pair $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ (respectively $(\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$), and let Q be a C-generic k-ary query over $\mathbb{D}$.

Recall the definition of $\mathbb{D}^*$ and Q^* based on $\mathbb{D}$ and Q. Recall that Q^* is generic over $\mathbb{D}^*$ and remark that $[\![\,]\!]^*$ is given by the pair $(\mathcal{R}^*_{\text{val}}, \mathcal{R}^*_{\text{sem}})$ (respectively $(\boldsymbol{\mathcal{R}}^*_{\text{val}}, \boldsymbol{\mathcal{R}}^*_{\text{sem}})$) where

$$\mathcal{R}^*_{\text{val}} = \{((D,t),(D',t)) \mid (D,D') \in \mathcal{R}_{\text{val}} \text{ and } t \text{ is a } k\text{-tuple over } \mathsf{Const}\}$$

$$\mathcal{R}^*_{\text{sem}} = \{((D,t),(D',t)) \mid (D,D') \in \mathcal{R}_{\text{sem}} \text{ and } t \text{ is a } k\text{-tuple over } \mathsf{Const}\}$$

Similarly $\boldsymbol{\mathcal{R}}^*_{\text{val}} = \{(\,(D,t),(\mathcal{X},t)\,) \mid (D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\text{val}} \text{ and } t \text{ is a } k\text{-tuple over } \mathsf{Const}\}$ and $\boldsymbol{\mathcal{R}}^*_{\text{sem}} = \{(\,(\mathcal{X},t),(D,t)\,) \mid (\mathcal{X},D) \in \boldsymbol{\mathcal{R}}_{\text{sem}} \text{ and } t \text{ is a } k\text{-tuple over } \mathsf{Const}\}$

If $\mathcal{M}$ is a mapping type, we let $\mathcal{R}^*_{\mathcal{M}} = \{(x,y) \in \mathcal{C}^* \times \mathcal{C}^* \mid x = (D,t),\ y = (h(D),t),$ the mapping $h \in \mathcal{M}(D)$ and h is the identity on $C \cup t\}$.

Similarly if $\boldsymbol{\mathcal{M}}$ is a powerset mapping type, we let $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}} = \{(x,\mathcal{X}) \in \mathcal{C}^* \times 2^{\mathcal{C}^*} \mid x = (D,t),\ \mathcal{X} = (\mathcal{H}(D),t),$ the set of mappings $\mathcal{H} \in \boldsymbol{\mathcal{M}}(D)$ and $\mathcal{H}$ is the identity on $C \cup t\}$.

The above notion of $\mathcal{R}^*_{\mathcal{M}}$ (respectively $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$) is easily related to $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms (respectively $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms).

Claim 5. $((D,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}} \circ \mathcal{R}^*_{\text{sem}}$ *(respectively* $((D,t),(D',t)) \in \boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}} \circ \boldsymbol{\mathcal{R}}^*_{\text{sem}}$*) if and only if there exists an* $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$*-homomorphism (respectively an* $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$*-homomorphism) from* D *to* D' *which is the identity on* $C \cup t$*.*

Given a class $\mathcal{T}$ of $\mathcal{R}_{\text{sem}}$-homomorphisms (respectively $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms), we say that a k-ary query $\tilde{Q}$ over $\mathbb{D}$ is *weakly preserved under* $\mathcal{T}$ if $t \in \tilde{Q}(D)$ implies $t \in \tilde{Q}(D')$ whenever t is a k-tuple over Const, and in $\mathcal{T}$ there exists an $\mathcal{R}_{\text{sem}}$-homomorphism (respectively an $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphism) from D to D' which is the identity on t.

