Intuitionistic Data Exchange

Gösta Grahne, Ali Moallemi, and Adrian Onet

Concordia University, Montreal, Canada
grahne@cs.concordia.ca,moa_ali@encs.concordia.ca,adrian_onet@yahoo.com

Abstract. The field of Data Exchange has overcome many obstacles, but when it comes to negation in rule bodies combined with existentially quantified variables in rule heads, to unequalities $\neq$, as well as to inconsistency management, the same intractability barriers that plague the area of incomplete information arise.
In this paper we develop an intuitionistic relevance-logic based semantics that allows us to extend the polynomial time "naive evaluation" technique to Data Exchange dependencies (TGDs and EGDs) with negated atoms and unequalities, and to Data Exchange Target Queries consisting of unions of conjunctive queries with negation and unequalities. The semantics is also paraconsistent, and avoids the intractability barriers encountered in inconsistency management as well.

1 Introduction

The problem of *data exchange* poses one of the major challenges in distributed information processing environments. A connection in such an environment can be viewed as a labeled directed edge from a node representing a *source database* to a node representing a *target database*. The edge-label denotes a *schema mapping* that guides the middle-ware in restructuring the data from the source database to fit the requirements of the target database. Since its inception in 2003 by Fagin et al. in [8], the field of data exchange has been intensely investigated, and many functionalities are mature for technology transfer. In this paper we focus on the problems of

- computing and materializing target instances when the schema mapping is expressed as a set of embedded dependencies (TGDs and EGDs), and
- computing certain answers to unions of conjunctive queries expressed on the target schema (target UCQs).

These problems have been shown to admit efficient implementation, based on a property colloquially called the *naive evaluation property*. The property roughly says that the incompleteness of some domain values can be ignored, as long as these values are distinguished from each other, and from the "ordinary" domain values that denote named and known objects. The price to pay is that we have to restrict the schema-mappings and target queries to be monotone. Most attempts to include non-monotone features, such as negation ($\neg$) and unequality

($\neq$), soon run into intractability barriers, due to the underlying issues of incomplete information. For a comparison between different closed world semantics the reader should consult [22].

We recall (see [1]) that a *tuple generating dependency* (TGD) is a first order formula of the form $\forall \mathbf{XY} \left(\Phi(\mathbf{XY}) \rightarrow \exists \mathbf{Z}\, \Psi(\mathbf{XZ}) \right)$, where Φ and Ψ are conjunctions of relational atoms, and $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ are sequences of variables. We refer to Φ as the *body* of the dependency, and to Ψ as the *head*. In an *equality generating dependency* (EGD), there is no existential quantification, and Ψ is an equality $X_1 = X_2$, where X_1 and X_2 are variables from $\mathbf{XY}$.

When it comes to dependencies with negation in the body, the main proposals are the *stratified model* the *stable model* and the *well-founded model* (for definitions, see [1]). We argue that these semantics are not well suited for data exchange, mainly because they "favor" negative information. Sometimes this is desirable, as for example in the classical *Tweety*-example:

$$\forall X \left(BIRD(X), \neg PENGUIN(X) \rightarrow FLY(X) \right).$$

If the source database is $\{BIRD(\texttt{tweety})\}$ it might be desirable to "defeasibly" conclude $FLY(\texttt{tweety})$. However, if we use the same semantics for

$$\forall X \left(COUNTRY(X), \neg NUKES(X) \rightarrow FRIEND(X) \right),$$

and the source database is $\{COUNTRY\ (\texttt{sylvania})\}$, it might not be prudent to conclude $FRIEND(\texttt{sylvania})$.

Intuitionistic logic rejects the law of the excluded middle and non-constructive existence proofs. The motivation is usually epistemological, but it can also be computational, as in Kleene's intuitionistic logic based on recursive predicates [20]. In an intuitionistic approach we would assign both $FLY(\texttt{tweety})$ and $FRIEND(\texttt{sylvania})$ the value *unknown* or *undetermined*. This does not preclude the truth-value to later change, as was for example the case with (the knowledge of) the truth-value of the statement expressing Fermat's Last Theorem, that changed from *unknown* to *true* in 1994.

If non-constructive existence proofs are epistemologically susceptible, what can one say about defeasible reasoning? We contend that negative facts should be explicitly derived, just as the positive ones are. In other words, only if we have explicitly decided the fact $\neg PENGUIN(\texttt{tweety})$ can we draw the conclusion $FLY(\texttt{tweety})$. In this spirit we propose to use Nuel Belnap's four-valued relevance logic **R** [5] as foundation for negation in data exchange. This allows us to derive efficient algorithms for:

- computing and materializing target instances when the schema mapping is expressed as a set of TGDs with negated atoms in the body and in the head, and EGDs with negation in the body, and
- computing certain answers to target unions of conjunctive queries with negation and unequality ($\neq$).

Belnap's logic also is *paraconsistent*, meaning that "if we have inconsistent information about ducks, it is possible that our information about decimals can still be trusted" (see Fitting [10], page 10). This feature allows efficient inconsistency management, which is not in general possible in the *repair*-approach of [3].

The rest of this paper is organized as follows. The next section introduces the notions used throughout the paper. This is followed by the section dedicated to Belnap's Logic $\mathcal{FOUR}$ that plays a central role in this paper. Section 4 describes the "naive evaluation" of conjunctive queries with negation, and Section 5 adds negation to the bodies of tuple generating dependencies, in a way that allows us to extend the "naive chase" (i.e. the standard chase [8]). In Section 6 we study how our techniques affect the termination of the chase procedure. We end with conclusions in Section 7.

2 Preliminaries

For basic definitions and concepts we refer to [1]. Let $\mathbf{a} = a_1, a_2, \ldots, a_n$ and $\mathbf{b} = b_1, b_2, \ldots, b_m$ be *sequences* of elements from a semigroup (a set with a binary associative operation $\cdot$). Then $\mathbf{ab}$ denotes the *concatenation sequence* $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m$, and if $n = m$ we can form the *product sequence* $\mathbf{a} \cdot \mathbf{b} = a_1 \cdot b_1, a_2 \cdot b_2, \ldots, a_n \cdot b_n$. The *length* of a sequence $\mathbf{a} = a_1, a_2, \ldots, a_n$, denoted $|\mathbf{a}|$, is n. The *empty sequence* is denoted ϵ. Let $\mathbf{X} = X_1, X_2, \ldots, X_n$ be a sequence and A a set. Then a mapping $f : \mathbf{X} \to A^n$, where $f(X_i) = a_i$, is listed as $f = \{X_1/a_1, X_2/a_2, \ldots, X_n/a_n\}$.

