Foundations of Information Integration under Bag Semantics

André Hernich
University of Liverpool

Phokion G. Kolaitis
UC Santa Cruz and IBM Research - Almaden

Abstract—**During the past several decades, the database theory community has successfully investigated several different facets of the principles of database systems, including the development of various data models, the systematic exploration of the expressive power of database query languages, and, more recently, the study of the foundations of information integration via schema mappings. For the most part, all these investigations have been carried out under set semantics, that is, both the database relations and the answers to database queries are sets. In contrast, SQL deploys bag (multiset) semantics and, as a result, theory and practice diverge at this crucial point.**

Our main goal in this paper is to embark on the development of the foundations of information integration under bag semantics, thus taking the first step towards bridging the gap between theory and practice in this area. Our first contribution is conceptual, namely, we give rigorous bag semantics to GLAV mappings and to the certain answers of conjunctive queries in the context of data exchange and data integration. In fact, we introduce and explore two different versions of bag semantics that, intuitively, correspond to the maximum-based union of bags and to the sum-based union of bags. After this, we establish a number of technical results, including results about the computational complexity of the certain answers of conjunctive queries under bag semantics and about the existence and computation of universal solutions under these two versions of bag semantics. Our results reveal that the adoption of more realistic semantics comes at a price, namely, algorithmic problems in data exchange and data integration that were tractable under set semantics become intractable under bag semantics.

I. Introduction and Summary of Results

Since the introduction of the relational data model by Codd [8] in 1970, the database theory community has successfully investigated several different facets of the principles of database systems. These investigations include the development of various data models, the exploration of the expressive power of database query languages, the examination of computational aspects of query processing and optimization, and, more recently, the study of the foundations of information integration.

For the most part, the aforementioned theoretical investigations have been carried out under set semantics, that is, both the database relations and the results returned by database queries are sets. In practice, however, SQL uses bag (multisets) semantics as the default, which means that duplicates are not eliminated from the answers to queries, unless explicitly specified. The adoption of bag semantics in SQL is due to reasons of both efficiency and convenience. First, duplicate elimination on large databases is regarded as a costly operation; moreover, even though aggregate operations, such as average and sum, can be computed without using bag semantics [17], it is easier to express aggregate queries correctly when bag semantics is used. This discrepancy between theory and practice was already pointed out by Chaudhuri and Vardi [7] in 1993, who initiated the study of the query containment problem under bag semantics by focusing on conjunctive queries. Recall that a conjunctive query $q(\bar{x})$ is defined by a first-order formula of the form $\exists\bar{y}\theta(\bar{x},\bar{y})$, where $\theta(\bar{x},\bar{y})$ is a conjunction of atoms with variables among those in $\bar{x}$ and $\bar{y}$. Under set semantics, the conjunctive query containment problem is NP-complete [6]. In contrast, the status of the conjunctive query containment problem under bag semantics is not well understood and the main question raised by Chaudhuri and Vardi remains open to date: is the conjunctive query containment problem under bag semantics decidable? At present, only special cases of this problem are known to be decidable (see, e.g., [2], [22]). A striking difference between set semantics and bag semantics has been established for extensions of conjunctive queries with unions and inequalities ($\neq$). Specifically, under set semantics, the containment problem for each of these two classes of queries is decidable [26], [18], [30], while, under bag semantics, it is undecidable [15], [16].

Beyond the query containment problem, some other topics in data management have been investigated under bag semantics. These investigations include early work on giving precise bag semantics to relational algebra operators [3], [13], techniques for query processing under bag semantics [25], the expressive power and complexity of database query languages under bag semantics [14], and the query equivalence problem under bag semantics [9].

During the past decade, an extensive investigation of the foundations of information integration has taken place. A notable outcome of this investigation is the formalization and study of critical data inter-operability tasks, such as data exchange and data integration (see the surveys [20], [21], [23] and the books [4], [10]). So far, the study of the foundations of information integration has been carried out

978-1-5090-3018-7/17/$31.00 ©2017 IEEE

entirely under set semantics. In this paper, we develop the foundations of information integration under bag semantics, thus taking the first step towards bridging the gap between theory and practice in this area of research. Towards this goal, we make a number of both conceptual and technical contributions that we now summarize.

The formalization and study of data exchange and data integration (under set semantics) has been based on the systematic use of schema mappings, which are high-level specifications describing the relationship between two database schemas. Undoubtedly, GLAV mappings constitute the most widely used and extensively studied class of schema mappings. A GLAV mapping is specified by a finite set of source-to-target tuple generating dependencies (in short, s-t tgds), i.e., by first-order formulas of the form

$$\forall \bar{x}(q_1(\bar{x}) \rightarrow q_2(\bar{x})),$$

where $q_1(\bar{x})$ is a conjunctive query over the source schema, $q_2(\bar{x})$ is a conjunctive query over the target schema, and $\bar{x}$ are the free variables of both q_1 and q_2.

Our first conceptual contribution is to give rigorous bag semantics to s-t tgds and to GLAV mappings. We view the above s-t tgd as asserting the containment of a source conjunctive query in a target conjunctive query and give it the following bag semantics: a pair (I, J) consisting of a source bag database I and a target bag database J satisfies the s-t tgd $\forall \bar{x}(q_1(\bar{x}) \rightarrow q_2(\bar{x}))$ if the result of evaluating q_1 over I under bag semantics is contained in the result of evaluating q_2 over J under bag semantics. We present several examples demonstrating that this is the "right" bag semantics for s-t tgds. We then turn attention to the bag semantics of GLAV mappings. We give two equally natural, but, in general, different, bag semantics to GLAV mappings, which we call *incognizant* semantics and *cognizant* semantics. Intuitively, these two semantics arise from the two different semantics of the union of bags, the maximum-based union and the sum-based union. The maximum-based union of two bags is the bag in which the multiplicity of a tuple is the maximum of the multiplicities of the tuple in the two bags, while the sum-based union is the bag in which the multiplicity of a tuple is the sum of the multiplicities of the tuple in the two bags. The motivation and definitions of these notions are given in Section III.

Our second conceptual contribution is to define the notion of the certain answers of target conjunctive queries under bag semantics. Under set semantics, the certain answers $\text{certain}_{\mathcal{M}}(q, I)$ of a target conjunctive q w.r.t. a schema mapping $\mathcal{M}$ on a set source instance I is the intersection $\bigcap q(J)$, as J ranges over the solutions for I w.r.t. $\mathcal{M}$, i.e., set target instances J such that the pair (I, J) satisfies $\mathcal{M}$. Here, we have to deal with two complications. First, depending on the bag semantics of GLAV mappings considered, there are two types of solutions, incognizant solutions (i-solutions) and cognizant solutions (c-solutions), which give rise to the i-certain answers $\text{i-certain}_{\mathcal{M}}(q, I)$ and to the c-certain answers $\text{c-certain}_{\mathcal{M}}(q, I)$, respectively. Second, here $\bigcap q(J)$ is the bag intersection of the $q(J)$'s, which means that a tuple $\bar{a}$ appears in $\text{i-certain}_{\mathcal{M}}(q, I)$ with multiplicity m if m is the minimum of the multiplicities of $\bar{a}$ over all $q(J)$'s, as J ranges over the i-solutions for I (and analogously for $\text{c-certain}_{\mathcal{M}}(q, I)$).

As regards technical contributions, we explore the data complexity of the certain answers of target conjunctive queries. Under set semantics, every fixed schema mapping $\mathcal{M}$ and every fixed target query q, give rise to the following decision problem: given a source instance I and a tuple $\bar{a}$, is $\bar{a}$ in $\text{certain}_{\mathcal{M}}(q, I)$? For GLAV mappings and for conjunctive queries, this problem is solvable in polynomial time [11]. Under bag semantics, multiplicities enter the picture and have to be taken into account in formulating algorithmic problems about the certain answers. For every fixed schema mapping $\mathcal{M}$ and every target conjunctive query q, the decision problem $\text{i-QA}(\mathcal{M}, q)$ asks: given a (bag) source instance I, a tuple $\bar{a}$, and a positive integer m, is the multiplicity of $\bar{a}$ in $\text{i-certain}_{\mathcal{M}}(q, I)$ at least m? The decision problem $\text{c-QA}(\mathcal{M}, q)$ is defined analogously.

We show that if $\mathcal{M}$ is a GLAV mapping and q is a target conjunctive query, then both decision problems $\text{i-QA}(\mathcal{M}, q)$ and $\text{c-QA}(\mathcal{M}, q)$ are in coNP. Furthermore, if $\mathcal{M}$ is a GAV mapping (i.e., for each s-t tgd $\forall \bar{x}(q_1(\bar{x}) \rightarrow q_2(\bar{x}))$ of $\mathcal{M}$, the conjunctive query q_2 consists of a single non-existentially quantified atom), then for every target conjunctive query q, both $\text{i-QA}(\mathcal{M}, q)$ and $\text{c-QA}(\mathcal{M}, q)$ are solvable in polynomial time. Once syntactically minimal extensions of GAV mappings are considered, however, the picture changes. Specifically, both $\text{i-QA}(\mathcal{M}, q)$ and $\text{c-QA}(\mathcal{M}, q)$ can become coNP-complete for full mappings, i.e., GLAV schema mappings $\mathcal{M}$ such that in each s-t tgd of $\mathcal{M}$, the conjunctive query q_2 is a conjunction of non-existentially quantified atoms. We also consider elementary mappings, which form the other syntactically minimal extension of GAV mappings. These are the GLAV mappings $\mathcal{M}$ such that in each s-t tgd of $\mathcal{M}$, the conjunctive query q_2 consists of a single atom that may be existentially quantified. We show that for each elementary mapping $\mathcal{M}$ and each target conjunctive query q, the decision problem $\text{c-QA}(\mathcal{M}, q)$ is solvable in polynomial time, whereas the decision problem $\text{i-QA}(\mathcal{M}, q)$ may become coNP-complete.

