Stable Model Semantics for Tuple-Generating Dependencies Revisited

Stable Model Semantics for Tuple-Generating Dependencies Revisited

Mario Alviano
Dept. of Mathematics and CS
University of Calabria
alviano@mat.unical.it

Michael Morak
Inst. of Information Systems
TU Wien
morak@dbai.tuwien.ac.at

Andreas Pieris
School of Informatics
University of Edinburgh
apieris@inf.ed.ac.uk

ABSTRACT

Normal tuple-generating dependencies (NTGDs) are TGDs enriched with default negation, a.k.a. negation as failure. Query answering under NTGDs, where negation is interpreted according to the stable model semantics, is an intriguing new problem that gave rise to flourishing research activity in the database and KR communities. So far, all the existing works that investigate this problem, except for one recent paper that adopts an operational semantics based on the chase, follow the so-called logic programming (LP) approach. According to the LP approach, the existentially quantified variables are first eliminated via Skolemization, which leads to a normal logic program, and then the standard stable model semantics for normal logic programs is applied. However, as we discuss in the paper, Skolemization is not appropriate in the presence of default negation since it fails to capture the intended meaning of NTGDs, while the operational semantics mentioned above fails to overcome the limitations of the LP approach. This reveals the need to adopt an alternative approach to stable model semantics that is directly applicable to NTGDs with existentially quantified variables. We propose such an approach based on a recent characterization of stable models in terms of second-order logic, which indeed overcomes the limitations of the LP approach. We then perform an in-depth complexity analysis of query answering under prominent classes of NTGDs based on the main decidability paradigms for TGDs, namely weak-acyclicity, guardedness and stickiness. Interestingly, weakly-acyclic NTGDs give rise to robust and highly expressive query languages that allow us to solve in a declarative way problems in the second level of the polynomial hierarchy.

1. INTRODUCTION

Rule-based languages lie at the core of several areas of central importance to databases and artificial intelligence, such as data exchange and integration, deductive databases, and knowledge representation and reasoning, to name a few. The well-known *tuple-generating dependencies (TGDs)* [6] (a.k.a. *existential rules* [4] and *Datalog$^{\pm}$ rules* [9]) form a prominent rule-based formalism. TGDs are implications of the form $\forall \mathbf{X} \forall \mathbf{Y} (\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \psi(\mathbf{X}, \mathbf{Z}))$, where φ and ψ are conjunctions of (positive) atoms, and they essentially state that certain tuples in a database imply the presence of some other tuples in the database (hence the term "tuple-generating"). Notably, during the last decade, TGDs have found many uses and applications in different areas of database and AI research. For example, they have been used in formalizing and investigating inter-operability tasks such as data exchange [17]. TGDs have also been used for metadata management tasks, and in particular to formalize operations on schema mappings [16]. Finally, they have been employed for knowledge representation purposes, and in fact as an alternative way to model ontologies [4, 9].

The main algorithmic task that is relevant for the above applications is conjunctive query answering: given a database D, a set Σ of TGDs, a conjunctive query q, and a tuple $\mathbf{t}$ of constants, decide whether $(D, \Sigma) \models q(\mathbf{t})$, or, equivalently, whether each (possibly infinite) model of the logical theory $(D \wedge \Sigma)^1$ is also a model of $q(\mathbf{t})$. Unfortunately, query answering under TGDs is undecidable; see, e.g., [5], and [4, 7] for tight undecidable classes. This has led to a flurry of activity to identify syntactic restrictions on sets of TGDs that lead to decidable query answering; see, e.g., [4, 7, 8, 24]. In general, decidable classes of TGDs are based on the notions of weak-acyclicity [17], introduced in the context of data exchange, as well as guardedness [7] and stickiness [10], proposed in the context of ontological reasoning.

Although decidable classes of TGDs are well-suited for modeling positive information, none of them can express default negation, a.k.a. negation as failure, which is a key feature for deductive databases and knowledge representation.

Example 1. Consider the set Σ consisting of the rules

$$\begin{aligned}
&\forall X (\mathtt{person}(X) \rightarrow \exists Y\, \mathtt{hasFather}(X, Y))\\
&\forall X \forall Y (\mathtt{hasFather}(X, Y) \rightarrow \mathtt{sameAs}(Y, Y))\\
&\forall X \forall Y \forall Z (\mathtt{hasFather}(X, Y) \wedge \mathtt{hasFather}(X, Z) \wedge\\
&\qquad\qquad \neg \mathtt{sameAs}(Y, Z) \rightarrow \mathtt{abnormal}(X)),
\end{aligned}$$

which states that each person has at most one biological father; otherwise, (s)he is abnormal. The above rules cannot be equivalently expressed using (negation-free) TGDs. ■

One of the standard approaches to interpret negation is the stable model semantics [20], which is the main subject of this paper. Other approaches include perfect model [28] and well-founded semantics [19], which go beyond the scope of this work. Conjunctive query answering under TGDs extended with default negation, called *normal TGDs (NTGDs)*, w.r.t. the stable model semantics, is an intriguing new problem that gave rise to flourishing research activity in the database and KR communities. In [2, 25], acyclicity and stratification conditions for NTGDs have been considered, which give rise to formalisms that admit finite and/or unique stable models. In [3], a notion of stable models that can be directly applied to NTGDs is proposed, and NTGDs that satisfy certain acyclicity conditions are investigated. The classes that are based on guardedness and stickiness have been recently studied in [22] and [1], respectively. A closely related work is [15], which focuses on disjunctive logic programs, where the so-called $\mathbb{FDNC}$ programs are

^{1}By abuse of notation, D refers to the logical formula given by the conjunction of atoms occurring in D. This suffices since we adopt the open world assumption.

proposed. $\mathbb{FDNC}$ programs combine default negation with function symbols, and decidability is obtained by restricting the rule syntax to one of seven predefined forms.

The Logic Programming approach. All the above works, except for [3] that we discuss below, follow what we call the logic programming (LP) approach to stable model semantics for NTGDs. This means that the existentially quantified variables are first eliminated via Skolemization, which leads to a normal logic program, and then the standard stable model semantics for normal logic programs with function symbols is applied. Consider the set Σ of NTGDs given in Example 1. The first NTGD is replaced by the rule

$$\texttt{hasFather}(X, f(X)) \leftarrow \texttt{person}(X),$$

where f is a Skolem function, while all the other NTGDs of Σ can be directly conceived as normal rules since they do not have existentially quantified variables. The database $\{\texttt{person}(Alice)\}$ together with the obtained normal logic program have exactly one stable model M consisting of

$$\texttt{person}(Alice), \texttt{hasFather}(Alice, f(Alice)), \\ \texttt{sameAs}(f(Alice), f(Alice)).$$

It is easy to verify that the (Boolean) query

$$\exists X(\texttt{person}(X) \wedge \neg\texttt{abnormal}(X))$$

is entailed by M, while the query

$$\exists X(\texttt{person}(X) \wedge \texttt{abnormal}(X))$$

is refuted by M.

Is the LP Approach the Right One?

Although the LP approach has so far been considered as the standard approach to stable model semantics for NTGDs, we claim it is not the right way to reconcile existentially quantified variables with default negation. In fact, as we discuss below, we believe Skolemization is not appropriate in the presence of stable negation since it fails to capture the intuitive meaning of NTGDs. But let us first say a few words about Skolemization and (positive) TGDs.

Skolemization and $\neg$-free TGDs. In the absence of default negation, TGDs are essentially first-order theories, and conjunctive query answering can be reduced to a satisfiability check of a first-order formula. Given a database D, a set Σ of TGDs, a conjunctive query q, and a tuple of constants $\mathbf{t}$, $(D, \Sigma) \models q(\mathbf{t})$ iff $(D \wedge \Sigma \wedge \neg q(\mathbf{t}))$ is unsatisfiable. Furthermore, a classical result in first-order logic guarantees that the formula $(D \wedge \mathsf{sk}(\Sigma) \wedge \neg q(\mathbf{t}))$, where $\mathsf{sk}(\Sigma)$ is obtained after Skolemizing the TGDs of Σ, is equisatisfiable with $(D \wedge \Sigma \wedge \neg q(\mathbf{t}))$. Roughly, this is true since, for each tuple $\mathbf{u}$ of terms replacing the universally quantified variables occurring in a TGD σ, each existentially quantified variable Z occurring in σ can be satisfied by a different witness, which can be represented by $f_{\sigma,Z}(\mathbf{u})$, where $f_{\sigma,Z}$ is a Skolem function. Moreover, due to Herbrand's theorem, the satisfiability of a theory can be checked by considering only Herbrand interpretations, that is, interpretations associating constants and functions with themselves. Hence, whenever $\mathbf{u} \neq \mathbf{u}'$, it is safe to assume that $f(\mathbf{u})$ and $f(\mathbf{u}')$ are different objects. Similarly, we can assume that Skolem terms using different function symbols represent different objects, and also that they are different from any constant in the theory. Due to the above key results, it is easy to see that Skolemization can be safely applied in the absence of default negation.

Skolemization and normal TGDs. In contrast to $\neg$-free TGDs, Skolemization is not appropriate in the presence of default negation. In fact, a result which guarantees that Skolemization provides an equisatisfiable theory in the case of NTGDs is missing. This is illustrated by the following example.

Example 2. Let $D = \{\texttt{person}(Alice)\}$, and Σ be the set of NTGDs in Example 1. From D and Σ there is no evidence that Bob is not the father of $Alice$, and thus, it is intuitive to say that

$$q = \neg\texttt{hasFather}(Alice, Bob)$$

is not entailed by (D, Σ). In fact, it is natural to consider

$$\{\texttt{person}(Alice), \texttt{hasFather}(Alice, Bob), \\ \texttt{sameAs}(Bob, Bob)\},$$

as a stable model of $(D \wedge \Sigma)$, which does not satisfy q. However, $(D, \mathsf{sk}(\Sigma)) \models q$, where $\mathsf{sk}(\Sigma)$ is the set of normal rules obtained by Skolemizing Σ, since the unique stable model of $(D \wedge \mathsf{sk}(\Sigma))$, where $Bob \neq f(Alice)$, satisfies q. ■

From the above discussion, it is apparent that the LP approach to stable model semantics for NTGDs does not capture the intended meaning of NTGDs. This reveals the need to adopt an alternative approach to stable model semantics for NTGDs, which is more general than the LP approach in the sense that is directly applicable to normal rules with existentially quantified variables. This has also been recognized by other researchers, and there are attempts in the literature to resolve this issue. Notably, Baget et al. [3] propose a notion of stable models, which relies on the well-known chase procedure, that can be directly applied to NTGDs without requiring Skolemization. Roughly, given a database D and a set Σ of NTGDs, a (possibly infinite) set of atoms M is a stable model of $(D \wedge \Sigma)$ if it can be obtained by the chase procedure starting from D and applying the TGDs of Σ^+, where Σ^+ is obtained from Σ by eliminating the negative literals, under the assumptions that (i) all TGD applications are sound, i.e., none of the negative literals of a TGD can be found in M (or, in other words, the TGD is not blocked), and (ii) the chase is complete, i.e., all the applicable TGDs that are not blocked are eventually being applied. Although this is an interesting approach, there are still simple cases in which we infer unexpected answers. For instance, if we consider again Example 2, and we apply the operational semantics of [3], then we unexpectedly conclude that $(D, \Sigma) \models q$. The reason for this is the fact that the chase procedure always invents a new null value, but never a constant, in order to satisfy an existentially quantified variable in the head of a TGD. This implies that there is no way to have the atom $\texttt{hasFather}(Alice, Bob)$ in a stable model of $(D \wedge \Sigma)$, which in turn implies that $(D, \Sigma) \models q$.