Signatures, schemas, languages etc. A *signature* σ consists of a finite set $\mathsf{R} = \{R, S, \ldots\}$ of *relation symbols* (called the *schema*), a countably infinite set $\mathsf{Vars} = \{X, Y, Z, \ldots\}$ of *variables*, a countably infinite set $\mathsf{Nulls} = \{x, y, z, \ldots\}$ of *nulls*, and a countably infinite set $\mathsf{Cons} = \{a, b, c, \ldots\}$ of *constants*. We assume that Nulls and Cons are semigroups under the $\cdot$ operation. Each *relational symbol* $R \in \mathsf{R}$ has an associated natural number $ar(R)$, called the *arity* of R.

Let σ be a signature. The set of *well-formed formulas (wff's)* of the language $\mathcal{L}_\sigma$ is defined inductively by stating that each atom $R(\mathbf{X})$, where $R \in \mathsf{R}$ and $\mathbf{X} \in (\mathsf{Vars} \cup \mathsf{Cons})^{ar(R)}$, is a wff, and then closing the set under $\vee, \wedge, \neg, \exists$, and $\forall$. If $\Phi(\mathbf{X})$ is a $\mathcal{L}_\sigma$-formula, then $vars(\Phi)$ denotes its variables, the sequence $\mathbf{X}$ denotes the free variables, and $cons(\Phi)$ the constants in Φ. If $\Phi(\mathbf{X})$ is a formula and v is the mapping $\mathbf{X} \mapsto \mathbf{a}$, then $\Phi(\mathbf{a})$ denotes the sentence $\Phi(v(\mathbf{X}))$. If a wff Φ does not have any free variables, it is called a *sentence*.

Boolean valued instances. In algebraic semantics (see e.g. [23]) an interpretation of sentences in the object language is obtained through a homomorphism h into a structure $(\mathbf{A}, \varphi)$, where $\mathbf{A}$ is an algebra, and φ is a subset of the carrier set A of $\mathbf{A}$. The elements in φ represent "truth." The homomorphism maps atomic sentences to elements in A, and the connectives and quantifiers are mapped into operators in the algebra. Under interpretation h, a sentence Φ is then "true" in $\mathbf{A}$, if $h(\Phi) \in \varphi$.

The well known *Boolean valued* models [19] are obtained by taking the algebra to be a Boolean algebra $\mathbf{B} = (B, \vee, \wedge, \neg, 0, 1)$, and φ to be an ultrafilter of $\mathbf{B}$. Recall that an ultrafilter is a subset φ of B, such that $1 \in \varphi$; for all $b_1, b_2 \in B$, $b_1 \wedge b_2 \in \varphi$ iff $b_1 \in \varphi$ and $b_2 \in \varphi$; and for all $b \in B$ either $b \in \varphi$ or $\neg b \in \varphi$, but $\{b, \neg b\} \notin \varphi$. Adapting this approach to database instances (models), we get the following definitions.

Definition 1. *Let $\sigma = (\mathsf{R}, \mathsf{Vars}, \mathsf{Nulls}, \mathsf{Cons})$ be a database signature, and let $\mathbf{B} = (B, \vee, \wedge, \neg, 0, 1)$ be a Boolean algebra. A* $\mathbf{B}$-instance *$\mathcal{I}$ consists of the following parts:*

1. *A finite set of elements $dom(\mathcal{I}) \subseteq \mathsf{Cons} \cup \mathsf{Nulls}$, the* domain *of $\mathcal{I}$,*
2. *$cons(\mathcal{I}) = dom(\mathcal{I}) \cap \mathsf{Cons}$, the* constants *of $\mathcal{I}$,*
3. *The set $nulls(\mathcal{I}) = dom(\mathcal{I}) \cap \mathsf{Nulls}$, the* nulls *of $\mathcal{I}$, and*
4. *For each $R \in \mathbf{R}$ and $\mathbf{a} \in dom(\mathcal{I})^{ar(R)}$, an* interpretation *$R(\mathbf{a})^{\mathcal{I}} \in B$.*

Definition 2. *Let Φ be an $\mathcal{L}$-sentence, $\mathbf{B}$ a Boolean algebra, $\varphi \subset B$ an ultrafilter in $\mathbf{B}$, and $\mathcal{I}$ a $\mathbf{B}$-interpretation. Then the* satisfaction *relation $\mathcal{I} \models_{\mathbf{B},\varphi} \Phi$ is defined inductively as follows.*

1. *$\mathcal{I} \models_{\mathbf{B},\varphi} R(\mathbf{a})$, if $R(\mathbf{a})^{\mathcal{I}} \in \varphi$,*
2. *$\mathcal{I} \models_{\mathbf{B},\varphi} \Phi \wedge \Psi$, if $\Phi^{\mathcal{I}} \wedge \Psi^{\mathcal{I}} \in \varphi$,*
3. *$\mathcal{I} \models_{\mathbf{B},\varphi} \neg\Phi$, if $\neg(\Phi^{\mathcal{I}}) \in \varphi$,*
4. *$\mathcal{I} \models_{\mathbf{B},\varphi} \forall \mathbf{X}\Phi(\mathbf{X})$, if $\bigwedge_{\mathbf{a} \in dom(\mathcal{I})^{ar(R)}} (\Phi(\mathbf{a})^{\mathcal{I}}) \in \varphi$,*
5. *$\mathcal{I} \models_{\mathbf{B},\varphi} \Phi \to \Psi$, if $\Phi^{\mathcal{I}} \in \varphi$ entails that $\Psi^{\mathcal{I}} \in \varphi$.*

The *natural partial order* $\leq$ in $\mathbf{B}$ is defined as $\boldsymbol{\alpha} \leq \boldsymbol{\beta}$ iff $\boldsymbol{\alpha} \vee \boldsymbol{\beta} = \boldsymbol{\beta}$. The partial order can be lifted to $(\mathbf{B}, \varphi)$-instances $\mathcal{I}$ and $\mathcal{J}$ by stipulating that $\mathcal{I} \leq \mathcal{J}$ if $cons(\mathcal{I}) \subseteq cons(\mathcal{J})$, and there is a mapping $h : dom(\mathcal{I}) \to dom(\mathcal{J})$, identity on the constants, such that $R(\mathbf{a})^{\mathcal{I}} \leq R(h(\mathbf{a}))^{\mathcal{J}}$, for all $R \in \mathbf{R}$ and $\mathbf{a} \in dom(\mathcal{I})^{ar(R)}$. Such a mapping h is said to be a *homomorphism* from $\mathcal{I}$ to $\mathcal{J}$. It is easily shown that $\leq$ is a partial order on the equivalence classes generated by the relation $\mathcal{I} \simeq \mathcal{J}$, defined to hold iff $\mathcal{I} \leq \mathcal{J}$ and $\mathcal{J} \leq \mathcal{I}$.