From the above claim it follows that weak preservation of Q can be characterized as follows:

Claim 6. *Q^* is preserved under $\mathcal{R}^*_{\mathcal{M}} \circ \mathcal{R}^*_{\text{sem}}$ (respectively under under $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}} \circ \boldsymbol{\mathcal{R}}^*_{\text{sem}}$) iff Q is weakly preserved under $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms (respectively under $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms) which are the identity on C.*

We now use the above claim and apply Lemma 6 and Lemma 12 to the database domain $\mathbb{D}^*$ and the generic query Q^*. We obtain the following corollary

Claim 7. *If the semantics $[\![\ ]\!]$ in $\mathbb{D}$ is given by $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$ and $\mathcal{R}^*_{\mathcal{M}}$ is $\approx^*$-equivalent to $\mathcal{R}^*_{\text{val}}$, then Q is weakly monotone iff it is weakly preserved under $\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms which are the identity on C.*
If $[\![\ ]\!]$ is given by $(\boldsymbol{\mathcal{R}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$ and $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ is $\approx^*$-equivalent to $\boldsymbol{\mathcal{R}}^*_{\text{val}}$, then Q is weakly monotone iff it is weakly preserved under $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms which are the identity on C.

We now consider mapping types $\mathcal{M} = \text{all}$ and $\mathcal{M} = \min$, as well as $\boldsymbol{\mathcal{M}} = \boldsymbol{all}$ (defined as $\mathcal{P}(\text{all})$) and $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ for powerset semantics.

Claim 8. *1) If $[\![\ ]\!]$ is based on $\mathcal{R}_{\text{val}} = \mathcal{R}^{\text{all}}_{\text{val}}$ then $\mathcal{R}^*_{\mathcal{M}}$ is strongly $\approx^*$-equivalent to $\mathcal{R}^*_{\text{val}}$ for $\mathcal{M} = \text{all}$;*

2) If $[\![\ ]\!]$ is based on $\mathcal{R}_{\text{val}} = \mathcal{R}^{\min}_{\text{val}}$ then $\mathcal{R}^*_{\mathcal{M}}$ is $\approx^*$-equivalent to $\mathcal{R}^*_{\text{val}}$ for $\mathcal{M} = \min$;

3) If $[\![\ ]\!]$ is based on $\boldsymbol{\mathcal{R}}_{\text{val}} = \boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}}$ (respectively $\boldsymbol{\mathcal{R}}_{\text{val}} = \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$) then $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ is $\approx^*$-equivalent to $\boldsymbol{\mathcal{R}}^*_{\text{val}}$ for $\boldsymbol{\mathcal{M}} = \boldsymbol{all}$ (respectively $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$).

PROOF. 1) Consider a pair $((D,t),(D',t))$ where $(D,D') \in \mathcal{R}^{\text{all}}_{\text{val}}$ and t is a k-tuple over Const. We prove that there exists $(E,t) \in \mathcal{C}^*$ such that $(D,t) \approx^* (E,t)$ and $((E,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}}$. The instance E is obtained from D by replacing nulls of D with new distinct constants not occurring in $\mathsf{Const}(D) \cup C \cup t$. Clearly there exists an isomorphism $i : E \to D$ such that both i and i^- are the identity on $C \cup t$. We let i' be the mapping obtained by extending i with the identity mapping on t. It is easy to see that i' is a bijection $\text{adom}(E) \cup t \to \text{adom}(D) \cup t$, such that $i'(E) = D$ and $i'(t) = t$. Moreover both i' and i'^- are the identity on C. Therefore $(E,t) \approx^* (D,t)$.

We know that there exists a valuation v on D such that $v(D) = D'$. Let $h = v \circ i$; then $h(E) = v(D) = D'$ and h is the identity on $C \cup t$ (because both v and i are). This implies $((E,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}}$, for $\mathcal{M} = all$. Remark that (E,t) only depends on (D,t) (and not on v).

Conversely consider a pair $((E,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}}$. Let $D' = h(E)$ where h is the identity on $C \cup t$. We prove that there exists $(D,t) \in \mathcal{D}^*$ such that $(D,t) \approx^* (E,t)$ and $(D,D') \in \mathcal{R}^{\text{all}}_{\text{val}}$. The instance D is obtained from E by replacing each element of $\text{adom}(E)$ not occurring in $C \cup t$ with a new distinct null. Clearly this replacement defines an isomorphism $i : D \to E$ such that both i and i^- are the identity on $C \cup t$. As in the previous case i can be extended to show $(E,t) \approx^* (D,t)$.

