When is naive evaluation possible?

Citation for published version:
Gheerbrant, A, Libkin, L & Sirangelo, C 2013, When is naive evaluation possible? in Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2013. ACM, pp. 75-86. DOI: 10.1145/2463664.2463674

Digital Object Identifier (DOI):
10.1145/2463664.2463674

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Early version, also known as pre-print

Published In:
Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2013

When is Naïve Evaluation Possible?

AMÉLIE GHEERBRANT, University of Paris 7 and University of Edinburgh
LEONID LIBKIN, University of Edinburgh
CRISTINA SIRANGELO, LSV at ENS-Cachan, INRIA & CNRS

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.

Categories and Subject Descriptors: H.2.1 [**Database Management**]: Logical Design—*Data Models*; H.2.1 [**Database Management**]: Languages—*Query Languages*; H.2.4 [**Database Management**]: Systems—*Query Processing*

General Terms: Theory, Languages, Algorithms

Additional Key Words and Phrases: Incompleteness, naive tables/evaluation, certain answers, orderings, homomorphisms

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:

Authors' addresses: A. Gheerbrant, LIAFA, Université Paris Diderot – Paris 7, Case 7014, 75205 Paris Cedex 13, France;
L. Libkin, School of Informatics, University of Edinburgh, Informatics Forum, 10 Crichton Street, Edinburgh EH8 9AB, United Kingdom;
C. Sirangelo, LSV at ENS de Cachan, INRIA & CNRS. 61, avenue du Président Wilson, 94235 Cachan Cedex, France.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

— In SQL, the design of null-related features is one of the most criticized aspects of the language [Date and Darwin 1996], due to the oversimplification of the model. This 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. Indeed, this is what happens with queries like `select R.A from R where R.A not in (select S.A from S)` due to the 3-valued semantics of SQL.

— In theory, we understand that the proper way of evaluating queries on incomplete databases is to find *certain answers* to them [Imielinski and Lipski 1984]. Unfortunately, for many classes of queries, even within first-order logic, this is an intractable problem [Abiteboul et al. 1991], and even when it is tractable, there is no guarantee the algorithms can be easily implementable on top of commercial DBMSs [Gheerbrant et al. 2012].

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 on the incomplete database itself (i.e. as if nulls were the usual data values) does produce certain answers. This is usually referred to as *naïve evaluation* [Abiteboul et al. 1995; Imielinski and Lipski 1984]. 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 $\bot$ with subscripts:

R:

A	B
1	$\bot_1$
$\bot_2$	$\bot_3$

S:

B	C
$\bot_1$	4
$\bot_3$	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: evaluate the query directly on R and S, proceed as if nulls were usual values; they are equal only if they are syntactically the same (for instance $\bot_1 = \bot_1$ but $\bot_1 \neq \bot_2$). Then evaluating the above query results in two tuples: $(1, 4)$, and $(\bot_2, 5)$. Tuples with nulls cannot be certain answers, so we only keep the tuple $(1, 4)$.

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 in general, naïve evaluation need not compute certain answers. Recall that these are answers which hold true in all possible complete databases represented by the incomplete one, under some semantics of incompleteness.

For the query above, the tuple $(1, 4)$ is however the certain answer, under the common open-world semantics (to be properly defined later). This is true because [Imielinski and Lipski 1984] showed 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, [Libkin 2011] showed that under the open-world semantics, 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 [Chang and Keisler 1990], and in particular on its version over finite structures [Rossman 2008].

This observation suggests that the limits of naïve evaluation depend on the semantics of incompleteness, and that syntactic restrictions on queries admitting such evaluation might 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 with respect to 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 under very mild conditions, naïve evaluation works for a query iff it is monotone with respect to the semantic ordering. In fact, under even milder conditions, naïve evaluation corresponds to a weak notion of monotonicity, that only considers going from an object x to a more informative object $y \in [\![x]\!]$.

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 [Chang and Keisler 1990], 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 [Ajtai and Gurevich 1987; Stolboushkin 1995], or are notoriously hard to establish (a well-known example is Rossman's theorem [Rossman 2008], 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 8).

In Sections 6 and 7, we offer a more detailed study of other relational semantics of incompleteness. We take a closer look at semantic orderings, explain their justification via updates that incrementally improve informativeness of a database, and compare them with known orderings on Codd databases, that model SQL's null features. We show that capturing one of such well known orderings on Codd databases leads to a new class of *powerset* semantics, and we provide preservation results for that class, using the general methodology established earlier.

The key property of the semantics considered up to that point is what we call the saturation property: for each incomplete object, there is an isomorphic complete one in its semantics. For most standard semantics, this is trivially so, simply by substituting distinct constants for nulls. However, there is a class of semantics, that originated in AI and found applications in data exchange (see [Minker 1982; Hernich 2011]) for which this property fails.

To deal with them, we present a general tool for handling such non-saturated semantics in Section 9. It shows that the previous results apply as long as we have a subdomain that possesses the saturation property, and for queries that can be posed over objects from that domain. Then, in Section 10, we look at concrete examples of non-saturated semantics. These are *minimal* semantics that find their justification in the study of various forms of the closed world assumption. For them, the saturated subdomain is the set of relational cores (see [Hell and Nešetřil 1992]); in particular, previous results do apply, but only over cores. We conclude the paper by showing how to adjust the lifting tool to obtain results for non-Boolean queries under non-saturated semantics.

New material. This paragraph is inserted here to help the reviewers see where the additional material is. Perhaps this paragraph will be easier to follow once the paper has been read as it (due to its very nature) contains forward references.

In addition to including all the proofs (the conference version had none), we added the following concepts and results. The conference version dealt with one semantics based on minimal homomorphisms, and showed that, while the saturation property fails for it, results can be recovered for cores.

Here we show that this is an instance of a much more general phenomenon. Basically, for an arbitrary non-saturated semantics, we need the existence of a saturated subdomain to recover results relative to that subdomain; it just happens to be the set of cores for the minimal semantics. We present this general notion which also leads to a principled way of lifting results to non-Boolean queries.

Furthermore, we study a non-powerset based minimal semantics, and carry out the same program for it as for other semantics, thereby showing that there are other reasonable non-saturated semantics.

Specifically, the following results are completely new here:

- Proposition 7.2.
- The entire Section 9, including Theorem 9.1 and Corollary 9.3.
- The notion of the $[\![\ ]\!]^{\min}_{\text{CWA}}$ semantics.
- Theorem 10.2.
- Corollary 10.6, Proposition 10.7, and Corollaries 10.11 and 10.12: items referring to the $[\![\ ]\!]^{\min}_{\text{CWA}}$ semantics.
- Proposition 10.13.
- Section 11, including Lemma 11.1 and Theorem 11.5 for minimal semantics.

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 semantic orderings on incomplete databases, and in Section 7 we study naïve evaluation for the resulting new class of semantics. Section 8 shows how to lift all the results for Boolean queries to queries with free variables. Section 9 deals with non-saturated semantics in general, and two concrete cases of such, the minimal semantics, are handled in Section 10. Finally, Section 11 shows how to lift results to non-Boolean queries in non-saturated semantics.

2. PRELIMINARIES

2.1. 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 [Abiteboul et al. 1995; Imielinski and Lipski 1984]; 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 [Abiteboul et al. 1995; Imielinski and Lipski 1984]. 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 $\text{adom}(D) = \mathsf{Const}(D) \cup \mathsf{Null}(D)$. A *complete* database D has no nulls, i.e., $\text{adom}(D) \subseteq \mathsf{Const}$.

2.2. Homomorphisms

Homomorphisms 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, i.e., 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'.

2.3. 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 image contains only values in Const.

In general, the semantics $[\![D]\!]$ of an incomplete database is a set of *complete* databases D', i.e., databases D' with $\mathrm{adom}(D') \subseteq \mathsf{Const}$. The semantics under the closed-world assumption (or CWA *semantics*) is defined as

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

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

$$[\![D]\!]_{\mathrm{OWA}} \;=\; \{D' \;\mid\; D' \text{ is complete and there is a valuation } h : D \to D'\}.$$

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

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

2.4. 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 under the given semantics. When $[\![\;]\!]$ is $[\![\;]\!]_{\mathrm{OWA}}$ or $[\![\;]\!]_{\mathrm{CWA}}$, we write $\mathsf{certain}_{\mathrm{OWA}}(Q, D)$ or $\mathsf{certain}_{\mathrm{CWA}}(Q, D)$. Even for first-order queries, the standard 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 [Abiteboul et al. 1991].

Naïve evaluation of a query Q refers to a two-step procedure: first, evaluate Q on the incomplete database itself, as if nulls were values (i.e., equal iff they are syntactically the same: e.g., $\bot_1 = \bot_1,\;\; \bot_1 \neq \bot_2,\;\; \bot_1 \neq c$ 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 [Imielinski and Lipski 1984; Libkin 2011]) *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.

Note that in this paper we assume the active domain semantics for relational first-order queries.

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 show that it guarantees to compute certain answers for monotone queries.

3.1. 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 $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$, where $\mathcal{D}$ is a set, $\mathcal{C}$ is a subset of $\mathcal{D}$, the function $[\![\,]\!]$ is 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 [Gunter 1992], and such orderings have been used to provide semantics of incompleteness in the past [Buneman et al. 1991; Libkin 1995; 2011; Ohori 1990; Rounds 1991]. Note that $\preceq$ is a *preorder*.

Queries and certain answers. For now we look at Boolean queries in the most abstract setting (we will generalize them later). For a database domain $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$, 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 $[\![\,]\!]$) 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.

Saturation property. We now impose an additional property on database domains saying, essentially, that there are enough complete objects. A database domain $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ is *saturated* if every object has a complete object in its semantics that is isomorphic to it: that is, for each $x \in \mathcal{D}$ there is $y \in [\![x]\!]$ such that $x \approx y$.

In the case of the usual semantics of incompleteness, this property trivially holds: if we have an instance D with nulls $\bot_1, \ldots, \bot_n$, we simply replace them with distinct constants $c_1, \ldots, c_n$ that do not occur elsewhere in D, to obtain a complete database isomorphic to D. Nonetheless, there are other semantics, primarily motivated by AI considerations, that are not saturated; we shall deal with them in Section 9.

3.2. Naïve evaluation and monotonicity

We say that a query Q is *weakly monotone* if

$$y \in [\![x]\!] \quad \Rightarrow \quad Q(x) \leqslant Q(y).$$

That is, if y is a complete object representing x, and Q is already true on x, then Q must be true on y. This property characterizes naïve evaluation over saturated database domains.

THEOREM 3.1. *Let $\mathbb{D}$ be a database domain with the saturation property, and Q a generic Boolean query. Then naïve evaluation works for Q iff Q is weakly monotone.*

PROOF. The statement follows immediately from the more general Theorem 9.1 which will be proved in Section 9. However we provide a direct simple proof here for completeness.

Let Q be a Boolean generic query. Assume that naïve evaluation works for Q; then weak monotonicity of Q immediately follows.

Conversely assume that Q is weakly monotone, and let $x \in \mathcal{D}$. By weak monotonicity we have $Q(x) \leqslant \mathsf{certain}(Q, x)$. To prove the converse, assume $\mathsf{certain}(Q, x) = 1$. By the saturation property there exists $c \in [\![x]\!]$ such that $c \approx x$. We know $Q(c) = 1$; then by genericity $Q(x) = 1$. Hence $\mathsf{certain}(Q, x) = Q(x)$ for all $x \in \mathcal{D}$, i.e. naïve evaluation works for Q. □

Of course one can also look at the natural definition of monotonicity: a query Q is *monotone* if $x \preceq y$ implies $Q(x) \leqslant Q(y)$. Recall that $x \preceq y$ means that $[\![y]\!] \subseteq [\![x]\!]$. This condition turns out to be equivalent to weak monotonicity in database domains that satisfy one additional property. To state it, note that there is a natural duality between preorders and semantics: each semantics $[\![\ ]\!]$ gives rise to the ordering $x \preceq y \ \Leftrightarrow [\![y]\!] \subseteq [\![x]\!]$, and conversely any preorder $\preccurlyeq$ on $\mathcal{D}$ gives a semantics $[\![x]\!]_{\preccurlyeq} = \{y \in \mathcal{C} \mid x \preccurlyeq y\}$. We say that a database domain is *fair* if $[\![\ ]\!]$ and its ordering $\preceq$ agree: that is, the semantics that the ordering $\preceq$ gives rise to is $[\![\ ]\!]$ itself. Fair domains can be easily characterized:

PROPOSITION 3.2. *A database domain $\mathbb{D}$ is fair iff the following conditions hold:*

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

PROOF. Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\ ]\!], \approx \rangle$ be a database domain and let $\preceq$ be the preorder obtained from it.