Going back to the satisfaction relation $\models_{\mathbf{B},\varphi}$, note that if $B = \{0, 1\}$, then $\mathbf{B} = \mathbf{2}$ (the two-element Boolean algebra), and the unique ultrafilter is $\{1\}$. Thus we can write simply $\models_{\mathbf{2}}$, and $\mathcal{I} \models_{\mathbf{2}} \Phi$ becomes the standard two-valued semantics in which $\to$ behaves as material implication. Thus $\mathcal{I} \models_{\mathbf{2}} \Phi \to \Psi$ if and only if $\Phi^{\mathcal{I}} \leq \Psi^{\mathcal{I}}$. Here $\leq$ is the natural order, i.e. $0 \leq 1$, and $\mathcal{I} \leq \mathcal{J}$ means that there is a classical database homomorphism [1] from $\mathcal{I}$ to $\mathcal{J}$, and $\mathcal{I} \simeq \mathcal{J}$ means that $\mathcal{I}$ and $\mathcal{J}$ are homomorphically equivalent (in the classical database sense).

Conjunctive queries and certain answers. A *conjunctive query (CQ)* is a first order formula of the form $q = \exists \mathbf{Y}\Phi(\mathbf{XY})$, where Φ is a conjunction of atoms of $\mathcal{L}_\sigma$. For an atom R of arity k in Φ, for notational convenience we write $R(\mathbf{XY})$ instead of $R(x_{i_1}, \ldots, x_{i_k})$, where $x_{i_1}, \ldots, x_{i_k}$ is a subsequence of $\mathbf{XY}$. The free variables of Φ are those in $\mathbf{X}$. If $\mathbf{X} = \epsilon$, the expression denotes a *Boolean* CQ. A query q is called a *union of conjunctive queries (UCQ)* if it is of the form $q = \exists \mathbf{Y}(\Phi_1(\mathbf{XY}) \vee \ldots \vee \Phi_m(\mathbf{XY}))$, where each Φ_i is a conjunction of atoms. It is

a well known fact that UCQs are monotonic in the $0 \leq 1$ partial order, meaning that if $\mathcal{I} \leq \mathcal{J}$, then $q(\mathcal{I}) \leq q(\mathcal{J})$, for all UCQs q. It has further been shown in [14] that $\mathcal{I} \simeq \mathcal{J}$ if and only if $q(\mathcal{I}) \simeq q(\mathcal{J})$, for all UCQs q. Thus, if a user only has UCQs as query language, then the user cannot distinguish between $\simeq$-equivalent (homomorphically equivalent) instances. This leads to the important notion of *incomplete instances* and *certain answers*, instrumental in data exchange [8]. Conceptually, an *incomplete database instance* is a set $\mathcal{X}$ of ordinary instances $\mathcal{I}$. The information that is certain is the information that holds in *all* possible instances (possible worlds). This leads to the definition of the *certain answer* to a UCQ q on an incomplete instance $\mathcal{X}$ as $\bigwedge q(\mathcal{X}) = \bigwedge\{q(\mathcal{I}) : \mathcal{I} \in \mathcal{X}\}$. Here $\bigwedge$ denotes the $\leq$-greatest lower-bound (modulo $\simeq$) of a set of ordinary instances. Likewise, $\bigvee$ denotes the $\leq$-least upper-bound (modulo $\simeq$). The least upper-bound is clearly given by the disjoint (rename nulls apart) union. The lower bound $\bigwedge$ has been shown to exist by Hell and Nesetril [16], whose results were generalized by Libkin in [21]. In particular, Libkin showed that if $\mathcal{X}$ is a finite set of instances, then $\bigwedge(\mathcal{X})$ can always be represented as a finite instance (up to $\simeq$-equivalence).

Usually the certain answer is further restricted to only the named domain values (the constants). For an instance $\mathcal{I}$, define $\mathcal{I}\!\uparrow$ as function $\mathcal{I}$ restricted to sequences of objects in $cons(\mathcal{I})$. Thus, the usual certain answer is

$$Cert(q, \mathcal{X}) = (\bigwedge\{q(\mathcal{I}) : \mathcal{I} \in \mathcal{X}\})\!\uparrow . \tag{1}$$

Note that $q(\mathcal{X})$ is what Libkin [21] calls *answers as knowledge*, while (1) is his *answers as objects*. Since an instance $\mathcal{I}$ can contain nulls, it can be viewed as a "naive table" [18], that represents the set of complete (ground) instances $\mathcal{X}_{\mathcal{I}} = \{\mathcal{J} : \mathcal{I} \leq \mathcal{J} \text{ and } dom(\mathcal{J}) \subseteq \mathsf{Cons}\}$. Consider now an instance $\mathcal{I}$ as a naive table representing the set $\mathcal{X}_{\mathcal{I}}$ of possible worlds. Then the "naive evaluation theorem" says that $\bigwedge q(\mathcal{X}_{\mathcal{I}}) \simeq q(\mathcal{I})$. Consequently for a UCQ q, we have [18, 8]:

$$Cert(q, \mathcal{X}_{\mathcal{I}}) = \Big(q(\mathcal{I})\Big)\!\uparrow . \tag{2}$$

3 Belnap's logic $\mathcal{FOUR}$

We are interested in Belnap's intuitionistic logic $\mathcal{FOUR}$. This logic has a sound and complete axiomatization in terms of a system **R** of relevance logic, and $\Phi \to \Psi$ becomes relevant entailment of first degree (see [2, 5]).

The logic $\mathcal{FOUR}$ is a generalization of Kleene's strong three-valued logic. $\mathcal{FOUR}$ has four truth-values $\varnothing, \{\mathbf{f}\}, \{\mathbf{t}\}$, and $\{\mathbf{f}, \mathbf{t}\}$. Intuitively, assigning a sentence Φ the value $\{\mathbf{t}\}$ means that "the computer has been told that Φ is true," (see [5], page 11) and assigning the value $\{\mathbf{f}\}$ means that "the computer has been told that Φ is false." Furthermore, the value $\{\mathbf{f}, \mathbf{t}\}$ means that "the computer has been told that Φ is false, and the computer has been told that Φ is true." In other words, Φ is *inconsistent*. Finally, assigning Φ the value $\varnothing$ means that "the computer has not been told anything about the truth-value of Φ," that is, the truth-value of Φ is *unknown*.