Let $v = h \circ i$. Remark that v is the identity on $\mathsf{Const}(D)$ (because $\mathsf{Const}(D) \subseteq C \cup t$ and both i and h are the identity on $C \cup t$). Then v is a valuation on D, and hence $(D,D') \in \mathcal{R}^{\text{all}}_{\text{val}}$. Note that D depends only on (E,t) (and not on h).

Thus $\mathcal{R}^*_{\mathcal{M}}$ is strongly $\approx$-equivalent to $(\mathcal{R}^{\text{all}}_{\text{val}})^*$, for $\mathcal{M} = \text{all}$.

2) Consider a pair $((D,t),(D',t))$ where $(D,D') \in \mathcal{R}^{\min}_{\text{val}}$ and t is a k-tuple over Const. We then know that $D' = h(D)$ where h is a D-minimal valuation. We prove that there exists $(E,t) \in \mathcal{C}^*$ such that $(D,t) \approx^* (E,t)$ and $((E,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}}$. The instance E is obtained from D by replacing nulls of D with new distinct constants not occurring in $\mathsf{Const}(D) \cup C \cup t$. Clearly there exists an isomorphism $i : E \to D$ such that both i and i^- are the identity on $\mathsf{Const}(D) \cup C \cup t$. It is easy to check that i can be extended over t to show $(E,t) \approx^* (D,t)$.

Now using Lemma 15, the mapping $h' = h \circ i$ is E-minimal. Moreover $h'(E) = D'$ and h' is the identity on $C \cup t$. It follows that $((E,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}}$ for $\mathcal{M} = \min$.

Conversely consider a pair $((E,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}}$ (where $\mathcal{M} = \min$). Let $D' = h(E)$, where h is an E-minimal and h is the identity on $C \cup t$. We prove that there exists $(D,t) \in \mathcal{D}^*$ such that $(D,t) \approx^* (E,t)$ and $(D,D') \in \mathcal{R}^{\min}_{\text{val}}$. The instance D is obtained from E by replacing each element of $\text{adom}(E)$ not occurring in $\text{fix}(h,E)$ with a new distinct null. Clearly this replacement defines an isomorphism $i : D \to E$ such that both i and i^- are the identity on $\text{fix}(h,E)$. Remark that i and i^- are also the identity on $C \cup t$. Indeed i is the identity on all constants, and i^- is the identity on $C \cup t$ because $(C \cup t) \cap \text{adom}(E) \subseteq \text{fix}(h,E)$. Thus as in the previous case, i can be extended to show $(E,t) \approx^* (D,t)$. Now let $h' = h \circ i$. By Lemma 15 h' is D-minimal. Moreover $h'(D) = h(E) = D'$, and h' is the identity on $\text{fix}(h,E)$. Now remark that $\mathsf{Const}(D) = \text{fix}(h,E)$, therefore h' is a valuation on D. We then conclude that $(D,D') = (D,h'(D)) \in \mathcal{R}^{\min}_{\text{val}}$.

3) We fist prove that if $[\![\ ]\!]$ is based on $\boldsymbol{\mathcal{R}}_{\text{val}} = \boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}}$ then $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ is $\approx^*$-equivalent to $\boldsymbol{\mathcal{R}}^*_{\text{val}}$ for $\boldsymbol{\mathcal{M}} = \boldsymbol{all}$.