Finally, we consider the existence and the computation of universal solutions in the context of data exchange under bag semantics. Universal solutions play an important role in data exchange under set semantics [11]. Their main feature is that they are the "most general" solutions in data exchange; furthermore, they are CQ-*adequate*, which means that if $\mathcal{M}$ is a GLAV mapping and q is a target conjunctive query, then for every source instance I and every universal solution J for I w.r.t. $\mathcal{M}$, the certain answers $\text{certain}_{\mathcal{M}}(q, I)$ can be obtained by simply evaluating q on J and removing from the result all tuples that contain at least one null value. The aforementioned coNP-completeness results for full mappings rule out the possibility of having a polynomial-time procedure for computing CQ-adequate i-solutions or c-solutions; moreover, the same holds true for elementary mappings and for CQ-adequate i-solutions. We amplify these negative results by showing that for such schema mappings every CQ-*adequate basis* may be of exponential size. On the positive side, we prove that for every elementary mapping $\mathcal{M}$, there is a polynomial-time procedure that computes universal c-solutions.

II. Preliminaries

Bags A *bag* (or *multiset*) is a collection of elements, where each element occurs one or more times; the *multiplicity* of an element x in a bag B, denoted by $|x|_B$, is the number of occurrences of x in B. An element x *belongs* to a bag B, denoted by $x \in B$, if $|x|_B \geq 1$. We will often denote a bag B as a set of pairs $x : m$, where x is an element and m is its multiplicity in B (instead of $x : 1$ we will often write just x). For example, $B = \{a, b : 3, c : 2\}$ denotes the bag with $|a|_B = 1$, $|b|_B = 3$, $|c|_B = 2$, and $|x|_B = 0$ for all other elements x. A bag B is *contained* in a bag B', denoted $B \subseteq B'$, if $|x|_B \leq |x|_{B'}$, for every element $x \in B$.

Bag Databases A k-ary relation is a bag whose elements are k-tuples. A *schema* is a finite non-empty set of relation symbols, where each relation symbol R has an associated positive integer $\text{ar}(R)$ as its *arity*. An *instance* K of a schema $\mathbf{R}$ is a collection of relations such that K contains a relation R^K of arity $\text{ar}(R)$, for each relation symbol R in $\mathbf{R}$. A *set-instance* is an instance whose relations are sets, i.e., each tuple occurs in a relation of K with multiplicity at most 1. The *active domain* of an instance K, denoted by $\text{adom}(K)$, is the set of all elements that occur in a tuple of some relation in K. We assume that $\text{adom}(K)$ is a subset of a fixed set Dom, which is the union of two disjoint infinite sets—the set Const of all *constants*, and the set Null of all *(labeled) nulls*. Constants will be denoted by lower-case letters, and nulls by upper-case letters. Every instance K can be viewed as a bag of *facts* of the form $R(\bar{a})$, where the multiplicity of $R(\bar{a})$ in K is $|\bar{a}|_{R^K}$. An instance K is a *subinstance* of an instance L if $K \subseteq L$, that is, if $|R(\bar{a})|_K \leq |R(\bar{a})|_L$ for all atoms $R(\bar{a}) \in K$.

Query Answering under Bag Semantics A *conjunctive query (CQ)* over a schema $\mathbf{R}$ is a first-order formula $q(\bar{x}) = \exists \bar{y} \left(R_1(\bar{z}_1) \wedge \ldots \wedge R_n(\bar{z}_n)\right)$, where each R_i belongs to $\mathbf{R}$, each variable occurs in $\bar{x}$ or $\bar{y}$, and each variable in $\bar{x}$ occurs in some $\bar{z}_i$; the notation $q(\bar{x})$ indicates that the tuple $\bar{x}$ consists of all the free variables of the formula. A *match* of q in K is a function μ from the variables of q to $\text{adom}(K)$ such that for each $i \in \{1, \ldots, n\}$, the tuple $\mu(\bar{z}_i)$ occurs in R_i^K; here, $\mu(\bar{z}_i)$ is the result of applying μ component-wise to $\bar{z}_i$. The *answer* to q on K is the $|\bar{x}|$-ary relation $q(K)$ such that for all $|\bar{x}|$-tuples $\bar{a}$:

$$|\bar{a}|_{q(K)} := \sum_{\substack{\mu\colon \text{match of } q \text{ in } K \\ \text{such that } \mu(\bar{x}) = \bar{a}}} \prod_{i=1}^{n} |\mu(\bar{z}_i)|_{R_i^K}. \tag{1}$$

Note that, if K is a set-instance, then (1) is the number of matches μ of q in K such that $\mu(\bar{x}) = \bar{a}$. Note also that $q(K)$ is the answer obtained when evaluating the SQL query expressing q on a standard DBMS.

If the query q is Boolean (i.e., q has no free variables), we will often identify $q(K)$ with the number $\{()\}_{q(K)}$, where $()$ stands for the empty tuple; it will always be clear whether $q(K)$ denotes the relation as defined above, or the number $\{()\}_{q(K)}$. In particular, $q(K) = 0$ means that the relation $q(K)$ is empty (which may be interpreted as "false"), and $q(K) = m > 0$ means that the relation $q(K)$ is non-empty (which may be interpreted as "true") and the empty tuple $()$ occurs in $q(K)$ exactly m times.

III. Schema Mappings and Certain Answers Under Bag Semantics

Schema mappings are high-level specifications that describe the relationship between two schemas, a source schema and a target schema. These high-level specifications are typically expressed using database dependencies, the most widely used of which are tuple-generating dependencies [5]. Here, we give bag semantics to tuple-generating dependencies and to schema mappings specified by such dependencies. We also define the notion of the certain answers of target queries under bag semantics.

A. Bag Semantics of Database Dependencies

A *tuple-generating dependency* (in short, *tgd*) over a schema $\mathbf{R}$ is an expression of the form

$$\forall \bar{x} \left(q_1(\bar{x}) \to q_2(\bar{x})\right), \tag{2}$$

where $q_1(\bar{x})$ and $q_2(\bar{x})$ are conjunctive queries over $\mathbf{R}$. We call $q_1(\bar{x})$ the *body* of the tgd, and $q_2(\bar{x})$ its *head.* Note that the conjunctive queries q_1 and q_2 have the same free variables, namely, $\bar{x}$, since, as stated earlier, the notation $q(\bar{x})$ for a conjunctive query q indicates that the tuple $\bar{x}$ consists of all the free variables of q. It is often the case that the universal quantifiers at the beginning of tgds are omitted; thus, we may write $q_1(\bar{x}) \to q_2(\bar{x})$, instead of (2).

Under set semantics, an instance K satisfies a tgd of the form (2) if each tuple in the answer to q_1 on K also occurs in the answer to q_2 on K. In other words, the tgd asserts that the answer to q_1 is contained in the answer to q_2, where containment refers to the usual notion of containment of one set in another set. This is consistent with Lenzerini's framework for data integration [23], which considers the formula (2) as a logical representation of the assertion that $q_1 \rightsquigarrow q_2$ with the squiggly arrow indicating containment of the answers to q_1 in the answers to q_2.

Under bag semantics, the answer to a conjunctive query is a bag of tuples rather than a set. To account for this change, we modify the interpretation of the arrow in a tgd from set containment to bag containment.

Definition 1. Let K be an instance of a schema $\mathbf{R}$, and let d be a tgd over $\mathbf{R}$ of the form (2). We say that K *satisfies*

d if $q_1(K) \subseteq q_2(K)$, i.e., the bag $q_1(K)$ is contained in the bag $q_2(K)$.

In data exchange, there is a source schema $\mathbf{S}$ and a target schema $\mathbf{T}$ that is disjoint from $\mathbf{S}$. A *source-to-target tuple-generating dependency* (in short, *s-t tgd*) is an expression

$$\forall \bar{x} \left(q_1(\bar{x}) \rightarrow q_2(\bar{x}) \right), \tag{3}$$

where $q_1(\bar{x})$ is a conjunctive query over $\mathbf{S}$ and $q_2(\bar{x})$ is a conjunctive query over $\mathbf{T}$. An instance K over the combined schema $\mathbf{S} \cup \mathbf{T}$ can be identified with a pair (I, J), where I is an $\mathbf{S}$-instance and J is a $\mathbf{T}$-instance. Thus, such a pair (I, J) satisfies the s-t tgd in (3) if $q_1(I) \subseteq q_2(J)$. From now on, $\mathbf{S}$-instances will be referred to as *source* instances, and $\mathbf{T}$-instances as *target* instances. In data exchange, the relationship between the source schema and the target schema is typically specified by a finite set of s-t tgds. For this reason, we will focus on s-t tgds in what follows.

Before proceeding further, we wish to reconcile the definition of an s-t tgd presented here with that in the literature about data exchange under set semantics (e.g., in [11]). Specifically, an s-t tgd is typically defined as an expression of the form

$$\forall \bar{x} \forall \bar{y} (\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})),$$

where $\varphi(\bar{x}, \bar{y})$ is a conjunction of atoms over the source schema such that the variables occurring in these atoms are the variables in $\bar{x}$ and $\bar{y}$, and $\psi(\bar{x}, \bar{z})$ is a conjunction of atoms over the target schema such that the variables occurring in these atoms are the variables in $\bar{x}$ and $\bar{z}$. Such an expression does not express that a conjunctive query over the source is contained in a conjunctive query over the target under set semantics, because the free variables of the body and the head do not match. For this reason, we do not give bag semantics to such expressions. However, the universally quantified variables $\bar{y}$ occurring in φ and not in ψ can be turned into existentially quantified variables in the antecedent of the above expression and, thus, such an expression is logically equivalent under set semantics to an s-t tgd as defined here. For example,under set semantics, the expression $\forall x \forall y (S(x, y) \rightarrow T(x))$ is logically equivalent to the expression $\forall x (\exists y S(x, y) \rightarrow T(x))$. We give bag semantics to the latter expression, but not to the former.