Since truth-values are sets, they are naturally ordered by set-inclusion. The more elements the set contains, the more information the computer has about the truth. This gives rise to the *information ordering*, which we in the sequel will denote $\leq$. We can still retain the $\leq$-order, which is called the *truth order*, by stipulating that $\{\mathbf{f}\} \leq \{\mathbf{t}\}$, and that $\varnothing$ and $\{\mathbf{f}, \mathbf{t}\}$ are incomparable and both lie strictly between $\{\mathbf{f}\}$ and $\{\mathbf{t}\}$. We shall use the following symbols: $\mathtt{0}$ for $\{\mathbf{f}\}$, $\mathtt{1}$ for $\{\mathbf{t}\}$, $\bot$ for $\varnothing$, and $\top$ for $\{\mathbf{f}, \mathbf{t}\}$. Furthermore, $\top$ and $\bot$ are incomparable wrt $\leq$, and $\mathtt{0}$ and $\mathtt{1}$ are incomparable in $\leqslant$. All of this can be put together into a structure $\mathbf{4} = (4, \vee, \wedge, \curlyvee, \curlywedge, \neg, \mathtt{0}, \mathtt{1}, \bot, \top)$, where $4 = \{\bot, \top, \mathtt{0}, \mathtt{1}\}$. Here $\vee$ is the least upper bound in the $\leq$-order, and $\wedge$ the greatest lower bound (i.e. $\top \vee \bot = \mathtt{1}$, and $\top \wedge \bot = \mathtt{0}$, etc). The least upper bound in the $\leqslant$-order is $\curlyvee$, and the greatest lower bound is $\curlywedge$ (e.g. $\mathtt{0} \curlyvee \mathtt{1} = \top$, and $\mathtt{0} \curlywedge \mathtt{1} = \bot$, etc). For negation, $\neg \mathtt{1} = \mathtt{0}, \neg \mathtt{0} = \mathtt{1}, \neg \top = \top$, and $\neg \bot = \bot$. Consequently, $(4, \vee, \wedge, \neg, \mathtt{0}, \mathtt{1})$ is a Boolean algebra.

Belnap's structure $\mathbf{4}$ has subsequently been generalized by Ginsberg and Fitting to so called *bilattices* [12, 9]. The simplest non-trivial bilattice is $\mathbf{4}$, just as $\mathbf{2}$ is the simplest non-trivial Boolean algebra. Bilattices have been thoroughly investigated by Fitting and Avron, among others (see e.g. [9, **?**,**?**,**?**]). The algebra $\mathbf{4}$ has been shown to provide a principled and uniform account of various non-monotonic semantics of logic programming with negation, including the stable models and the well-founded model.

Since $\mathbf{4}$ is an algebra, it can be used to give meaning to $\mathcal{L}_\sigma$ sentences, by choosing the designated values φ. Arieli and Avron [4] have argued that φ should be a *ultrabifilter*[1] and they have shown that $\varphi = \{\top, \mathtt{1}\}$ is the unique ultrabifilter in $\mathbf{4}$. We shall thus work with the algebraic semantics $(\mathbf{4}, \{\top, \mathtt{1}\})$.

A $\mathbf{4}$-instance is then a $\mathbf{B}$-interpretation of Definition 1, where $\mathbf{B} = \mathbf{4}$, and by noticing that $(4, \vee, \wedge, \neg, \mathtt{0}, \mathtt{1})$ is a Boolean algebra, $\mathcal{I} \models_{\mathbf{4}} \Psi$ can be read from Definition 2 of $\models_{(\mathbf{B},\varphi)}$, by choosing $\mathbf{B} = \mathbf{4}$, and $\varphi = \{\top, \mathtt{1}\}$.

Let $\mathcal{I}$ and $\mathcal{J}$ be $\mathbf{4}$-instances, such that $cons(\mathcal{I}) \subseteq cons(\mathcal{J})$ and let h be a mapping from $dom(\mathcal{I})$ to $dom(\mathcal{J})$, such that h is identity on the constants. The mapping h is said to be a *homomorphism* from $\mathcal{I}$ to $\mathcal{J}$, provided that $R(\mathbf{a})^{\mathcal{I}} \leqslant R(h(\mathbf{a}))^{\mathcal{J}}$, for all $R \in \mathsf{R}$ and $\mathbf{a} \in dom(\mathcal{I})^{ar(R)}$. The partial order $\leqslant$ of $\mathbf{4}$ is lifted to $\mathbf{4}$-instances in the following definition.

Definition 3. *Let $\mathcal{I}$ and $\mathcal{J}$ be $\mathbf{4}$-instances such that $cons(\mathcal{I}) \subseteq cons(\mathcal{J})$. We say that $\mathcal{J}$ is* more informative *than $\mathcal{I}$, denoted $\mathcal{I} \leqslant \mathcal{J}$, if there is a homomorphism from $\mathcal{I}$ to $\mathcal{J}$. Furthermore, if both $\mathcal{I} \leqslant \mathcal{J}$ and $\mathcal{J} \leqslant \mathcal{I}$, we say that $\mathcal{I}$ and $\mathcal{J}$ are* information equivalent*, and denote it $\mathcal{I} \doteq \mathcal{J}$. Let $\mathscr{I}$ denote the set of all instances over σ. Then $\mathscr{I}/_{\doteq}$ denotes the set of equivalence classes of instances induced by $\doteq$.*

Clearly $\leqslant$ is a partial order on $\mathscr{I}/_{\doteq}$. Due to space limitation, the definitions for relations $\curlywedge$ and $\curlyvee$ are omitted, they can be found in the full version [13]. We need to verify that $\curlywedge$ and $\curlyvee$ indeed are the meet and join in the lattice induced by $\leqslant$.

[1] For a technical definition, see [4]

Lemma 1. *Let $\mathcal{I}$ and $\mathcal{J}$ be* **4***-instances. Then $\mathcal{I} \leq \mathcal{J}$ if and only if $\mathcal{I} \wedge \mathcal{J} \doteq \mathcal{I}$. and $\mathcal{I} \leq \mathcal{J}$ if and only if $\mathcal{I} \vee \mathcal{J} \doteq \mathcal{J}$.*

Reducing $\mathcal{FOUR}$ to $\mathcal{TWO}$. We now losslessy represent a **4**-instance as a two-valued instance as follows.

Definition 4. *1. Let $\mathcal{I}$ be a* **4***-instance over schema R. The* **2** -projection *of $\mathcal{I}$, denoted $\mathcal{I}^{\downarrow\mathbf{2}}$, is a two-valued Boolean instance over schema $\bigcup_{R\in\mathsf{R}}\{R^+, R^-\}$, where*

(a) $cons(\mathcal{I}^{\downarrow\mathbf{2}}) = cons(\mathcal{I})$, $nulls(\mathcal{I}^{\downarrow\mathbf{2}}) = nulls(\mathcal{I})$, and $dom(\mathcal{I}^{\downarrow\mathbf{2}}) = dom(\mathcal{I})$.

