Revisiting Chase Termination for Existential Rules and their Extension to Nonmonotonic Negation

Jean-François Baget, Fabien Garreau, Marie-Laure Mugnier, Swan Rocher

Revisiting Chase Termination for Existential Rules and their Extension to Nonmonotonic Negation

Jean-François Baget
INRIA

Fabien Garreau
University of Angers

Marie-Laure Mugnier
University of Montpellier

Swan Rocher
University of Montpellier

Abstract

Existential rules have been proposed for representing ontological knowledge, specifically in the context of Ontology-Based Data Access. Entailment with existential rules is undecidable. We focus in this paper on conditions that ensure the termination of a breadth-first forward chaining algorithm known as the chase. Several variants of the chase have been proposed. In the first part of this paper, we propose a new tool that allows to extend existing acyclicity conditions ensuring chase termination, while keeping good complexity properties. In the second part, we study the extension to existential rules with nonmonotonic negation under stable model semantics, discuss the relevancy of the chase variants for these rules and further extend acyclicity results obtained in the positive case.

Introduction

Existential rules (also called Datalog+/-) have been proposed for representing ontological knowledge, specifically in the context of Ontology-Based Data Access, that aims to exploit ontological knowledge when accessing data (Calì, Gottlob, and Lukasiewicz 2009a; Baget et al. 2009). These rules allow to assert the existence of unknown individuals, a feature recognized as crucial for representing knowledge in an open domain perspective. Existential rules generalize lightweight description logics, such as DL-Lite and $\mathcal{EL}$ (Calvanese et al. 2007; Baader, Brandt, and Lutz 2005) and overcome some of their limitations by allowing any predicate arity as well as cyclic structures.

Entailment with existential rules is known to be undecidable (Beeri and Vardi 1981; Chandra, Lewis, and Makowsky 1981). Many sufficient conditions for decidability, obtained by syntactic restrictions, have been exhibited in knowledge representation and database theory (see e.g., the overview in (Mugnier 2011)). We focus in this paper on conditions that ensure the termination of a breadth-first forward chaining algorithm, known as the *chase* in the database literature. Given a knowledge base composed of data and existential rules, the chase saturates the data by application of the rules. When it is ensured to terminate, inferences enabled by the rules can be materialized in the data, which can then be queried like a classical database, thus allowing to benefit from any database optimizations technique. Several variants of the chase have been proposed, which differ in the way they deal with redundant information (Fagin et al. 2005; Deutsch, Nash, and Remmel 2008; Marnette 2009). It follows that they do not behave in the same way with respect to termination. In the following, when we write *the* chase, we mean one of these variants. Various acyclicity notions have been proposed to ensure the halting of some chase variants.

Nonmonotonic extensions to existential rules were recently considered in (Calı, Gottlob, and Lukasiewicz 2009b) with stratified negation, (Gottlob et al. 2012) with well-founded semantics and (Magka, Krötzsch, and Horrocks 2013) with stable model semantics. This latter work studies skolemized existential rules and focuses on cases where a finite unique model exists.

In this paper, we tackle the following issues : Can we still extend known acyclicity notions ? Would any chase variant be applicable to existential rules provided with nonmonotonic negation, a useful feature for ontological modeling ?

1. Extending acyclicity notions. Acyclicity conditions can be classified into two main families : the first one constrains the way existential variables are propagated during the chase (e.g. (Fagin et al. 2003; Fagin et al. 2005; Marnette 2009; Krötzsch and Rudolph 2011)) and the second one encodes dependencies between rules, i.e., the fact that a rule may lead to trigger another rule (e.g. (Baget 2004; Deutsch, Nash, and Remmel 2008; Baget et al. 2011)). These conditions are based on different graphs, but all of them can be seen as can as forbidding "dangerous" cycles in the considered graph. We define a new family of graphs that allows to extend these acyclicity notions, while keeping good complexity properties.

2. Processing rules with nonmonotonic negation. We define a notion of stable models on nonmonotonic existential rules and provide a derivation algorithm that instantiate rules "on the fly" (Lefèvre and Nicolas 2009; Dao-Tran et al. 2012). This algorithm is parametrized by a chase variant. We point out that, differently to the positive case, not all variants of the chase lead to sound procedures in presence of nonmonotonic negation ; furthermore, skolemizing existential variables or not makes a semantic difference, even when both computations terminate. Finally, we further extend acyclicity results obtained on positive rules by exploiting negative information as well.