Recall the notation $\mathcal{P}(\,)$ for powerset semantics. Notice that $(\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}})^* = \mathcal{P}((\mathcal{R}^{\text{all}}_{\text{val}})^*)$. We also know by the first item that $\mathcal{R}^*_{\mathcal{M}}$ is strongly $\approx^*$-equivalent to $(\mathcal{R}^{\text{all}}_{\text{val}})^*$ for $\mathcal{M} = \text{all}$. Then by Lemma 13, $\mathcal{P}(\mathcal{R}^*_{\mathcal{M}})$, for $\mathcal{M} = \text{all}$, is $\approx^*$-equivalent to $(\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}})^*$. Now remark that for $\mathcal{M} = \text{all}$ we have $\mathcal{P}(\mathcal{R}^*_{\mathcal{M}}) = \boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$, where $\boldsymbol{\mathcal{M}} = \boldsymbol{all}$.

We now prove that If $[\![\ ]\!]$ is based on $\boldsymbol{\mathcal{R}}_{\text{val}} = \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$, then $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ is $\approx^*$-equivalent to $\boldsymbol{\mathcal{R}}^*_{\text{val}}$ for $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$.

Let $((D,t),(\mathcal{X},t)) \in (\boldsymbol{\mathcal{R}}^{\min}_{\text{val}})^*$; then $(D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$. We prove that there exists a complete relational instance E such that $(E,t) \approx^* (D,t)$ and $((E,t),(\mathcal{X},t)) \in \boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ (where $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$). Let $\mathsf{Const}(\mathcal{X})$ be the union of $\mathsf{Const}(D')$, for all $D' \in \mathcal{X}$. The instance E is obtained from D by replacing nulls of D with new distinct constants not occurring in $\mathsf{Const}(D) \cup \mathsf{Const}(\mathcal{X}) \cup C \cup t$. Clearly there exists an isomorphism $i : E \to D$. Note that both i and i^- are the identity on $\mathsf{Const}(D) \cup \mathsf{Const}(\mathcal{X}) \cup C \cup t$. Therefore i can be extended to show $(E,t) \approx^* (D,t)$.

For each $D' \in \mathcal{X}$ there exists a D-minimal valuation v such that $v(D) = D'$. Let $h = v \circ i$, then $h(E) = D'$ and, by Lemma 15, h is E-minimal. Note also that $\text{fix}(h,E) = \mathsf{Const}(D)$, and h is the identity on $C \cup t$. Since such an h exists for all $D' \in \mathcal{X}$, the set of all h mappings, when D' ranges over $\mathcal{X}$, is E-minimal, as well as the identity on $C \cup t$. Then $((E,t),(\mathcal{X},t)) \in \boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ (for $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$).

$(E,\mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}}$. This proves one direction.

Conversely assume $((E,t),(\mathcal{X},t)) \in \boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ for $\mathcal{M} = \boldsymbol{min}$, then $\mathcal{X} = \{h_1(E), \ldots h_n(E)\}$ where where $\{h_1, \ldots h_n\}$ is E-minimal and the identity on $C \cup t$. We prove that there exists a relational instance D such that $(D,t) \approx^* (E,t)$ and $(D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$. Let $K = \text{fix}(h_i, E)$ (which is the same for all $i \in 1..n$).

The instance D is obtained from E by replacing each element of $\text{adom}(E) \backslash K$ with a new distinct null. Clearly this replacement defines an isomorphism $i : D \to E$. Note that both i and i^- are the identity on K; thus they are the identity on $C \cup t$. Indeed i is the identity on all constants; moreover $(C \cup t) \cap \text{adom}(E) \subseteq K$, then i^- is the identity on $C \cup t$. Then we can extend i to show $(D,t) \approx^* (E,t)$.

The mappings $v_j = h_j \circ i$, for $j \in 1..n$ are all D-minimal, by Lemma 15. Moreover notice that $\mathsf{Const}(D) = K$, then v_j is the identity on $\mathsf{Const}(D)$, and therefore a D-minimal valuation on D. Moreover $v_j(D) = h_j(E)$, for all $j = 1..n$. It follows that $(D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$.