First assume that (1) and (2) hold of $\mathbb{D}$. Recall that by definition for all $x, y \in \mathcal{D}$, $x \preceq y$ iff $[\![y]\!] \subseteq [\![x]\!]$ and so $[\![x]\!]_{\preceq} = \{c \in \mathcal{C} \mid [\![c]\!] \subseteq [\![x]\!]\}$. We want to show that $\mathbb{D}$ is fair, i.e., for all $x \in \mathcal{D}$, $[\![x]\!] = [\![x]\!]_{\preceq}$. So let $x \in \mathcal{D}$ and $c \in \mathcal{C}$ such that $c \in [\![x]\!]$. By condition 2, $[\![c]\!] \subseteq [\![x]\!]$. But then $c \in [\![x]\!]_{\preceq}$ 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, we have $[\![x]\!]_{\preceq} \subseteq [\![x]\!]$.

Conversely assume $\mathbb{D}$ is fair, i.e., for all $x \in \mathcal{D}$, $[\![x]\!] = [\![x]\!]_{\preceq} = \{c \in \mathcal{C} \mid [\![c]\!] \subseteq [\![x]\!]\}$. So in particular for all $c \in \mathcal{C}$, $[\![c]\!] = \{c' \in \mathcal{C} \mid [\![c']\!] \subseteq [\![c]\!]\}$. As $[\![c]\!] \subseteq [\![c]\!]$, it follows that $c \in [\![c]\!]$, that is, condition (1) holds. Condition (2) follows immediately from $[\![x]\!] = \{c \in \mathcal{C} \mid [\![c]\!] \subseteq [\![x]\!]\}$. □

The standard semantics – including all those seen in the previous section – satisfy these conditions. The first condition 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. Note also that in a fair domain, $y \in [\![x]\!]$ implies $x \preceq y$, so weak monotonicity is indeed weaker than monotonicity.

In fair database domains, we can extend Theorem 3.1:

PROPOSITION 3.3. *Let $\mathbb{D}$ be a fair database domain with the saturation property, and Q a generic Boolean query. Then the following are equivalent:*

(*1*) *Naïve evaluation works for Q;*
(*2*) *Q is monotone;*
(*3*) *Q is weakly monotone.*

PROOF. We need to prove that in a fair database domain naive evaluation works for Q iff Q is monotone. Assume that naïve evaluation works for Q, 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. Let x be in $\mathcal{D}$, we prove that $Q(x) = \mathsf{certain}(Q, x)$. Let $c \in [\![x]\!]$. Since the database domain is fair, $x \preceq c$. 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 $\mathsf{certain}(Q, x) = 1$. By the saturation property there exists $c' \in [\![x]\!]$ such that $c' \approx x$. We know $Q(c') = 1$, then by genericity, $Q(x) = 1$.

This shows $Q(x) = \mathsf{certain}(Q, x)$ – i. e. naïve evaluation works for Q – and concludes the proof of the proposition. □

Theorem 3.1 and Proposition 3.3 establish the promised connection between monotonicity and naïve evaluation. Extension to non-Boolean queries is given in Section 8.

4. SEMANTICS, RELATIONS, AND HOMOMORPHISMS

We have seen that getting naïve evaluation to work (at least for Boolean queries), is equivalent to their (weak) monotonicity. To apply this to concrete semantics, 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.

4.1. 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. Intuitively, a pair (x, c) is in $\mathcal{R}_{\text{val}}$ if c is obtained from x by replacing nulls by constants. The restriction of $\mathcal{R}_{\text{val}}$ to $\mathcal{C}$ is the identity: $\mathcal{R}_{\text{val}} \cap (\mathcal{C} \times \mathcal{C}) = \{(c, c) \mid c \in \mathcal{C}\}$

(if there are no nulls, there is no substitution). And since for every object there is some way to replace nulls by constants, $\mathcal{R}_{\text{val}}$ 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 4.1. *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 fair iff $\mathcal{R}_{\text{sem}}$ is transitive.*

PROOF. 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 3.2, $\mathbb{D}$ is fair.

Conversely assume that $\mathbb{D}$ is fair, 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 3.2, $e \in [\![c]\!]$. Then $(c, e) \in \mathcal{R}_{\text{sem}}$. This proves that $\mathcal{R}_{\text{sem}}$ is transitive. □

Relational databases. When we deal with relational databases, the most natural valuation relation is $\mathcal{R}_{\text{val}}^{\text{rdb}}$ defined as follows:

$$(D, D') \in \mathcal{R}_{\text{val}}^{\text{rdb}} \quad \Leftrightarrow \quad D' = v(D) \text{ for some valuation } v.$$

So we assume, for now, that in relational semantics of incompleteness, the valuation relation is $\mathcal{R}_{\text{val}}^{\text{rdb}}$, and thus such semantics are defined by relation $\mathcal{R}_{\text{sem}}$. For OWA and CWA, these are particularly easy:

- For CWA, $\mathcal{R}_{\text{sem}}$ is the identity (i.e., $=$);
- For OWA, $\mathcal{R}_{\text{sem}}$ is the subset relation (i.e., $\subseteq$).

The special form of relation $\mathcal{R}_{\text{val}}^{\text{rdb}}$ implies the saturation property. Indeed, it does allow us to replace nulls by distinct constants that do not occur elsewhere in the instance. Therefore, by Theorem 3.1 we have:

PROPOSITION 4.2. *For an arbitrary relational semantics given by relation $\mathcal{R}_{\text{sem}}$, and an arbitrary generic Boolean query Q, naïve evaluation works for Q iff Q is weakly monotone.*

4.2. Naïve evaluation via homomorphism preservation

We shall now relate weak monotonicity and preservation under homomorphisms (at least for relational semantics).

Consider relational databases over constants. Given two such databases D and D', a mapping h defined on the active domain 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 from D to D', if Q is true in D, then Q is true in D'.

PROPOSITION 4.3. *If a relational semantics is given by a relation $\mathcal{R}_{\text{sem}}$ and Q is a generic Boolean query, then Q is weakly monotone iff it is preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms.*

PROOF. We prove a slightly more general result holding for arbitrary relational semantics given by a pair $(\mathcal{R}_{\text{val}}, \mathcal{R}_{\text{sem}})$. We first need to introduce some definitions and notations. If $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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 $(x, y) \in \mathcal{R}$. If $\mathcal{R}$ 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 4.4. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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 4.5. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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 is 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 4.4. □

We are now ready to move to the relational setting and finish the proof of the proposition. In what follows, we say that $\mathbb{D}$ is a *relational database domain* if $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$, 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 $\text{adom}(D)$ such that $\pi(D) = D'$).

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 $\{(D, h(D)) \mid D \text{ is a complete relational instance and } h \in \mathcal{M}(D)\}$. 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 4.5 we have:

COROLLARY 4.6. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx\rangle$ be a relational database domain whose semantics $[\![\,]\!]$ is given by a pair $(\mathcal{R}_{\mathrm{val}}, \mathcal{R}_{\mathrm{sem}})$ and let $\mathcal{M}$ be a mapping type. Assume that $\mathcal{R}_{\mathcal{M}}$ is $\approx$-equivalent to $\mathcal{R}_{\mathrm{val}}$. Then a generic Boolean query over $\mathbb{D}$ is weakly monotone iff it is preserved under $\mathcal{M}$-$\mathcal{R}_{\mathrm{sem}}$-homomorphisms.*

Proposition 4.3 will be obtained as a special case of Corollary 4.6. To prove it, we consider the mapping type $\mathcal{M} = \mathrm{all}$, associating with each complete relational instance D the set of all mappings adom$(D) \to \mathsf{Const}$, and we prove the following lemma:

LEMMA 4.7. *If $\mathcal{M} = \mathrm{all}$ and $\approx$ is relational isomorphism, then $\mathcal{R}_{\mathcal{M}}$ is strongly $\approx$-equivalent to $\mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$.*

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) $(D, D') \in \mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$ 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 $(D, D') \in \mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$. 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 $\mathcal{M} = \mathrm{all}$). 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 $(D, D') \in \mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$.

The instance D is obtained from E by replacing each element of 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 adom$(E) \to \mathsf{Const}$. Let $v = h \circ i$. Then v is a valuation on D (because adom(D) contains no constants, and D' is complete) and hence $(D, D') \in \mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$. □

Now remark that with $\mathcal{M} = \mathrm{all}$, $\mathcal{M}$-$\mathcal{R}_{\mathrm{sem}}$-homomorphisms coincide with $\mathcal{R}_{\mathrm{sem}}$-homomorphisms. Then Proposition 4.3 follows immediately from Corollary 4.6 with $\mathcal{M} = \mathrm{all}$. □

Putting together Proposition 4.2 and Proposition 4.3, we have our first key result for naïve evaluation over incomplete databases.

THEOREM 4.8. *For a relational incompleteness semantics given by a semantic relation $\mathcal{R}_{\mathrm{sem}}$, and a generic Boolean query Q, naïve evaluation works for Q iff Q is preserved under $\mathcal{R}_{\mathrm{sem}}$-homomorphisms.*

4.3. Homomorphisms for relational semantics

Theorem 4.8 connects naïve evaluation with homomorphism preservation. We now investigate what these $\mathcal{R}_{\mathrm{sem}}$-homomorphisms are.

- CWA semantics. In this case $\mathcal{R}_{\mathrm{sem}}$ is the identity, and the definition states that h is an $\mathcal{R}_{\mathrm{sem}}$-homomorphism from D to D' if $D' = h(D)$. That is, under CWA, $\mathcal{R}_{\mathrm{sem}}$-homomorphisms are the *strong onto* homomorphisms, i.e., homomorphisms from D to $h(D)$.
- OWA semantics. In this case $\mathcal{R}_{\mathrm{sem}}$ is $\subseteq$, and the definition states that h is an $\mathcal{R}_{\mathrm{sem}}$-homomorphism from D to D' if $h(D) \subseteq D'$. That is, under OWA, $\mathcal{R}_{\mathrm{sem}}$-homomorphisms are just the usual homomorphisms.

Another well known notion of homomorphisms is that of *onto* homomorphisms. When used in the database context, an onto homomorphism h from D to D' is a homomorphism between D and D' so that $h(\mathrm{adom}(D)) = \mathrm{adom}(D')$. For instance, if

$D = \{(1,2)\}$, and $h(1) = 3, h(2) = 4$, then h is a strong onto homomorphism from D to $D' = \{(3,4)\}$, and an onto homomorphism to $D'' = \{(3,4),(4,3)\}$. Note that while D'' contains more than $h(D)$, all the tuples in D'' only use elements that occur in $h(D)$.

A semantics of incompleteness that corresponds to this notion, that we refer to as *weak* CWA, or WCWA semantics, was actually previously studied [Reiter 1977] (in a slightly different, deductive-database context). We define it as follows:

$$[\![D]\!]_{\text{WCWA}} = \left\{ D' \;\middle|\; \begin{array}{l} D' \text{ is complete and there is a valuation } h : D \to D' \\ \text{so that } \mathrm{adom}(D') = \mathrm{adom}(h(D)) \end{array} \right\}.$$

In other words, it is not completely closed world: a database can be extended, but still in a rather limited fashion, only with the tuples that use values already stored in the database.

For this semantics, $\mathcal{R}_{\text{sem}}$ contains all pairs (D, D') so that $D \subseteq D'$ and $\mathrm{adom}(D) = \mathrm{adom}(D')$. That is, D can be expanded only within its active domain. Thus, $\mathcal{R}_{\text{sem}}$-homomorphisms are exactly onto homomorphisms.

For this relation $\mathcal{R}_{\text{sem}}$, the notion of preservation under $\mathcal{R}_{\text{sem}}$-homomorphisms is exactly the notion of preservation 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 present in $\{1,2\}$.

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

COROLLARY 4.9. *Let Q be a Boolean generic 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 4.9 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 [Chang and Keisler 1990]. 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 [Rossman 2008]. In database language, this means being a union of conjunctive queries, which led to an observation [Libkin 2011] 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 [Keisler 1965b] 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 *positive* formulae use all the FO connectives except negation (i.e., $\wedge, \vee, \forall, \exists$). Formally, 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 $\forall x \varphi$ are in Pos.

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

Rossman's theorem [Rossman 2008] 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 [Chang and Keisler 1990] 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 [Ajtai and Gurevich 1987; Stolboushkin 1995] but the implication from being positive to preservation is still valid.