We now present several examples that illustrate the bag semantics for s-t tgds. In all these examples, universal quantifiers in front of s-t tgds have been omitted.

Example 1. Let d be the s-t tgd $P(x) \rightarrow R(x)$. A pair (I, J) satisfies d if each element that occurs m times in P^I occurs *at least* m times in R^J. Thus, d asserts that P^I is contained (as a bag) in R^J. Concrete pairs that satisfy d are $(I_1 = \{P(a) : 2\}, J_1 = \{R(a) : 2\})$ and $(I_2 = \{P(a) : 2\}, J_2 = \{R(a) : 3\})$. A pair that does not satisfy d is $(I_3 = \{P(a) : 2\}, J_3 = \{R(a) : 1\})$.

Example 2. Let d be the s-t tgd $\exists y\, S(x, y) \rightarrow R(x)$. The body of d is $q_1(x) = \exists y\, S(x, y)$, and its head is $q_2(x) = R(x)$. A pair (I, J) satisfies d if each element that occurs m times in $q_1(I)$ occurs *at least* m times in the answer to $q_2(J)$. Let I be an $\mathbf{S}$-instance containing $S(a, b) : 1$ and $S(a, c) : 2$. Then, $q_1(I)$ contains $a : 3$, so if J is a $\mathbf{T}$-instance such that the pair (I, J) satisfies d, then R^J must contain at least 3 occurrences of a.

Example 3. Let d be the s-t tgd $S(x, y) \rightarrow P(x) \wedge Q(y)$, and let I be an $\mathbf{S}$-instance that contains two occurrences of $S(a, b)$. If J is a $\mathbf{T}$-instance such that the pair (I, J) satisfies d, then J must contain at least two occurrences of $P(a)$ or at least two occurrences of $Q(b)$.

Example 4. Let d be the s-t tgd

$$R(x, y) \rightarrow \exists z \big(S(x, z) \wedge S(z, y)\big)$$

and let I be an $\mathbf{S}$-instance that contains two occurrences of $R(a, b)$ with $a \neq b$. If J is a $\mathbf{T}$-instance such that the pair (I, J) satisfies d, then J must contain one of the following $\mathbf{T}$-instances as subinstances, where X and Y are values (constants or labelled nulls) such that $X \neq Y$:

$$J_1 := \{S(a, X) : 2, S(X, b) : 1\},$$
$$J_2 := \{S(a, X) : 1, S(X, b) : 2\},$$
$$J_3 := \{S(a, X) : 1, S(X, b) : 1, S(a, Y) : 1, S(Y, b) : 1\}.$$

B. Bag Semantics of Schema Mappings

Schema mappings are high-level descriptions of the transformation of source instances into target instances. More precisely, a schema mapping is a tuple $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ is a *source schema*, $\mathbf{T}$ is a *target schema* that is disjoint from $\mathbf{S}$, and Σ is a finite set of assertions about $\mathbf{S}$ and $\mathbf{T}$. Here, we focus on *GLAV mappings* [11], that is, schema mappings $(\mathbf{S}, \mathbf{T}, \Sigma)$ in which Σ is a finite set of s-t tgds. Arguably, GLAV mappings form the most extensively studied class of schema mappings in data exchange.

In what follows, we will introduce two natural bag semantics of GLAV mappings, which we call *incognizant semantics* and *cognizant semantics*. These two semantics coincide on GLAV mappings $(\mathbf{S}, \mathbf{T}, \Sigma)$ in which Σ is a singleton, but, in general, differ if Σ contains more than one s-t tgds. As we shall see, the difference between these two semantics has to do with the difference between two possible semantics for the union of bags. We begin by first discussing the semantics for the union of bags and then presenting an example that motivates the incognizant and the cognizant semantics of GLAV mappings.

Knuth [19] discusses the history of bags (under the name multisets) and defines several different operations on bags. As regards the union operation, Knuth considers the *maximum-based union (max-union)* of bags, which, given two bags B_1 and B_2, returns the bag

$$B_1 \cup B_2 := \{x : \max(|x|_{B_1}, |x|_{B_2}) \mid x \text{ in } B_1 \text{ or } B_2\}, \tag{4}$$

i.e., an element appears in the max-union $B_1 \cup B_2$ with multiplicity equal to the maximum of its multiplicities in B_1 and B_2. In the literature on bag semantics, however, another natural notion of union of bags has been identified, called *sum-based union (sum-union)* [3], [13]. Given two bags B_1 and B_2, the sum-union returns the bag

$$B_1 \uplus B_2 := \{x : (|x|_{B_1} + |x|_{B_2}) \mid x \text{ in } B_1 \text{ or } B_2\}, \quad (5)$$

i.e., an element appears in the sum-union $B_1 \uplus B_2$ with multiplicity equal to the sum of its multiplicities in B_1 and B_2. It should be noted that the sum-union is the semantics of the UNION ALL operator in SQL.

We now come to the example that motivates the incognizant and the cognizant semantics for schema mappings. Let $\mathcal{M}_u = (\{P, Q\}, \{R\}, \Sigma_u)$ be the schema mapping such that P, Q, R are unary relational symbols and Σ consists of the two s-t tgds (with the universal quantifiers omitted):

$$P(x) \to R(x), \quad Q(x) \to R(x).$$

Let I be a source instance and J a target instance. What does it mean to say that the pair (I, J) *satisfies* Σ_u? Intuitively, we should have that (I, J) satisfies Σ_u if and only if the bag R^J contains the union of the bags P^I and Q^I. We just saw, however, that there are two different, yet equally natural, semantics for the union of bags. This suggests two different bag semantics of GLAV mappings. The first, which we call *incognizant* semantics corresponds to the max-union and the second, which we call *cognizant* semantics, corresponds to the sum-union.

Definition 2. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a GLAV mapping and let I be a source instance. An *incognizant solution (i-solution)* for I w.r.t. $\mathcal{M}$ is a target instance J such that the pair (I, J) satisfies every s-t tgd in Σ.

Consider the schema mapping $\mathcal{M}_u = (\{P, Q\}, \{R\}, \Sigma_u)$. By unraveling the bag semantics of individual s-t tgds, it is easy to see that J is an i-solution for J w.r.t. $\mathcal{M}_u$ if and only if for every element a, the multiplicity of a in R^I is greater than or equal to the maximum of the multiplicities of a in P^I and Q^I. Thus, in this case, the incognizant semantics corresponds to the max-union of P and Q. More generally, if $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ is an arbitrary GLAV mapping and I is a source instance, we have that a target instance J is an i-solution for I w.r.t. $\mathcal{M}$ if and only if for each tgd $d \in \Sigma$, there is a target instance J_d such that (I, J_d) satisfies d and $J_d \subseteq J$; note that the condition $J_d \subseteq J$, for each d in Σ, is equivalent to the condition $\bigcup_{d\in\Sigma} J_d \subseteq J$, where $\bigcup$ denotes the max-union of bags.

Definition 3. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma_u)$ be a GLAV mapping and let I be a source instance. A *cognizant solution (c-solution)*[1] for I w.r.t. $\mathcal{M}$ is a target instance J such for each $d \in \Sigma_{\text{st}}$, there is a target instance J_d such that (I, J_d) satisfies d and $\biguplus_{d\in\Sigma_{\text{st}}} J_d \subseteq J$, where $\biguplus$ denotes the sum-union of bags.

[1]The dictionary defines "incognizant" as "oblivious", and "cognizant" as "aware". We chose these terms for the two semantics we introduced here because a target instance J is an i-solution for I as long as (I, J) satisfies each s-t tgd of the GLAV mapping at hand individually - there is no other condition on the combined effect of the s-t tgds; in contrast, a c-solution must, in addition, be aware of the combined effect of the s-t tgds.

Consider the schema mapping $\mathcal{M}_u = (\{P, Q\}, \{R\}, \Sigma_u)$. By unraveling the bag semantics of individual s-t tgds, it is easy to see that J is a c-solution for J w.r.t. $\mathcal{M}_u$ if and only if for every element a, the multiplicity of a in R^I is greater than or equal to the sum of the multiplicities of a in P^I and Q^I. Thus, in this case, the cognizant semantics corresponds to the max-union of P and Q.

Since the max-union of bags is always contained in the sum-union of bags, it follows that every c-solution is an i-solution. It is clear, however, that the converse need not be true; thus, in general, the incognizant semantics differs from the cognizant semantics. As mentioned earlier, the incognizant semantics coincides with the cognizant semantics on individual s-t tgds. The next simple result, whose proof follows immediately from the definitions, yields a useful sufficient condition for the incognizant semantics to coincide with the cognizant ones.

Proposition 1. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a GLAV mapping such that for every two distinct tgds $d, d' \in \Sigma$, no relation symbol that occurs in the head of d also occurs in the head of d'. Then for each source instance I, the i-solutions for I w.r.t. $\mathcal{M}$ are precisely the c-solutions for I w.r.t. $\mathcal{M}$.*

The following special cases of GLAV mappings play an important role in the context of data integration [23] and will be of interest to us in the sequel.

- A *full* mapping is a GLAV mapping $(\mathbf{S}, \mathbf{T}, \Sigma)$ in which every s-t tgd in Σ is *full*, i.e., its head is quantifier-free.
- A *GAV mapping* is a GLAV mapping $(\mathbf{S}, \mathbf{T}, \Sigma_{\text{st}})$ in which every s-t tgd in Σ is *Horn*, i.e., its head consists of a single atom that is not existentially quantified.
- An *elementary mapping* is a GLAV mapping in which every s-t tgd in Σ is elementary, i.e., its head consists of a single atom (which may be existentially quantified).

For example, the s-t tgds in Examples 1 and 2 are Horn (hence also both full and elementary). The tgd in Example 3 is full, but not elementary, while the tgd $P(x) \to \exists y\, R(x, y)$ is elementary, but not full. Finally, the tgd in Example 4 is neither elementary nor full.