Another promising approach to default negation, which avoids Skolemization, is the so-called *equality-friendly well-founded semantics (EFWFS)* [21]. Although this approach deviates from the idea of stable models, we would like to include it in our discussion since in some cases it yields the intended query answer. The key idea is that, given a database D and a set Σ of NTGDs, the meaning of (D, Σ) may be captured by the set of all normal programs Π obtained by (i) unifying constants occurring in D (note that the unique name assumption is not adopted), and (ii) replacing each NTGD $\sigma \in \Sigma$ by arbitrary instances of σ, at least one for each possible variable assignment for its body; an instance of a normal TGD $\forall\mathbf{X}\forall\mathbf{Y}(\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists\mathbf{Z}\,\psi(\mathbf{X}, \mathbf{Z}))$ is simply a rule $\varphi(\mathbf{a}, \mathbf{b}) \rightarrow \psi(\mathbf{a}, \mathbf{c})$, where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are tuples of constants. Let $\mathcal{I}(D, \Sigma)$ be the set of all programs Π obtained from (D, Σ) as described above. The equality-friendly well-founded models of (D, Σ) are defined as $\{ WFS(\Pi) \mid \Pi \in \mathcal{I}(D, \Sigma)\}$, where $WFS(\Pi)$ are the well-founded models of Π. Interestingly, if we apply the EFWFS to Example 2, then we get the expected answer. Unfortunately, as we explain below, this is not always the case.

Example 3. Let $D = \{\mathtt{person}(Alice)\}$, and Σ be the set of NTGDs given in Example 1. One expects that the query

$$q = \neg\mathtt{abnormal}(Alice)$$

is entailed by (D, Σ) since there is no evidence that $Alice$ has two biological fathers, and thus that she is abnormal. However, by following the EFWFS, q is not entailed. Observe that at least one normal program $\Pi \in \mathcal{I}(D, \Sigma)$ contains the rules

$$\mathtt{person}(Alice) \rightarrow \mathtt{hasFather}(Alice, Bob)$$
$$\mathtt{person}(Alice) \rightarrow \mathtt{hasFather}(Alice, John),$$

where $Bob \neq John$. Therefore, there exists an equality-friendly well-founded model of (D, Σ) where $Alice$ has two fathers, and thus the atom $\mathtt{abnormal}(Alice)$ is entailed. ■

Research Challenges. In order to overcome the limitations discussed previously, we focus on the following questions:

- How can we define a new approach to stable model semantics for NTGDs, such that it is directly applicable to rules with existentially quantified variables without requiring Skolemization?
- What is the data and combined complexity of conjunctive query answering under the main classes of NTGDs (based on weak-acyclicity, guardedness and stickiness) w.r.t. our new approach to stable models? Notice that the query may also contain negation.
- How is conjunctive query answering affected if, in addition, we allow disjunction to appear in rule-heads?
- How is the expressive power affected by allowing default negation and existentially quantified variables to coexist? Does disjunction in rule-heads increase the expressive power of the various formalisms?

Our Contributions

Our answers to the above questions, and the main contributions of the present paper, can be summarized as follows:

▶ We provide a precise definition of our new approach to stable models for normal TGDs, which in turn is based on a recent characterization of stable models via second-order logic [18]. The stable models of a database D and a set Σ of NTGDs are defined as the classical models of a second-order formula $\mathsf{SM}[D, \Sigma]$, which goes beyond MSO and encodes the vital properties of stable models. We formally show that our new approach is indeed a generalization of the LP approach (Theorem 1). We then establish a sufficient criterion for the decidability of query answering under stable models via the so-called *stable tree model property* (Theorem 2).

▶ We proceed to understand how conjunctive query answering behaves under NTGDs interpreted according to our new approach. We show that weak-acyclicity preserves the decidability of query answering (Theorem 3). However, if we focus on stickiness and guardedness, then query answering is undecidable (Theorems 4 and 5, respectively). Although for sticky NTGDs this was expected, for guarded NTGDs this is a rather surprising result. We show that the tree model property, the key property of guarded-based logics, is not preserved when stable negation and existentially quantified variables coexist. The reason for this can be found in the fact that we can "guess" an appropriate guard, whose domain can be restricted to an arbitrary set of values, and thus the property of guardedness that only atoms derived from a common guard ancestor can interact, is destroyed.

▶ Query answering under weakly-acyclic sets of NTGDs is Π_2^P-complete in data complexity, and coN2EXPTIME$^{\text{NP}}$-complete in combined complexity, even for predicates of bounded arity, and $\neg$-free atomic queries (Theorem 6).

▶ Interestingly, weak-acyclicity can be enriched with disjunction in the rule-heads without paying a price in terms of complexity (Theorem 12). The reason for this unexpected outcome is the fact that we can simulate disjunction using existentially quantified variables and stable negation.

▶ The class of weakly-acyclic NTGDs gives rise to powerful query languages that express exactly the queries with Π_2^P (resp., Σ_2^P) data complexity when the cautious (resp., brave) semantics is adopted (Theorem 17); this implies that disjunction in rule-heads does not add expressive power (Theorem 18). This expressivity result exposes an additional advantage of our approach compared to the LP approach, the operational semantics of [3], and the EFWFS of [21], that is, we obtain languages that can be used for declarative solving of problems that lie at the second level of the polynomial hierarchy. To justify this statement, we devise novel encodings for central problems in the second level of the polynomial hierarchy: (i) consistent query answering under weakly-acyclic TGDs relative to set-based repairs [30]; (ii) satisfiability for quantified Boolean formulas with two alternations of quantifiers (2-QBF); and (iii) an interesting variation of graph k-colorability, which generalizes the well-known problem CERT3COL [29].

2. PRELIMINARIES

General Definitions. We define the following pairwise disjoint countably infinite sets of symbols: a set $\mathbf{C}$ of *constants*, a set $\mathbf{N}$ of *labeled nulls* (used as placeholders for unknown values), and a set $\mathbf{V}$ of *variables* (used in queries and dependencies). Different constants represent different values (*unique name assumption*), while different nulls may represent the same value. We denote by $\mathbf{X}$ sequences (or sets, with a slight abuse of notation) of variables $X_1, \ldots, X_k$ with $k \geqslant 0$. Let $[n] = \{1, \ldots, n\}$, for any $n \geqslant 1$.

A (*relational*) *schema* $\mathcal{R}$ is a (finite) set of *relational symbols* (or *predicates*). We write p/n for the fact that p is an n-ary predicate. A *term* is a constant, null or variable. An *atomic formula* over $\mathcal{R}$ (or $\mathcal{R}$*-atom*) has the form $p(\mathbf{t})$, where $p \in \mathcal{R}$ and $\mathbf{t}$ is a tuple of terms. An $\mathcal{R}$*-literal* is either an $\mathcal{R}$-atom (i.e., a positive literal), or an $\mathcal{R}$-atom preceded by the negation symbol "$\neg$" (i.e., a negative literal). For a literal ℓ, we write $dom(\ell)$ for the set of its terms; this notation naturally extends to sets of literals. For brevity, conjunctions of literals are often identified with the sets of their literals. A (*total* or *two-valued*) $\mathcal{R}$*-interpretation* I is a set of $\mathcal{R}$-literals which contain only constants and nulls such that, for every $\mathcal{R}$-atom $p(t_1, \ldots, t_n)$, where $(t_1, \ldots, t_n) \in dom(I)^n$, either $p(t_1, \ldots, t_n) \in I$ or $\neg p(t_1, \ldots, t_n) \in I$. We write I^+ (I^-) for the set of positive (negative) literals of I.

A *homomorphism* from a set of literals L to a set of literals L' is a mapping $h : \mathbf{C} \cup \mathbf{N} \cup \mathbf{V} \rightarrow \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ that is defined on $dom(L)$ and is the identity on $\mathbf{C}$, and $p(t_1, \ldots, t_n) \in L$ (resp., $\neg p(t_1, \ldots, t_n) \in L$) implies $p(h(t_1), \ldots, h(t_n)) \in L'$ (resp., $\neg p(h(t_1), \ldots, h(t_n)) \in L'$).

A *database* D over a schema $\mathcal{R}$ is a finite set of $\mathcal{R}$-atoms such that $dom(D) \subset \mathbf{C}$. We sometimes treat a database D as a logical formula given by $\bigwedge_{p(\mathbf{t}) \in D} p(\mathbf{t})$, that is, the conjunction of atoms occurring in D. An $\mathcal{R}$-interpretation I is a *model* of D, denoted

$I \models D$, if there exists a homomorphism h such that $h(D) \subseteq I$. Since h is the identity on $dom(D)$, $I \models D$ iff $D \subseteq I$.

Normal TGDs. A *normal tuple-generating dependency (NTGD)* σ is a constant-free[2] first-order formula of the form

$$\forall \mathbf{X} \forall \mathbf{Y} (\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \psi(\mathbf{X}, \mathbf{Z})),$$

where φ (resp., ψ) is a conjunction of literals (resp., atoms), with variables from $\mathbf{X} \cup \mathbf{Y}$ (resp., $\mathbf{X} \cup \mathbf{Z}$). If there are no negative literals in φ, then σ is a TGD. We focus on *safe* NTGDs, i.e., every variable in a negative literal occurs also in a positive literal in φ. Formula φ is the *body* of σ, denoted $B(\sigma)$, while ψ is the *head* of σ, denoted $H(\sigma)$. The schema of a set Σ of NTGDs, denoted $sch(\Sigma)$, is the set of predicates occurring in Σ. We sometimes treat a set Σ of NTGDs as a logical formula given by $\bigwedge_{\sigma \in \Sigma} \sigma$, that is, the conjunction of NTGDs in Σ. A $sch(\Sigma)$-interpretation I is a model of an NTGD $\sigma \in \Sigma$, denoted $I \models \sigma$, if, whenever there exists a homomorphism h such that $h(B(\sigma)) \subseteq I$, then there exists $h' \supseteq h$, called *extension* of h, such that $h'(H(\sigma)) \subseteq I$. I is a *model* of Σ, denoted $I \models \Sigma$, if $I \models \sigma$ for each $\sigma \in \Sigma$. The class of all (finite) sets of NTGDs is denoted $\mathsf{TGD}^{\neg}$.

Normal (Boolean) Conjunctive Queries. An n-ary *normal conjunctive query (NCQ)* q over a schema $\mathcal{R}$, where $n \geqslant 0$, is a first-order formula of the form

$$\exists \mathbf{Y} \left(\bigwedge_{i=1}^{m} p_i(\mathbf{X}, \mathbf{Y}) \wedge \bigwedge_{j=m+1}^{m+k} \neg p_j(\mathbf{X}, \mathbf{Y}) \right),$$

where $m \geqslant 1$, $k \geqslant 0$, $\{p_i\}_{i \in [m+k]} \subseteq \mathcal{R}$, each atom contains variables from $(\mathbf{X} \cup \mathbf{Y}) \subset \mathbf{V}$ (and possibly constants of $\mathbf{C}$), and $|\mathbf{X}| = n$. A 0-ary NCQ is called *normal Boolean conjunctive query (NBCQ)*. We focus on *safe* queries, i.e., every variable in a negative literal occurs also in a positive literal. The *answer* to an n-ary NCQ q of the form $\exists \mathbf{Y}\, \varphi(\mathbf{X}, \mathbf{Y})$ over an interpretation I, denoted $q(I)$, is the set of all tuples $\mathbf{t} \in \mathbf{C}^n$ for which there exists a homomorphism h such that $h(\varphi(\mathbf{X}, \mathbf{Y})) \subseteq I$ and $h(\mathbf{X}) = \mathbf{t}$. A NBCQ q has only the empty tuple as possible answer, and it has a *positive* answer over I, denoted $I \models q$, if $q(I) \neq \varnothing$. The class of all (finite) normal (B)CQs is denoted $(\mathsf{B})\mathsf{CQ}^{\neg}$.

3. STABLE MODEL SEMANTICS: A NEW APPROACH

We proceed to introduce our new approach to stable model semantics for NTGDs that adopts a recent characterization of stable models in terms of second-order logic [18]. We first recall the logic programming (LP) approach considered so far (see, e.g., [1, 22, 25]). Since minimality is one of the key properties of stable models, we then recall, by means of a simple example, how the (subset) minimal models of a database D and a set Σ of NTGDs can be characterized via a second-order formula. We then proceed to explain why this formula fails to precisely capture the stable models of D and Σ, and how it can be modified in order to accurately encode the properties of stable models; this leads to our new approach to stable model semantics. Finally, we introduce the main problem tackled in this work, that is, query answering under NTGDs interpreted according to our new approach.