A technical report with the proofs omit-

ted for space restriction reasons is available http://www2.lirmm.fr/~baget/publications/nmr2014-long-paper.pdf.

Preliminaries

Atomsets We consider first-order vocabularies with constants but no other function symbols. An *atom* is of the form $p(t_1, \ldots, t_k)$ where p is a predicate of arity k and the t_i are terms, i.e., variables or constants (in the paper we denote constants by $a, b, c, ...$ and variables by $x, y, z, ...$). An *atomset* is a set of atoms. Unless indicated otherwise, we will always consider *finite* atomsets. If F is an atom or an atomset, we write $terms(F)$ (resp. $vars(F)$, resp. $csts(F)$) the set of terms (resp. variables, resp. constants) that occur in F. If F is an atomset, we write $\phi(F)$ the formula obtained by the conjunction of all atoms in F, and $\Phi(F)$ the existential closure of $\phi(F)$. We say that an atomset F *entails* an atomset Q (notation $F \models Q$) if $\Phi(F) \models \Phi(Q)$. It is well-known that $F \models Q$ iff there exists a *homomorphism* from Q to F, i.e., a *substitution* $\sigma : vars(F) \to terms(Q)$ such that $\sigma(Q) \subseteq F$. Two atomsets F and F' are said to be *equivalent* if $F \models F'$ and $F' \models F$. If there is a homomorphism σ from an atomset F to itself (i.e., an *endomorphism* of F) then F and $\sigma(F)$ are equivalent. An atomset F is a *core* if there is no homomorphism from F to one of its strict subsets. Among all atomsets equivalent to an atomset F, there exists a unique core (up to isomorphism). We call this atomset *the* core of F.

Existential Rules An *existential rule* (and simply a rule hereafter) is of the form $B \to H$, where B and H are atomsets, respectively called the *body* and the *head* of the rule. To an existential rule $R : B \to H$ we assign a formula $\Phi(R) = \forall \vec{x} \forall \vec{y}(\phi(B) \to \exists \vec{z} \phi(H))$, where $vars(B) = \vec{x} \cup \vec{y}$, and $vars(H) = \vec{x} \cup \vec{z}$. Variables $\vec{x}$, which appear in both B and H, are called *frontier variables*, while variables $\vec{z}$, which appear only in H are called *existential variables*. E.g., $\Phi(b(x,y) \to h(x,z)) = \forall x \forall y(b(x,y) \to \exists z h(x,z))$. The presence of existential variables in rule heads is the distinguishing feature of existential rules.

A *knowledge base* is a pair $K = (F, \mathcal{R})$ where F is an atomset (the set of facts) and $\mathcal{R}$ is a finite set of existential rules. We say that $K = (F, \{R_1, \ldots, R_k\})$ *entails* an atomset Q (notation $K \models Q$) if $\Phi(F), \Phi(R_1), \ldots, \Phi(R_k) \models \Phi(Q)$. The fundamental problem we consider, denoted by ENTAILMENT, is the following : given a knowledge base K and an atomset Q, is it true that $K \models Q$? When $\Phi(Q)$ is seen as a Boolean conjunctive query, this problem is exactly the problem of determining if K yields a positive answer to this query.

A rule $R : B \to H$ is *applicable* to an atomset F if there is a homomorphism π from B to F. Then the *application of R to F according to π* produces an atomset $\alpha(F, R, \pi) = F \cup \pi(safe(H))$, where $safe(H)$ is obtained from H by replacing existential variables with fresh ones. An $\mathcal{R}$-derivation from F is a (possibly infinite) sequence $F_0 = \sigma_0(F), \ldots, \sigma_k(F_k), \ldots$ of atomsets such that $\forall 0 \leq i$, σ_i is an endomorphism of F_i (that will be used to remove redundancy in F_i) and $\forall 0 < i$, there is a rule $(R : B \to H) \in \mathcal{R}$ and a homomorphism π_i from B to $\sigma_i(F_{i-1})$ such that $F_i = \alpha(\sigma_i(F_{i-1}), R, \pi_i)$.