Note that under set semantics, every full mapping $\mathcal{M}$ is equivalent to a GAV mapping $\mathcal{M}'$ obtained from $\mathcal{M}$ by "decomposing" every full s-t tgd of $\mathcal{M}$ to finitely many GAV s-t tgds of $\mathcal{M}'$ (one for each atom in the head of the full s-t tgd under consideration). Here, $\mathcal{M}$ being *equivalent* to $\mathcal{M}'$ means that for every source set-instance I, the target set-instances that are solutions for I w.r.t. $\mathcal{M}$ coincide with the target set-instances that are solutions for I w.r.t. $\mathcal{M}'$. As the following example shows, however, the equivalence of $\mathcal{M}$ and $\mathcal{M}'$ need not hold under bag semantics, whether incognizant or cognizant. In fact, this difference between full mappings and GAV mappings is a harbinger of more profound differences between full mappings and GAV mappings mappings that will be unveiled in Section IV.

Example 5. Let $\mathcal{M} = (\{S\}, \{P, Q\}, \Sigma)$ be the GLAV mapping, where Σ consists of the s-t tgd in Example 3, i.e., $\Sigma = \{S(x, y) \to P(x) \wedge Q(x)\}$. Under set semantics, d can be "decomposed" into the two Horn s-t tgds

$$d_1\colon\ \exists y S(x, y) \to P(x) \qquad d_2\colon\ \exists x S(x, y) \to Q(y)$$

so that the set-instances that satisfy d are precisely those that satisfy d_1 and d_2. This is not true under bag semantics. First, notice that, by Proposition 1, the incognizant semantics coincide with the cognizant ones. Consider now the source instance $I = \{S(a, b) : 2\}$. Then, the target instance $J = \{P(a) : 2, Q(b) : 1\}$ is both an i-solution and a c-solution for I w.r.t. $\mathcal{M}$. However, J is neither an i-solution nor a c-solution for I w.r.t. the GAV mapping $\mathcal{M}' = (\{S\}, \{P, Q\}, \{d_1, d_2\})$. In fact, while $\mathcal{M}$ only requires the presence of at least two occurrences of either $P(a)$ or $Q(b)$ (and only one occurrence of the other atom), the mapping $\mathcal{M}'$ imposes the much stricter restriction that both $P(a)$ and $Q(b)$ have to occur at least two times in a solution for I.

C. *Bag Semantics of Certain Answers*

To answer target queries with respect to a schema mapping, we adopt the *certain answers* semantics. However, since the answer to a query is a *bag* of tuples, we replace intersection of sets by intersection of bags. Formally, let $\mathcal{M}$ be a schema mapping, let I be a source instance, and let q be a conjunctive query over the target schema. The *incognizant certain answers of q with respect to $\mathcal{M}$ on I* are defined as

$$\text{i-certain}_{\mathcal{M}}(q, I) := \bigcap \{q(J) \mid J \text{ is an i-solution for } I\},$$

where the intersection denotes bag intersection. This means that the multiplicity of a tuple $\bar{a}$ in $\text{i-certain}_{\mathcal{M}}(q, I)$ is the minimum of the multiplicities of $\bar{a}$ in $q(J)$, where J ranges over the i-solutions for I w.r.t. $\mathcal{M}$, or 0 if there is no solution for I w.r.t. $\mathcal{M}$. If q is a Boolean conjunctive query, then we identify $\text{i-certain}_{\mathcal{M}}(q, I)$ with the multiplicity of the empty tuple () in $\text{certain}_{\mathcal{M}}(q, I)$. Similarly, the *cognizant certain answers of q with respect to $\mathcal{M}$ on I* are defined as

$$\text{c-certain}_{\mathcal{M}}(q, I) := \bigcap \{q(J) \mid J \text{ is a c-solution for } I\}.$$

Example 6. Let $\mathcal{M} = (\{R\}, \{S\}, \Sigma)$ be the GLAV mapping, where Σ consists of the tgd in Example 4, i.e., $\Sigma = \{R(x, y) \to \exists z(S(x, z) \wedge S(z, y))\}$. We aim to compute the (incognizant and cognizant) certain answers of the following conjunctive queries with respect to $\mathcal{M}$ on the source instance $I = \{R(a, b) : 2\}$:

$$\begin{aligned} q_1(x) &:= \exists y\, S(x, y), \\ q_2(x, y) &:= \exists z\, (S(x, z) \wedge S(z, y)), \\ q_3 &:= \exists x\, \exists y\, S(x, y). \end{aligned}$$

The i-solutions for I w.r.t. $\mathcal{M}$ are precisely the target instances that contain J_1, J_2, or J_3 from Example 4; moreover, the same holds true for c-solutions. This implies that J_1, J_2, and J_3 are the minimal i-solutions and c-solutions for I w.r.t. $\mathcal{M}$, and, in particular, that $\text{i-certain}_{\mathcal{M}}(q_i, I) = \text{c-certain}_{\mathcal{M}}(q_i, I)$, for each $i \in \{1, 2, 3\}$. It is therefore enough to compute only the incognizant certain answers. For q_1, we have $\text{i-certain}_{\mathcal{M}}(q_1, I) = \{a : 1\}$, since a occurs once in $q_1(J_2)$ and at least one time in the answer to q_1 on J_1 and J_3. For q_2, we obtain $\text{i-certain}_{\mathcal{M}}(q_2, I) = \{(a, b) : 2\}$, since (a, b) occurs exactly two times in the answer to q_2 on each of the instances J_1, J_2, and J_3. Finally, $\text{i-certain}_{\mathcal{M}}(q_3, I) = 3$,[2] because $q_3(J_i) \geq 3$, for $i \in \{1, 2, 3\}$, and $q_3(J_1) = 3$.

IV. Complexity of Certain Answers

Answering queries with respect to schema mappings is a fundamental task in both data integration and data exchange [23], [20]. The complexity of this problem under *set* semantics has been studied in depth. In particular, for many different types of schema mappings, including GLAV mappings, it is known that the data complexity of computing the certain answers of a conjunctive query (CQ) is tractable [23], [20]. This section explores the complexity of computing the certain answers of CQs under *bag* semantics. We will focus on data complexity (i.e., the complexity of computing the certain answers for a fixed schema mapping and query) and on Boolean CQs. Since CQs may contain constants, all results extend to non-boolean CQs in a straightforward way.

For every schema mapping $\mathcal{M}$ and every Boolean CQ q over the target schema of $\mathcal{M}$, we consider the following decision problems about the certain answers of q w.r.t. $\mathcal{M}$:

- $\text{i-QA}(\mathcal{M}, q)$: given a source instance I and a positive integer m, decide whether $\text{i-certain}_{\mathcal{M}}(q, I) \geq m$.
- $\text{c-QA}(\mathcal{M}, q)$: given a source instance I and a positive integer m, decide whether $\text{c-certain}_{\mathcal{M}}(q, I) \geq m$.

Note that both $\text{i-QA}(\mathcal{M}, q)$ and $\text{c-QA}(\mathcal{M}, q)$ generalize computing the certain answers under set semantics. In particular, $\text{i-certain}_{\mathcal{M}}(q, I) \geq 1$ holds if, and only if, the certain answers to q w.r.t. $\mathcal{M}$ on I are non-empty, and analogously for the cognizant certain answers.

A. *Upper Bounds for GLAV Mappings*

We will show that, for arbitrary GLAV mappings, the data complexity of computing the incognizant certain answers, as well as the cognizant certain answers, is in coNP. The notion of the *canonical solution* is a key tool in obtaining this complexity bound.

Given a GLAV mapping $\mathcal{M}$ and a source instance I, the *canonical solution* for I w.r.t. $\mathcal{M}$, denoted $\textsc{CanSol}_{\mathcal{M}}(I)$, is obtained by applying all the s-t tgds of $\mathcal{M}$ "in parallel". More precisely, we start with an empty target instance J. Then, for each s-t tgd $q_1(\bar{x}) \to \exists \bar{y}\, \psi(\bar{x}, \bar{y})$ of $\mathcal{M}$, each tuple $\bar{a} \in q_1(I)$, and each $i \in \{1, \ldots, |\bar{a}|_{q_1(I)}\}$, we pick a tuple $\bar{Y}_i$

[2]Recall that the notation $\text{i-certain}_{\mathcal{M}}(q, I) = m$ denotes the fact that $\text{i-certain}_{\mathcal{M}}(q, I) = \{() : m\}$.

of pairwise distinct fresh nulls, and increase the multiplicity of each atom of $\psi(\bar{a}, \bar{Y}_i)$ in J by 1. The resulting target instance is the canonical solution $\textsc{CanSol}_{\mathcal{M}}(I)$.

It is easy to see that $\textsc{CanSol}_{\mathcal{M}}(I)$ is uniquely defined up to renaming of nulls, and that it is both an i-solution and a c-solution for I w.r.t. $\mathcal{M}$.

Example 7. Let $\mathcal{M}$ be the GLAV mapping specified by the following s-t tgds (universal quantifiers are omitted):

$$\begin{aligned} d_1 &: \; P(x) \to Q(x) \wedge R(x,x) \wedge Q(x), \\ d_2 &: \; P(x) \to \exists y \left(Q(x) \wedge R(x,y) \wedge Q(y)\right). \end{aligned}$$

We want to construct $\textsc{CanSol}_{\mathcal{M}}(I)$ for $I = \{P(a) : 2\}$. Let J to be empty target instance. We apply the tgd d_1 two times with $a \in P^I$. In the first round, we add the atoms $Q(a)$ and $R(a,a)$, that is, we increase their multiplicity to 1. In the second round, we increase the multiplicity of $Q(a)$ and $R(a,a)$ by one once more, so we obtain two occurrences of $Q(a)$ and $R(a,a)$. Similarly, we apply the tgd d_2 two times with $a \in P^I$. In the first round, we pick a fresh null Y_1, and increase the multiplicities of $Q(a)$, $R(a, Y_1)$, and $Q(Y_1)$ by 1. In the second round, we pick another fresh null Y_2, and increase the multiplicities of $Q(a)$, $R(a, Y_2)$, and $Q(Y_2)$ by 1. Altogether, we obtain

$$J = \{Q(a) : 4, R(a,a) : 2\} \cup \{R(a, Y_i), Q(Y_i) \mid i \in \{1,2\}\}$$

as the canonical solution $\textsc{CanSol}_{\mathcal{M}}(I)$.