3.1 The Logic Programming Approach

Consider a database D, and a set $\Sigma \in \mathsf{TGD}^{\neg}$. Following the LP approach to stable model semantics for NTGDs, we first need to transform D and Σ into a normal logic program $\Pi_{D,\Sigma}$. If existentially quantified variables occur in the head of an NTGD of Σ, then they are eliminated via the standard process of *Skolemization*. The Skolemization of an NTGD σ of the form $\forall \mathbf{X} \forall \mathbf{Y} (\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \psi(\mathbf{X}, \mathbf{Z}))$ is the normal rule $\psi(\mathbf{X}, \mathbf{f}_\sigma(\mathbf{X}, \mathbf{Y})) \leftarrow \varphi(\mathbf{X}, \mathbf{Y})$, where $\mathbf{f}_\sigma$ is a vector of function symbols $f_{\sigma,Z}$, one for each $Z \in \mathbf{Z}$. Then, the so-called grounding of $\Pi_{D,\Sigma}$, denoted $ground(\Pi_{D,\Sigma})$, is computed by constructing all the ground instances of a rule ρ in $\Pi_{D,\Sigma}$, i.e., all the rules that can be constructed by replacing each variable of ρ with a term from the Herbrand universe of $\Pi_{D,\Sigma}$, that is, all the terms that can be formed using constants and function symbols occurring in D and Σ.

Consider a $sch(\Sigma)$-interpretation I. I is a stable model of $\Pi_{D,\Sigma}$ if I is a (classical) model of $\Pi_{D,\Sigma}$, and a (subset) minimal model (w.r.t. positive literals) of the so-called reduct of $\Pi_{D,\Sigma}$ w.r.t. I, written $\Pi^I_{D,\Sigma}$, obtained from $ground(\Pi_{D,\Sigma})$ as follows: first, remove all the rules with a negative literal not in I, and then eliminate the remaining negative literals. Intuitively, I is a stable model of $\Pi_{D,\Sigma}$ if it can be obtained by "executing" $\Pi_{D,\Sigma}$ using I as an oracle for the negative literals. The stable models of D and Σ are defined as the stable models of the normal logic program $\Pi_{D,\Sigma}$.

3.2 Minimal Models

The minimality of a (classical) model of a database D and a set Σ of NTGDs can be captured via a second-order formula $\mathsf{MM}[D, \Sigma]$. Consider the database $D = \{p(0)\}$, and the set Σ of NTGDs consisting of

$$\forall X (p(X) \wedge \neg t(X) \rightarrow r(X)) \qquad \forall X (r(X) \rightarrow t(X)).$$

The second-order formula $\mathsf{MM}[D, \Sigma]$ is defined as follows:

$$\begin{aligned} & D \wedge \Sigma \wedge \\ & \quad \neg \exists p^\star \exists t^\star \exists r^\star \Bigg(\bigwedge_{u \in \{p,t,r\}} (\forall X (u^\star(X) \rightarrow u(X))) \wedge \\ & \qquad \neg \Bigg(\bigwedge_{u \in \{p,t,r\}} (\forall X (u(X) \rightarrow u^\star(X))) \Bigg) \wedge \\ & \quad p^\star(0) \wedge \forall X (p^\star(X) \wedge \neg t^\star(X) \rightarrow r^\star(X)) \wedge \\ & \qquad\qquad\qquad \forall X (r^\star(X) \rightarrow t^\star(X)) \Bigg), \end{aligned}$$

where $p^\star, t^\star, r^\star$ are predicate variables. An interpretation I is a model of the above formula if (i) I is a model of $(D \wedge \Sigma)$, and (ii) there is no way to obtain a model of $(D \wedge \Sigma)$ from I by eliminating a positive literal. In other words, the models of $\mathsf{MM}[D, \Sigma]$ are precisely the minimal models of $(D \wedge \Sigma)$.

The syntactic transformation from $(D \wedge \Sigma)$ into $\mathsf{MM}[D, \Sigma]$ is actually *circumscription*, a logical approach, introduced by McCarthy [26], suitable for modeling what normally holds. The idea is to define, using first-order logic, both domain knowledge and so-called abnormality predicates that identify instances of a class that violate the normal properties of that class; e.g., a bird that cannot fly is abnormal, stored in a unary predicate $\mathtt{ab}_{\mathtt{bird}}$. To capture the intuition that abnormality is exceptional, inference is restricted to those models where the extension of the abnormality predicates is minimal w.r.t. set inclusion. So, to transform $(D \wedge \Sigma)$ into $\mathsf{MM}[D, \Sigma]$ we basically apply circumscription where all the predicates in D and Σ are conceived as abnormality predicates.

3.3 The New Approach

Our ultimate goal is to transform a database D and a set Σ of

[2]Constants are excluded for technical clarity; however, our results can be extended to NTGDs with constants.

NTGDs into a second-order formula $\mathsf{SM}[D,\Sigma]$ that characterizes the stable models of D and Σ. Interestingly, $\mathsf{SM}[D,\Sigma]$ is obtained by slightly modifying $\mathsf{MM}[D,\Sigma]$. But let us first expose the key reason why $\mathsf{MM}[D,\Sigma]$ fails to recognize that an interpretation is not a stable model of $(D \wedge \Sigma)$.

Consider the database D and the set Σ of NTGDs given in Section 3.2. It can be verified that, according to the LP approach, D and Σ do not have a stable model. Furthermore, since Σ does not contain existentially quantified variables, the LP approach and our new approach should coincide, and thus, the formula $\mathsf{SM}[D,\Sigma]$ should not have a model. Consider now the interpretation $J = \{p(0), t(0), \neg r(0)\}$, which is clearly a model of $\mathsf{MM}[D,\Sigma]$. During the minimality check, the content of the predicate t that appears in a negative literal may change. Indeed, $J \not\models \mathsf{MM}[D,\Sigma]$ since the interpretation $K = \{p(0), \neg t(0), \neg r(0)\}$, which converts $t(0)$ into $\neg t(0)$, is not a model of D and Σ. But, this is in a conflict with the idea of stable models, where the reduct $\Pi^J_{D,\Sigma}$ is obtained by essentially fixing in J the content of the predicates that appear in negative literals of $\Pi_{D,\Sigma}$. This can be resolved by stating that $\neg t(X)$ holds instead of $\neg t^\star(X)$, i.e., $\mathsf{SM}[D,\Sigma]$ is obtained from $\mathsf{MM}[D,\Sigma]$ by replacing the atom $t^\star(X)$ in $\forall X(p^\star(X) \wedge \neg t^\star(X) \rightarrow r^\star(X))$ with $t(X)$; clearly, J is not a model of $\mathsf{SM}[D,\Sigma]$.

The syntactic transformation from $(D \wedge \Sigma)$ into $\mathsf{SM}[D,\Sigma]$ discussed above can be extended to arbitrary NTGDs. In what follows, we formalize this transformation, which will be at the basis of our new approach to stable model semantics for NTGDs. For convenience, we employ the following notation. For predicates p and s of the same arity $m \geqslant 0$, $(p \leqslant s)$ stands for the formula $\forall \mathbf{X}(p(\mathbf{X}) \rightarrow s(\mathbf{X}))$, where $\mathbf{X}$ is a tuple of m distinct variables. If $\mathbf{p} = (p_1,\ldots,p_n)$ and $\mathbf{s} = (s_1,\ldots,s_n)$ are tuples of predicates, where p_i and s_i have the same arity, then $(\mathbf{p} \leqslant \mathbf{s})$ stands for $\bigwedge_{i=1}^n (p_i \leqslant s_i)$, while $(\mathbf{p} < \mathbf{s})$ stands for $(\mathbf{p} \leqslant \mathbf{s}) \wedge \neg(\mathbf{s} \leqslant \mathbf{p})$. In second-order logic, we apply the same notation to tuples of predicate variables. Consider a set $\Sigma \in \mathsf{TGD}^\neg$, and a database D over $sch(\Sigma)$. Let $\mathbf{p} = (p_1,\ldots,p_n)$ be the list of predicates of $sch(\Sigma)$, and $\mathbf{s} = (s_1,\ldots,s_n)$ a list of n distinct predicate variables. For a literal ℓ occurring in D or Σ, we define

$$\tau_{\mathbf{p}\triangleright\mathbf{s}}(\ell) \;=\; \begin{cases} s_i(\mathbf{t}), & \text{if } \ell = p_i(\mathbf{t}),\\[2ex] \neg p_i(\mathbf{t}), & \text{if } \ell = \neg p_i(\mathbf{t}).\end{cases}$$

We define $\tau_{\mathbf{p}\triangleright\mathbf{s}}(D)$ and $\tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma)$ as the database and the set of NTGDs obtained by applying $\tau_{\mathbf{p}\triangleright\mathbf{s}}$ to every literal in D and Σ, respectively. $\mathsf{UNA}[D]$ is the formula $\bigwedge_{c,d \in dom(D), c \neq d} \neg(c = d)$, which encodes the unique name assumption. $\mathsf{SM}[D,\Sigma]$ is defined as:

$$\mathsf{UNA}[D] \wedge D \wedge \Sigma \wedge \neg\exists \mathbf{s}((\mathbf{s} < \mathbf{p}) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(D) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma)).$$

We are now ready to introduce our new approach to stable model semantics for NTGDs.

Definition 1. (**Stable Models**) Consider a set $\Sigma \in \mathsf{TGD}^\neg$, and a database D over $sch(\Sigma)$. A $sch(\Sigma)$-interpretation I is a *stable model* of D and Σ if I is a model of $\mathsf{SM}[D,\Sigma]$. We define $SMS(D,\Sigma)$ as the set of stable models of D and Σ. ■

The conceptual advantage of our new approach over the classical LP approach is shown by the following example.

Example 4. Let $D = \{\mathtt{person}(\mathit{Alice})\}$, and Σ be the set of NTGDs given in Example 1. As discussed in Example 2, it is natural to consider the interpretation I, where

$$I^+ \;=\; \{\mathtt{person}(\mathit{Alice}), \mathtt{hasFather}(\mathit{Alice}, \mathit{Bob}), \mathtt{sameAs}(\mathit{Bob}, \mathit{Bob})\},$$

as a stable model of D and Σ, which in turn implies that

$$q = \neg\mathtt{hasFather}(\mathit{Alice}, \mathit{Bob})$$

is not entailed by (D,Σ), which is what we expect since from D and Σ there is no evidence that Bob is not the father of Alice. Recall that, according to the LP approach, I is not a stable model of $(D \wedge \Sigma)$. However, $I \models \mathsf{SM}[D,\Sigma]$, and thus, according to our new approach, I is a stable model. ■

Our new approach directly deals with existentially quantified variables without requiring Skolemization or grounding. The crucial question though that we need to answer, in order to safely conclude that the new approach is a generalization of the LP approach, is whether the two approaches coincide if we focus on dependencies that can be treated by both of them, that is, Skolemized NTGDs; note that Definition 1 can be directly applied to normal logic programs. The answer to this question is affirmative, and immediately follows from Corollary 1 in [18]. To avoid notational clutter, $SMS_{LP}(\cdot)$ and $SMS_{SO}(\cdot)$ is the set of stable models according to the LP and the new approach (which relies on second-order logic, hence the subscript "SO"), respectively.[3] Recall that $\Pi_{D,\Sigma}$ is the program obtained from D and Σ by applying Skolemization.