A characterization of preservation under strong onto homomorphisms was stated in [Keisler 1965a; 1965b], 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 [Keisler 1965b]. 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 [Compton 1983], extends Pos with universal guards. It is defined as follows:

— *true* and *false* are in $\mathsf{Pos} + \forall\mathsf{G}$;
— every positive atomic formula (i.e., $R(\bar{x})$ or $x = y$) is in $\mathsf{Pos} + \forall\mathsf{G}$;
— if $\varphi, \psi \in \mathsf{Pos} + \forall\mathsf{G}$, then $\varphi \vee \psi$ and $\varphi \wedge \psi$ are in $\mathsf{Pos} + \forall\mathsf{G}$;
— if φ is in Pos, then $\exists x \varphi$ and $\forall x \varphi$ are in $\mathsf{Pos} + \forall\mathsf{G}$.
— if $\varphi(\bar{x}, \bar{y})$ is in $\mathsf{Pos} + \forall\mathsf{G}$, and R is an n-ary relation symbol, then the formula $\forall x_1, \ldots, x_n \big(R(x_1, \ldots, x_n) \ \rightarrow\ \varphi(x_1, \ldots, x_n, \bar{y})\big)$ is in $\mathsf{Pos} + \forall\mathsf{G}$ if $x_1, \ldots, x_n$ are pairwise distinct variables;
— if $\varphi(x, z, \bar{y})$ is in $\mathsf{Pos} + \forall\mathsf{G}$, and x, z are distinct variables, then the formula $\forall x, z \big(x = z \ \rightarrow\ \varphi(x, z, \bar{y})\big)$ is in $\mathsf{Pos} + \forall\mathsf{G}$.

Note that the first four rules are the same as for Pos, so we have $\exists\mathsf{Pos} \subsetneq \mathsf{Pos} \subsetneq \mathsf{Pos} + \forall\mathsf{G}$.

PROPOSITION 5.1. *Sentences in* $\mathsf{Pos} + \forall\mathsf{G}$ *are preserved under strong onto homomorphisms.*

PROOF. We prove preservation for arbitrary formulas with free variables in the fragment. 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 $h(\bar{a}) \in Q(D')$.

Now we show that $\mathsf{Pos} + \forall\mathsf{G}$ formulae (and thus sentences) are preserved under strong onto homomorphisms. We proceed by structural induction on the formula φ. If $\varphi =$*false* or $\varphi =$*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 $b \in \text{adom}(D)$. 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 $b \in \text{adom}(D')$, we have to prove $D' \models \varphi'(b, h(\bar{a}))$. Since $D' = h(D)$, there exists $a \in \text{adom}(D)$ 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 $D' \models \varphi'(h(a), h(\bar{a})) = \varphi'(b, h(\bar{a}))$.

We next assume that $\varphi(\bar{x}, \bar{y}) \in \mathsf{Pos} + \forall\mathsf{G}$ is preserved under strong onto homomorphisms and show that $\forall \bar{x}\ (R(\bar{x}) \to \varphi)$ is, where $\bar{x} = (x_1, \ldots, x_n)$ is a tuple of pairwise distinct variables. Let $D \models \forall \bar{x}(R(x_1, \ldots, x_n) \to \varphi(\bar{x}, \bar{a}))$ and let $D' = h(D)$ where h is a homomorphism. We must show $D' \models \forall \bar{x}(R(x_1, \ldots, x_n) \to \varphi(\bar{x}, h(\bar{a}))$. Let $\bar{b} = (b_1, \ldots, b_n)$ be a tuple such that $D' \models R(\bar{b})$. As $D' = h(D)$, there are $c_1, \ldots, c_n$ in adom(D) such that $\bar{b} = h(\bar{c})$ (i.e., $b_i = h(c_i)$ for each $i \in \{1, \ldots, n\}$) and $D \models R(c_1, \ldots, c_n)$. Since the x_is are pairwise distinct, this means that $D \models R(x_1, \ldots, x_n)$ under any valuation sending x_i to c_i for each $i \leqslant n$. By $D \models \forall \bar{x}(R(x_1, \ldots, x_n) \to \varphi(\bar{x}, \bar{a}))$, we conclude that $D \models \varphi(\bar{c}, \bar{a})$ and so, by the inductive hypothesis, $D' \models \varphi(h(\bar{c}), h(\bar{a}))$, which implies $D' \models \forall \bar{x}(R(x_1, \ldots, x_n) \to \varphi(\bar{x}, h(\bar{a}))$.

The case of the equality atom in the guarded formula is exactly the same as the above case of the relational atom. This concludes the proof of Proposition 5.1. □

We remark that the condition that the variables x_is be pairwise distinct is essential. Consider, for example, a formula $\varphi = \forall x\ (R(x, x) \to S(x))$, and databases D and D' so that R is interpreted as $\{(1, 2)\}$ in D, as $\{(3, 3)\}$ in D', and S is empty in both. Then $D \models \varphi$, while $D' \models \neg\varphi$, even though $D' = h(D)$ under the homomorphism h that sends both 1 and 2 to 3.

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 5.2. *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 [Libkin 2011] 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, under other semantics, 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 8, once we have looked at other semantics to which such lifting results will apply as well.

6. SEMANTIC ORDERINGS

In this section we study semantic orderings arising from the usual relational semantics of incompleteness. We recall known results about the study of such orderings in the context of Codd databases [Buneman et al. 1991; Libkin 1995; Ohori 1990; Rounds 1991]. Such results are of two kinds: they connect orderings based on incompleteness with well-known orderings from the field of programming semantics, and they describe those via elementary *updates* that increase the information content of an instance.

Codd databases. SQL uses a single value `null` for missing information. As comparisons of a null with other values in SQL do not evaluate to true (technically, they evaluate to *unknown*, as SQL uses three-valued logic), this is properly modeled by a special kind of naïve databases, called *Codd databases*, in which nulls do not repeat.

For tuples $t = (a_1, \ldots, a_n)$ and $t' = (a'_1, \ldots, a'_n)$ over $\mathsf{Const} \cup \mathsf{Null}$ in which nulls do not repeat, we write $t \sqsubseteq t'$ if $a_i \in \mathsf{Const}$ implies $a'_i = a_i$. The meaning is that t' is at least as informative as t. There are two standard ways of lifting $\sqsubseteq$ to sets:

$$D \sqsubseteq^{\mathrm{H}} D' \;\Leftrightarrow\; \forall t \in D\, \exists t' \in D' : t \sqsubseteq t'$$
$$D \sqsubseteq^{\mathrm{P}} D' \;\Leftrightarrow\; \forall t' \in D'\, \exists t \in D : t \sqsubseteq t' \text{ and } D \sqsubseteq^{\mathrm{H}} D'$$

Superscripts H and P stand for Hoare and Plotkin, who first studied these orderings in the context of the semantics of concurrent processes, cf. [Gunter 1992].

These had been previously accepted as the correct orderings to represent the OWA and the CWA semantics over Codd databases [Buneman et al. 1991; Libkin 1995; Ohori 1990; Rounds 1991]. This can be justified by considering updates that affect informativeness of incomplete databases. Consider, for example, two tuples $(1, 2)$ and $(2, 2)$, and assume that we 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 $\{(1, 2), (2, 2)\}$ by replacing the null twice. In the case of OWA, one can have updates that add arbitrary new tuples.

Let D be a database, R a relation in it, t a tuple, and i a position in that tuple that contains a null $\bot$. Then by $D[v/R(t.i)]$ we mean D in which that occurrence of $\bot$ is replaced by $v \in \mathsf{Const} \cup \mathsf{Null}$, and by $D^+[v/R(t.i)]$ we mean D to which a tuple obtained from t by replacing the occurrence of $\bot$ in the ith position with v is added (i.e., the original t is retained). Now we consider updates $D \rightarrowtail^{\mathrm{codd}} D'$ of two kinds:

- Codd CWA updates: $D \rightarrowtail^{\mathrm{codd}}_{\mathrm{CWA}} D[v/R(t.i)]$ and and $D \rightarrowtail^{\mathrm{codd}}_{\mathrm{CWA}} D^+[v/R(t.i)]$;
- OWA update: $D \rightarrowtail^{\mathrm{codd}}_{\mathrm{OWA}} D \cup R(t)$ that adds a tuple to a relation in a database.

It is known [Libkin 1995] that the reflexive-transitive closure

- of $\rightarrowtail^{\mathrm{codd}}_{\mathrm{CWA}} \cup \rightarrowtail^{\mathrm{codd}}_{\mathrm{OWA}}$ is exactly $\sqsubseteq^{\mathrm{H}}$; and
- of $\rightarrowtail^{\mathrm{codd}}_{\mathrm{CWA}}$ is exactly $\sqsubseteq^{\mathrm{P}}$,

over Codd databases. Our next goal is to describe orderings corresponding to OWA and CWA for naïve databases, and to give an update semantics for them.

Naïve databases. Firstly we describe the semantic orderings $\preceq_*$ given by the semantics $[\![\,]\!]_*$, where $*$ is OWA, CWA, or WCWA. They are characterized via database homomorphisms as follows (the first item was already shown in [Libkin 2011]).

PROPOSITION 6.1. *$D \preceq_{\mathrm{OWA}} D'$ (respectively $D \preceq_{\mathrm{CWA}} D'$ or $D \preceq_{\mathrm{WCWA}} D'$) iff there is a database homomorphism (respectively, strong onto, or onto database homomorphism) from D to D'.*

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

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

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

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 $(E, D) \in \mathcal{R}^{\text{rdb}}_{\text{val}} \circ \mathcal{R}_{\text{sem}}$, 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, i.e.,

— Under OWA: $h(D) \subseteq E$, therefore $h'(D) = i(h(D)) \subseteq i(E) = D'$;
— Under CWA: $h(D) = E$, therefore $h'(D) = i(h(D)) = i(E) = D'$;
— 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:

— Under OWA, h' is a database homomorphism $D \to D'$;
— Under CWA, h' is a database strong onto homomorphism $D \to D'$;
— Under WCWA, h' is a database onto homomorphism $D \to D'$.

Conversely assume that there exists a database $*$-homomorphism $D \to D'$ where $*$ is "arbitrary", if $[\![\,]\!] =$ OWA; "strong onto" $[\![\,]\!] =$ CWA; and "onto" if $[\![\,]\!] =$ WCWA. Note 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''. Note 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 $D \preceq_{[\![\,]\!]} D'$. □

Next, we provide update justification for these orderings. OWA updates just add tuples as before; we denote them by $\rightarrowtail_{\text{OWA}}$. CWA updates are different, to account for repetition of nulls. In particular, once a null is replaced by some value v, *all* its occurrences must be replaced. Formally, if $\bot$ is a null that occurs in D, then $D[v/\bot]$ is D in which $v \in \mathsf{Const} \cup \mathsf{Null}$ replaces $\bot$ everywhere. The CWA update is now an update $D \rightarrowtail_{\text{CWA}} D[v/\bot]$.

THEOREM 6.2. *The transitive-reflexive closure of $\rightarrowtail_{\text{CWA}}$ is $\preceq_{\text{CWA}}$; and the transitive-reflexive closure of $\rightarrowtail_{\text{CWA}} \cup \rightarrowtail_{\text{OWA}}$ is $\preceq_{\text{OWA}}$.*

In other words, D is less informative than D' iff D' is obtained from D by a sequence of CWA updates, under CWA, and both CWA and OWA updates, under OWA.

These results will be shown inside the proof of Theorem 7.1 in the next Section.

What are the orderings $\preceq_{\text{OWA}}$ and $\preceq_{\text{CWA}}$ when we restrict them to Codd databases? One would expect them to be $\sqsubseteq^{\text{H}}$ and $\sqsubseteq^{\text{P}}$, corresponding to OWA and CWA for the Codd semantics, but this is only partly true. In fact, [Libkin 2011] 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 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 $\llparenthesis\ \rrparenthesis$ brackets.

We start with a semantics defined as follows:

$$\llparenthesis D \rrparenthesis_{\text{CWA}} \ = \ \{h_1(D) \cup \ldots \cup h_n(D) \ \mid \ h_1, \ldots, h_n \text{ are valuations}, n \geqslant 1\}.$$

That is, $D' \in \llparenthesis D \rrparenthesis_{\text{CWA}}$ 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 semantic.

Next, we describe the ordering $\Sqsubset_{\text{CWA}}$ induced by this semantics: that is, $D \Sqsubset_{\text{CWA}} D'$ iff $\llparenthesis D' \rrparenthesis_{\text{CWA}} \subseteq \llparenthesis D \rrparenthesis_{\text{CWA}}$).

To updates used as the justification of orderings in the previous section, we now add a new type. A *copying* CWA *update* is of the form

$$D \twoheadrightarrowtail_{\text{CWA}} D[v/\bot] \cup D^{\text{fresh}},$$

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.

It turns out that the ordering $\Sqsubset_{\text{CWA}}$ can be seen as a sequence of regular and copying CWA updates, and that when restricted to Codd databases, it coincides precisely with $\sqsubseteq^{\text{P}}$. That is, we have the following.

THEOREM 7.1.