It is worth pointing out that the set-instance J obtained from $\textsc{CanSol}_{\mathcal{M}}(I)$ by eliminating duplicates coincides with the notion of a canonical universal solution defined in [24], and also with the result of the naive chase [28]. Under set semantics, the certain answers of a Boolean CQ q can be obtained by first evaluating q on the canonical universal solution J and then removing all tuples containing at least one null. In contrast, it is not true in general that, under bag semantics, the certain answers of q can be computed by evaluating q on $\textsc{CanSol}_{\mathcal{M}}(I)$. To see this, observe that in Example 6 we have that $\textsc{CanSol}_{\mathcal{M}}(I) = J_3$, but $q_3(J_3) = 4 \neq 3 = \text{i-certain}_{\mathcal{M}}(q_3, I) = \text{c-certain}_{\mathcal{M}}(q_3, I)$.

Our procedure for computing the certain answers to Boolean CQs makes essential use of the fact that all minimal solutions can be obtained from the canonical solution in the way made precise next.

An i-solution J for a source instance I w.r.t. a schema mapping $\mathcal{M}$ is called *minimal* if there is no i-solution J' for I w.r.t. $\mathcal{M}$ with $J' \subsetneq J$.[3] Minimal c-solutions are defined analogously. Note that the answers to q on all minimal solutions are enough to compute the certain answers to q.

Given an instance K and a function $h\colon \text{adom}(K) \to \mathsf{Dom}$, we let $h(K)$ be the instance

$$h(K) := \{\beta : m \mid \beta = h(\alpha) \text{ for some fact } \alpha \in K, \text{ and } m = \textstyle\sum_{\alpha \in K, h(\alpha) = \beta} |\alpha|_K\},$$

[3]Here, $\subsetneq$ denotes strict inclusion under bag semantics, i.e., $|\alpha|_{J'} \leq |\alpha|_J$ for all facts α, and there is at least one fact β with $|\beta|_{J'} < |\beta|_J$.

where for all facts $\alpha = R(a_1, \ldots, a_k)$, we define $h(\alpha) := R(h(a_1), \ldots, h(a_k))$. A function $h\colon \text{adom}(K) \to \mathsf{Dom}$ is said to be *constant-preserving* if for all constants a in $\text{adom}(K)$, we have that $h(a) = a$.

Lemma 1. *Let $\mathcal{M}$ be a GLAV mapping, let I be a source instance, and let $J_0 := \textsc{CanSol}_{\mathcal{M}}(I)$. For every minimal i-solution or c-solution J for I w.r.t. $\mathcal{M}$, there exists a constant-preserving function $h\colon \text{adom}(J_0) \to \mathsf{Dom}$ such that $J \subseteq h(J_0)$.*

As an immediate consequence of Lemma 1, we obtain a complexity-theoretic upper bound for the two decision problems associated with the computation of the certain answers under bag semantics.

Theorem 1. *Let $\mathcal{M}$ be a GLAV mapping and let q be a target Boolean CQ. Then the decision problems* i-QA$(\mathcal{M}, q)$ *and* c-QA$(\mathcal{M}, q)$ *are in* coNP.

B. Tractable Cases

We now show that the decision problems associated with the computation of the certain answers become tractable for natural special cases of GLAV mappings.

Theorem 2. *Let $\mathcal{M}$ be a schema mapping and let q be a target Boolean CQ.*

1) *If $\mathcal{M}$ is a GAV mapping, then the decision problem* i-QA$(\mathcal{M}, q)$ *is in* PTIME.
2) *If $\mathcal{M}$ is an elementary mapping, then the decision problem* c-QA$(\mathcal{M}, q)$ *is in* PTIME.

For the first part of Theorem 2, we use Lemma 1 to show that for every GAV mapping $\mathcal{M}$ and every source instance I, there is a unique minimal i-solution J for I w.r.t. $\mathcal{M}$, which in addition satisfies $J \subseteq \textsc{CanSol}_{\mathcal{M}}(I)$. The unique minimal i-solution for I w.r.t. $\mathcal{M}$ can be computed in polynomial time by starting with $J := \textsc{CanSol}_{\mathcal{M}}(I)$ and exhaustively applying the following rule: if there is a fact $\alpha \in J$ such that the instance J' obtained from J by decreasing the multiplicity of α by 1 is still an i-solution for I w.r.t. $\mathcal{M}$, then replace J by J'. This leads to a proof of the first part of Theorem 2.

The proof of the second part of Theorem 2 is based on the concept of a universal solution under bag semantics. We defer the definition of universal solutions and the proof to Section V.

We note that 'GAV mapping' in the first part of Theorem 2 cannot be replaced by 'elementary mapping'. In the next section, we establish coNP-hardness lower bounds for the cases that are not covered by Theorem 2.

C. Lower Bounds

We just saw that, as regards GAV mappings, computing the incognizant or cognizant certain answers of CQs is a tractable problem. We now show that these problems may become coNP-complete if we consider full mappings, that is, GLAV mappings in which the heads of the s-t tgds are quantifier-free but may contain more than one atoms.

Theorem 3. *There exists a full mapping $\mathcal{M}$ and a target Boolean CQ q such that* i-QA$(\mathcal{M}, q)$ *and* c-QA$(\mathcal{M}, q)$ *are* coNP*-complete. Moreover, $\mathcal{M}$ is specified by one s-t tgd with a two-atom head and one Horn s-t tgd.*

Proof. We construct $\mathcal{M}$ and q such that the problem Positive Not-All-Equal 3-Sat (Pnae3Sat) is polynomial-time reducible to the complement of i-QA$(\mathcal{M}, q)$. Moreover, $\mathcal{M}$ satisfies the hypothesis of Proposition 1, hence the i-certain answers coincide with the c-certain answers. Given a propositional formula $\varphi = \bigwedge_{i=1}^{m}(x_1^i \vee x_2^i \vee x_3^i)$, where each x_j^i is a variable, Pnae3Sat asks whether there is a truth assignment $\alpha\colon \operatorname{var}(\varphi) \to \{t, f\}$ such that not all variables of a clause have the same truth value (i.e., for each $i \in \{1, \dots, m\}$, we have $|\{j \in \{1,2,3\} \mid \alpha(x_j^i) = t\}| \in \{1, 2\}$). If this is the case, we call φ *nae-satisfiable*, and α a *nae-assignment* for φ. It is known that Pnae3Sat is NP-complete [27].

We first describe how we encode a formula φ as an instance I_φ. Let $\varphi = \bigwedge_{i=1}^{m}(x_1^i \vee x_2^i \vee x_3^i)$. Then, I_φ consists of the following atoms:

- $V(x, t, f)$ with multiplicity 2, for each variable x of φ;
- $C(x_1^i, x_2^i, x_3^i)$ with multiplicity 1, for $i \in \{1, \dots, m\}$.

Here, we regard the variables of φ as constants, and use distinguished constants t and f to denote 'true' and 'false'.

We now turn to the construction of $\mathcal{M}$ and q. The source schema of $\mathcal{M}$ consists of V and C, while its target schema consists of a binary relation symbol A and a copy C' of C. The s-t tgds of $\mathcal{M}$ are:

$$\begin{aligned} V(x, y_t, y_f) &\to A(x, y_t) \wedge A(x, y_f), \\ C(x_1, x_2, x_3) &\to C'(x_1, x_2, x_3). \end{aligned}$$

The first s-t tgd assigns the truth value t or f to each variable x of φ. This is done by assigning the multiplicity 2 of the atom $V(x, t, f)$ in I_φ to either $A(x, t)$ or $A(x, f)$, and a multiplicity of 1 to the other atom, in a minimal i-solution for I_φ w.r.t. $\mathcal{M}$. We will interpret the fact that $A(x, t)$ has multiplicity 2 as x being true, and likewise for $A(x, f)$. We then consider the query

$$q := \exists x_1 \exists x_2 \exists x_3 \exists v \left(C'(x_1, x_2, x_3) \wedge \bigwedge_{j=1}^{3} A(x_j, v) \right)$$

that essentially counts the number of clauses in which all variables have the same truth value.

Let $m_\varphi := 6m + 1$. We show that φ is nae-satisfiable if, and only if, i-certain$_\mathcal{M}(q, I_\varphi) < m_\varphi$.

We first show that there is a one-to-one correspondence between truth assignments for φ and minimal i-solutions for I_φ w.r.t. $\mathcal{M}$. Let α be a truth assignment for φ. Then,

$$\begin{aligned} J_\alpha \;=\; &\{A(x,t) : 2, A(x,f) : 1 \mid x \in \operatorname{var}(\varphi),\, \alpha(x) = 1\} \\ \cup\; &\{A(x,t) : 1, A(x,f) : 2 \mid x \in \operatorname{var}(\varphi),\, \alpha(x) = 0\} \\ \cup\; &\{C'(x_1^i, x_2^i, x_3^i) : 1 \mid 1 \le i \le m\} \end{aligned}$$

is a minimal i-solution for I_φ w.r.t. $\mathcal{M}$. It is not hard to see that for every minimal i-solution J, there is a truth assignment α with $J = J_\alpha$.