THEOREM 1. *Consider a database D, and $\Sigma \in \mathsf{TGD}^\neg$. It holds that, $SMS_{LP}(\Pi_{D,\Sigma}) = SMS_{SO}(\Pi_{D,\Sigma})$.*

3.4 Query Answering

The answer to an n-ary NCQ q over D and Σ under the stable model semantics is defined as the set of tuples

$$\bigcap_{M \in SMS(D,\Sigma)} \{\mathbf{t} \in \mathbf{C}^n \mid \mathbf{t} \in q(M)\}.$$

A NBCQ q has a *positive* answer over D and Σ under the stable model semantics, denoted $(D,\Sigma) \models_{SMS} q$, if $M \models q$, for each $M \in SMS(D,\Sigma)$. For clarity, we focus on NBCQs; however, our results can be extended to NCQs. The main decision problem tackled in this work is defined as follows; $\mathbb{C}$ is a class of sets of NTGDs (e.g., weakly-acyclic, sticky, etc., which are defined below):

PROBLEM:	$\mathsf{SMS\text{-}QAns}(\mathbb{C})$
INPUT:	Database D, $\Sigma \in \mathbb{C}$, and $q \in \mathsf{BCQ}^\neg$.
QUESTION:	Does $(D,\Sigma) \models_{SMS} q$?

We assume, w.l.o.g., that both the database and the query use only predicates that already occur in the set of NTGDs.

A Criterion for Decidability

It is possible to establish a sufficient (semantic) condition for the decidability of SMS-QAns via the so-called stable tree model property. To introduce this property, we first need to recall what is the treewidth of an interpretation. A *tree decomposition* of a (possibly infinite) interpretation I is a labeled tree $T = (V,E,\lambda)$, where λ is the labeling function $V \rightarrow 2^{dom(I)}$, such that: (i) for a (positive) literal $p(t_1,\ldots,t_n) \in I$, there exists $v \in V$ such that $\lambda(v) \supseteq \{t_1,\ldots,t_n\}$; and (ii) for every term $t \in dom(I)$, the set of nodes $\{v \in V \mid t \in \lambda(v)\}$ induces a connected subtree of T. The *width* of T is $\max_{v \in V}\{|\lambda(v)| - 1\}$. The *treewidth* of I is the minimum width among all tree decompositions. Intuitively, the treewidth of an interpretation I, which can be represented as a graph via its Gaifman graph, measures how similar I is to a tree; the smaller the treewidth, the closer the interpretation is to a tree. We are now ready to introduce the stable tree model property:

[3]In the rest of the paper, $SMS(\cdot)$ without a subscript refers to the set of stable models according to our new approach.

Definition 2. We say that a class $\mathbb{C}$ of NTGDs enjoys the *stable tree model property* if the following holds: for every database D, $\Sigma \in \mathbb{C}$, and $q \in \mathsf{BCQ}^{\neg}$, if $(\mathsf{SM}[D,\Sigma] \wedge \neg q)$ has a model, then it has a model of finite treewidth. ■

The next result establishes that the stable tree model property implies the decidability of query answering:

THEOREM 2. $\mathsf{SMS\text{-}QAns}(\mathbb{C})$ *is decidable if* $\mathbb{C}$ *is a class of NTGDs that enjoys the stable tree model property.*

Let us sketch the proof of the above result. Consider a database D, a set $\Sigma \in \mathbb{C}$, where $\mathbb{C}$ enjoys the stable tree model property, and $q \in \mathsf{BCQ}^{\neg}$. Since $(D,\Sigma) \models_{SMS} q$ iff $(\mathsf{SM}[D,\Sigma] \wedge \neg q)$ is unsatisfiable, it suffices to show that the problem whether $(\mathsf{SM}[D,\Sigma] \wedge \neg q)$ is satisfiable is decidable. We can construct an MSO formula $\mathsf{SM}[D,\Sigma]_1$ such that: (i) $(\mathsf{SM}[D,\Sigma] \wedge \neg q)$ and $(\mathsf{SM}[D,\Sigma]_1 \wedge \neg q)$ are equisatisfiable; and (ii) if $(\mathsf{SM}[D,\Sigma]_1 \wedge \neg q)$ is satisfiable, then it has a model of finite treewidth. Therefore, by Courcelle's classical result, which states that the satisfiability problem for fragments of MSO that enjoy the tree model property is decidable [11], we get that the problem whether $(\mathsf{SM}[D,\Sigma] \wedge \neg q)$ is satisfiable is decidable, as needed. The key idea underlying the construction of $\mathsf{SM}[D,\Sigma]_1$ can be described as follows: we attach, via a first-order formula, to every tuple of terms $\mathbf{t}$ occurring in a model of $(D \wedge \Sigma)$ a unique identifier; this allows us to access $\mathbf{t}$ via a unary predicate that stores all the valid identifiers. Then, we can rewrite $\mathsf{SM}[D,\Sigma]$ in such a way that only unary predicates are quantified.

4. QUERY ANSWERING UNDER STABLE MODELS: A CASE STUDY

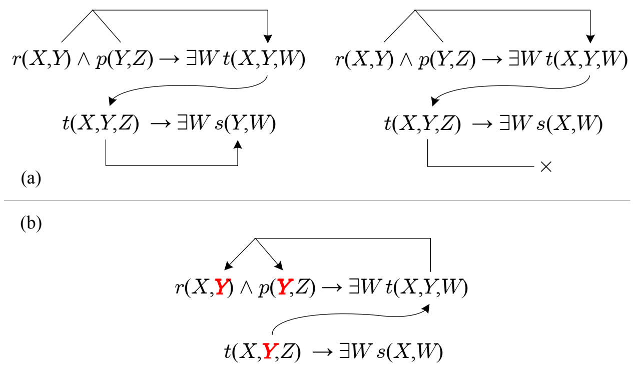
The goal of this section is to investigate how the problem of query answering under stable models behaves. We consider classes of NTGDs that are based on the main decidability paradigms proposed for TGDs, namely weak-acyclicity [17], stickiness [10] and guardedness [7], and we study query answering.

4.1 Weak-Acyclicity

Weak-acyclicity is defined by posing an acyclicity condition on the position (dependency) graph, introduced in [17], which encodes how terms are propagated during the "execution" of the program. A position in a schema $\mathcal{R}$, written $p[i]$, is a pair of an n-ary predicate $p \in \mathcal{R}$ and an integer $i \in [n]$, which represents the i-th attribute of p. The set of positions of $\mathcal{R}$, denoted $pos(\mathcal{R})$, is defined as $\{p[i] \mid p/n \in \mathcal{R} \text{ and } i \in [n]\}$.

Definition 3. The *position graph* of a set $\Sigma \in \mathsf{TGD}$ is a directed graph $PoG(\Sigma) = (V,E)$, where $V = pos(sch(\Sigma))$, and E is defined as follows: for each $\sigma \in \Sigma$ of the form $\forall \mathbf{X}(\varphi(\mathbf{X}) \rightarrow \exists \mathbf{Y}\, \psi(\mathbf{X},\mathbf{Y}))$, for each $X \in \mathbf{X}$ that occurs in ψ, and for each X in φ in position π: (1) for each occurrence of X in ψ in position π', there is a regular edge $(\pi,\pi') \in E$; (2) for each $Y \in \mathbf{Y}$, and for each occurrence of Y in ψ in position π', there is a special edge $(\pi,\pi') \in E$; and (3) no other edges occur in E. ■

Roughly speaking, a regular edge (π,π') keeps track of the fact that a term may propagate from π to π' during the "execution" of Σ. A special edge (π,π'') keeps track of the fact that the propagation of a value from π to π' also creates a new value at position π''. We are now ready to define weakly-acyclic NTGDs. A set Σ of NTGDs is *weakly-acyclic* if no cycle in $PoG(\Sigma^+)$, where Σ^+ is obtained from Σ by eliminating the negative literals, containing a special edge exists. The corresponding class is denoted $\mathsf{WATGD}^{\neg}$.



$r(X,Y) \wedge p(Y,Z) \rightarrow \exists W\, t(X,Y,W)$
$r(X,Y) \wedge p(Y,Z) \rightarrow \exists W\, t(X,Y,W)$
$t(X,Y,Z) \rightarrow \exists W\, s(Y,W)$
$t(X,Y,Z) \rightarrow \exists W\, s(X,W)$
(a)
×
(b)
$r(X,Y) \wedge p(Y,Z) \rightarrow \exists W\, t(X,Y,W)$
$t(X,Y,Z) \rightarrow \exists W\, s(X,W)$

Figure 1: Stickiness and marking; for brevity, the universal quantifiers are omitted.

As one may expect, if a database and a weakly-acyclic set of NTGDs admit a stable model, then they admit a stable model of finite size, and thus, $\mathsf{WATGD}^{\neg}$ enjoys the stable tree model property. Hence, by Theorem 2, we get that:

THEOREM 3. $\mathsf{SMS\text{-}QAns}(\mathsf{WATGD}^{\neg})$ *is decidable.*

The fact that $\mathsf{WATGD}^{\neg}$ enjoys the stable tree model property is shown in two steps: (i) we establish that a stable model M of a database D and a set Σ of NTGDs (not necessarily weakly-acyclic) is obtained by "executing" Σ, starting from D, using M as an oracle for its negative literals, and (ii) we show that whenever the input set Σ of NTGDs is weakly-acyclic, then its "execution" terminates. In fact, the same approach is followed in Section 5 to pinpoint the complexity of query answering under stable models for weakly-acyclic sets of NTGDs, which is one of the main contributions of this work; thus, all the details are deferred to Section 5.

4.2 Stickiness

Let us first recall the key property underlying the class of sticky sets of (positive) TGDs [10]. During the execution of the TGDs, or, in other words, during the chase procedure, terms that are associated (via a homomorphism) with variables that appear more than once in the body of a TGD (i.e., join variables) are always propagated (or "stick") to the inferred atoms. This is illustrated in Figure 1(a); the first set of TGDs is sticky, while the second is not. The formal definition is based on an inductive marking procedure that marks the variables that may violate the semantic property of the chase described above [10]. Roughly, during the base step of this procedure, a variable that appears in the body of a TGD σ but not in every head-atom of σ is marked. Then, the marking is inductively propagated from head to body as shown in Figure 1(b). Finally, a set of TGDs Σ is *sticky* if no TGD in Σ contains two occurrences of a marked variable. The notion of stickiness can be straightforwardly extended to NTGDs: a set Σ of NTGDs is sticky if the set of TGDs obtain by converting every literal $\neg p(\mathbf{t})$ occurring in Σ into the atom $p(\mathbf{t})$ is sticky [1]; the corresponding class is denoted $\mathsf{STGD}^{\neg}$. Although stickiness guarantees not only the decidability, but also the data tractability of CQ answering under TGDs [10], the situation changes for NTGDs under the stable model semantics:

THEOREM 4. $\mathsf{SMS\text{-}QAns}(\mathsf{STGD}^{\neg})$ *is undecidable, even if the query is* $\neg$*-free.*

The above negative result is not surprising since the problem is undecidable even if we follow the LP approach [1]. The reason for this undesirable behavior should be found in the fact that sticky sets of NTGDs are powerful enough for expressing cartesian products, that is, rules of the form $\forall \mathbf{X} \forall \mathbf{Y}(p(\mathbf{X}) \wedge s(\mathbf{Y}) \rightarrow t(\mathbf{X}, \mathbf{Y}))$, with $\mathbf{X} \cap \mathbf{Y} = \varnothing$. Using cartesian products we can build infinite grids, while the stable negation gives us the power of guessing. This combination allows us to simulate the behavior of a Turing machine.