- *$D \Sqsubset_{\text{CWA}} D'$ iff there exists a set of database homomorphisms $h_1, \ldots, h_n$ defined on D so that $D' = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$.*
- *The transitive-reflexive closure of $\rightarrowtail_{\text{CWA}} \cup \twoheadrightarrowtail_{\text{CWA}}$ is $\Sqsubset_{\text{CWA}}$.*
- *Over Codd databases, $\Sqsubset_{\text{CWA}}$ and $\sqsubseteq^{\text{P}}$ coincide.*

PROOF. We first show the first item of the Theorem. Let D and D' be two relational instances such that $D \Sqsubset_{\text{CWA}} D'$, i.e., $\llparenthesis D' \rrparenthesis_{\text{CWA}} \subseteq \llparenthesis D \rrparenthesis_{\text{CWA}}$. Let $E \in \llparenthesis D' \rrparenthesis_{\text{CWA}}$ be an instance having a bijection $b : \text{adom}(E) \to \text{adom}(D')$ which is the identity on $\mathsf{Const}(D)$ and such that $b(E) = D'$. By $E \in \llparenthesis D \rrparenthesis'_{\text{CWA}}$, also $E \in \llparenthesis D \rrparenthesis_{\text{CWA}}$ and so there exists a set of valuations $h_1, \ldots, h_n$ with $n \geqslant 1$ such that $E = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$. It follows that $D' = \bigcup\{b \circ h_i(D) \mid 1 \leqslant i \leqslant n\}$ where the $b \circ h_i$'s are database homomorphisms.

Conversely assume that there exists a set of database homomorphisms $h_1, \ldots, h_n$ defined on D so that $D' = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$. Remark that the existence of a set of database homomorphism is a transitive relation, i.e. if there exists a set of database homomorphism from D to D' and a set of database homomorphism from D' to D'', then there exists a set of database homomorphism from D to D''. Remark also that

$\llparenthesis D' \rrparenthesis_{\mathrm{CWA}}$ is precisely the set of complete relational instance E such that there exists a set of database homomorphisms from D' to E. Then, by transitivity, there exists a set of database homomorphisms from D to each $E \in \llparenthesis D' \rrparenthesis_{\mathrm{CWA}}$. Hence $E \in \llparenthesis D \rrparenthesis_{\mathrm{CWA}}$ for all $E \in \llparenthesis D' \rrparenthesis_{\mathrm{CWA}}$. In other words, $\llparenthesis D' \rrparenthesis_{\mathrm{CWA}} \subseteq \llparenthesis D' \rrparenthesis_{\mathrm{CWA}}$, and therefore $D \sqsubseteq_{\mathrm{CWA}} D'$.

We will show the second item of the Theorem last and so we show now its last item. Let D and D' be two Codd databases. Assume $D \sqsubseteq_{\mathrm{CWA}} D'$, i.e., there exists a set of homomorphisms $h_1, \dots, h_n$ from D so that $D' = \bigcup\{h_i(D) \mid 1 \leqslant i \leqslant n\}$. So for every tuple $(a_1, \dots, a_m) \in D$, there is some $1 \leqslant i \leqslant n$ such that $(h_i(a_1), \dots, h_i(a_m)) \in D'$, i.e., $(a_1, \dots, a_m) \sqsubseteq (h_i(a_1), \dots, h_i(a_m))$. It follows that $D \sqsubseteq^{\mathrm{H}} D'$. Similarly for every tuple $(b_1, \dots, b_m) \in D'$, there exists i such that $(b_1, \dots, b_m) \in h_i(D)$, which entails that there is $(a_1, \dots, a_m) \in D$ such that for every $1 \leqslant j \leqslant m$, $h_i(a_j) = b_j$ and so $(a_1, \dots, a_m) \sqsubseteq (b_1, \dots, b_m)$. It follows that $D \sqsubseteq^{\mathrm{P}} D'$.

Conversely, assume $D \sqsubseteq^{\mathrm{P}} D'$. 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 $H_t = t'_1, \dots, t'_k$ 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 $t \in D$, $H_t = t'_1, \dots, t'_m$ for some $t'_1, \dots, t'_m \in D'$. For every $1 \leqslant i \leqslant m$, we can now put:

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

Observe that by $D \sqsubseteq^{\mathrm{P}} D'$, $\bigcup_{1 \leqslant i \leqslant m} D'_i = D'$. 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 $D \sqsubseteq^{\mathrm{P}} D'$, h_i is a homomorphism and moreover $h_i(D) = D_i$. It follows that $D \sqsubseteq_{\mathrm{CWA}} D'$.

We finally show the last item of the Theorem. We first show $\twoheadrightarrow^*_{\mathrm{CWA}} = \sqsubseteq_{\mathrm{CWA}}$.

⇒ Let $D \sqsubseteq_{\mathrm{CWA}} D'$, i.e., there exists a set of homomorphisms $h_1, \dots, h_n$ from D so that $D' = \bigcup_{1 \leqslant j \leqslant n} h_j(D)$. Now let $\{\bot_1, \dots, \bot_k\}$ be the set of nulls occurring in D. We inductively define a sequence $D_0 \twoheadrightarrow_{\mathrm{CWA}} D_1 \twoheadrightarrow_{\mathrm{CWA}} \dots \twoheadrightarrow_{\mathrm{CWA}} D_k$ of $\twoheadrightarrow_{\mathrm{CWA}}$-updates of length k where $D_0 = D$ and for all $1 \leqslant i \leqslant k$:

$$D_i = \bigcup_{1 \leqslant j \leqslant n} D_{i-1}[h_j(\bot_i)/\bot_i]$$

Observe that $D_k = D'$ entails $D \twoheadrightarrow^*_{\mathrm{CWA}} D'$ and assume as inductive hypothesis that $D_k = D'$ whenever $k \leqslant m$. Now let $k = m + 1$ be the number of nulls occurring in D. Let also $D^c = D[c/\bot_{m+1}]$ be the result of substituting a fresh constant c for $\bot_{m+1}$ everywhere in D and let $D^{c'} = \bigcup_{1 \leqslant j \leqslant n} h_j(D^c)$. Homomorphisms being always the identity on constants, each $h_j : D_c \to h_j(D^c)$ is also a homomorphism and so $D^c \sqsubseteq_{\mathrm{CWA}} D^{c'}$. Now by inductive hypothesis, $D^{c'} = D^c_m$, where $D^c_m = \bigcup_{1 \leqslant j \leqslant n} D^c_{m-1}[h_j(\bot_m)/\bot_m]$. It follows immediately that $\bigcup_{1 \leqslant j \leqslant n} h_j(D^c[\bot_{m+1}/c]) = \bigcup_{1 \leqslant j \leqslant n} D^c_m[h_j(\bot_{m+1})/c]$. Finally, as $\bigcup_{1 \leqslant j \leqslant n} h_j(D^c[\bot_{m+1}/c]) = \bigcup_{1 \leqslant j \leqslant n} h_j(D) = D'$ and as $\bigcup_{1 \leqslant j \leqslant n} D^c_m[h_j(\bot_{m+1})/c] = D_{m+1}$, it follows that $D_{m+1} = D'$.

⇐ Assume $D \twoheadrightarrow^*_{\mathrm{CWA}} D'$. So there is a set of nulls $\{\bot_1, \dots, \bot_k\}$, a set of ordered sequences of constants and nulls $\{S_1, \dots, S_k\}$ (i.e., sequences over $\mathsf{Const} \cup \mathsf{Null}$) and a sequence $D_0 \twoheadrightarrow_{\mathrm{CWA}} D_1 \twoheadrightarrow_{\mathrm{CWA}} \dots \twoheadrightarrow_{\mathrm{CWA}} D_k$ of $\twoheadrightarrow_{\mathrm{CWA}}$-updates of length k where $D_0 = D$,

$D' = D_k$ and for all $1 \leqslant i \leqslant k$:

$$D_i = \bigcup_{x \in S_i} D_{i-1}[x/\bot_i].$$

Without loss of generality we can assume that there is some m big enough so that for every $1 \leqslant i \leqslant k$ there exist some $x^i_1, \ldots, x^i_m$ such that $S_i = x^i_1, \ldots, x^i_m$. Indeed, take m to be the length of the longest S_i sequence. If there is some $S_j = x^j_1, \ldots, x^j_n$ with $n < m$, then we can simply add to S_j a sequence of identical elements $x^j_{n+1}, \ldots, x^j_m$ all equal to x^j_n without altering the construction, i.e., we will obtain exactly the same database D_j by replacing multiple times the null $\bot_j$ by the same element x^j_n. The reason for that is simply that $D_{j-1}[x^j_n/\bot_j] \cup D_{j-1}[x^j_n/\bot_j] = D_{j-1}[x^j_n/\bot_j]$.
Out of this sequence of $\twoheadrightarrowtail_{\text{CWA}}$-updates of length k, we will now construct for every $0 \leqslant i \leqslant k$ and for every $1 \leqslant j \leqslant m^i$ a family of homomorphisms h^i_j's from D to D_i so that $D_i = \bigcup_{1 \leqslant j \leqslant m^i} h^i_j(D)$, which will entail in particular that $D' = \bigcup_{1 \leqslant j \leqslant m^k} h^k_j(D)$, i.e., $D \sqsubseteq_{\text{CWA}} D'$ and will achieve the proof of $\twoheadrightarrowtail^*_{\text{CWA}} = \sqsubseteq_{\text{CWA}}$.
We construct the h^i_j's by induction on k, first ordering them in a sibling-ordered tree of depth k and rank m to ease the construction. We start by defining h^0_1 and use it to label the root of the tree. We then label the rest of the nodes so that each homomorphism h^i_j 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^i_j in function of some previously defined homomorphism lying at depth $i - 1$. Now for each h^i_j with $i \neq 0$, observe that there is a unique r and a unique s such that h^i_j is the r^{th} child of h^{i-1}_s. We can now proceed to defining the h^i_j's. We let h^0_1 be the identity and for all $i \neq 0$ we let h^i_j be exactly as its parent h^{i-1}_s, except that it assigns the value x^i_r to all the preimages of $\bot_i$ by h^{i-1}_s.
We now show the correctness of the construction. Assume as inductive hypothesis that for all $i < k$, the following property holds:

- for every $1 \leqslant j \leqslant m^i$, $h^i_j : D \to D_i$ is a homomorphism and moreover $D_i = \bigcup_{1 \leqslant j \leqslant m^i} h^i_j(D)$.

(Notice in particular that the property holds trivially for $i = 0$.) We now derive that it also holds for $i = k$. For each $1 \leqslant j \leqslant m^k$, the fact that $h^k_j : D \to D_k$ is a homomorphism follows from the fact that $h^{k-1}_s : D \to D_{k-1}$ is a homomorphism (recall that h^k_j is the r^{th} child of h^{k-1}_s). Indeed, h^k_j is exactly as its parent h^{k-1}_s, except that it assigns the value x^k_r to all the preimages of $\bot_k$ by h^{k-1}_s. So $h^k_j(D) = D_{k-1}[x^k_r/\bot_k]$, which by assumption is a subinstance of D_k. But given that $D_{k-1} = \bigcup_{1 \leqslant j \leqslant m^{k-1}} h^{k-1}_j(D)$, this also implies that $D_k = \bigcup_{1 \leqslant j \leqslant m^k} h^k_j(D)$.

Observe now that a CWA update $D \rightarrowtail_{\text{CWA}} D[v/\bot]$ can be seen as a special case of multiple CWA update $D \twoheadrightarrowtail_{\text{CWA}} \bigcup\{D[v/\bot] \mid v \in S\}$ where S is a singleton. The proof of $\twoheadrightarrowtail^*_{\text{CWA}} = \sqsubseteq_{\text{CWA}}$ then 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 special case where $n = 1$ for every $D_i = \bigcup_{1 \leqslant j \leqslant n} D_{i-1}[h_j(\bot_i)/\bot_i]$, while showing $\rightarrowtail^*_{\text{CWA}} \subseteq \preceq_{\text{CWA}}$ amounts to restricting in the second direction to the special case where the length of the longest S_i sequence is $m = 1$.

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 homomorphism h from D and a sequence of $(\rightarrowtail_{\mathrm{OWA}} \cup \rightarrowtail_{\mathrm{CWA}})^*$ updates $D \rightarrowtail_{\mathrm{CWA}} \ldots \rightarrowtail_{\mathrm{CWA}} h(D) \rightarrowtail_{\mathrm{OWA}} \ldots \rightarrowtail_{\mathrm{OWA}} D'$ where all the OWA updates are performed last. Adding new tuples to $h(D)$ does not alter the tuples in it and so $h(D)$ is a subinstance of D', i.e., $D \preceq_{\mathrm{OWA}} D'$. Finally it can be shown using a similar reasoning that $(\rightarrowtail_{\mathrm{OWA}} \cup \rightarrowtail_{\mathrm{CWA}})^* = (\rightarrowtail_{\mathrm{OWA}} \cup \mathrel{\rightarrowtail\!\!\!\twoheadrightarrow}_{\mathrm{CWA}})^*$, which achieves the proof of the Theorem.