Next, we take a closer look at the possible answers to q on each minimal i-solution for I_φ w.r.t. $\mathcal{M}$. Let α be a truth assignment for φ. It is easy to verify that

$$q(J_\alpha) \;=\; \sum_{i=1}^{m} \sum_{v \in \{t,f\}} \prod_{j=1}^{3} |A(x_j^i, v)|_{J_\alpha}. \tag{6}$$

Let $k_i := |\{j \in \{1,2,3\} \mid \alpha(x_j^i) = 1\}|$ for $1 \le i \le m$. Then,

$$\sum_{v \in \{t,f\}} \prod_{j=1}^{3} |A(x_j^i, v)|_{J_\alpha} \;=\; 2^{k_i} + 2^{3-k_i},$$

which is 9 if $k_i \in \{0, 3\}$, and 6 if $k_i \in \{1, 2\}$. Thus, letting s_α be the number of $i \in \{1, \dots, m\}$ with $k_i \in \{1, 2\}$, we can rewrite (6) as

$$q(J_\alpha) = 6s_\alpha + 9(m - s_\alpha) = 9m - 3s_\alpha. \tag{7}$$

We are now ready to prove that φ is nae-satisfiable iff i-certain$_\mathcal{M}(q, I_\varphi) < m_\varphi$.

For the "only if" direction, let α be a nae-assignment for φ. Then, $s_\alpha = m$, so that (7) yields i-certain$_\mathcal{M}(q, I_\varphi) \le q(J_\alpha) = 6m < m_\varphi$.

For the "if" direction, let J be a minimal i-solution for I_φ w.r.t. $\mathcal{M}$ with $q(J) < m_\varphi$, and let α be a truth assignment for φ with $J = J_\alpha$. By (7), we have that $6m = m_\varphi - 1 \ge q(J) = 9m - 3s_\alpha$. Hence, $s_\alpha \ge m$, which implies that α is a nae-assignment for φ. ☐

The proof of Theorem 3 actually yields a stronger result, namely, that the schema mapping $\mathcal{M}$ and the query q constructed in the course of the proof can be used to solve a *maximization version* of the Pnae3Sat problem. Given a Pnae3Sat instance φ, let s_φ be the maximum number of clauses of φ that can be simultaneously nae-satisfied by a single truth assignment for φ (i.e., s_φ is the maximum of s_α, where α ranges over all truth assignments for φ, and s_α is defined as in the proof of Theorem 3). Then, certain$_\mathcal{M}(q, I_\varphi) = 9m - 3s_\varphi$, where m is the number of clauses of φ. It follows that s_φ can be computed as $(9m - \text{certain}_\mathcal{M}(q, I_\varphi))/3$. This observation raises the question about connections between the approximability properties of optimization problems and the computation of the certain answers of conjunctive queries with respect to schema mappings under bag semantics. We leave this question as a direction of future research.

It remains to consider the extension of GAV mappings by s-t tgds whose heads contain existential quantifiers.

Theorem 4. *There exists an elementary mapping $\mathcal{M}$ and a target Boolean CQ q such that* i-QA$(\mathcal{M}, q)$ *is* coNP*-complete. Moreover, $\mathcal{M}$ is specified by two elementary tgds, and five Horn tgds.*

Proof. (*Sketch*) As in Theorem 3, we construct $\mathcal{M}$ and q in such a way that Pnae3Sat is polynomial-time reducible to

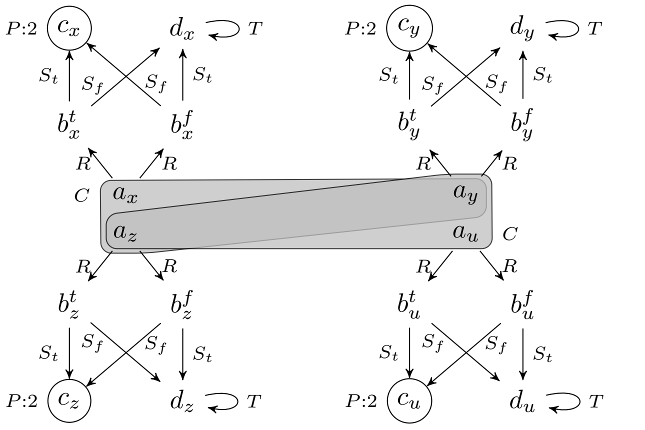

Fig. 1. The instance I_φ for $\varphi = (x \vee y \vee z) \wedge (y \vee z \vee u)$. For each variable w, the R-labeled edge from a_w to b_w^v represents the atom $R(v, a_w, b_w^v)$, the label $P:2$ at c_w means that c_w belongs to P with multiplicity 2, and the T-labeled loop at d_w represents $T(d_w, d_w, d_w)$. The gray areas containing a_x, a_y, a_z and a_y, a_z, a_u represent the atoms $C(a_x, a_y, a_z)$ and $C(a_y, a_z, a_u)$, respectively.

the complement of i-QA$(\mathcal{M}, q)$. The construction is much more elaborate than the one in the proof of Theorem 3, due to the restriction to s-t tgds with single-atom heads.

To encode a PNAE3SAT instance $\varphi = \bigwedge_{i=1}^{m}(x_1^i \vee x_2^i \vee x_3^i)$ as an instance I_φ, we represent each variable x of φ by five constants a_x, b_x^t, b_x^f, c_x, and d_x, and connect these constants by adding the following atoms to I_φ:

- $R(v, a_x, b_x^v)$ with multiplicity 1, for each $v \in \{t, f\}$;
- $S_t(b_x^t, c_x)$ and $S_f(b_x^t, d_x)$ with multiplicity 1;
- $S_t(b_x^f, d_x)$ and $S_f(b_x^f, c_x)$ with multiplicity 1;
- $T(d_x, d_x, d_x)$ with multiplicity 1;
- $P(c_x)$ with multiplicity 2.

In addition, I_φ contains the atom $C(a_{x_1^i}, a_{x_2^i}, a_{x_3^i})$ with multiplicity 1, for each clause $x_1^i \vee x_2^i \vee x_3^i$ of φ. See Figure 1 for an illustration.

The schema mapping $\mathcal{M}$ has $\mathbf{S} = \{P, R, S_t, S_f, C, T\}$ as its source schema; its target schema consists of copies R', S_t', S_f', C', T' of the relation symbols in $\mathbf{S} \setminus \{P\}$. The s-t tgds of $\mathcal{M}$ are:

$$\begin{aligned} &P(x) \to \exists y\, T'(x, x, y);\\ &P(x) \to \exists z\, T'(x, z, x);\\ &U(\bar{x}) \to U'(\bar{x}), \quad \text{for each } U \in \mathbf{S} \setminus \{P\}.\end{aligned}$$

The first two s-t tgds assign a truth value to each variable x of φ, depending on the pattern of T'-atoms generated by them. In a minimal i-solution J for I_φ, the "loop" $T'(c_x, c_x, c_x)$ can occur at most two times. If it occurs two times, then there is no other atom of the form $T'(c_x, c_x, a)$, $a \neq c_x$, which means that the answers to the queries

$$\begin{aligned} \theta_t(x) &:= \exists x' \exists x''\, T'(x, x', x''),\\ \theta_f(x) &:= \exists x' \exists x'' \big(T'(x, x, x') \wedge T'(x, x'', x')\big)\end{aligned}$$

on J contain c_x with multiplicity 2 and 4, respectively. The other extreme is that J does not contain $T'(c_x, c_x, c_x)$. One can show that in this case we only need to focus on the case that J contains exactly two atoms of the form $T'(c_x, a, c_x)$ and two atoms of the form $T'(c_x, c_x, b)$, each with multiplicity 1. Thus, the multiplicities of c_x in the answers to θ_t and θ_f on J are swapped, that is, 4 and 2, respectively. We use θ_t and θ_f to assign a truth value to each variable x, based on whether $T'(c_x, c_x, c_x)$ occurs with multiplicity 2 or not.

The desired query q uses the subquery

$$\theta(x, v) := \exists y \exists z_t \exists z_f \big(R'(v, x, y) \wedge \bigwedge_{u \in \{t,f\}} (S_u'(y, z_u) \wedge \theta_u(z_u))\big)$$

to switch between θ_t and θ_f, depending on the value of v. In fact, it is straightforward to verify that in a minimal i-solution J for I_φ, for each variable x of φ and each $v \in \{t, f\}$ we have $|(a_x, v)|_{\theta(J)} = |c_x|_{\theta_v(J)}$. We now obtain q from the query in Theorem 3 by replacing each atom of the form $A(x, v)$ with $\theta(x, v)$:

$$q := \exists x_1 \exists x_2 \exists x_3 \exists v \left(C'(x_1, x_2, x_3) \wedge \bigwedge_{j=1}^{3} \theta(x_j, v)\right).$$

One can show that φ is nae-satisfiable if, and only if, i-certain$_{\mathcal{M}}(q, I_\varphi) < 48m + 1$. □

V. Data Exchange

The goal of data exchange is to materialize a "good" solution J for a given source instance I. Under set semantics, the notion of a "good" solution was made precise in [11] via the notion of a *universal* solution for I w.r.t. a schema mapping $\mathcal{M}$. Intuitively, a universal solution J for I w.r.t. $\mathcal{M}$ is a "most general" solution for I, in the sense that, for every other solution J' for I w.r.t. $\mathcal{M}$, there is a homomorphism from J to J'. Moreover, every universal solution J is *CQ-adequate* for I w.r.t. $\mathcal{M}$, which means that the certain answers of any conjunctive query q w.r.t. $\mathcal{M}$ on I can be obtained by first evaluating q on J and then removing all tuples with nulls from the set of answers. If $\mathcal{M}$ is a GLAV mapping, then every source instance has a universal solution. Moreover, it can be shown that, in this case, the universal solutions for I w.r.t. $\mathcal{M}$ are precisely the target instances that are CQ-adequate for I w.r.t. $\mathcal{M}$.

In this section, we show that the state of affairs changes dramatically once we turn to bag semantics. We start by introducing a suitable notion of universal solution.