(b) For all relations $R \in \mathsf{R}$ and $\mathbf{a} = a_1, \ldots, a_{\alpha(R)}$, where $a_i \in dom(\mathcal{I})$ for all $i \in \{1, \ldots, \alpha(R)\}$, $R^+(\mathbf{a})^{\mathcal{I}^{\downarrow\mathbf{2}}} = \mathtt{1}$ if $R(\mathbf{a})^{\mathcal{I}} \geq \mathtt{1}$, and $\mathtt{0}$ otherwise; and $R^-(\mathbf{a})^{\mathcal{I}^{\downarrow\mathbf{2}}} = \mathtt{1}$ if $R(\mathbf{a})^{\mathcal{I}} \geq \mathtt{0}$, and $\mathtt{0}$ otherwise.

2. Let $\mathcal{I}$ be a **2***-instance. Then the* **4***-instance corresponding to $\mathcal{I}$ is $\mathcal{I}^{\uparrow\mathbf{4}}$, where*

(a) $cons(\mathcal{I}^{\uparrow\mathbf{4}}) = cons(\mathcal{I})$, $nulls(\mathcal{I}^{\uparrow\mathbf{4}}) = nulls(\mathcal{I})$, and $dom(\mathcal{I}^{\uparrow\mathbf{4}}) = dom(\mathcal{I})$.

(b) For all $R \in \mathsf{R}$ and $\mathbf{a} = a_1, \ldots, a_{\alpha(R)}$, for all $i \in \{1, \ldots, \alpha(R)\}$,

$$R(\mathbf{a})^{\mathcal{I}^{\uparrow\mathbf{4}}} = \begin{cases} \mathtt{T} \text{ if } R^+(\mathbf{a})^{\mathcal{I}} = \mathtt{1} \text{ and } R^-(\mathbf{a})^{\mathcal{I}} = \mathtt{1} \\ \mathtt{\bot} \text{ if } R^+(\mathbf{a})^{\mathcal{I}} = \mathtt{0} \text{ and } R^-(\mathbf{a})^{\mathcal{I}} = \mathtt{0} \\ \mathtt{1} \text{ if } R^+(\mathbf{a})^{\mathcal{I}} = \mathtt{1} \text{ and } R^-(\mathbf{a})^{\mathcal{I}} = \mathtt{0} \\ \mathtt{0} \text{ if } R^+(\mathbf{a})^{\mathcal{I}} = \mathtt{0} \text{ and } R^-(\mathbf{a})^{\mathcal{I}} = \mathtt{1} \end{cases}$$

4 UCQs with negation

The intuitionistic semantics $(\mathbf{4}, \{\mathtt{T}, \mathtt{1}\})$ allows us to extend the (polynomial time) "naive evaluation" from unions of conjunctive queries to unions of conjunctive queries with negation. We next define CQs with negation. UCQs with negation is defined similarly.

A *literal* is an expression of the form $R(\mathbf{X})$ or $\neg R(\mathbf{X})$, where $R(\mathbf{X})$ is an atom in $\mathcal{L}_\sigma$. A *conjunctive query with negation (NCQ)* is a first order formula of the form $q = \exists\mathbf{Y}\Phi(\mathbf{XY})$, where Φ is a conjunction of literals. Consider now an NCQ

$$q = \exists\mathbf{Y}\Big(\bigwedge_{i=1}^{n} R_i(\mathbf{XY}) \wedge \bigwedge_{i=n+1}^{n+m} \neg R_i(\mathbf{XY})\Big). \tag{3}$$

The NCQ q, when applied on an instance $\mathcal{I}$, extends the interpretation $\mathcal{I}$ to also include the relation Q, defined as follows.

Definition 5. *Let $\mathcal{I}$ be a* **4***-instance and $q = \exists\mathbf{Y}\Phi(\mathbf{XY})$ an NCQ of the form* (3)*. Let v range over all mappings from $\mathbf{XY}$ to $dom(\mathcal{I})$. Then*

$$Q(\mathbf{a})^{\mathcal{I}} = \bigvee_{\{v\,:\,v(\mathbf{X})=\mathbf{a}\}}\Big(\bigwedge_{i=1}^{n} R_i(v(\mathbf{XY}))^{\mathcal{I}} \wedge \bigwedge_{i=n+1}^{n+m} \neg\big(R_i(v(\mathbf{XY}))^{\mathcal{I}}\big)\Big),$$

for each $\mathbf{a} \in dom(\mathcal{I})^{|\mathbf{X}|}$.

We next show that NCQs are monotonic in the information order $\leq$. By a recent result of Gheerbrant et al. [11], such monotonicity is a necessary and sufficient condition for a class of queries to admit naive evaluation. Indeed, the following holds.

Theorem 1. *Let q be an NCQ, and let $\mathcal{I}$ and $\mathcal{J}$ be $\mathbf{4}$-instances, such that $\mathcal{I} \leq \mathcal{J}$ by homomorphism h. Then $Q(\mathbf{a})^{\mathcal{I}} \leq Q(h(\mathbf{a}))^{\mathcal{J}}$, for all $\mathbf{a} \in dom(\mathcal{I})^{ar(Q)}$.*

Furthermore, the naive evaluation property of CQs on $\mathbf{2}$-instances can now be extended to NCQs on $\mathbf{4}$-instances. It turns out that an NCQ q can be evaluated naively on $\mathcal{I}^{\downarrow\mathbf{2}}$ by turning q into a UCQs, denoted $q^{\downarrow\mathbf{2}}$. Assume q is of the form (3). Then

$$q^{\downarrow\mathbf{2}} = \exists\mathbf{Y}\Big(\Big(\bigwedge_{i=1}^{n} R_i^+(\mathbf{XY}) \wedge \bigwedge_{i=n+1}^{n+m} R_i^-(\mathbf{XY})\Big) \vee \bigvee_{i=1}^{n} R_i^-(\mathbf{XY}) \vee \bigvee_{j=n+1}^{m} R_j^+(\mathbf{XY})\Big). \quad (4)$$

In the next theorem we view an NCQ query q as mapping an instance $\mathcal{I}$ to the instance $q(\mathcal{I})$, where $q(\mathcal{I})$ is a $\mathbf{4}$-instance over schema $\{Q\}$, with $Q(\mathbf{a})^{q(\mathcal{I})} = Q(\mathbf{a})^{\mathcal{I}}$.