□

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 $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}_{\mathrm{val}}, \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$ of relations with $\boldsymbol{\mathcal{R}}_{\mathrm{val}} \subseteq \mathcal{D} \times 2^{\mathcal{C}}$ and $\boldsymbol{\mathcal{R}}_{\mathrm{sem}} \subseteq 2^{\mathcal{C}} \times \mathcal{C}$. 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 $\boldsymbol{\mathcal{R}}_{\cup} = \{(\mathcal{X}, X) \mid X = \bigcup \mathcal{X}\}$. The semantics given by $\boldsymbol{\mathcal{R}}$ is again the composition of two relations: $D' \in \llbracket D \rrbracket_{\boldsymbol{\mathcal{R}}}$ iff $D'(\boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}})D$.

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 $\llbracket\ \rrbracket_{\boldsymbol{\mathcal{R}}}$ is given by $\boldsymbol{\mathcal{R}}$ if both relations are total, relation $\boldsymbol{\mathcal{R}}_{\mathrm{val}}$ equals id_{ℓ} when restricted to $\mathcal{C}$, relation $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ contains id_r, and $D' \in \llbracket D \rrbracket_{\boldsymbol{\mathcal{R}}}$ iff $D(\boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}})D'$. Previously we just used identity instead of id_{ℓ} and id_r.

We say that $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ is transitive if $\boldsymbol{\mathcal{R}}_{\mathrm{sem}} \circ \mathsf{id}_{\ell} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}} \subseteq \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$. Note that $\boldsymbol{\mathcal{R}}_{\cup}$ is transitive. Now we have an analog of Proposition 4.1.

PROPOSITION 7.2. *A pair $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}_{\mathrm{val}}, \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$ gives rise to a fair database domain if $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ is transitive.*

PROOF. We prove a more general necessary and sufficient condition for fairness:

LEMMA 7.3. *A powerset semantics given by $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}_{\mathrm{val}}, \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$ gives rise to a fair database domain iff $\boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}} \circ \mathsf{id}_{\ell} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}} \subseteq \boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$. In particular if $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ is transitive then the database domain is fair.*

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

(1) $c \in \llbracket c \rrbracket_{\boldsymbol{\mathcal{R}}}$.
Indeed we know $(c, \{c\}) \in \boldsymbol{\mathcal{R}}_{\mathrm{val}}$ and $(\{c\}, c) \in \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$, then $c \in \llbracket c \rrbracket_{\boldsymbol{\mathcal{R}}}$.

(2) $c \in \llbracket x \rrbracket_{\boldsymbol{\mathcal{R}}}$ implies $\llbracket c \rrbracket_{\boldsymbol{\mathcal{R}}} \subseteq \llbracket x \rrbracket_{\boldsymbol{\mathcal{R}}}$.
Indeed if $c \in \llbracket x \rrbracket_{\boldsymbol{\mathcal{R}}}$ there exists $y \subseteq \mathcal{C}$ such that $(x, y) \in \boldsymbol{\mathcal{R}}_{\mathrm{val}}$ and $(y, c) \in \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$. Moreover if $c' \in \llbracket c \rrbracket_{\boldsymbol{\mathcal{R}}}$ then $(c, c') \in \mathsf{id}_{\ell} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ (because $\boldsymbol{\mathcal{R}}_{\mathrm{val}}$ is id_{ℓ} when restricted to $\mathcal{C}$). Hence $(x, c') \in \boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}} \circ \mathsf{id}_{\ell} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$. This implies $(x, c') \in \boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$, and therefore $c' \in \llbracket x \rrbracket_{\boldsymbol{\mathcal{R}}}$.

By Proposition 3.2, the database domain is fair.

Conversely assume that the database domain is fair, and $(x, c) \in \boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}} \circ \mathsf{id}_{\ell} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$, then there exist c' such that $(x, c') \in \boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ and $(c', c) \in \mathsf{id}_{\ell} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$. Then $c' \in \llbracket x \rrbracket_{\boldsymbol{\mathcal{R}}}$ and $c \in \llbracket c' \rrbracket_{\boldsymbol{\mathcal{R}}}$ (because $\boldsymbol{\mathcal{R}}_{\mathrm{val}}$ coincides with id_{ℓ} over $\mathcal{C}$). Then by fairness, $c \in \llbracket x \rrbracket_{\boldsymbol{\mathcal{R}}}$, and hence $(x, c) \in \boldsymbol{\mathcal{R}}_{\mathrm{val}} \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$. □

Proposition 7.2 immediately follows from the lemma above. □

Preservation for powerset semantics. Our next goal is to understand how we can make naïve evaluation work under the powerset semantics. For the standard semantics of incompleteness, we related naïve evaluation to preservation of queries under homomorphisms. We shall do the same here, but the setting for homomorphisms will be a bit different.

Recall that before we looked at relational semantics defined by two relations, relation $\mathcal{R}_{\text{val}}^{\text{rdb}} = \{(D, v(D)) \mid v \text{ is a valuation}\}$ and relation $\mathcal{R}_{\text{sem}}$ between complete databases. Now we deal with relations $\boldsymbol{\mathcal{R}}_{\text{val}}$ and $\boldsymbol{\mathcal{R}}_{\text{sem}}$. The natural powerset-based analog of $\mathcal{R}_{\text{val}}^{\text{rdb}}$ is the relation

$$\boldsymbol{\mathcal{R}}_{\text{val}}^{\text{rdb}} = \{(D, \{v_1(D), \ldots, v_n(D)\}) \mid v_i\text{'s are valuations}\}.$$

Hence, we now look at the semantics where the valuation relations are $\boldsymbol{\mathcal{R}}_{\text{val}}^{\text{rdb}}$, and thus the semantics is determined by $\boldsymbol{\mathcal{R}}_{\text{sem}}$ (e.g., by $\boldsymbol{\mathcal{R}}_{\cup} = \{(\mathcal{X}, X) \mid X = \bigcup \mathcal{X}\}$).

Consider complete relational databases D and D'. An $\boldsymbol{\mathcal{R}}_{\text{sem}}$-*homomorphism* between D and D' is a set $\{h_1, \ldots, h_n\}$ of mappings defined on $\text{adom}(D)$ so that $\{h_1(D), \ldots, h_n(D)\}\boldsymbol{\mathcal{R}}_{\text{sem}}D'$. Note that if $n = 1$, this is exactly the notion of $\mathcal{R}_{\text{sem}}$-homomorphisms seen earlier. The connection between naïve evaluation and homomorphism preservation now extends to powerset semantics.

PROPOSITION 7.4. *For every powerset semantics given by a relation $\boldsymbol{\mathcal{R}}_{\text{sem}}$, naïve evaluation works for a generic Boolean query Q iff Q is preserved under $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms.*

PROOF. We prove the proposition by proving some slightly more general results which will be useful later, when dealing with other forms of powerset semantics.

We start by defining a notion of $\approx$-equivalence for powerset semantics. This is the analog of the notion of $\approx$-equivalence (and strong $\approx$-equivalence) introduced for proving Proposition 4.3.

If $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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 4.5:

LEMMA 7.5. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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 $\{\mathcal{H}_1, \ldots \mathcal{H}_n, \ldots\}$, 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 $(D, \{h_1(D), \ldots h_k(D)\})$ such that D is a complete instance and $\{h_1, \ldots h_k\} \in \boldsymbol{\mathcal{M}}(D)$. 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 $\{h_1, \ldots h_k\}$ from D to D' which belongs to $\boldsymbol{\mathcal{M}}(D)$.

The following claim follows directly from definitions:

CLAIM 2. *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 7.5 we have:

COROLLARY 7.6. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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. As-*

sume 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 is preserved under $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms.

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 $y \in \mathcal{X}$.

Similarly 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 $\boldsymbol{\mathcal{R}}_{\boldsymbol{\mathcal{M}}} = \mathcal{P}(\mathcal{R}_{\mathcal{M}})$.

We now consider a special case when $\boldsymbol{\mathcal{R}}_{\text{val}} = \mathcal{P}(\mathcal{R}_{\text{val}})$.

LEMMA 7.7. *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 7.6, a generic Boolean query is weakly monotone iff it is preserved under $\boldsymbol{\mathcal{M}}$-$\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms. □

We are now ready to prove Proposition 7.4. Remark that $\boldsymbol{\mathcal{R}}_{\text{val}}^{\text{rdb}} = \mathcal{P}(\mathcal{R}_{\text{val}}^{\text{rdb}})$. Moreover by Lemma 4.7, if $\mathcal{M} = \text{all}$, then $\mathcal{R}_{\mathcal{M}}$ is strongly $\approx$-equivalent to $\mathcal{R}_{\text{val}}^{\text{rdb}}$. 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 7.7 that, for every powerset semantics given by a relation $\boldsymbol{\mathcal{R}}_{\text{sem}}$, a generic Boolean query is weakly monotone iff it is preserved under $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms. Now note that, under all relational semantics given by a relation $\boldsymbol{\mathcal{R}}_{\text{sem}}$ the database domain has the saturation property. Then the statement of Proposition 7.4 follows from Theorem 3.1. □

Let us now look at the semantics $(\!|\ |\!)_{\text{CWA}}$ given by relation $\boldsymbol{\mathcal{R}}_{\cup}$. The notion of preservation under $\boldsymbol{\mathcal{R}}_{\cup}$-homomorphisms is 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 (possibly the equality relation), then $\forall\bar{x}\ (R(\bar{x}) \rightarrow \varphi(\bar{y}))$ is in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$.

LEMMA 7.8. *Sentences in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$ are preserved under unions of strong onto homomorphisms.*

PROOF. We first define the notion of preservation under unions of strong onto homomorphisms for non-Boolean queries. This will allow us to prove the preservation property by structural induction on formulas in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$.

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 $\mathrm{adom}(D)$), we say that Q is preserved under unions of strong onto homomorphisms if, whenever there exists a union of strong onto homomorphisms $\{h_1 \ldots h_n\}$ from an instance D to an instance D', and $\bar{a} \in Q(D)$, then $h_i(\bar{a}) \in Q(D')$, for all $i \in 1, \ldots, k$.

To prove Lemma 7.8, we show that formulae in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\mathrm{bool}}$ are preserved under unions of strong onto homomorphisms. We proceed by structural induction on the formula φ. If φ =*false* or φ =*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 $D' = h_1(D) \cup \cdots \cup h_k(D)$ one has that $D' \models \varphi(h_i(\bar{a}))$ for all $i = 1, \ldots, k$.

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 $D' = h_1(D) \cup \cdots \cup h_k(D)$. Then $D \models \varphi'(b, \bar{a})$ for some value $b \in \mathrm{adom}(D)$. Since φ' is preserved under unions of strong onto homomorphisms, $D' \models \varphi'(h_i(b), h_i(\bar{a}))$ for each $i \in 1, \ldots, k$. Thus $D' \models \exists y \varphi'(y, h_i(\bar{a}))$, i.e. $D' \models \varphi(h_i(\bar{a}))$, for each $i \in 1, \ldots, k$.

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

This also concludes the proof of Lemma 7.8. □

Combining with Proposition 7.4, we get the following result.

COROLLARY 7.9. *If Q is a Boolean query from the class $\exists\mathsf{Pos} + \forall\mathsf{G}^{\mathrm{bool}}$, then naïve evaluation works for Q under the $(\!|\ |\!)_{\mathrm{CWA}}$ semantics.*

Semantics similar to $(\!|\ |\!)_{\mathrm{CWA}}$ did appear in the literature. In fact, the closest comes from the study of CWA in the context of data exchange [Arenas et al. 2010]. It was presented in [Hernich 2011] (and based in turn on a semantics from [Minker 1982]), and essentially boils down to the $(\!|\ |\!)_{\mathrm{CWA}}$ semantics, but based on a restricted notion of valuations, namely minimal valuations. We shall study those in Section 10.

8. LIFTING 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 $\mathrm{adom}(D)^k$. It is *generic* if, for each one-to-one map $f : \mathrm{adom}(D) \rightarrow \mathsf{Const} \cup \mathsf{Null}$, 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 *weakly monotone* if $Q^{\mathsf{C}}(D) \subseteq Q^{\mathsf{C}}(D')$ whenever $D' \in [\![D]\!]$.

We will need a stronger form of saturation property. A relational database domain is *strongly saturated* if every database has "sufficiently" many complete instances in its semantics that are isomorphic to it. More precisely, for each database D, and each finite set $C \subset \mathsf{Const}$, there is an isomorphic instance $D' \in [\![D]\!]$ such that both the isomorphism from D to D' and its inverse are the identity on C.