A. Universal Solutions Under Bag Semantics

Under set semantics, a solution J for a source instance I w.r.t. a schema mapping $\mathcal{M}$ is *universal* if for every solution J' for I w.r.t. $\mathcal{M}$, there is a homomorphism from J to J', where a homomorphism from J to J' is a mapping $h\colon \text{adom}(J) \to \text{adom}(J')$ such that $R(\bar{a}) \in I$ implies $R(h(\bar{a})) \in J$, and h is the identity on constants. Such a solution is particularly useful for query answering, since the certain answers to any CQ $q(\bar{x})$ can be computed as $q(J)_\downarrow$, where $q(J)_\downarrow$ is obtained from $q(J)$ by removing all tuples that contain at least one null. The following example shows that this notion of universal solution is not suitable for bag semantics.

Example 8. Let $\mathcal{M}$ be the GLAV mapping specified by the s-t tgd $R(x,y) \to \exists z(S(x,z) \wedge S(z,y))$ from Example 6. Then, $J = \{S(a,X):1, S(X,b):1, S(a,Y):1, S(Y,b):1\}$ is both an i-solution and a c-solution for the source instance $I = \{R(a,b):2\}$. It is easy to see that for every i-solution (resp., c-solution) J' for I, there is a homomorphism from J to J', but it is not the case that the i-certain answers or the c-certain answers to any CQ $q(\bar{x})$ can be obtained by computing $q(J)_{\downarrow}$. For instance, consider the CQ $q(x) := \exists y\, S(x,y)$. Then, $|a|_{q(J)} = 2$, but $a:1 \in$ i-certain$_{\mathcal{M}}(q,I) =$ c-certain$_{\mathcal{M}}(q,I)$, since $K = \{S(a,X):1, S(X,b):2\}$ is an i-solution and a c-solution for I.

The reason for this behavior is that homomorphisms do not preserve CQs under bag semantics. Given instances I, J and a homomorphism h, we say that h *preserves* a CQ $q(\bar{x})$ if for all tuples $\bar{a} \in q(I)$, we have $|\bar{a}|_{q(I)} \le |h(\bar{a})|_{q(J)}$; we say that h *preserves CQs* if h preserves every CQ. It follows from Example 8 that homomorphisms do not preserve CQs. We can strengthen this observation as follows. A bag homomorphism [16] from an instance I to an instance J is a homomorphism h from I to J such that for all facts $R(\bar{a})$, we have $|R(\bar{a})|_I \le |R(h(\bar{a}))|_J$. Note that there is a bag homomorphism from the instance J to the instance K in Example 8, but that this homomorphism does not preserve $q(x) = \exists y\, S(x,y)$. It turns out that the following stricter condition on homomorphisms is sufficient to preserve CQs.

Definition 4. Let I and J be instances. A homomorphism h from I to J is *additive* if for all atoms $\beta \in J$,

$$\sum_{\substack{\alpha \in I \\ h(\alpha)=\beta}} |\alpha|_I \le |\beta|_J.$$

Lemma 2. *If h is an additive homomorphism from an instance I to an instance J, then h preserves CQs.*

Proof. Let $q(\bar{x}) = \exists \bar{y} \bigwedge_{i=1}^{n} R_i(\bar{z}_i)$ be a CQ, and let $\bar{a} \in q(I)$. Let M_I be the set of all matches μ of q in I with $\mu(\bar{x}) = \bar{a}$. Then,

$$|\bar{a}|_{q(I)} = \sum_{\mu \in M_I} \prod_{i=1}^{n} |\mu(\bar{z}_i)|_{R_i^I}. \tag{8}$$

Now, let M_J be the set of all matches μ' of q in J with $\mu'(\bar{x}) = h(\bar{a})$. Observe that for every match $\mu \in M_I$, the composition $\mu' := h \circ \mu$ belongs to M_J. Thus, the sets $M_{I,\mu'} := \{\mu \in M_I \mid h \circ \mu = \mu'\}$, where μ' ranges over the matches in M_J, form a partition of M_I (note, however, that some sets of the partition may be empty). This allows us to rewrite (8) as:

$$|\bar{a}|_{q(I)} = \sum_{\mu' \in M_J} \sum_{\mu \in M_{I,\mu'}} \prod_{i=1}^{n} |\mu(\bar{z}_i)|_{R_i^I}. \tag{9}$$

Let us focus on a match $\mu' \in M_J$. It is easy to see that the inner summation in (9) can be upper-bounded by:

$$\sum_{\mu \in M_{I,\mu'}} \prod_{i=1}^{n} |\mu(\bar{z}_i)|_{R_i^I} \le \prod_{i=1}^{n} \sum_{\mu \in M_{I,\mu'}} |\mu(\bar{z}_i)|_{R_i^I}. \tag{10}$$

Furthermore, for each $i \in \{1,\dots,n\}$, the fact that h is an additive homomorphism from I to J, and that $h \circ \mu = \mu'$ for each $\mu \in M_{I,\mu'}$ implies:

$$\sum_{\mu \in M_{I,\mu'}} |\mu(\bar{z}_i)|_{R_i^I} \le |h(\mu(\bar{z}_i))|_{R_i^J} = |\mu'(\bar{z}_i)|_{R_i^J}. \tag{11}$$

Altogether, (9)–(11) imply that:

$$|\bar{a}|_{q(I)} \le \sum_{\mu' \in M_J} \prod_{i=1}^{n} |\mu'(\bar{z}_i)|_{R_i^J} = |h(\bar{a})|_{q(J)},$$

as desired. □

We now define universal solutions in terms of additive homomorphisms.

Definition 5. Let $\mathcal{M}$ be a GLAV mapping, and let I be a source instance. A *universal i-solution* for I w.r.t. $\mathcal{M}$ is an i-solution J for I w.r.t. $\mathcal{M}$ such that for all i-solutions J' for I in $\mathcal{M}$, there is an additive homomorphism from J to J'. *Universal c-solutions* are defined analogously.

A target instance J is called *CQ-adequate for I w.r.t. $\mathcal{M}$ under the incognizant semantics* if for every target CQ $q(\bar{x})$, we have i-certain$_{\mathcal{M}}(q,I) = q(J)_{\downarrow}$. It can be shown that every such target instance is an i-solution for I w.r.t. $\mathcal{M}$, thus in the following we call such instances *CQ-adequate i-solutions*. We define *CQ-adequate c-solutions* analogously.

Proposition 2. *Let $\mathcal{M}$ be a GLAV mapping, and let I be a source instance. Every universal i-solution (resp., c-solution) for I w.r.t. $\mathcal{M}$ is CQ-adequate. Moreover, every universal i-solution (resp., c-solution) is unique up to renaming of nulls.*

Proof. CQ-adequacy follows from Lemma 2. Uniqueness follows from the fact that if there is an additive homomorphism from I to J and a homomorphism from J to I, then I and J are isomorphic. □

Under set semantics, GLAV mappings have the property that every source instance has a CQ-adequate instance. The results of the previous section imply that, for GLAV mappings under bag semantics, it is not, in general, possible to compute such an instance in polynomial time, even if one exists. In fact, for the schema mappings $\mathcal{M}$ constructed in the proofs of Theorems 3 and 4, there is no polynomial-time algorithm that takes a source instance I for $\mathcal{M}$ as input, and computes a CQ-adequate i-solution (or c-solution in the case of the schema mapping in the proof of Theorem 3) for I w.r.t. $\mathcal{M}$, unless $\mathsf{PTIME} = \mathsf{coNP}$. We strengthen this result by showing that there are simple GLAV mappings such that essentially none of their source instances admits a CQ-adequate solution.

Theorem 5. *The following statements are true.*

1) *There exists a full mapping $\mathcal{M}$ specified by a single full s-t tgd such that no source instance I containing at least one fact with multiplicity at least 2 has a CQ-adequate i-solution or c-solution w.r.t. $\mathcal{M}$.*
2) *There exists an elementary mapping $\mathcal{M}$ such that no non-empty source instance has a CQ-adequate i-solution w.r.t. $\mathcal{M}$.*

Note that the schema mappings used in Theorem 5 go only slightly beyond the class of GAV mappings. The first part of the theorem shows that full tgds with more than one atom on the head are capable of destroying the existence of CQ-adequate i-solutions and c-solutions, whereas the second part shows that tgds whose head contains only a single atom, but with existentially quantified variables are capable of destroying the existence of CQ-adequate i-solutions. In contrast, if we restrict our attention to GAV mappings $\mathcal{M}$, then every source instance I has a CQ-adequate i-solution and c-solution w.r.t. $\mathcal{M}$. Indeed, as we have argued in Section IV-B, the unique minimal i-solution (resp., c-solution) contained in $\textsc{CanSol}_{\mathcal{M}}(I)$ is CQ-adequate for I w.r.t. $\mathcal{M}$. We now show that, for elementary mappings, the canonical solution is a universal c-solution, hence also a CQ-adequate c-solution.

Theorem 6. *For every elementary mapping $\mathcal{M}$ and every source instance I, the canonical solution for I w.r.t. $\mathcal{M}$ is a universal c-solution for I w.r.t. $\mathcal{M}$.*

By combining Proposition 2 and Theorem 6, we obtain a proof of the second part of Theorem 2 in Section IV.

Proof of Theorem 2 (2). Let $\mathcal{M}$ be an elementary mapping, and let $q(\bar{x})$ be a CQ. Given a source instance I for an elementary mapping $\mathcal{M}$, we first compute $\textsc{CanSol}_{\mathcal{M}}(I)$ and then output $q(\textsc{CanSol}_{\mathcal{M}}(I))_{\downarrow}$. By Theorem 6, we know that $\textsc{CanSol}_{\mathcal{M}}(I)$ is a universal c-solution for I, hence, by Proposition 2, we have that $q(\textsc{CanSol}_{\mathcal{M}}(I))_{\downarrow} =$ c-certain$_{\mathcal{M}}(q, I)$. It is easy to see that this algorithm runs in time polynomial in the size of I. □

B. CQ-Adequate Bases

In the previous section, we saw that CQ-adequate instances might not exist even for very simple GLAV mappings under bag semantics. A similar situation also arises under set semantics, but in contexts in which more complex schema mappings are used, such as schema mappings in peer data exchange [12] and in data exchange with arithmetic comparisons [1], [29]. One approach to deal with this problem is to relax the notion of a CQ-adequate instance to that of a CQ-universal basis.