Theorem 2. *Let q be an NCQ and $\mathcal{I}$ a $\mathbf{4}$-instance. Then $q(\mathcal{I}) \doteq (\mathbf{q}^{\downarrow\mathbf{2}}(\mathcal{I}^{\downarrow\mathbf{2}}))^{\uparrow\mathbf{4}}$.*

Similarly to the two-valued case, we can regard a $\mathbf{4}$-instance $\mathcal{I}$ as a naive table defining the set $\mathcal{X}_{\mathcal{I}} = \{\mathcal{J} : \mathcal{I} \leq \mathcal{J} \text{ and } dom(\mathcal{J}) \subseteq \mathsf{Cons}\}$ of possible worlds. We can now extended the naive evaluation property to NCQs on $\mathbf{4}$-instances, by using the result (proved in the full version) that $\bigwedge q(\mathcal{X}_{\mathcal{I}}) \doteq q(\mathcal{I})$. In analogue with (1) we define the *certain answer* as $Cert_{\mathbf{4}}(q, \mathcal{X}) = (\bigwedge\{q(\mathcal{I}) : \mathcal{I} \in \mathcal{X}\})\restriction$. To wrap this section up formally, we conclude

Corollary 1. *Let q be an NCQ and $\mathcal{I}$ a $\mathbf{4}$-instance. Then $Cert_{\mathbf{4}}(q, \mathcal{X}_{\mathcal{I}}) = \Big((\mathbf{q}^{\downarrow\mathbf{2}}(\mathcal{I}^{\downarrow\mathbf{2}})^{\uparrow\mathbf{4}}\Big)\restriction$.*

Note that all the previous results hold as well for the larger class NUCQ of unions of conjunctive queries with negation. We have only presented the transformation of NCQs q into UCQs $q^{\downarrow\mathbf{2}}$. In a very similar fashion we can transform any NUCQ query into a UCQ, while preserving all theorems of this section, and in particular Corollary 1.

5 Data Exchange in $\mathcal{TWO}$ and $\mathcal{FOUR}$

A *tuple generating dependency (TGD)* is a first order formula having the form $\forall\mathbf{XY}(\Phi(\mathbf{XY}) \rightarrow \exists\mathbf{Z}\ R(\mathbf{XZ}))$. An *equality generating dependency (EGD)* is a first order formula of the form $\forall\mathbf{X}(\Phi(\mathbf{X}) \rightarrow X_1 = X_2)$, where X_1 and X_2 are variables in $\mathbf{X}$. A *TGD with negation and unequality (NTGD$^{\not\approx}$)* is a TDG that can have atomic negations in the body and in the head, and that also includes a special equality symbol $\approx$, possibly negated (denoted $\not\approx$). Note that NTGD$^{\approx}$s include the EGDs. Also, the notion of a $\mathbf{4}$-instance has to be extended to account

for the explicit equality. Details appear in the full paper [13]. The salient point is that an atom $a \approx b$ can have truth-values in $\mathbf{4}$, meaning that $(a \approx b)^{\mathcal{I}}$ can be *inconsistent* or *unknown*, in addition to *true* or *false*. Let $\mathbf{n} \in \{\mathbf{2}, \mathbf{4}\}$. A dependency ξ is said to be *satisfied* in an $\mathbf{n}$-instance $\mathcal{I}$ if $\mathcal{I} \models_{\mathbf{n}} \xi$. A set of dependencies is denoted Σ, and $\mathcal{I} \models_{\mathbf{n}} \Sigma$, if $\mathcal{I} \models_{\mathbf{n}} \xi$ for all $\xi \in \Sigma$. The following is a central concept in data exchange.

Definition 6. *Let $\mathcal{I}$ be a $\mathbf{2}$-instance (a $\mathbf{4}$-instance) and Σ a set of TGDs and EGDs (a set of NTGD$^{\approx}$s). A* $\mathbf{2}$-universal model *(a* $\mathbf{4}$-universal model*) of $(\mathcal{I}, \Sigma)$ is a $\mathbf{2}$-instance $\mathcal{J}$ (a $\mathbf{4}$-instance $\mathcal{J}$), such that*

1. *$\mathcal{I} \preceq \mathcal{J}$ ($\mathcal{I} \preccurlyeq \mathcal{J}$),*
2. *$\mathcal{J} \models_{\mathbf{2}} \Sigma$ ($\mathcal{J} \models_{\mathbf{4}} \Sigma$), and*
3. *for all instances $\mathcal{K}$, if $\mathcal{I} \preceq \mathcal{K}$ and $\mathcal{K} \models_{\mathbf{2}} \Sigma$, then $J \preceq \mathcal{K}$ (if $\mathcal{I} \preccurlyeq \mathcal{K}$ and $\mathcal{K} \models_{\mathbf{4}} \Sigma$, then $J \preccurlyeq \mathcal{K}$).*

Let $\mathcal{I}$ be a $\mathbf{2}$-instance and Σ a set of dependencies. The definition of the procedure $Chase(\mathcal{I}, \Sigma)$ can be found in [8, 7]. The chase procedure is said to *successfully terminate* (or simply terminate) [8], if it does not fail due to some EGD and it stops after a finite number of steps. In this case $Chase(\mathcal{I}, \Sigma)$ denotes the instance returned by the chase procedure. It is known (see [8, 7]) that $Chase(\mathcal{I}, \Sigma)$ is a $\mathbf{2}$-universal model of $(\mathcal{I}, \Sigma)$.

Data Exchange. A *data exchange schema* is a schema of the form $\mathbf{S} \cup \mathbf{T}$, where $\mathbf{S} \cap \mathbf{T} = \varnothing$. The schema $\mathbf{S}$ is called the *source schema*, and $\mathbf{T}$ is called the *target schema*. Let $\mathbf{S} \cup \mathbf{T}$ be a data exchange schema. A *source-to-target TGD (st-TGD)* is a TGD where the relation symbols in the body come from $\mathbf{S}$ and the relation symbols in the head come from $\mathbf{T}$. Source-to-target NTGD$^{\approx}$s (s-t NTGD$^{\approx}$s) are defined similarly. A *target EGD (t-EGD)* is an EGD where all relation symbols come from $\mathbf{T}$. A *data exchange setting* is a pair $(\mathbf{S} \cup \mathbf{T}, \Sigma)$, where $\mathbf{S} \cup \mathbf{T}$ is a data exchange schema, and Σ is a set of s-t TGDs (or s-t NTGD$^{\approx}$s) and t-EGDs.