4.3 Guardedness

An NTGD σ is guarded if there exists an atom, called guard, in $B^+(\sigma)$ that contains all the variables in $B(\sigma)$. A set Σ of NTGDs is called *guarded* if every NTGD of Σ is guarded, and the corresponding class is denoted $\mathsf{GTGD}^\neg$. Guardedness is a well-accepted paradigm that gives rise to robust rule-based languages. As shown in [7], query answering under guarded TGDs is decidable, while this decidability result has been recently extended to guarded NTGDs interpreted according to the LP approach to stable model semantics [22]. Contrary to what one might expect, this is not the case for guarded NTGDs interpreted according to our new approach to stable model semantics. As for sticky NTGDs, we can build grids of unbounded size, while the stable negation gives us the power of guessing. This allows us to simulate the behavior of a Turing machine, and show that:

THEOREM 5. $\mathsf{SMS\text{-}QAns}(\mathsf{GTGD}^\neg)$ *is undecidable, even if the query is* $\neg$*-free.*

The above result, together with Theorem 2, immediately implies that $\mathsf{GTGD}^\neg$ does not enjoy the stable tree model property. This is a surprising negative outcome since guardedness usually gives rise to robust and decidable computational logics, such as the guarded fragment of first-order logic, while the model-theoretic reason for this desirable behavior is the tree model property; see, e.g., [23]. The reason for this negative outcome can be informally described as follows. In the classical setting, the guard atoms occurring in rule-bodies force the existence of a model that follows an inherent tree structure, where its branches cannot interact. This holds also for the LP approach to stable model semantics, which refers to Skolemization and grounding, since the obtained reduct can be seen as a set of (positive first-order) guarded TGDs [22]. However, by following our new approach to stable negation for NTGDs, it is possible to "guess" an appropriate guard, whose domain can be restricted to an arbitrary set, and thus the key property of guardedness, i.e., only atoms derived from a common guard ancestor can interact, is destroyed. In fact, by restricting the domain of an (existentially guessed) guard in an appropriate way, joins between any two atoms can be forced, destroying the inherent tree structure of the underlying models.

4.4 Our Conclusions

From the above analysis, we conclude that:

- The class that is based on weak-acyclicity can be reconciled with the new approach to stable model semantics, without sacrificing the decidability of query answering. This is because, whenever a database and a set of weakly-acyclic NTGDs have a model, then they admit a finite model, which implies that $\mathsf{WATGD}^\neg$ enjoys the stable tree model property.
- It is not possible to reconcile stickiness and guardedness with our new approach to stable negation for NTGDs, since query answering becomes undecidable. Although for sticky sets of NTGDs this was expected, for guarded NTGDs this is a surprising outcome.

The former gives rise to the problem of closing the complexity of $\mathsf{SMS\text{-}QAns}(\mathsf{WATGD}^\neg)$, which will be the subject of Section 5. The latter demonstrates the need to provide a more refined definition of stickiness and guardedness. This task goes beyond the scope of the present work.

5. WEAK-ACYCLICITY: ALGORITHMS AND COMPLEXITY

We proceed to pinpoint the complexity of NBCQ answering under stable models focussing on weakly-acyclic sets of NTGDs. We consider the standard complexity measures, i.e., *data complexity*, calculated by considering only the database as input, and *combined complexity*, which considers everything as input. We show that:

THEOREM 6. $\mathsf{SMS\text{-}QAns}(\mathsf{WATGD}^\neg)$ *is*

- Π_2^P*-complete in data complexity, and*
- coN2EXPTIME$^{\text{NP}}$*-complete in combined complexity.*

The lower bounds hold for predicates of bounded arity, and CQs consisting of a single 0*-ary predicate.*

The above result shows that the complexity of query answering significantly increases in the presence of default negation; recall that for weakly-acyclic sets of TGDs the problem is PTIME-complete in data complexity [17], and 2EXPTIME-complete in combined complexity [10]. The rest of this section is devoted to establishing the above result. The desired upper bounds are obtained via a guess and check algorithm for the complement of the problem, which guesses a stable model, and checks that it does not entail the query. However, in order to apply such a simple procedure, we need to solve two non-trivial subtasks: (i) establish an upper bound on the size of stable models under $\mathsf{WATGD}^\neg$, and (ii) check the stability of a model in an optimal way.

5.1 Bounding the Size of Stable Models

In this section, we establish an upper bound on the size of stable models under $\mathsf{WATGD}^\neg$, which will then allow us to obtain optimal data and combined complexity upper bounds for query answering. Let us first introduce our technical tool.

Immediate Consequence Operator

Consider a set Σ of NTGDs, a set S of $sch(\Sigma)$-atoms, and a $sch(\Sigma)$-interpretation I. An atom $p(\mathbf{t}) \in I$ is an *immediate consequence* for S and Σ relative to I if there exists $\sigma \in \Sigma$, and a homomorphism h such that $h(B(\sigma)) \subseteq S \cup I^-$ and $p(\mathbf{t}) \in h(H(\sigma))$. The *immediate consequence operator* of Σ relative to I follows:

$$T_{\Sigma,I}(S) \;=\; \{p(\mathbf{t}) \in I^+ \mid p(\mathbf{t}) \text{ is an immediate consequence for } S \text{ and } \Sigma \text{ relative to } I\}.$$

We write $T^i_{\Sigma,I}(S)$ for the result obtained after i applications of the operator $T_{\Sigma,I}$ starting from S. Formally,

$$T^0_{\Sigma,I}(S) \;=\; S \qquad T^{i+1}_{\Sigma,I}(S) \;=\; T_{\Sigma,I}(T^i_{\Sigma,I}(S)) \cup T^i_{\Sigma,I}(S)$$

and

$$T^\infty_{\Sigma,I}(S) \;=\; \bigcup_{i=0}^{\infty} T^i_{\Sigma,I}(S).$$

Note that $T_{\Sigma,I}$ is monotone; thus, $T^\infty_{\Sigma,I}(S)$ is defined as its least fixpoint. The next result formalizes the idea that a stable model M of a database D and $\Sigma \in \mathsf{TGD}^\neg$ can be obtained by "executing" Σ, starting from D, using M as an oracle for its negative literals.

LEMMA 7. *Consider a database D, and $\Sigma \in \mathsf{TGD}^\neg$. For every $M \in SMS(D, \Sigma)$, $M^+ = T^\infty_{\Sigma,M}(D)$.*

It is important to clarify that the $T_{\Sigma,I}$ operator cannot be used to characterize the stable models of D and Σ. In particular, given a $sch(\Sigma)$-interpretation I, $I^+ = T^{\infty}_{\Sigma,I}(D)$ does not imply $I \in SMS(D,\Sigma)$. Consider, for example,

$$D = \{s(a)\} \quad \text{and} \quad \Sigma = \{\forall X(s(X) \rightarrow \exists Y\, p(X,Y))\}.$$

For the interpretation I with $I^+ = \{s(a), p(a,b), p(a,c)\}$ it is clear that $I^+ = T^{\infty}_{\Sigma,I}(D)$, but $I \notin SMS(D,\Sigma)$. However, as we shall see, the immediate consequence operator is a useful technical tool for bounding the size of stable models.

Establishing a Bound

Consider a model $M \in SMS(D,\Sigma)$, where $\Sigma \in \mathsf{WATGD}^{\neg}$. The least fixpoint of $T_{\Sigma,M}$, starting from D, can be reached after a finite number of applications.

LEMMA 8. *Consider a database D, a set $\Sigma \in \mathsf{WATGD}^{\neg}$, and let $M \in SMS(D,\Sigma)$. It holds that,*

$$|T^{\infty}_{\Sigma,M}(D)| \ \leqslant\ f(D,\Sigma),$$

where f is polynomial in D, and doubly exponential in Σ.

The above lemma relies on the fact that, given a database D and a weakly-acyclic set Σ of (positive) TGDs, the result of every chase sequence of D and Σ has size at most polynomial in D, and doubly exponential in Σ; implicit in [17].[4] In particular, given a database D and a set $\Sigma \in \mathsf{WATGD}^{\neg}$, we show that $T^{\infty}_{\Sigma,M}(D)$ induces a chase sequence C of D and Σ^+ such that $|T^{\infty}_{\Sigma,M}(D)|$ is bounded by the size of the result of C, which in turn implies that $|T^{\infty}_{\Sigma,M}(D)|$ is at most polynomial in D, and doubly exponential in Σ, as needed. The next key result follows from Lemma 7 and 8:

PROPOSITION 9. *Consider a database D, and a set $\Sigma \in \mathsf{WATGD}^{\neg}$. For every $M \in SMS(D,\Sigma)$,*

$$|M^+| \ \leqslant\ f(D,\Sigma),$$

where f is polynomial in D, and doubly exponential in Σ.

5.2 Checking Stability

We now investigate the stability problem, i.e., checking whether a model M of a database D and $\Sigma \in \mathsf{WATGD}^{\neg}$ satisfies the formula $\neg\exists \mathbf{s}((\mathbf{s} < \mathbf{p}) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(D) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma))$. Unfortunately, if we perform this check blindly, without having available any additional information about the model M in question, then we cannot do better than Π^P_2, which in turn does not lead to the desired upper bounds stated in Theorem 6. This is because such a check involves model checking for NTGDs, that is, to decide whether an interpretation is a model of a set of NTGDs, and it is known that already for TGDs model checking is Π^P_2-hard, even if we focus on a single non-recursive TGD [27]. The crucial question that we should answer is the following: having available some additional information about the model M, which can be computed on-the-fly during the execution of our guess and check algorithm, is the above checking feasible in coNP? The answer to the above question is affirmative, which, as we shall see in the next section, will allows us to obtain the worst-case optimal upper bounds stated in Theorem 6. In particular, the additional information about the model M that we need is the set of homomorphisms that acts as a witness for the fact that M is a model of Σ. The notion of the witness is defined as follows:

[4]Here, we refer to the standard (a.k.a. the restricted) version of the chase, where a TGD is being applied only if its necessary.

Definition 4. Consider an NTGD σ, and an interpretation I. The *witness* for I w.r.t. σ, denoted $\mathcal{W}^{\sigma}_{I}$, is defined as

$$\left\{\left(h, E^{h,\sigma}_{I}\right) \mid h \text{ is a homomorphism s.t. } h(B(\sigma)) \subseteq I\right\},$$

where $E^{h,\sigma}_{I} = \{\mu \mid \mu \supseteq h \text{ and } \mu(H(\sigma)) \subseteq I\}$. The witness $\mathcal{W}^{\sigma}_{I}$ is called *negative* if there exists $(h,E) \in \mathcal{W}^{\sigma}_{I}$ such that $E = \varnothing$; otherwise, $\mathcal{W}^{\sigma}_{I}$ is called *positive*. ■

It is clear that having the positive witness $\mathcal{W}^{\sigma}_{I}$ in place, we know precisely via which homomorphisms $B(\sigma)$ is satisfied by I (if there exists one), and how they are extended in order to satisfy $H(\sigma)$, and therefore σ itself. The next auxiliary lemma follows by definition:

LEMMA 10. *Consider a set $\Sigma \in \mathsf{TGD}^{\neg}$, and a $sch(\Sigma)$-interpretation I. The following are equivalent:*

1. *$I \models \Sigma$.*
2. *$\mathcal{W}^{\sigma}_{I}$ is positive, for every $\sigma \in \Sigma$.*

We now introduce a refined version of the stability problem, based on the notion of witness; let $\mathbb{C}$ be a class of NTGDs, and $\Phi_{D,\Sigma} = \neg\exists\mathbf{s}((\mathbf{s} < \mathbf{p}) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(D) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma))$:

PROBLEM:	W-Stability($\mathbb{C}$)
INPUT:	Datab. D, $\Sigma \in \mathbb{C}$, model M of $(D \wedge \Sigma)$, and positive witnesses $\{\mathcal{W}^{\sigma}_{M}\}_{\sigma\in\Sigma}$.
QUESTION:	Does $M \models \Phi_{D,\Sigma}$?

We establish the following technical result, which is crucial for our later complexity analysis:

PROPOSITION 11. W-Stability($\mathsf{WATGD}^{\neg}$) *is in* coNP.

We can decide the complement of the problem in question via the following non-deterministic algorithm:

1. Guess $J \subset M^+$ such that $J \supseteq D$, and let I be the total interpretation with $dom(I) = dom(J)$ and $I^+ = J$.
2. If for each $\sigma \in \tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma)$ the witness $\mathcal{W}^{\sigma}_{(M \cup \tau_{\mathbf{p}\triangleright\mathbf{s}}(J))}$ is positive, then $accept$; otherwise, $reject$.