If we deal, as before, with relational semantics given by pairs $\mathcal{R} = (\mathcal{R}^{\text{rdb}}_{\text{val}}, \mathcal{R}_{\text{sem}})$, 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.

LEMMA 8.1. *Let $\mathbb{D}$ be a relational database domain with the strong saturation property, and Q a k-ary generic query. Then the following are equivalent:*

(*1*) *naïve evaluation works for Q;*
(*2*) *Q is weakly monotone; and*
(*3*) *(if the semantics is given by a relation $\mathcal{R}_{\text{sem}}$): Q is weakly preserved under $\mathcal{R}_{\text{sem}}$-homomorphisms.*

We postpone the proof of Lemma 8.1 until Section 11 where it will be proved together with its analog for minimal semantics.

One can develop similar transfer techniques for powerset semantics. Specifically Lemma 8.1 remains true if one replaces (3) by:

(*3*) *(if the semantics is given by a relation $\boldsymbol{\mathcal{R}}_{\text{sem}}$): Q is weakly preserved under $\boldsymbol{\mathcal{R}}_{\text{sem}}$-homomorphisms*.

In addition, one can then check that for all the classes of FO formulae considered here, preservation results hold when extended to formulae with free variables. One can then conclude that all the results remain true for non-Boolean queries.

THEOREM 8.2. *Let Q be a k-ary FO query, $k \geqslant 0$. 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.
—*If Q is in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$, then naïve evaluation works for Q under* $(\!|\ |\!)_{\text{CWA}}$.

PROOF. One can easily verify that all relational semantics given by a relation $\mathcal{R}_{\text{sem}}$ (respectively $\boldsymbol{\mathcal{R}}_{\text{sem}}$) have the strong saturation property. Moreover every k-ary FO query is generic. Then using Lemma 8.1 we have

CLAIM 3. *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 showed (see proofs of Proposition 5.1 and Lemma 7.8) that k-ary formulae of $\mathsf{Pos} + \forall\mathsf{G}$ are preserved under strong onto homomorphisms, and k-ary formulae of $\exists\mathsf{Pos} + \forall\mathsf{G}^{\mathrm{bool}}$ 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 way defined in the proof of Proposition 5.1.

Now notice that, for all these notions of homomorphism, preservation of k-ary formulae implies weak preservation. Then the statement of Theorem 8.2 immediately follows. □

9. NON-SATURATED DOMAINS

So far we dealt with saturated domains, those in which every object x has an isomorphic object y in its semantics: $y \in [\![x]\!]$ and $y \approx x$. While the semantics allowing arbitrary valuations of nulls are such, there are others. Such semantics, originating in AI, restrict possible valuations to minimal ones, i.e., valuations that produced results that cannot be smaller by using other valuations instead. We shall formally define and study them in Section 10. For now, our goal is to see what happens with non-saturated semantics, since all the equivalences we used previously require that domains be saturated.

The key idea is that to recover all the results, we need two conditions:

- the existence of a saturated subdomain, which we shall call a *representative* set, and
- the existence of a *canonical* function selecting a representative for each element of the domain.

Recall that a database domain was defined as a structure $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$, where $\mathcal{D}$ is a set and $\mathcal{C}$ one of its subset, $[\![\,]\!]$ is a function from $\mathcal{D}$ to nonempty subsets of $\mathcal{C}$, and $\approx$ is an equivalence relation on $\mathcal{D}$. A set $\mathcal{S} \subseteq \mathcal{D}$ is *representative* if

- $\mathcal{C} \subseteq \mathcal{S}$ (it contains all complete objects);
- $\mathcal{S}$ is saturated, i.e., 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).

Over relational database domains, if moreover $\mathcal{S}$ is strongly saturated, we say that S is a *strong representative* set.

In all the examples encountered so far we had $\mathcal{S} = \mathcal{D}$, but as we just said (and will study in detail in the following section) this need not always be the case.

If $\mathcal{S} \neq \mathcal{D}$, the equivalence between naïve evaluation and weak monotonicity need not work any more. However, we have the following generalization.

THEOREM 9.1. *Let $\mathbb{D}$ be a database domain with a representative set $\mathcal{S}$, and Q a generic Boolean query. Then naïve evaluation works for Q iff Q is weakly monotone and $Q(x) = Q(\chi_{\mathcal{S}}(x))$ for every $x \in \mathcal{D}$.*

PROOF. Theorem 9.1 follows immediately from the lemma below. We say that naïve evaluation works over $\mathcal{D}' \subseteq \mathcal{D}$ if $\mathsf{certain}(Q, x) = Q(x)$ for every $x \in \mathcal{D}'$.

LEMMA 9.2. *Let $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ 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 $\mathcal{S} = \mathcal{D}$ (i.e. if $\mathbb{D}$ is saturated) 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. □

This ends the proof of Theorem 9.1. □

Thus, our recipe for finding out when naïve evaluation works continues to apply, but with one extra condition: the query (Boolean, in this case), should not distinguish between an object x and its representative $\chi_{\mathcal{S}}(x)$ in $\mathcal{S}$.

Immediately from the above theorem, we have:

COROLLARY 9.3. *Let $\mathbb{D}$ be a database domain with a representative set $\mathcal{S}$, and Q a generic Boolean query. Then naïve evaluation works for Q over $\mathcal{S}$ iff Q is weakly monotone over $\mathcal{S}$.*

Thus, for instances restricted to those in the representative set, our previous recipe applies without any changes. We shall next see an example of non-saturated semantics where representative instances are a well known object, namely cores.

10. MINIMAL VALUATIONS 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 [Hernich 2011], based on earlier work in the area of logic programming [Minker 1982], proposed a powerset semantics that is based on *minimal* valuations. We now introduce it in our context (as [Hernich 2011] defined it in the context of data exchange). In this section, we again only look at Boolean queries, and in Section 11 we show how to lift results to non-Boolean ones.

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.

Not every valuation (or homomorphism) is minimal. Consider an incomplete table $D = \{(\bot, \bot), (\bot, \bot')\}$ and a valuation $v(\bot) = 1,\ v(\bot') = 2$. This is not minimal: take for instance $v'(\bot) = v'(\bot') = 1$ and we have $v'(D) \subsetneq v(D)$. The valuation v' is minimal.

The semantics of [Hernich 2011] is defined as

$$(\!|D|\!)^{\min}_{\text{CWA}} \ = \ \{\bigcup_{h \in \mathcal{H}} h(D) \ \mid \ \mathcal{H} \text{ is a nonempty set of } D\text{-minimal valuations}\}.$$

This is a powerset-based semantics, and the semantic relation it uses is the union relation $\mathcal{R}_{\cup}$, the same as in Section 7. However the valuation relation is no longer

$\boldsymbol{\mathcal{R}}^{\mathrm{rdb}}_{\mathrm{val}}$, allowing all valuations, but rather $\boldsymbol{\mathcal{R}}^{\min}_{\mathrm{val}}$ containing all 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 $\mathcal{R}^{\min}_{\mathrm{val}} = \{(D, h(D)) \mid h \text{ is a } D\text{-minimal valuation}\}$. For the identity relation, playing the role of $\mathcal{R}_{\mathrm{sem}}$ for CWA, this gives us

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

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

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

The fact that we no longer allow all valuations makes the equivalence of naïve evaluation and preservation of $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$-homomorphisms invalid. However, we can apply the results of the previous section to recover results on naïve evaluation. The main goal is then to find out what the representative sets are. This is what we do next.

10.1. 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 $h : D \to D'$ but there is no homomorphism $g : D \to D''$ 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 $\mathrm{core}(D)$. A structure is called a core if $D = \mathrm{core}(D)$. The cores are commonly used over graphs [Hell and Nešetřil 2004]; here we use them with the database notion of homomorphism that preserves constants (for which all results about cores remain true [Fagin et al. 2005]).

Even if minimal homomorphisms are related to cores, their images cannot be described precisely in terms of cores, as shown next. We strengthen results given in several examples in [Hernich 2011] (where constants were used in an essential way):

PROPOSITION 10.1. *If h is D-minimal, then $h(D)$ is a core and $h(D) = h(\mathrm{core}(D))$. However, there is a core D and a homomorphism h defined on it so that $h(D)$ is a core, but h is not D-minimal. This also holds if both D and $h(D)$ contain only nulls, and if D is a graph.*

PROOF. 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 $\mathrm{core}(D)$. Clearly $h \circ h_{\mathrm{core}}$ is a database homomorphism on D and $h_{\mathrm{core}}(h(D)) = h(\mathrm{core}(D)) \subsetneq h(D)$. Again this contradicts the D-minimality of h.

We now prove that there exists a core D and a database homomorphism $h : \mathrm{adom}(D) \to \mathsf{Null}$ 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

$\bot_1$	$\bot_1$	$\bot_2$	$\bot_3$
$\bot_4$	$\bot_5$	$\bot_2$	$\bot_2$

$h(D)$

$\bot_6$	$\bot_6$	$\bot_7$	$\bot_7$
$\bot_6$	$\bot_7$	$\bot_7$	$\bot_7$

where $h :\ \bot_1 \to \bot_6,\ \bot_2 \to \bot_7,\ \bot_3 \to \bot_7,\ \bot_4 \to \bot_6,\ \bot_5 \to \bot_7$

It is easy to check that both D and $h(D)$ are cores. However h is not D-minimal. In fact there exists a mapping $h' :\ \bot_1 \to \bot_6,\ \bot_2 \to \bot_7,\ \bot_3 \to \bot_7,\ \bot_4 \to \bot_6,\ \bot_5 \to \bot_6$ 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 $[\![D]\!]^{\min}_{\text{CWA}} \neq [\![\text{core}(D)]\!]_{\text{CWA}}$. 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, $C_3^{\mathsf{C}} + C_2^{\mathsf{C}} \notin [\![D]\!]^{\min}_{\text{CWA}}$. □

Proposition 10.1 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 10.2. *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 $\boldsymbol{\mathcal{R}} = (\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$.

PROOF. In order to easily work with minimal homomorphisms we extend the D-minimality notion to arbitrary mappings, and prove some technical facts about D-minimal 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 valuation, resp.) is a database homomorphism (valuation, resp.) which is also a D-minimal mapping.

We now prove a technical lemma about minimal mappings.

LEMMA 10.3. *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 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 : 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 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. □

We are now ready to prove the theorem. Indeed we prove the following more general proposition:

PROPOSITION 10.4. *If a relational semantics is given by a pair* $(\mathcal{R}_{\text{val}}^{\min}, \mathcal{R}_{\text{sem}})$ *or* $(\boldsymbol{\mathcal{R}}_{\text{val}}^{\min}, \boldsymbol{\mathcal{R}}_{\text{sem}})$*, the set* $\mathcal{S}$ *of cores is a strong representative set, and* $\chi_{\mathcal{S}}(D) = core(D)$ *for every instance* D*.*

PROOF. We first prove that for a semantics $[\![\,]\!]$ given by $(\mathcal{R}_{\text{val}}^{\min}, \mathcal{R}_{\text{sem}})$ the set of cores is a strong representative set.

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 [Hernich 2011] (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 10.3 with $D = D' = E$ and conclude that v is D-minimal.

Thus $(D, v(D)) \in \mathcal{R}_{\text{val}}^{\min}$ 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. $(D, D') \in \mathcal{R}_{\text{val}}^{\min}$ iff $(\text{core}(D), D') \in \mathcal{R}_{\text{val}}^{\min}$, for all D'. This will imply $[\![D]\!] = [\![core(D)]\!]$. We observe that this property has been proved in [Hernich 2011] (Lemma 6.9) restricted to the canonical solution in data exchange and its core.

Assume first that $(D, D') \in \mathcal{R}_{\text{val}}^{\min}$, then there exists a D-minimal valuation h such that $D' = h(D)$. We know by Proposition 10.1 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 $(core(D), D') \in \mathcal{R}_{\text{val}}^{\min}$.

Conversely assume that $(\text{core}(D), D') \in \mathcal{R}_{\text{val}}^{\min}$, then there exists a $\text{core}(D)$-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 $\text{core}(D)$-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. We need to extend this result to powerset semantics given by $(\boldsymbol{\mathcal{R}}_{\text{val}}^{\min}, \boldsymbol{\mathcal{R}}_{\text{sem}})$.