A $\mathbf{2}$-*instance* of a data exchange setting is a pair $(\mathcal{I}, \Sigma)$, such that $nulls(\mathcal{I}) = \varnothing$, $R(\mathbf{a})^{\mathcal{I}} = \mathsf{0}$, for all $R \in \mathbf{T}$ and $\mathbf{a} \in dom(\mathcal{I})^{ar(R)}$, and $\Sigma = \Sigma_{st} \cup \Sigma_t$, where Σ_{st} is a set of s-t TGDs and Σ_t is a set of t-EGDs.

A $\mathbf{4}$-*instance* of a data exchange setting is like a $\mathbf{2}$-instance, except that Σ_{st} can contain NTGD$^{\approx}$'s, and that $R(\mathbf{a})^{\mathcal{I}} = \bot$, for all $R \in \mathbf{T}$ and $\mathbf{a} \in dom(\mathcal{I})^{ar(R)}$.

Let $\mathbf{n} \in \{\mathbf{2}, \mathbf{4}\}$ and $\preccurlyeq \in \{\preceq, \preccurlyeq\}$. An $\mathbf{n}$-*solution* for an $\mathbf{n}$-instance $(\mathcal{I}, \Sigma)$ of a data exchange setting $(\mathbf{S} \cup \mathbf{T}, \Sigma)$, is an $\mathbf{n}$-instance $\mathcal{J}$, such that $\mathcal{I} \preccurlyeq \mathcal{J}$, $\mathcal{J} \models_{\mathbf{n}} \Sigma$, and $\mathcal{J}{\restriction_{\mathbf{S}}} = \mathcal{I}{\restriction_{\mathbf{S}}}$. The notation $\mathcal{J}{\restriction_{\mathbf{S}}}$ stands for the instance $\mathcal{J}$ restricted to the relation symbols in $\mathbf{S}$, and similarly for $\mathcal{I}{\restriction_{\mathbf{S}}}$. The set of all $\mathbf{n}$-solutions to an $\mathbf{n}$-instance $\mathcal{I}$ in setting $(\mathbf{S} \cup \mathbf{T}, \Sigma)$ is denoted $Sol_{\mathbf{n}}(\mathcal{I}, \Sigma)$. A $\mathbf{n}$-solution $\mathcal{J} \in Sol_{\mathbf{n}}(\mathcal{I}, \Sigma)$ is said to be an $\mathbf{n}$-*universal solution* if $\mathcal{J} \preccurlyeq \mathcal{K}$, for all $\mathcal{K} \in Sol_{\mathbf{n}}(\mathcal{I}, \Sigma)$.

Let $\mathcal{I}$ be a $\mathbf{2}$ instance of a data exchange setting $(\mathbf{S} \cup \mathbf{T}, \Sigma)$, and let $q = \exists \mathbf{Y} \Phi(\mathbf{XY})$ be a *target* UCQ, i.e. such that all relation symbols in Φ come from $\mathbf{T}$. Then the *certain answer* [8] to q on $(\mathcal{I}, \Sigma)$ is defined as

$$Cert_{\mathbf{2}}(q, \mathcal{I}, \Sigma) \;=\; \Big(\bigwedge\{q(\mathcal{J}) : \mathcal{J} \in Sol_{\mathbf{2}}(\mathcal{I}, \Sigma)\}\Big){\restriction} \,. \tag{5}$$

In other words, $Cert_{\mathbf{2}}(q, \mathcal{I}, \Sigma)$ is the greatest lower bound in the partial order $\preceq$ of the set $\{q(\mathcal{J}) : \mathcal{J} \in Sol_{\mathbf{2}}(\mathcal{I}, \Sigma)\}$, restricted to the constants. The following result is at the foundation of data exchange.

Theorem 3. [8] *Let $(\mathcal{I}, \Sigma)$ be a data exchange setting, such that the chase with Σ of $\mathcal{I}$ terminates, and let q be a target UCQ. Then*

1. *$\mathcal{J} = Chase(\mathcal{I}, \Sigma)$ is a $\mathbf{2}$-universal solution for $(\mathcal{I}, \Sigma)$.*
2. *$Cert(q, \mathcal{I}, \Sigma) = (q(\mathcal{J}))\!\uparrow$.*

In the full paper [13] we show that by a suitable reduction, any set Σ of NTGD$^{\approx}$s can be transformed into a set $\Sigma^{\downarrow\mathbf{2}}$ of TGDs, such that for all $\mathbf{4}$-instances $\mathcal{I}, \mathcal{I} \models_{\mathbf{4}} \Sigma$ if and only if $\mathcal{I}^{\downarrow\mathbf{2}} \models_{\mathbf{2}} \Sigma^{\downarrow\mathbf{2}}$. The following theorem makes the connection between the data exchange solutions, as defined in [8], and solutions for the intuitionistic data exchange problem.

Theorem 4. *Let $(\mathbf{S} \cup \mathbf{T}, \Sigma)$ be a data exchange setting where Σ consists of st-NTGD$^{\approx}$s and t-EGDs, and let $\mathcal{I}$ be a $\mathbf{4}$-instance. Then*

$$Sol_{\mathbf{4}}(\mathcal{I}, \Sigma) = Sol_{\mathbf{2}}(I^{\downarrow\mathbf{2}}, \Sigma^{\downarrow\mathbf{2}})^{\uparrow\mathbf{4}}.$$

Armed with this result we now show that we can use the chase process defined in [8] to find a $\mathbf{4}$-universal model. As the next theorem shows, a $\mathbf{4}$-universal solution is a good candidate to be materialized and used to compute the certain answer for any NUCQ-query.

Theorem 5. *Let $(\mathcal{I}, \Sigma)$ be a $\mathbf{4}$-instance of a data exchange setting, where Σ consists of st-NTGD$^{\approx}$s and t-EGDs, Then $(Chase(\mathcal{I}^{\downarrow\mathbf{2}}, \Sigma^{\downarrow\mathbf{2}}))^{\uparrow\mathbf{4}}$ is a $\mathbf{4}$-universal solution for $(\mathcal{I}, \Sigma)$. Moreover, for every NUCQ query q we have*

$$Cert_{\mathbf{4}}(q, \mathcal{I}, \Sigma) = (q^{\downarrow\mathbf{2}}((Chase(\mathcal{I}^{\downarrow\mathbf{2}}, \Sigma^{\downarrow\mathbf{2}}))))\!\uparrow^{\uparrow\mathbf{4}}.$$

6 On Universal Solution Existence

In the previous sections we have shown that for a given $\mathbf{4}$-instance $\mathcal{I}$ and a set of NTGDs Σ, (or NTGD$^{\not\approx}$s), the $\mathbf{4}$-universal solution is the best candidate for target materialization in data exchange, and that as such it can be used to efficiently compute certain answers to target UNCQ queries (or UNCQ$^{\approx}$ queries). The question then arises whether a $\mathbf{4}$-universal solution exists. Since our results include the lossless decomposition $Sol_{\mathbf{4}}(\mathcal{I}, \Sigma) = Sol_{\mathbf{2}}(\mathcal{I}^{\downarrow\mathbf{2}}, \Sigma^{\downarrow\mathbf{2}})^{\uparrow\mathbf{4}}$, the answer follows directly from from [7] and [15]. Formally, we have

Theorem 6. 1. *The problem of testing if there exists a $\mathbf{4}$-universal solution for a given $\mathbf{4}$-instance and a set of NTGD$^{\approx}$s is* RE*-complete.*
2. *The problem of testing if a set of NTGD$^{\approx}$s has a $\mathbf{4}$-universal solution for every $\mathbf{4}$-instance is* coRE*-complete.*

The definition of the *core-chase* procedure can be found in [7], from which the next theorem follows.