It is easy to see that, by Lemma 10, the above algorithm accepts iff $M \models \exists\mathbf{s}((\mathbf{s} < \mathbf{p}) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(D) \wedge \tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma))$. It is clear that both steps are feasible in polynomial time. In particular, the witness for $(M \cup \tau_{\mathbf{p}\triangleright\mathbf{s}}(I))$ w.r.t. some $\sigma \in \tau_{\mathbf{p}\triangleright\mathbf{s}}(\Sigma)$ can be obtained in polynomial time from $\mathcal{W}^{\hat{\sigma}}_{M}$, where $\hat{\sigma} \in \Sigma$ and $\sigma = \tau_{\mathbf{p}\triangleright\mathbf{s}}(\hat{\sigma})$. Thus, W-Stability($\mathsf{WATGD}^{\neg}$) is in coNP, as needed.

5.3 Finalizing the Complexity Analysis

We are now ready to close the complexity of SMS-QAns for $\mathsf{WATGD}^{\neg}$, and obtain Theorem 6.

Upper Bounds

The upper bounds stated in Theorem 6 are obtained in a uniform way via a guess and check algorithm, which exploits Propositions 9 and 11. Fix a database D, $\Sigma \in \mathsf{WATGD}^{\neg}$, and $q \in \mathsf{BCQ}^{\neg}$. Let $\mathbf{p} = (p_1,\ldots,p_n)$ be the list of predicates of $sch(\Sigma)$, and $\mathbf{s} = (s_1,\ldots,s_n)$ a list of n predicate variables. We write $\delta_{D,\Sigma}$ for the bound provided by Proposition 9. The following procedure decides if $(D,\Sigma) \not\models_{SMS} q$, i.e., the complement of our problem:

1. Guess a set of atoms J, where $|J| \leqslant \delta_{D,\Sigma}$, and let I be the total $sch(\Sigma)$-interpretation with $dom(I) = dom(J)$ and $I^+ = J$ such that $I \models (D \wedge \Sigma \wedge \neg q)$; if such a set of atoms does not exist, then $reject$.

2. If $I \models \neg\exists \mathbf{s}((\mathbf{s} < \mathbf{p}) \wedge \tau_{\mathbf{p} \triangleright \mathbf{s}}(D) \wedge \tau_{\mathbf{p} \triangleright \mathbf{s}}(\Sigma))$, then *accept*; otherwise, *reject*.

The above procedure runs in polynomial time in $\delta_{D,\Sigma}$ with an $\mathcal{S}$-oracle, where $\mathcal{S}$ is a complexity class powerful enough for solving W-Stability; $\{\mathcal{W}_I^\sigma\}_{\sigma \in \Sigma}$ is implicitly constructed during the first step of the algorithm. Consequently, the complement of SMS-QAns is in $\mathrm{NP}^{\mathcal{S}}$ in data complexity, and in $\mathrm{N2ExpTime}^{\mathcal{S}}$ in combined complexity. The desired upper bounds follow by Proposition 11 since $\mathcal{O}^{\mathrm{coNP}} = \mathcal{O}^{\mathrm{NP}}$, where $\mathcal{O} \in \{\mathrm{NP}, \mathrm{N2ExpTime}\}$.

Lower Bounds

We give the reduction for the Π_2^P-hardness in data complexity, while for the combined complexity we simply mention the problem from which we provide a reduction.

Π_2^P-hard Data Complexity

We reduce satisfiability of 2-QBF$_\exists$ formulas to the complement of SMS-QAns(WATGD$^\neg$). Let $\varphi = \exists \mathbf{X} \forall \mathbf{Y}\, \psi(\mathbf{X}, \mathbf{Y})$ be a 2-QBF$_\exists$ formula, and $\psi = \vee_{i=1}^k (\ell_i^1 \wedge \ell_i^2 \wedge \ell_i^3)$ be a 3DNF formula. We construct a database D_φ, and $\Sigma \in$ WATGD$^\neg$, which does not depend on φ, such that φ is satisfiable iff $(D_\varphi, \Sigma) \not\models_{SMS}$ `error`, where `error` is a 0-ary predicate.

Database. For a variable V, let $\pi(V) = \nu(\neg V) = V$ and $\pi(\neg V) = \nu(V) = \star$, where $\star$ is a special constant. The database D_φ is defined as follows:

$$\{\mathtt{exists}(X) \mid X \in \mathbf{X}\} \cup \{\mathtt{forall}(Y) \mid Y \in \mathbf{Y}\} \cup$$
$$\{\mathtt{cl}(\pi(\ell_i^1), \pi(\ell_i^2), \pi(\ell_i^3), \nu(\ell_i^1), \nu(\ell_i^2), \nu(\ell_i^3))\}_{i \in [k]} \cup \{\mathtt{nil}(\star)\}.$$

This completes the definition of D_φ.

NTGDs. The set $\Sigma \in$ WATGD$^\neg$ is defined as follows. First, we guess an assignment for the variables of φ:

$$\begin{array}{l}
\rightarrow \exists X\, \mathtt{zero}(X) \quad \rightarrow \exists X\, \mathtt{one}(X)\\
\forall X (\mathtt{zero}(X) \wedge \mathtt{one}(X) \rightarrow \mathtt{error})\\
\forall X (\mathtt{zero}(X) \rightarrow \mathtt{truthVal}(X))\\
\forall X (\mathtt{one}(X) \rightarrow \mathtt{truthVal}(X))\\
\forall X (\mathtt{exists}(X) \rightarrow \exists Y\, \mathtt{assign}(X,Y))\\
\forall X (\mathtt{forall}(X) \rightarrow \exists Y\, \mathtt{assign}(X,Y))\\
\forall X \forall Y (\mathtt{assign}(X,Y) \wedge \neg \mathtt{truthVal}(Y) \rightarrow \mathtt{error}).
\end{array}$$

By exploiting the well-known technique of saturation [13], we proceed to perform the universal check as follows:

$$\begin{array}{l}
\neg \mathtt{saturate} \rightarrow \mathtt{saturate}\\
\forall X \forall Y (\mathtt{forall}(X) \wedge \mathtt{truthVal}(Y) \wedge\\
\qquad\qquad\qquad \mathtt{saturate} \rightarrow \mathtt{assign}(X,Y))\\
\forall X \forall Y (\mathtt{nil}(X) \wedge \mathtt{truthVal}(Y) \rightarrow \mathtt{assign}(X,Y))\\
\forall \mathbf{P} \forall \mathbf{N} \forall O \forall Z (\mathtt{cl}(P_1, P_2, P_3, N_1, N_2, N_3) \wedge\\
\qquad \bigwedge_{i=1}^{3} \mathtt{assign}(P_i, O) \wedge \mathtt{one}(O) \wedge\\
\qquad\qquad \bigwedge_{i=1}^{3} \mathtt{assign}(N_i, Z) \wedge \mathtt{zero}(Z) \rightarrow \mathtt{saturate}).
\end{array}$$

This completes the definition of Σ.

We can show that φ is satisfiable iff $(D_\varphi, \Sigma) \not\models_{SMS}$ `error`, and the claim follows.

coN2ExpTime$^{\mathrm{NP}}$-hard Combined Complexity

By reduction from the complement of finite tiling extension: for a grid of double-exponential size, decide whether a top-row tiling exists that cannot be extended into a grid tiling.[5] We define a database D, and $\Sigma \in$ WATGD$^\neg$ such that the instance of finite tiling extension is negative iff $(D, \Sigma) \models_{SMS} q$, where q is 0-ary predicate.

6. ADDING DISJUNCTION

An interesting question that comes up is how SMS-QAns is affected if we extend WATGD$^\neg$ with disjunction in rule-heads. Disjunction is a crucial feature for KR and database query languages; thus, WATGD$^{\neg,\vee}$ deserves our attention. We consider *normal disjunctive tuple-generating dependencies (NDTGDs)* of the form

$$\forall \mathbf{X} \forall \mathbf{Y} \left(\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \bigvee_{i=1}^{n} \exists \mathbf{Z}_i\, \psi_i(\mathbf{X}, \mathbf{Z}_i) \right),$$

where φ is a conjunction of literals, and ψ_i is a conjunction of atoms; we write TGD$^{\neg,\vee}$ for the corresponding class. Extending weak-acyclicity with disjunction is done in the obvious way. A set $\Sigma \in$ TGD$^{\neg,\vee}$ is weakly-acyclic if no cycle in $PoG(\Sigma^{+,\wedge})$, where $\Sigma^{+,\wedge}$ is obtained from Σ by removing the negative literals and converting the disjunction into a conjunction, contains a special edge. The corresponding class is denoted WATGD$^{\neg,\vee}$. Similarly to NTGDs, we exploit the notion of homomorphism to define when an interpretation is a model of a set of NDTGDs. For a database D and a set $\Sigma \in$ TGD$^{\neg,\vee}$, $SMS(D, \Sigma)$ is defined, as for NTGDs, via the second-order formula SM$[D, \Sigma]$, which is obtained by simply applying the operator $\tau_{\mathbf{p} \triangleright \mathbf{s}}$ to every literal occurring in D and Σ.

6.1 Complexity of Query Answering

Weak-acyclicity can be enriched with disjunction in the rule-heads without paying a price in terms of complexity. As we shall see, the reason for this is the fact that we can simulate disjunction using existential quantification and stable negation. We show that:

THEOREM 12. SMS-QAns(WATGD$^{\neg,\vee}$) *is*

- Π_2^P*-complete in data complexity, and*
- coN2ExpTime$^{\mathrm{NP}}$*-complete in combined complexity.*

The lower bounds hold for predicates of bounded arity, and CQs consisting of a single 0*-ary predicate.*

The lower bounds are immediately inherited from Theorem 6; thus, in the rest of this section, we concentrate on the upper bounds. We show that the problem in question can be reduced in polynomial time into query answering under (non-disjunctive) NTGDs. Fix a database D, a set $\Sigma \in$ WATGD$^{\neg,\vee}$, and a query $q \in$ BCQ$^\neg$:

LEMMA 13. *A database D' and a set $\Sigma' \in$ TGD$^\neg$, which does not depend on D, can be constructed in polynomial time such that $(D, \Sigma) \models_{SMS} q$ iff $(D', \Sigma') \models_{SMS} q$.*

Before we proceed with the construction of D' and Σ', let us clarify that Σ' is, in general, *not* weakly-acyclic. Once we define Σ' below, we will explain, via an example, the reason why Σ' violates weak-acyclcity, and we will discuss why Lemma 13 is still useful for our purposes.

[5]This problem is known to be Σ_2^P-hard for a grid of polynomial size; see, e.g., [12]. Interestingly, the same construction shows that for a grid of double-exponential size it is N2ExpTime$^{\mathrm{NP}}$-hard.