Recall that we say that $\boldsymbol{\mathcal{R}} = \mathcal{P}(\mathcal{R})$ if $\boldsymbol{\mathcal{R}}$ consists of precisely the pairs $(x, \mathcal{X})$ such that $\mathcal{X} \neq \emptyset$ and $(x, y) \in \mathcal{R}$ for all $y \in \mathcal{X}$. Remark that $\boldsymbol{\mathcal{R}}_{\text{val}}^{\min} = \mathcal{P}(\mathcal{R}_{\text{val}}^{\min})$. We have:

CLAIM 4. *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 $\boldsymbol{\mathcal{R}}_{\mathrm{val}} = \mathcal{P}(\mathcal{R}_{\mathrm{val}})$.

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

The following lemma easily follows:

LEMMA 10.5. *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* $(\mathcal{R}_{\mathrm{val}}, \mathsf{id})$*. Then* $\mathcal{S}$ *is a strong representative set also under any powerset semantics given by* $(\mathcal{P}(\mathcal{R}_{\mathrm{val}}), \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$*.*

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 $\mathcal{R}_{\mathrm{val}}$ to precisely the same instances. Then by Claim 4, D and $\chi_{\mathcal{S}}(D)$ have the same semantics under $(\mathcal{P}(\mathcal{R}_{\mathrm{val}}), \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$.

We further know that for all $D \in \mathcal{S}$ and for all $K \subseteq \mathsf{Const}$ there exists D' with $(D, D') \in \mathcal{R}_{\mathrm{val}}$ and a bijection $i : \mathsf{adom}(D) \to \mathsf{adom}(D')$ with $i(D) = D'$ such that i and i^{-} are the identity on K. Again by Claim 4, $(D, D') \in \mathcal{P}(\mathcal{R}_{\mathrm{val}}) \circ \boldsymbol{\mathcal{R}}_{\mathrm{sem}}$.

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

The lemma above implies that the set of cores is a strong representative set also under any powerset semantics given by $(\boldsymbol{\mathcal{R}}^{\min}_{\mathrm{val}}, \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$. This concludes the proof of Proposition 10.4 and of Theorem 10.2. □

Recall that a generic Boolean query Q is weakly monotone under $(\!|\ |\!)^{\min}_{\mathrm{CWA}}$ if $Q(D) = 1$ and $D' \in (\!|D|\!)^{\min}_{\mathrm{CWA}}$ imply $Q(D') = 1$. Immediately from Theorem 10.2 and Theorem 9.1, we obtain:

COROLLARY 10.6. *Let Q be a generic Boolean relational query. Then naïve evaluation works for Q under the* $[\![\ ]\!]^{\min}_{\mathrm{CWA}}$ *or the* $(\!|\ |\!)^{\min}_{\mathrm{CWA}}$ *semantics iff Q is weakly monotone (under the corresponding semantics), and* $Q(D) = Q(\mathrm{core}(D))$ *for every D.*

(Indeed by Theorem 10.2 and Theorem 9.1 the above corollary holds in general for arbitrary semantics given by $(\mathcal{R}^{\min}_{\mathrm{val}}, \mathcal{R}_{\mathrm{sem}})$ or $(\boldsymbol{\mathcal{R}}^{\min}_{\mathrm{val}}, \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$.)

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

10.2. 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 $\mathrm{fix}(h, D) = \{c \in \mathsf{Const}(D) \mid h(c) = c\}$. In the same way as for arbitrary mappings, h is called D-minimal if there is no homomorphism g with $\mathrm{fix}(h, D) \subseteq \mathrm{fix}(g, D)$ 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 $\mathrm{fix}(h, D) = \mathrm{fix}(g, D)$ whenever $f, g \in \mathcal{H}$, we have that $Q(D) = 1$ implies $Q(\bigcup\{h(D) \mid h \in \mathcal{H}\}) = 1$.

PROPOSITION 10.7. *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).*

PROOF. We derive the relationship between weak monotonicity and preservation for general semantics based on $\mathcal{R}^{\min}_{\text{val}}$ and $\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$. The proposition will follow as a special case.

Recall the notion of mapping type and $\approx$-equivalence used to prove Proposition 4.3 and Proposition 7.4. We now consider the mapping type $\mathcal{M} = \min$ which associates to each complete relational instance D the set of all D-minimal mappings $\mathsf{adom}(D) \to \mathsf{Const}$.

We prove the following lemma:

LEMMA 10.8. *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 10.3 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 $\mathsf{adom}(E) \backslash \text{fix}(h, E)$ 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 10.3, 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 $\mathsf{adom}(D)$ is D-minimal if each h_i is D-minimal and $\text{fix}(h_i, D) = \text{fix}(h_j, D)$ for all $i, j \in \{1, \ldots, k\}$. 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 $\mathsf{adom}(D) \to \mathsf{Const}$

LEMMA 10.9. *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 10.3 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 $i \in \{1, \ldots, k\}$).

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 $j \in \{1, \ldots, k\}$ are all D-minimal, by Lemma 10.3. 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 $j = \{1, \ldots, k\}$. It follows that $(D, \mathcal{X}) \in \boldsymbol{\mathcal{R}}^{\min}_{\text{val}}$. □

$\mathcal{M}$-$\mathcal{R}_{\text{sem}}$-homomorphisms with $\mathcal{M} = \min$ 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*.

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 $\boldsymbol{\mathcal{R}}_{\text{sem}} = \boldsymbol{\mathcal{R}}_{\cup}$.

Using Corollary 4.6 with $\mathcal{M} = \min$, and Corollary 7.6 with $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ we then have:

COROLLARY 10.10. *If a relational semantics is given by a pair* $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$ *(or* $(\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$*, 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.*

Proposition 10.7 is a special case of Corollary 10.10 where $\mathcal{R}_{\text{sem}}$ is the identity and $\boldsymbol{\mathcal{R}}_{\text{sem}} = \boldsymbol{\mathcal{R}}_{\cup}$. □

Combining this with Corollary 10.6 and results in Section 5 and Section 7, and observing that minimal homomorphisms are a special case of strong onto homomorphisms, we obtain:

COROLLARY 10.11. *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, by Corollary 9.3, if we only need to compute queries on cores, then the condition is not necessary. More precisely, recall that 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 10.12. *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.

PROPOSITION 10.13. *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.*

PROOF. from Proposition 5.1 and Lemma 7.8, and the fact that minimal homomorphisms are a special case of strong onto homomorphisms, we have that queries in $\mathsf{Pos} + \forall\mathsf{G}$ (respectively in $\exists\mathsf{Pos} + \forall\mathsf{G}^{\text{bool}}$) are weakly monotone under the $[\![\ ]\!]^{\min}_{\text{CWA}}$ (respectively the $(\!|\ |\!)^{\min}_{\text{CWA}}$) semantics. By definition of weak monotonicity , if $Q(D) = 1$ then $Q(D') = 1$ for all D' in the semantics of D (under the corresponding semantics). Therefore the certain answer to Q over D is true. □

11. LIFTING TO NON-BOOLEAN QUERIES FOR MINIMAL SEMANTICS

To lift results to non-Boolean queries for minimal (or, more generally, non-saturated) semantics requires a bit more work than in the saturated case, but the results still hold.

Recall that a representative set $\mathcal{S}$ is called *strong* if $\mathcal{S}$ is also strongly saturated. 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.

With these concepts, we can lift results to non-Boolean queries.

LEMMA 11.1. *Let $\mathbb{D}$ be a 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 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 $(\mathcal{R}^{\min}_{\text{val}}, \mathcal{R}_{\text{sem}})$, 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.

Moreover Lemma 11.1 also holds for powerset semantics given by $(\boldsymbol{\mathcal{R}}^{\min}_{\text{val}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$. We now prove it together with its analog Lemma 8.1:

PROOF OF LEMMA 11.1 AND LEMMA 8.1. We prove a more general version of these results, also accounting for the possible presence of constants in queries. To this end 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 relational queries.

In order to relate the notions of naïve evaluation, weak monotonicity and preservation for k-ary queries, we proceed as follows. 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 weak 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 a 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 $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$, and a C-generic k-ary query Q over $\mathbb{D}$, we define $\mathbb{D}^* = \langle \mathcal{D}^*, \mathcal{C}^*, [\![\,]\!]^*, \approx^* \rangle$, 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 $i : \mathrm{adom}(D) \cup t \to \mathrm{adom}(D') \cup t'$ 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 5.

1) If $\mathbb{D}$ is fair, $\mathbb{D}^*$ is also fair;
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 weakly monotone iff Q is weakly monotone;
6) $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(D')$ iff for every k-tuple t over Const one has $Q^*(D, t) = Q^*(D', t)$;
7) 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 fair and consider $(D, t) \in \mathcal{C}^*$. Since $\mathbb{D}$ is fair, $D \in [\![D]\!]$. Then $(D, t) \in [\![(D, t)]\!]^*$. 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 fair $[\![D]\!] \subseteq [\![D']\!]$. Thus $[\![(D, t)]\!]^* \subseteq \{(E, t) \mid E \in [\![D']\!]\} = [\![(D', t)]\!]^*$.
By Proposition 3.2, $\mathbb{D}^*$ is fair.
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 on 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 saying 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 weakly monotone and consider $D, D' \in \mathcal{D}$ such that $D' \in [\![D]\!]$. We prove that $Q^{\mathsf{C}}(D) \subseteq Q^{\mathsf{C}}(D')$. By definition of $[\![\,]\!]^*$ we know that $(D', t) \in [\![(D,t)]\!]^*$, for all k-tuples t over Const. Since Q^* is weakly 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 weakly monotone and consider (D,t) and (D',t') in $\mathcal{D}^*$ such that $(D',t') \in [\![(D,t)]\!]^*$. Then $t' = t$ and $D' \in [\![D]\!]$. 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 immediately follows from the definition of Q^*.
7) 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 : adom$(D) \to$ 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 adom$(D) \cup t \to$ 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 and weak monotonicity over $\mathbb{D}^*$ and Q^*, we immediately get the following corollaries.

From Theorem 9.1 on $\mathbb{D}^*$ and Q^*, we have:

COROLLARY 11.2. *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 for Q if and only if*

— *Q is weakly monotone and*
— *$Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(\chi_{\mathcal{S}}(D))$ for all $D \in \mathcal{D}$*

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

This proves that (1) ⇔ (2) in Lemma 11.1, as well as in Lemma 8.1. We now need to prove the relationship between weak monotonicity and preservation for relational semantics based on $\mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$ and on $\mathcal{R}^{\mathrm{min}}_{\mathrm{val}}$ (as well as their powerset versions).

In the sequel we use the following additional notation.

If $\mathcal{H} = \{h_1, \ldots h_n\}$ is a set of mappings over 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 $i \in 1, \ldots, n$. 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 $\mathbb{D} = \langle \mathcal{D}, \mathcal{C}, [\![\,]\!], \approx \rangle$ be a relational database domain where $[\![\,]\!]$ is given by a pair $(\mathcal{R}_{\mathrm{val}}, \mathcal{R}_{\mathrm{sem}})$ (respectively $(\boldsymbol{\mathcal{R}}_{\mathrm{val}}, \boldsymbol{\mathcal{R}}_{\mathrm{sem}})$), 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 $\llbracket\,\rrbracket^*$ 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

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

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

Similarly $\boldsymbol{\mathcal{R}}^*_{\text{val}} = \{(\,(D,t),(\mathcal{X},t)\,) \mid (D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\text{val}}$ and t is a k-tuple over Const$\}$ and $\boldsymbol{\mathcal{R}}^*_{\text{sem}} = \{(\,(\mathcal{X},t),(D,t)\,) \mid (\mathcal{X},D) \in \boldsymbol{\mathcal{R}}_{\text{sem}}$ and t is a k-tuple over 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 6. *$((D,t),(D',t)) \in \mathcal{R}^*_{\mathcal{M}} \circ \mathcal{R}^*_{\text{sem}}$ (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 $C \cup t$.*

Recall that 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 7. *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 4.5 and Lemma 7.5 to the database domain $\mathbb{D}^*$ and the generic query Q^*. We obtain the following corollary

CLAIM 8. *If the semantics $\llbracket\,\rrbracket$ 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 $\llbracket\,\rrbracket$ 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 $\mathcal{M} = \text{min}$, as well as $\boldsymbol{\mathcal{M}} = \boldsymbol{all}$ (defined as $\mathcal{P}(\text{all})$) and $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$ for powerset semantics.

CLAIM 9.