Theorem 7. *The core chase procedure is complete for finding* **4***-universal solutions for* **4***-instances* $\mathcal{I}$ *and NTGD*$^{\approx}$*s*

One of the important classes for which universal solutions are guaranteed to exist, is the *guarded dependencies* of [6]. Applying the definition from [6], we say that an NTGD$^{\approx}$ is guarded if the body has an atom that contains all the universally quantified variables. It can easily be shown that if Σ is a set of guarded NTGD$^{\approx}$s, then $\Sigma^{\downarrow\mathbf{2}}$ is a set of guarded TGDs as well. From this and from the elegant result of [17] it follows that

Theorem 8. *A* **4***-universal solution always exists for a set of guarded NTGD*$^{\approx}$*s.*

In the literature one can find many classes of TGDs that are known to ensure uniform chase termination, and thus guaranteeing the existence of universal solutions for all instances (for an overview, see e.g. [22]). Let $\mathcal{C}$ be such a class. We then have

Corollary 2. *Let* Σ *be a set of NTGD*$^{\approx}$*s. If* $\Sigma^{\downarrow\mathbf{2}} \in \mathcal{C}$*, then a* **4***-universal solution for* $(\mathcal{I}, \Sigma)$ *exists, for all* **4***-instances* $\mathcal{I}$*.*

We note that most of the known classes guaranteeing uniform termination have a tractable membership problem.

7 Conclusions

In this paper we have put forth a new approach to Data Exchange based on Belnap's logic $\mathcal{FOUR}$. We showed that moving to a four-valued logic comes with the benefit of naturally extending the mapping language to NTGD$^{\approx}$s and the query language to NUCQ$^{\approx}$s, thus allowing negation over atoms, including equality atoms $a \approx b$. We also showed that well-known techniques from Data exchange (chase, universal solution, naive evaluation, etc) can be used in $\mathcal{FOUR}$ as well. Furthermore, the *core* of a **4**-instance $\mathcal{I}$ (minimal instance in size that is $\simeq$ equivalent with $\mathcal{I}$) is the same as $core(I^{\downarrow\mathbf{2}})^{\uparrow\mathbf{4}}$. Thus the core chase [7] is complete in finding **4**-universal solutions as well. In the special case where the instances are two-valued, the mappings are TGDs and EGDs, and the target queries are UCQs, the semantics reduces to the classical one [8].

There is still more work to be done as we haven't touched on for instance meta-data management problems, such as composition and inverse. Another aspect that needs to be further investigated is the chase termination problem in $\mathcal{TWO}$ for a set of TGDs corresponding to a set of NTGD$^{\approx}$s in $\mathcal{FOUR}$. Note that the set of TGDs in this case is special in the sense that it always contain the set of full TGDs expressed by Σ.

We believe that $\models_{\mathbf{4}}$ is a semantics with natural appeal, and one that can be practically implemented over existing DBMSs.

References

1. Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
2. Alan Anderson, Belnap R., D. Nuel, and J. Michael Dunn. *Entailment: The Logic of Relevance and Necessity, Vol. I.* Princeton University Press, 1992.
3. Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In Victor Vianu and Christos H. Papadimitriou, editors, *PODS*, pages 68–79. ACM Press, 1999.
4. Ofer Arieli and Arnon Avron. The value of the four values. *Artif. Intell.*, 102(1):97–141, 1998.
5. Jr. Belnap, Nuel D. A useful four-valued logic. In J.M̃ichael Dunn and George Epstein, editors, *Modern Uses of Multiple-Valued Logic*, volume 2 of *Episteme*, pages 5–37. Springer Netherlands, 1977.
6. Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. In *KR*, pages 70–80, 2008.
7. Alin Deutsch, Alan Nash, and Jeffrey B. Remmel. The chase revisited. In *PODS*, pages 149–158, 2008.
8. Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. In *ICDT*, pages 207–224, 2003.
9. Melvin Fitting. Bilattices and the semantics of logic programming. *J. Log. Program.*, 11(1&2):91–116, 1991.
10. Melvin Fitting. The family of stable models. *J. Log. Program.*, 17(2/3&4):197–225, 1993.
11. Amélie Gheerbrant, Leonid Libkin, and Cristina Sirangelo. When is naive evaluation possible? In *PODS*, pages 75–86, 2013.
12. Matthew L. Ginsberg. Bilattices and modal operators. In Rohit Parikh, editor, *TARK*, pages 273–287. Morgan Kaufmann, 1990.
13. Gösta Grahne, Ali Moallemi, and Adrian Onet. Intuitionistic data exchange. Technical report, `http://arxiv.org/`.
14. Gösta Grahne and Adrian Onet. Representation systems for data exchange. In *ICDT*, pages 208–221, 2012.
15. Gösta Grahne and Adrian Onet. The data-exchange chase under the microscope. *CoRR*, abs/1407.2279, 2014.
16. Pavol Hell and Jaroslav Neŝetřil. *Graphs And Homomorphisms*. Oxford University Press, 2004.
17. André Hernich. Computing universal models under guarded tgds. In *ICDT*, pages 222–235, 2012.
18. Tomasz Imielinski and Witold Lipski Jr. Incomplete information in relational databases. *J. ACM*, 31(4):761–791, 1984.
19. Thomas Jech. Boolean valued models. In *Handbook of Boolean Algebras*, pages 1197–1211, 1989.
20. Stephen Cole Kleene. *Introduction to metamathematics*. D. Van Norstrand, 1952.
21. Leonid Libkin. Incomplete data: what went wrong, and how to fix it. In *PODS*, pages 1–13, 2014.
22. Adrian Onet. *The chase procedure and its applications*. PhD thesis, Concordia University, 2012.
23. Alasdair Urquhart. Basic many-valued logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of philosophical logic*, volume 2, pages 249–295. Springer Netherlands, 2001.