Database. The database D' is obtained from D by adding sufficiently many indices for the disjuncts in Σ. In particular, assuming that $k > 0$ is the maximum number of disjuncts occurring in the head of a NDTGD of Σ,

$$D' = D \cup \{\mathtt{nil}(\star), \mathtt{idx}_1(c_1), \ldots, \mathtt{idx}_k(c_k)\}.$$

This completes the construction of D'

NTGDs. Before defining Σ', let us explain how a NDTGD $\sigma \in \Sigma$, which has the general form given at the beginning of the section, can be simulated via a set Σ_σ of (non-disjunctive) NTGDs. Clearly, if σ is already non-disjunctive, then $\Sigma_\sigma = \{\sigma\}$. Assume now that σ is disjunctive. First, we need to guess a disjunct of σ, which can be done via the set Σ_σ^{guess} consisting of the NTGDs:

$$\forall \mathbf{X} \forall \mathbf{Y} (\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists I \exists \mathbf{Z}\, t_\sigma(I,\mathbf{X},\mathbf{Z}))$$
$$\forall I \forall \mathbf{X} \forall \mathbf{Z} (t_\sigma(I,\mathbf{X},\mathbf{Z}) \wedge \neg \mathtt{idx}_1(I) \wedge \ldots \wedge \neg \mathtt{idx}_n(I) \rightarrow \mathtt{false}),$$

where $\mathbf{Z} = \mathbf{Z}_1, \ldots, \mathbf{Z}_n$, and $\mathtt{false}$ is a 0-ary predicate that is forced to be false in every stable model via the rule

$$\mathtt{false} \wedge \neg \mathtt{aux} \rightarrow \mathtt{aux},$$

which also belongs to Σ_σ^{guess}. Depending on the previous guess, we should infer the right disjunct from the head of σ, which is done via the set Σ_σ^{infer} consisting of the NTGDs:

$$\forall I \forall \mathbf{X} \forall \mathbf{Z} (t_\sigma(I,\mathbf{X},\mathbf{Z}) \wedge \mathtt{idx}_i(I) \rightarrow \psi_i(\mathbf{X},\mathbf{Z}_i)), \text{ for each } i \in [n].$$

Finally, we should ensure stability, that is, the first NTGD will not support any new disjunct from the head of σ if one is already true. This is done via the set Σ_σ^{stab} consisting of the NTGDs:

$$\forall \mathbf{X} \forall \mathbf{Y} \forall \mathbf{Z}_i \forall I \forall N (\varphi(\mathbf{X},\mathbf{Y}) \wedge \psi_i(\mathbf{X},\mathbf{Z}_i) \ \wedge\ \mathtt{idx}_i(I) \wedge \mathtt{nil}(N) \rightarrow t_\sigma(I,\mathbf{X},\mathbf{N}_1^{i-1},\mathbf{Z}_i,\mathbf{N}_{i+1}^{n})), \text{ for each } i \in [n],$$

where $\mathbf{N}_x^y = N, \ldots, N$ consisting of $(|\mathbf{Z}_x| + \ldots + |\mathbf{Z}_y|)$ variables. The set Σ_σ is defined as $(\Sigma_\sigma^{guess} \cup \Sigma_\sigma^{infer} \cup \Sigma_\sigma^{stab})$, while Σ' is defined as $\bigcup_{\sigma \in \Sigma} \Sigma_\sigma$. It can be shown that $(D,\Sigma) \models_{SMS} q$ iff $(D',\Sigma') \models_{SMS} q$, and Lemma 13 follows.

Although we have Lemma 13 in place, we cannot inherit yet the desired upper bounds from our results on $\mathsf{SMS\text{-}QAns}(\mathsf{WATGD}^\neg)$, since in general Σ' violates weak-acyclicity.

Example 5. Consider the set $\Sigma \in \mathsf{WATGD}^{\neg,\vee}$

$$\forall X (p(X) \rightarrow \exists Y\, s(X,Y))$$
$$\forall X (r(X) \rightarrow p(X) \vee s(X,X)).$$

According to our construction, Σ' contains (among others)

$$\forall X (p(X) \rightarrow \exists Y\, s(X,Y))$$
$$\forall I \forall X (t_\sigma(I,X) \wedge \mathtt{idx}_1(I) \rightarrow p(X))$$
$$\forall X \forall Y (r(X) \wedge s(X,X) \wedge \mathtt{idx}_2(I) \wedge \mathtt{nil}(N) \rightarrow t_\sigma(I,X)).$$

Due to the first rule, in the position graph we have the special edge $(p[1], s[2])$. In addition, due to the other rules, we have the regular edges $(t_\sigma[2], p[1])$ and $(s[2], t_\sigma[2])$. Therefore, we have a cycle in the position graph that contains a special edge, which implies that Σ' is not weakly-acyclic. ■

Even if Σ' is not weakly-acyclic, we can show that the new cycles in the position graph that destroy weak-acyclicity are harmless. As it can be verified in Example 5, these cycles do not encode the situation where the generation of a term in a certain position causes the generation of some other term in the same position, which eventually leads to an infinite generation of terms. By using this fact, we can show that Propositions 9 and 11, which were decisive for the analysis performed in the previous section, hold for Σ'. This, combined with Lemma 13, gives us the desired upper bounds.

7. EXPRESSIVE QUERY LANGUAGES

It turns out that the class of weakly-acyclic NTGDs, focussing on our new approach to stable negation, gives rise to expressive database query languages. Interestingly, the language based on $\mathsf{WATGD}^\neg$ with cautious (resp., brave) stable model semantics captures precisely Π_2^P (resp., Σ_2^P). Cautious (resp., brave) reasoning refers to the approach where the answer to the query is computed by considering the intersection (resp., union) of the underlying stable models. In the sequel, we introduce the new query languages, analyze their complexity, and demonstrate their practical relevance by exhibiting prominent queries that can be naturally expressed in $\mathsf{WATGD}^\neg$. We finally investigate their expressive power.

7.1 New Query Languages

A $\mathsf{WATGD}^\neg$ query is a pair (Σ, q), where $\Sigma \in \mathsf{WATGD}^\neg$ is the query program, and q/n is a predicate not occurring in the body of an NTGD. The *extensional (database) schema* of Σ, denoted $edb(\Sigma)$, consists of all the extensional predicates of $sch(\Sigma)$, whose values are given via an input database, while the *intensional schema* of Σ, denoted $idb(\Sigma)$, consists of all the intensional predicates of $sch(\Sigma)$, whose values are computed by the program. Given a database D over $edb(\Sigma)$, the answer to $Q = (\Sigma, q)$ over D under the *cautious* stable model semantics is defined as

$$Q(D) = \{\mathbf{t} \in \mathbf{C}^n \mid (D,\Sigma) \models_{SMS} q(\mathbf{t})\}.$$

The answer to Q over D under the *brave* stable model semantics is

$$Q(D) = \{\mathbf{t} \in \mathbf{C}^n \mid \exists M \in \mathit{SMS}(D,\Sigma), M \models q(\mathbf{t})\}.$$

Let $\mathsf{WATGD}^\neg_\mathsf{c}$ and $\mathsf{WATGD}^\neg_\mathsf{b}$ be the query language $\mathsf{WATGD}^\neg$ with cautious and brave stable model semantics, respectively. The query evaluation problem for a query language $\mathcal{Q}$ is as follows: given a database D, a query $Q \in \mathcal{Q}$, and a tuple $\mathbf{t}$, decide whether $\mathbf{t} \in Q(D)$; if it is $\mathcal{C}$-complete, then we say that $\mathcal{Q}$ is $\mathcal{C}$-complete.

Computational Complexity

By exploiting the complexity analysis performed in Section 5, it is not difficult to show that:

THEOREM 14. *$\mathcal{Q}$ is $\mathcal{D}$-complete in data complexity, and $\mathcal{C}$-complete in combined complexity, where*

$$\mathcal{D} = \begin{cases} \Pi_2^P, & \text{if } \mathcal{Q} = \mathsf{WATGD}^\neg_\mathsf{c}, \\ \Sigma_2^P, & \text{if } \mathcal{Q} = \mathsf{WATGD}^\neg_\mathsf{b}, \end{cases}$$

and

$$\mathcal{C} = \begin{cases} \textsc{coN2ExpTime}^{\text{NP}}, & \text{if } \mathcal{Q} = \mathsf{WATGD}^\neg_\mathsf{c}, \\ \textsc{N2ExpTime}^{\text{NP}}, & \text{if } \mathcal{Q} = \mathsf{WATGD}^\neg_\mathsf{b}. \end{cases}$$

The lower bounds hold even for predicates of bounded arity.

Applications

The obtained query languages can be employed to solve in a declarative way problems that lie at the second level of the polynomial hierarchy. For example, by using our languages, we can devise novel encodings for the following problems: (i) consistent query answering under weakly-acyclic TGDs relative to subset repairs [30]; (ii) satisfiability for 2-QBF; and (iii) an interesting variation of graph k-colorability, which generalizes the well-known problem CERT3COL [29]. Due to space reasons, we focus only on (ii).

Satisfiability of 2-QBF. As shown in Section 5.3, 2-QBF$_\exists$ can be reduced to $\mathsf{SMS\text{-}QAns}(\mathsf{WATGD}^\neg)$. The given formula φ is encoded in a database D_φ, and a fixed set $\Sigma \in \mathsf{WATGD}^\neg$ is constructed such that φ is satisfiable iff $(D_\varphi, \Sigma) \not\models_{SMS} \mathtt{error}$, where $\mathtt{error}$ is a propositional predicate. By using this construction, we can show that 2-QBF$_\exists$ can be decided using $\mathsf{WATGD}^\neg_\mathsf{b}$; notice that it cannot be decided using $\mathsf{WATGD}^\neg_\mathsf{c}$, unless the polynomial hierarchy collapses. We define the query

$$Q = (\Sigma \cup \{\neg\mathtt{error} \rightarrow \mathtt{ans}\}, \mathtt{ans}) \in \mathsf{WATGD}^\neg_\mathsf{b}.$$

For every 2-QBF$_\exists$ formula φ, φ is satisfiable iff the empty tuple belongs to $Q(D_\varphi)$, i.e., there exists $M \in SMS(D_\varphi, \Sigma)$ such that $M \models \mathtt{ans}$. Analogously, 2-QBF$_\forall$ can be decided using $\mathsf{WATGD}^\neg_\mathsf{c}$.

7.2 Expressive Power

We proceed to investigate the expressive power of our new query languages. For a query language $\mathcal{Q}$, we say that its *absolute expressive power* is $\mathcal{C}$, where $\mathcal{C}$ is a complexity class, written $\mathcal{Q} = \mathcal{C}$, if it expresses exactly the queries with $\mathcal{C}$ data complexity. For two languages $\mathcal{Q}$ and $\mathcal{Q}'$, we write $\mathcal{Q} \leq \mathcal{Q}'$, if for each query $Q \in \mathcal{Q}$, we can construct a query $Q' \in \mathcal{Q}'$ such that $Q(D) = Q'(D)$, for every database D. $\mathcal{Q}'$ is *more expressive* than $\mathcal{Q}$, written $\mathcal{Q} < \mathcal{Q}'$, if $\mathcal{Q} \leq \mathcal{Q}' \not\leq \mathcal{Q}$, while $\mathcal{Q}$ and $\mathcal{Q}'$ have the *same expressive power*, written $\mathcal{Q} = \mathcal{Q}'$, if $\mathcal{Q} \leq \mathcal{Q}' \leq \mathcal{Q}$.

Two query languages that are extremely important for our analysis are $\mathsf{DATALOG}^{\neg,\vee}_s$, where $s \in \{\mathsf{c}, \mathsf{b}\}$. These languages are defined in the same way as $\mathsf{WATGD}^\neg_s$ with the only difference that the query program is a set of NDTGDs of the form

$$\forall \mathbf{X} \forall \mathbf{Y} \left(\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \bigvee_{i=1}^{n} p_i(\mathbf{X}) \right),$$

where p_i is a single predicate, i.e., NDTGDs where the heads are existential-free disjunctions of atoms. Interestingly, $\mathsf{WATGD}^\neg_\mathsf{c}$ and $\mathsf{DATALOG}^{\neg,\vee}_\mathsf{c}$ have the same expressive power:

THEOREM 15. $\mathsf{WATGD}^\neg_\mathsf{c} = \mathsf{DATALOG}^{\neg,\vee}_\mathsf{c}$.

$\mathsf{DATALOG}^{\neg,\vee}_\mathsf{c}$ expresses exactly the queries with Π^P_2 data complexity [14]. By Theorem 14, $\mathsf{WATGD}^\neg_\mathsf{c}$ is in Π^P_2 in data complexity, and the "$\leq$" direction follows. Consider now a $\mathsf{DATALOG}^{\neg,\vee}_\mathsf{c}$ query $Q = (\Sigma, q)$. We construct the query $Q' = (\Sigma', q')$, where Σ' is defined as follows.