□

By combining the above two claims with Corollaries 14 and 15, we get a characterization of weak monotonicity under both standard an minimal semantics:

Theorem 8. *Assume that a relational semantics is given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, (respectively $(\boldsymbol{\mathcal{R}}^{\text{all}}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$) and let Q be a C-generic k-ary relational query. Then Q is weakly monotone iff Q is weakly preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms (respectively $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms) which are the identity on C.*
Moreover naïve evaluation works for Q iff Q is weakly preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms (respectively $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms) which are the identity on C.

Theorem 9. *Assume that a relational semantics is given by a pair $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$, (respectively $(\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$) and let Q be a C-generic k-ary relational query. Then Q is weakly monotone iff Q is weakly preserved under minimal $\mathcal{R}_{\text{sem}}$-homomorphisms (respectively minimal $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms) which are the identity on C.*
Moreover naïve evaluation works for Q iff Q is weakly preserved under minimal $\mathcal{R}_{\text{sem}}$-homomorphisms (respectively minimal $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms) which are the identity on C, and $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(\text{core}(D))$ for all relational instances D.

Proof of Theorem 6

By combining Corollaries 12 and 13, we have that naïve evaluation works for Q iff Q is monotone iff Q is weakly monotone and $Q^{\mathsf{C}}(x) = Q^{\mathsf{C}}(\chi_{\mathcal{S}}(x))$.

If moreover the semantics is given by a pair $(\mathcal{R}^{\text{all}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, as a special case of Theorem 8 with $C = \varnothing$, we have that naïve evaluation works for Q iff Q is weakly preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms. Similarly if the semantics is given by a pair $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$, we use Theorem 9.

Proof of Theorem 7

Since every k-ary FO query is generic, by using Theorem 8 we have

Claim 9. *If Q is a k-ary FO query, naïve evaluation works for Q iff Q is weakly preserved under*

- *homomorphisms, under* OWA
- *strong onto homomorphisms, under* CWA
- *onto homomorphisms, under* WCWA
- *unions of strong onto homomorphisms, under* $(\!|\ |\!)_{\text{CWA}}$

We proved (Lemma 8 and Lemma 14) that k-ary formulae of $\mathsf{Pos} + \forall\mathsf{G}$ are preserved under strong onto homomorphisms, and k-ary formulae of $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$ are preserved under unions of strong onto homomorphisms. Moreover it is known that k-ary formulae of $\exists\mathsf{Pos}$ (respectively Pos) are preserved under homomorphisms (respectively onto homomorphisms) in the sense of Lemma 8.

Now notice that, for all these notions of homomorphism, preservation of k-ary formulae implies weak preservation. Then we have

Claim 10. *Let Q be a k-ary FO query. If Q is in $\exists\mathsf{Pos}$ (respectively , Pos, $\mathsf{Pos} + \forall\mathsf{G}$, $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$) then naïve evaluation works for Q under* OWA *(respectively* WCWA, CWA, $(\!|\ |\!)_{\text{CWA}}$*).*

which is the analog of Theorem 3 and Corollary 3.

As for semantics based on minimal valuations, genericity of FO queries together with Theorem 9 imply the following claim.

Claim 11. *Let Q be a k-ary FO query such that $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(core(D))$ for each relational instance D. Then naïve evaluation works for Q under $[\![\ ]\!]^{\min}_{\text{CWA}}$ (respectively under $(\!|\ |\!)^{\min}_{\text{CWA}}$) iff Q is weakly preserved under minimal homomorphisms (their unions, respectively).*

Now remark that, for any k-ary FO query, preservation under strong onto homomorphisms (respectively their unions) implies weak preservation under minimal homomorphisms (respectively their unions). Then we have

Claim 12. *Let Q be a k-ary FO query such that $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(core(D))$ for each relational instance D. Then If Q is in $\mathsf{Pos} + \forall\mathsf{G}$ (respectively $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$) naïve evaluation works for Q under the $[\![\ ]\!]^{\min}_{\text{CWA}}$ semantics (respectively under the $(\!|\ |\!)^{\min}_{\text{CWA}}$ semantics).*

which is the analog of Corollary 5. This concludes the proof of the theorem.