1) *If $\llbracket\,\rrbracket$ is based on $\mathcal{R}_{\text{val}} = \mathcal{R}^{\text{rdb}}_{\text{val}}$ then $\mathcal{R}^*_{\mathcal{M}}$ is strongly $\approx^*$-equivalent to $\mathcal{R}^*_{\text{val}}$ for $\mathcal{M} = \text{all}$;*
2) *If $\llbracket\,\rrbracket$ is based on $\mathcal{R}_{\text{val}} = \mathcal{R}^{\text{min}}_{\text{val}}$ then $\mathcal{R}^*_{\mathcal{M}}$ is $\approx^*$-equivalent to $\mathcal{R}^*_{\text{val}}$ for $\mathcal{M} = \text{min}$;*
3) *If $\llbracket\,\rrbracket$ is based on $\boldsymbol{\mathcal{R}}_{\text{val}} = \boldsymbol{\mathcal{R}}^{\text{rdb}}_{\text{val}}$ (respectively $\boldsymbol{\mathcal{R}}_{\text{val}} = \boldsymbol{\mathcal{R}}^{\text{min}}_{\text{val}}$) 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 $(D,D') \in \mathcal{R}^{\text{rdb}}_{\text{val}}$ 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 $\mathsf{adom}(E) \cup t \to \mathsf{adom}(D) \cup t$, 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 $(D, D') \in \mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$. The instance D is obtained from E by replacing each element of $\mathsf{adom}(E)$ 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 $(D, D') \in \mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}}$. Note that D depends only on (E, t) (and not on h).

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

2) Consider a pair $((D, t), (D', t))$ where $(D, D') \in \mathcal{R}^{\min}_{\mathrm{val}}$ 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 10.3, 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 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 $(D, D') \in \mathcal{R}^{\min}_{\mathrm{val}}$. The instance D is obtained from E by replacing each element of $\mathrm{adom}(E)$ not occurring in $\mathrm{fix}(h, E)$ 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 $\mathrm{fix}(h, E)$. 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 $(C \cup t) \cap \mathrm{adom}(E) \subseteq \mathrm{fix}(h, E)$. 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 10.3 h' is D-minimal. Moreover $h'(D) = h(E) = D'$, and h' is the identity on $\mathrm{fix}(h, E)$. Now remark that $\mathsf{Const}(D) = \mathrm{fix}(h, E)$, therefore h' is a valuation on D. We then conclude that $(D, D') = (D, h'(D)) \in \mathcal{R}^{\min}_{\mathrm{val}}$.

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

Recall the notation $\mathcal{P}(\,)$ for powerset semantics. Notice that $(\boldsymbol{\mathcal{R}}^{\mathrm{rdb}}_{\mathrm{val}})^* = \mathcal{P}((\mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}})^*)$. We also know by the first item that $\mathcal{R}^*_{\mathcal{M}}$ is strongly $\approx^*$-equivalent to $(\mathcal{R}^{\mathrm{rdb}}_{\mathrm{val}})^*$ for $\mathcal{M} = \mathrm{all}$. Then by Lemma 7.7, $\mathcal{P}(\mathcal{R}^*_{\mathcal{M}})$, for $\mathcal{M} = \mathrm{all}$, is $\approx^*$-equivalent to $(\boldsymbol{\mathcal{R}}^{\mathrm{rdb}}_{\mathrm{val}})^*$. Now remark that for $\mathcal{M} = \mathrm{all}$ 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 $\boldsymbol{\mathcal{R}}_{\mathrm{val}} = \boldsymbol{\mathcal{R}}^{\min}_{\mathrm{val}}$, then $\boldsymbol{\mathcal{R}}^*_{\boldsymbol{\mathcal{M}}}$ is $\approx^*$-equivalent to $\boldsymbol{\mathcal{R}}^*_{\mathrm{val}}$ for $\boldsymbol{\mathcal{M}} = \boldsymbol{min}$.

Let $((D,t),(\mathcal{X},t)) \in (\boldsymbol{\mathcal{R}}_{\text{val}}^{\min})^*$; then $(D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\text{val}}^{\min}$. 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 10.3, 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 $\mathcal{X} = \{h_1(E), \dots h_n(E)\}$ where $\{h_1, \dots h_n\}$ 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 $(D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\text{val}}^{\min}$. Let $K = \text{fix}(h_i, E)$ (which is the same for all $i \in 1, \dots, n$).

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 $j \in 1, \dots, n$ are all D-minimal, by Lemma 10.3. 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 $j = 1, \dots, n$. It follows that $(D,\mathcal{X}) \in \boldsymbol{\mathcal{R}}_{\text{val}}^{\min}$.

□

We now combine the above two claims and get a characterization of weak monotonicity under both standard an minimal semantics:

COROLLARY 11.3. *Assume that a relational semantics is given by a pair* $(\mathcal{R}_{\text{val}}^{\text{rdb}}, \mathcal{R}_{\text{sem}})$*, (respectively* $(\boldsymbol{\mathcal{R}}_{\text{val}}^{\text{rdb}}, \boldsymbol{\mathcal{R}}_{\text{sem}})$*) 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*.*

COROLLARY 11.4. *Assume that a relational semantics is given by a pair* $(\mathcal{R}_{\text{val}}^{\min}, \mathcal{R}_{\text{sem}})$*, (respectively* $(\boldsymbol{\mathcal{R}}_{\text{val}}^{\min}, \boldsymbol{\mathcal{R}}_{\text{sem}})$*) 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*.*

The characterization of naïve evaluation in the above two corollaries is obtained by using Corollary 11.2 and the fact that semantics based on $\mathcal{R}_{\text{val}}^{\text{rdb}}$ as well as on $\boldsymbol{\mathcal{R}}_{\text{val}}^{\text{rdb}}$ are strongly saturated. Similarly semantics based on minimal valuations have a strong representative set, which is the set of cores (Proposition 10.4).

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

Fig. 1. Summary of naïve evaluation results for FO queries

Corollary 11.3 with $C = \varnothing$ completes the proof of Lemma 8.1. Similarly Corollary 11.4 with $C = \varnothing$ completes the proof of Lemma 11.1. □

Using Lemma 11.1, we can achieve the desired lifting result for minimal semantics, i.e. we can show that Corollary 10.11 continues to hold for k-ary FO queries.

THEOREM 11.5. *Let Q be a k-ary FO query such that $Q^{\mathsf{C}}(D) = Q^{\mathsf{C}}(\mathrm{core}(D))$ for all D.*

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

PROOF. The statement follows directly from Lemma 11.1, by recalling that $\mathcal{R}_{\mathrm{sem}}$ is the identity for $[\![\ ]\!]^{\min}_{\mathrm{CWA}}$ and therefore minimal $\mathcal{R}_{\mathrm{sem}}$-homomorphisms are just usual minimal homomorphisms. Similarly $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$ is $\boldsymbol{\mathcal{R}}_{\cup}$ for $(\!\mid\ \mid\!)^{\min}_{\mathrm{CWA}}$, and minimal $\boldsymbol{\mathcal{R}}_{\mathrm{sem}}$-homomorphisms are unions of minimal homomorphisms. By Proposition 5.1 and Lemma 7.8 the above fragments guarantee these preservation properties, and therefore the corresponding weak preservation properties. □

12. SUMMARY FUTURE WORK

The table in Figure 1 summarizes results on naïve evaluation for fragments of FO queries. The first line of course is the classical result of [Imielinski and Lipski 1984], proved to be optimal in [Libkin 2011]. Other results were shown using the methodology established here, that reduced naïve evaluation to monotonicity and preservation under homomorphisms.

There are several 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 [Abiteboul et al. 2006; Barceló et al. 2010; Gheerbrant et al. 2012] that can serve as a good starting point.

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 [Keisler 1965b] 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 [Atserias et al. 2006] which looked at bounded treewidth (but does not capture XML with data). We note in passing that [Atserias et al. 2006] 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. Constraints (e.g., keys and foreign keys) have a huge impact on the complexity of finding certain answers [Calì et al. 2003; Vardi 1986], so 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 [Imielinski and Lipski 1984] – that in general have higher complexity of query evaluation [Abiteboul et al. 1991] but are nonetheless useful in several applications [Arenas et al. 2011].

Applications. In applications such as data integration and exchange, finding certain answers is the standard query answering semantics [Arenas et al. 2010; Lenzerini 2002]. In fact one of our semantics came from data exchange literature [Hernich 2011]. 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 [Reiter 1977; 1982] 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. We thank reviews of the conference version of this paper for their helpful suggestions. Work partly supported by EPSRC grants G049165 and J015377.

REFERENCES

ABITEBOUL, S., HULL, R., AND VIANU, V. 1995. *Foundations of Databases*. Addison-Wesley.

ABITEBOUL, S., KANELLAKIS, P., AND GRAHNE, G. 1991. On the representation and querying of sets of possible worlds. *Theoretical Computer Science 78,* 1, 158–187.

ABITEBOUL, S., SEGOUFIN, L., AND VIANU, V. 2006. Representing and querying XML with incomplete information. *ACM Transactions on Database Systems 31,* 1, 208–254.

AJTAI, M. AND GUREVICH, Y. 1987. Monotone versus positive. *Journal of the ACM 34,* 4, 1004–1015.

ARENAS, M., BARCELÓ, P., LIBKIN, L., AND MURLAK, F. 2010. *Relational and XML Data Exchange*. Morgan&Claypool Publishers.

ARENAS, M., PEREZ, J., AND REUTTER, J. 2011. Data exchange beyond complete data. In *Proceedings of the 30th ACM Symposium on Principles of Database Systems (PODS)*. 83–94.

ATSERIAS, A., DAWAR, A., AND KOLAITIS, P. 2006. On preservation under homomorphisms and unions of conjunctive queries. *Journal of the ACM 53,* 2, 208–237.

BARCELÓ, P., LIBKIN, L., POGGI, A., AND SIRANGELO, C. 2010. XML with incomplete information. *Journal of the ACM 58,* 1.

BUNEMAN, P., JUNG, A., AND OHORI, A. 1991. Using Powerdomains to Generalize Relational Databases. *Theoretical Computer Science 91,* 1, 23–55.

CALÌ, A., LEMBO, D., AND ROSATI, R. 2003. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *ACM Symposium on Principles of Database Systems (PODS)*. 260–271.

CHANG, C. AND KEISLER, H. 1990. *Model Theory*. North Holland.

COMPTON, K. 1983. Some useful preservation theorems. *Journal of Symbolic Logic 48,* 2, 427–440.

DATE, C. AND DARWIN, H. 1996. *A Guide to the SQL Standard*. Addison-Wesley.

FAGIN, R., KOLAITIS, P., AND POPA, L. 2005. Data exchange: getting to the core. *ACM Transactions on Database Systems 30,* 1, 174–210.

GHEERBRANT, A., LIBKIN, L., AND TAN, T. 2012. On the complexity of query answering over incomplete xml documents. In *International Conference on Database Theory (ICDT)*. 169–181.

GUNTER, C. 1992. *Semantics of Programming Languages: Structures and Techniques*. MIT Press.

HELL, P. AND NEŠETŘIL, J. 1992. The core of a graph. *Discrete Mathematics 109,* 1-3, 127–126.

HELL, P. AND NEŠETŘIL, J. 2004. *Graphs and Homomorphisms*. Oxford University Press.

HERNICH, A. 2011. Answering non-monotonic queries in relational data exchange. *Logical Methods in Computer Science 7,* 3.

IMIELINSKI, T. AND LIPSKI, W. 1984. Incomplete information in relational databases. *Journal of the ACM 31,* 4, 761–791.

KEISLER, H. J. 1965a. Finite approximations of infinitely long formulas. In *Symposium on the Theory of Models*. North Holland, 158–169.

KEISLER, H. J. 1965b. Some applications of infinitely long formulas. *Journal of Symbolic Logic 30,* 3, 339–349.

LENZERINI, M. 2002. Data integration: a theoretical perspective. In *Proceedings of the 21st ACM Symposium on Principles of Database Systems, PODS'02*. 233–246.

LIBKIN, L. 1995. A semantics-based approach to design of query languages for partial information. In *Semantics in Databases*. Lecture Notes in Computer Science Series, vol. 1358. Springer-Verlag, 170–208.

LIBKIN, L. 2011. Incomplete information and certain answers in general data models. In *ACM Symposium on Principles of Database Systems (PODS)*. 59–70.

MINKER, J. 1982. On indefinite databases and the closed world assumption. In *CADE*. 292–308.

OHORI, A. 1990. Semantics of types for database objects. *Theoretical Computer Science 76*, 53–91.

REITER, R. 1977. On closed world data bases. In *Logic and Data Bases*. 55–76.

REITER, R. 1982. Towards a logical reconstruction of relational database theory. In *On Conceptual Modelling*. 191–233.

ROSSMAN, B. 2008. Homomorphism preservation theorems. *Journal of the ACM 55,* 3.

ROUNDS, B. 1991. Situation-theoretic aspects of databases. In *Situation Theory and Applications*. CSLI Series, vol. 26. 229–256.

STOLBOUSHKIN, A. 1995. Finitely monotone properties. In *IEEE Symposium on Logic in Computer Science (LICS)*. 324–330.

VARDI, M. 1986. On the integrity of databases with incomplete information. In *ACM Symposium on Principles of Database Systems (PODS)*. 252–266.

Received Month Year; revised Month Year; accepted Month Year