Simulate Predicates. We uniquely identify each predicate p in Σ via an atom $\mathtt{pred}_p(\cdot)$. This is done using the following set of rules: for each $p \in sch(\Sigma)$,

$$\rightarrow \exists X\, \mathtt{pred}_p(X),$$

and for each pair of distinct predicates $p, s \in sch(\Sigma)$,

$$\forall X (\mathtt{pred}_p(X) \wedge \mathtt{pred}_s(X) \rightarrow \mathtt{false}).$$

Notice that $\mathtt{false}$ is a special 0-ary predicate that is forced to be false in every stable model via the usual auxiliary rule $\mathtt{false} \wedge \neg\mathtt{aux} \rightarrow \mathtt{aux}$, where $\mathtt{aux} \notin sch(\Sigma)$.

Simulate Disjunction. We proceed to simulate disjunction. We could apply the construction given above for showing Lemma 13. However, since the query program Σ is existential-free, we can present a simplified version of it, which will allows us to easily verify that the obtained set Σ' of NTGDs is weakly-acyclic. Formally, for each rule ρ in Σ of the form

$$p_1(\mathbf{X}) \vee \ldots \vee p_m(\mathbf{X}) \leftarrow \varphi(\mathbf{X}, \mathbf{Y}),$$

we have in Σ' the following NTGDs:

$$\forall \mathbf{X} \forall \mathbf{Y} (\varphi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists Z\, t_\rho(Z, \mathbf{X}))$$
$$\forall Z \forall \mathbf{X} (t_\rho(Z, \mathbf{X}) \wedge \neg\mathtt{pred}_{p_1}(Z) \wedge \ldots \wedge \neg\mathtt{pred}_{p_m}(Z) \rightarrow \mathtt{false})$$
$$\forall Z \forall \mathbf{X} (t_\rho(Z, \mathbf{X}) \wedge \mathtt{pred}_{p_i}(Z) \rightarrow p_i(\mathbf{X})), \text{ for each } i \in [m]$$
$$\forall \mathbf{X} \forall \mathbf{Y} \forall Z (\varphi(\mathbf{X}, \mathbf{Y}) \wedge p_i(\mathbf{X}) \wedge \mathtt{pred}_{p_i}(Z) \rightarrow t_\rho(Z, \mathbf{X})), \text{ for each } i \in [m]$$
$$\forall \mathbf{X} (q(\mathbf{X}) \rightarrow q'(\mathbf{X})).$$

We can show that $Q(D) = Q'(D)$, for every database D. Moreover, in the position graph G of $(\Sigma')^+$ the only special edges are edges of the form $(\cdot, t_\rho[1])$, due to the first NTGD above, while there are no edges of the form $(t_\rho[1], \cdot)$ since there is no NTGD in Σ' with a body-variable that appears at position $t_\rho[1]$ and is also propagated to the head. Therefore, in G there is no cycle that contains a special edge, which implies that $\Sigma' \in \mathsf{WATGD}^\neg$, and Theorem 15 follows. By employing a similar construction, we can show that Theorem 15 holds even for the brave semantics:

THEOREM 16. $\mathsf{WATGD}^\neg_\mathsf{b} = \mathsf{DATALOG}^{\neg,\vee}_\mathsf{b}$.

As already mentioned above, $\mathsf{DATALOG}^{\neg,\vee}_\mathsf{c}$ expresses exactly the database queries that can be decided in Π^P_2 in data complexity [14]. Moreover, $\mathsf{DATALOG}^{\neg,\vee}_\mathsf{b}$ expresses exactly the queries that can be decided in Σ^P_2 in data complexity [14]. Consequently, by Theorem 14, which establishes the desired data complexity upper bounds, and Theorems 15 and 16, we obtain the following:

THEOREM 17. *It holds that,*

$$\mathsf{WATGD}^\neg_s = \begin{cases} \Pi^P_2, & \text{if } s = \mathsf{c}, \\ \Sigma^P_2, & \text{if } s = \mathsf{b}. \end{cases}$$

By exploiting the complexity analysis in Section 6, we can show that: $\mathsf{WATGD}^{\neg,\vee}_s$ is Π^P_2-complete, if $s = \mathsf{c}$, and Σ^P_2-complete, if $s = \mathsf{b}$, both in data complexity. Thus, an immediate consequence of Theorem 17 is that disjunction does not add expressive power:

THEOREM 18. *It holds that,*

$$\mathsf{WATGD}^\neg_\mathsf{c} = \mathsf{WATGD}^{\neg,\vee}_\mathsf{c} \qquad \mathsf{WATGD}^\neg_\mathsf{b} = \mathsf{WATGD}^{\neg,\vee}_\mathsf{b}.$$

It remains to understand how the different approaches to stable model semantics affect expressiveness. We would like to compare, in terms of expressive power, our new approach with the LP approach. To this end, we consider the class of normal rules obtained after Skolemizing weakly-acyclic sets of N(D)TGDs. Formally,

$$\mathsf{SWATGD}^{\neg(,\vee)} = \{\mathsf{sk}(\Sigma) \mid \Sigma \in \mathsf{WATGD}^{\neg(,\vee)}\},$$

where $\mathsf{sk}(\Sigma)$ is the set of normal rules obtained after Skolemizing the N(D)TGDs of Σ. By Theorem 1, the LP approach coincides with our approach when we focus on Skolemized NDTGDs,[6] and thus we simply need to compare $\mathsf{WATGD}^\neg_s$, where $s \in \{\mathsf{c}, \mathsf{b}\}$, with

[6]In fact, Theorem 1 considers only NTGDs, but it can be easily extended to NDTGDs.

$\mathsf{SWATGD}_s^{\neg}$ and $\mathsf{SWATGD}_s^{\neg,\vee}$ relative to our new stable model semantics. By employing a simple complexity-theoretic argument, it can be shown that our new approach gives rise to more expressive query languages than the LP approach when disjunction is not allowed, assuming that the polynomial hierarchy does not collapse. In particular, it is possible to show that $\mathsf{SWATGD}_{\mathsf{c}}^{\neg}$ is in coNP and $\mathsf{SWATGD}_{\mathsf{b}}^{\neg}$ is in NP, which immediately implies that:

THEOREM 19. *If* $\mathrm{NP} \subsetneq \Sigma_2^P$ *and* $\mathrm{coNP} \subsetneq \Pi_2^P$, *then*

$$\mathsf{WATGD}_{\mathsf{c}}^{\neg} > \mathsf{SWATGD}_{\mathsf{c}}^{\neg} \qquad \mathsf{WATGD}_{\mathsf{b}}^{\neg} > \mathsf{SWATGD}_{\mathsf{b}}^{\neg}.$$

Now, if disjunction is allowed to appear in rule-heads, then the LP approach gives rise to equally expressive languages:

THEOREM 20. *It holds that,*

$$\mathsf{WATGD}_{\mathsf{c}}^{\neg} = \mathsf{SWATGD}_{\mathsf{c}}^{\neg,\vee} \quad \mathsf{WATGD}_{\mathsf{b}}^{\neg} = \mathsf{SWATGD}_{\mathsf{b}}^{\neg,\vee}.$$

8. CONCLUSIONS

We propose a new approach to stable model semantics for NTGDs, based on a recent characterization of stable models in terms of second-order logic [18], that can be directly applied to rules with existentially quantified variables without requiring Skolemization. We show that the class of NTGDs based on weak-acyclicity can be reconciled with our new approach to stable negation, and we obtain precise complexity results for query answering. However, once we focus on stickiness and guardedness query answering becomes undecidable. Although for sticky sets of NTGDs this was expected, for guarded NTGDs this is a surprising negative result. Our next step is to understand whether there is a way to reconcile stickiness and guardedness with our new approach to stable models.

Acknowledgements: We thank the anonymous referees for their helpful feedback. Alviano has been partially supported by project "Smarter Solutions in the Big Data World (S2BDW)" funded by the Italian Ministry for Economic development (MISE), PON "Imprese e competitività" 2014-2020. Morak was supported by the Austrian Science Fund (FWF), project Y698. Pieris was supported by the EPSRC Programme Grant EP/M025268/ "VADA: Value Added Data Systems – Principles and Architecture".

9. REFERENCES

[1] M. Alviano and A. Pieris. Default negation for non-guarded existential rules. In *PODS*, pages 79–90, 2015.

[2] J. Baget, F. Garreau, M. Mugnier, and S. Rocher. Extending acyclicity notions for existential rules. In *ECAI*, pages 39–44, 2014.

[3] J. Baget, F. Garreau, M. Mugnier, and S. Rocher. Revisiting chase termination for existential rules and their extension to nonmonotonic negation. In *NMR*, 2014.

[4] J. Baget, M. Leclère, M. Mugnier, and E. Salvat. On rules with existential variables: Walking the decidability line. *Artif. Intell.*, 175(9-10):1620–1654, 2011.

[5] C. Beeri and M. Y. Vardi. The implication problem for data dependencies. In *ICALP*, pages 73–85, 1981.

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

[7] A. Calì, G. Gottlob, and M. Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res.*, 48:115–174, 2013.

[8] A. Calì, G. Gottlob, and T. Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *J. Web Sem.*, 14:57–83, 2012.

[9] A. Calì, G. Gottlob, T. Lukasiewicz, B. Marnette, and A. Pieris. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *LICS*, pages 228–242, 2010.

[10] A. Calì, G. Gottlob, and A. Pieris. Towards more expressive ontology languages: The query answering problem. *Artif. Intell.*, 193:87–128, 2012.

[11] B. Courcelle. The monadic second-order logic of graphs, II: infinite graphs of bounded width. *Mathematical Systems Theory*, 21(4):187–221, 1989.

[12] B. Durand and A. Fabret. On the complexity of deadlock detection in families of planar nets. *Theor. Comput. Sci.*, 215(1-2):225–237, 1999.

[13] T. Eiter and G. Gottlob. On the computational cost of disjunctive logic programming: Propositional case. *Ann. Math. Artif. Intell.*, 15(3-4):289–323, 1995.

[14] T. Eiter, G. Gottlob, and H. Mannila. Disjunctive datalog. *ACM Trans. Database Syst.*, 22(3):364–418, 1997.

[15] T. Eiter and M. Simkus. FDNC: decidable nonmonotonic disjunctive logic programs with function symbols. *ACM Trans. Comput. Log.*, 11(2), 2010.

[16] R. Fagin. Inverting schema mappings. *ACM Trans. Database Syst.*, 32(4), 2007.

[17] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: Semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[18] P. Ferraris, J. Lee, and V. Lifschitz. Stable models and circumscription. *Artif. Intell.*, 175(1):236–263, 2011.

[19] A. V. Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *J. ACM*, 38(3):620–650, 1991.

[20] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *ICLP/SLP*, pages 1070–1080, 1988.

[21] G. Gottlob, A. Hernich, C. Kupke, and T. Lukasiewicz. Equality-friendly well-founded semantics and applications to description logics. In *AAAI*, 2012.

[22] G. Gottlob, A. Hernich, C. Kupke, and T. Lukasiewicz. Stable model semantics for guarded existential rules and description logics. In *KR*, 2014.

[23] E. Grädel. On the restraining power of guards. *J. Symb. Log.*, 64(4):1719–1742, 1999.

[24] N. Leone, M. Manna, G. Terracina, and P. Veltri. Efficiently computable Datalog$^{\exists}$ programs. In *KR*, 2012.

[25] D. Magka, M. Krötzsch, and I. Horrocks. Computing stable models for nonmonotonic existential rules. In *IJCAI*, 2013.

[26] J. McCarthy. Circumscription - A form of non-monotonic reasoning. *Artif. Intell.*, 13(1-2):27–39, 1980.

[27] R. Pichler and S. Skritek. The complexity of evaluating tuple generating dependencies. In *ICDT*, pages 244–255, 2011.

[28] T. C. Przymusinski. On the declarative semantics of deductive databases and logic programs. In *Foundations of Deductive Databases and Logic Programming*, pages 193–216. Morgan Kaufmann, 1988.

[29] I. A. Stewart. Complete problems involving boolean labelled structures and projection transactions. *J. Log. Comput.*, 1(6):861–882, 1991.

[30] B. ten Cate, G. Fontaine, and P. G. Kolaitis. On the data complexity of consistent query answering. *Theory Comput. Syst.*, 57(4):843–891, 2015.