Definition 6. Let $\mathcal{M}$ be a schema mapping and I a source instance. An *incognizant CQ-adequate basis for I w.r.t. $\mathcal{M}$* is a set $\mathcal{B}$ of target instances such that for every target conjunctive query $q(\bar{x})$, we have that

$$\text{i-certain}_{\mathcal{M}}(q, I) = \bigcap\{q(J)_{\downarrow} \mid J \in \mathcal{B}\}.$$

We define a *cognizant CQ-adequate basis for I w.r.t. $\mathcal{M}$* analogously, using c-certain$_{\mathcal{M}}(q, I)$ instead of i-certain$_{\mathcal{M}}(q, I)$.

It can be shown that all instances in an incognizant CQ-adequate basis w.r.t. a GLAV mapping are i-solutions, and similarly for cognizant CQ-adequate bases.

We now examine the notion of an incognizant and cognizant CQ-adequate basis. Our first result shows that all GLAV mappings admit an incognizant and a cognizant CQ-adequate basis, and that such a basis can be computed by an exponential-time algorithm.

Theorem 7. *For every GLAV mapping $\mathcal{M}$, there is an exponential-time algorithm that takes a source instance I as input and outputs an incognizant CQ-adequate basis for I w.r.t. $\mathcal{M}$. An analogous result holds for computing a cognizant CQ-adequate basis.*

Proof. The set $S_{\min}$ of all minimal i-solutions for I w.r.t. $\mathcal{M}$ is an incognizant CQ-adequate basis for I w.r.t. $\mathcal{M}$. We can compute $S_{\min}$ by first computing $J_0 := \textsc{CanSol}_{\mathcal{M}}(I)$ in time polynomial in the size of I, and then generating the set of all minimal i-solutions J for I w.r.t. $\mathcal{M}$ such that $J \subseteq h(J_0)$ for some mapping $h\colon \text{adom}(J_0) \to \text{adom}(J_0)$. This procedure runs in time exponential in the size of I. To compute a cognizant CQ-adequate basis, we proceed analogously with the set of all minimal c-solutions. □

In general, the exponential running time for constructing a CQ-adequate basis is unavoidable, even for full mappings. Our final result shows that there is a full mapping such that every incognizant or cognizant CQ-adequate basis has exponential size on infinitely many source instances, and similarly for elementary mappings and incognizant CQ-adequate bases.

Theorem 8.

1) *There is a full mapping $\mathcal{M}$ with the property that for every positive integer n, there is a source instance I with $|I| = n$ such that every incognizant or cognizant CQ-adequate basis for I w.r.t. $\mathcal{M}$ has size at least 2^n.*
2) *There is an elementary mapping $\mathcal{M}$ with the property that for every positive integer n, there is a source instance I with $|I| = n$ such that every incognizant CQ-adequate basis for I w.r.t. $\mathcal{M}$ has size at least 2^n.*

VI. Concluding Remarks

In this paper, we developed the foundations of information integration under bag semantics, thus taking the first step towards bridging the gap between theory and practice in this area of data management. The technical results established here, as summarized in Figure 2, reveal that the adoption of this more realistic semantics comes at a price; in particular, algorithmic problems about GLAV mappings that were tractable under set semantics may become intractable under bag semantics, even for the special case of full mappings under either incognizant or cognizant semantics, and the special case of elementary

	Incognizant Semantics		**Cognizant Semantics**	
Type of Mapping	Query Answering	Size of a CQ-Adequate Basis	Query Answering	Size of a CQ-Adequate Basis
GAV	PTIME	1	PTIME	1
Elementary	coNP-complete	$2^{n^{O(1)}}$	PTIME	1
Full	coNP-complete	$2^{n^{O(1)}}$	coNP-complete	$2^{n^{O(1)}}$

Fig. 2. Summary of Results

mappings under incognizant semantics. While such results appear to cast a specter of pessimism about information integration under bag semantics, we believe that they had to be discovered and articulated, so that the research in this area can advance further. In particular, it remains an open problem to investigate approximation algorithms for computing the certain answers of target queries in data exchange under bag semantics.

On the positive side, we showed that elementary mappings under cognizant semantics have tractable algorithmic behavior as regards both universal solutions and certain answers of conjunctive queries. It should be noted that, their syntactic simplicity notwithstanding, elementary mappings can express many, if not most, of the transformations used in Extract-Transform-Load (ETL) tools, such as the IBM InfoSphere DataStage[4] and the Oracle Warehouse Builder[5]. Actually, our original goal was to develop a rigorous framework for ETL via the systematic use of schema mappings. It did not take us long to realize, though, that a rigorous framework for ETL has to be preceded by a study of schema mappings and data exchange under bag semantics. And this led us to the work reported here.

Acknowledgments. This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing.

References

[1] F. Afrati, C. Li, and V. Pavlaki. Data exchange in the presence of arithmetic comparisons. In *Proc. of the 11th International Conference on Extending Database Technology*, EDBT '08, 2008.

[2] F. N. Afrati, M. Damigos, and M. Gergatsoulis. Query containment under bag and bag-set semantics. *Inf. Process. Lett.*, 110(10):360–369, 2010.

[3] J. Albert. Algebraic properties of bag data types. In *Proc. of the 17th International Conference on Very Large Data Bases (VLDB 1991)*, pages 211–219, 1991.

[4] M. Arenas, P. Barceló, L. Libkin, and F. Murlak. *Foundations of Data Exchange*. Cambridge University Press, 2014.

[5] C. Beeri and M. Y. Vardi. A proof procedure for data dependencies. *J. ACM*, 31(4):718–741, 1984.

[6] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proc. of the 9th ACM Symposium on Theory of Computing*, pages 77–90, 1977.

[7] S. Chaudhuri and M. Y. Vardi. Optimization of *Real* conjunctive queries. In *Proc. of the 12th ACM Symposium on Principles of Database Systems (PODS 1993)*, pages 59–70, 1993.

[8] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.

[4]http://www-03.ibm.com/software/products/en/ibminfodata

[5]http://www.oracle.com/technetwork/developer-tools/warehouse/overview/introduction/index.html

[9] S. Cohen. Equivalence of queries that are sensitive to multiplicities. *VLDB J.*, 18(3):765–785, 2009.

[10] A. Doan, A. Y. Halevy, and Z. G. Ives. *Principles of Data Integration*. Morgan Kaufmann, 2012.

[11] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: Semantics and query answering. *Theoretical Computer Science*, 336(1):89–124, 2005.

[12] A. Fuxman, P. G. Kolaitis, R. J. Miller, and W.-C. Tan. Peer data exchange. *ACM Trans. Database Syst.*, 31(4):1454–1498, Dec. 2006.

[13] P. W. P. J. Grefen and R. A. de By. A multi-set extended relational algebra - a formal approach to a practical issue. In *ICDE*, pages 80–88. IEEE Computer Society, 1994.

[14] S. Grumbach, L. Libkin, T. Milo, and L. Wong. Query languages for bags: expressive power and complexity. *SIGACT News*, 27(2):30–44, 1996.

[15] Y. E. Ioannidis and R. Ramakrishnan. Containment of conjunctive queries: Beyond relations as sets. *ACM Trans. Database Syst.*, 20(3):288–324, 1995.

[16] T. S. Jayram, P. G. Kolaitis, and E. Vee. The containment problem for REAL conjunctive queries with inequalities. In *Proc. of the 25th ACM Symposium on Principles of Database Systems (PODS 2006)*, pages 80–89, 2006.

[17] A. C. Klug. Equivalence of relational algebra and relational calculus query languages having aggregate functions. *J. ACM*, 29(3):699–717, 1982.

[18] A. C. Klug. On conjunctive queries containing inequalities. *J. ACM*, 35(1):146–160, 1988.

[19] D. E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms, 2nd Edition*. Addison-Wesley, 1981.

[20] P. G. Kolaitis. Schema mappings, data exchange, and metadata management. In *Proc. of the 24th ACM Symposium on Principles of Database Systems (PODS 2005)*, pages 61–75, 2005.

[21] P. G. Kolaitis, M. Lenzerini, and N. Schweikardt, editors. *Data Exchange, Integration, and Streams*, volume 5 of *Dagstuhl Follow-Ups*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[22] S. Kopparty and B. Rossman. The homomorphism domination exponent. *Eur. J. Comb.*, 32(7):1097–1114, 2011.

[23] M. Lenzerini. Data integration: A theoretical perspective. In *Proc. of the 21st ACM Symposium on Principles of Database Systems (PODS 2002)*, pages 233–246, 2002.

[24] L. Libkin. Data exchange and incomplete information. In *Proc. of the 25th ACM Symposium on Principles of Database Systems (PODS 2006)*, pages 60–69, 2006.

[25] I. S. Mumick, H. Pirahesh, and R. Ramakrishnan. The magic of duplicates and aggregates. In *16th International Conference on Very Large Data Bases (VLDB 1990)*, pages 264–277, 1990.

[26] Y. Sagiv and M. Yannakakis. Equivalences among relational expressions with the union and difference operators. *J. ACM*, 27(4):633–655, 1980.

[27] T. J. Schaefer. The complexity of satisfiability problems. In *Proc. of the 10th ACM Symposium on Theory of Computing*, STOC '78, pages 216–226, 1978.

[28] B. ten Cate, L. Chiticariu, P. Kolaitis, and W.-C. Tan. Laconic schema mappings: Computing the core with sql queries. *Proc. VLDB Endow.*, 2(1):1006–1017, Aug. 2009.

[29] B. ten Cate, P. G. Kolaitis, and W. Othman. Data exchange with arithmetic operations. In *Proc. of the Joint 2013 EDBT/ICDT Conferences*, pages 537–548, 2013.

[30] R. van der Meyden. The complexity of querying indefinite data about linearly ordered domains. *J. Comput. Syst. Sci.*, 54(1):113–135, 1997.