Example 1 *Consider the existential rule $R = human(x) \to hasParent(x,y), human(y)$; and the atomset $F = \{human(a)\}$. The application of R to F produces an atomset $F' = F \cup \{hasParent(x, y_0), human(y_0)\}$ where y_0 is a fresh variable denoting an unknown individual. Note that R could be applied again to F' (mapping x to y_0), which would create another existential variable and so on.*

A finite $\mathcal{R}$-derivation $F_0, \ldots, F_k$ from F is said to be *from F to F_k*. Given a knowledge base $K = (F, \mathcal{R})$, $K \models Q$ iff there exists a finite $\mathcal{R}$-derivation from F to F' such that $F' \models Q$ (Baget et al. 2011).

Let R_i and R_j be rules, and F be an atomset such that R_i is applicable to F by a homomorphism π ; a homomorphism π' from B_j to $F' = \alpha(F, R_i, \pi)$ is said to be *new* if $\pi'(B_j) \not\subseteq F$. Given a rule $R = B \to H$, a homomorphism π from B to F is said to be *useful* if it cannot be extended to a homomorphism from $B \cup H$ to F ; if π is not useful then $\alpha(F, R, \pi)$ is equivalent to F, but this is not a necessary condition for $\alpha(F, R, \pi)$ to be equivalent to F.

Chase Termination

An algorithm that computes an $\mathcal{R}$-derivation by exploring all possible rule applications in a breadth-first manner is called a *chase*. In the following, we will also call chase the derivation it computes. Different kinds of chase can be defined by using different properties to compute $F'_i = \sigma_i(F_i)$ in the derivation (hereafter we write F'_i for $\sigma_i(F_i)$ when there is no ambiguity). All these algorithms are sound and complete w.r.t. the ENTAILMENT problem in the sense that $(F, \mathcal{R}) \models Q$ iff they provide in finite (but unbounded) time a finite $\mathcal{R}$-derivation from F to F_k such that $F_k \models Q$.

Different kinds of chase In the *oblivious chase* (also called naive chase), e.g., (Calì, Gottlob, and Kifer 2008), a rule R is applied according to a homomorphism π only if it has not already been applied according to the same homomorphism. Let $F_i = \alpha(F'_{i-1}, R, \pi)$, then $F'_i = F'_{i-1}$ if R was previously applied according to π, otherwise $F'_i = F_i$. This can be slightly improved. Two applications π and π' of the same rule add the same atoms if they map frontier variables identically (for any frontier variable x of R, $\pi(x) = \pi'(x)$) ; we say that they are frontier-equal. In the *frontier chase*, let $F_i = \alpha(F'_{i-1}, R, \pi)$, we take $F'_i = F'_{i-1}$ if R was previously applied according to some π' frontier-equal to π, otherwise $F'_i = F_i$. The *skolem chase* (Marnette 2009) relies on a skolemisation of the rules : a rule R is transformed into a rule $skolem(R)$ by replacing each occurrence of an existential variable y with a functional term $f^R_y(\vec{x})$, where $\vec{x}$ are the frontier variables of R. Then the oblivious chase is run on skolemized rules. It can easily be checked that frontier chase and skolem chase yield isomorphic results, in the sense that they generate exactly the same atomsets, up to a bijective renaming of variables by skolem terms.

The *restricted chase* (also called standard chase) (Fagin et al. 2005) detects a kind of local redundancy. Let $F_i = \alpha(F'_{i-1}, R, \pi)$, then $F'_i = F_i$ if π is useful, otherwise $F'_i = F'_{i-1}$. The *core chase* (Deutsch, Nash, and Remmel 2008) considers the strongest possible form of redundancy : for any F_i, F'_i is the core of F_i.

A chase is said to be *local* if $\forall i \leq j$, $F'_i \subseteq F'_j$. All chase variants presented above are local, *except for the core chase*. This property will be critical for nonmonotonic existential rules.

Chase termination Since ENTAILMENT is undecidable, the chase may not halt. We call *C-chase* a chase relying on some criterion C to generate $\sigma(F_i) = F'_i$. So C can be oblivious, skolem, restricted, core or any other criterion that ensures the equivalence between F_i and F'_i. A C-chase generates a possibly infinite $\mathcal{R}$-derivation $\sigma_0(F), \sigma_1(F_1), \ldots, \sigma_k(F_k), \ldots$

We say that this derivation *produces* the (possibly infinite) atomset $(F, \mathcal{R})^C = \cup_{0 \leq i \leq \infty} \sigma_i(F_i) \setminus \cup_{0 \leq i \leq \infty} \overline{(\sigma_i(F_i))}$, where $\overline{(\sigma_i(F_i))} = F_i \setminus \sigma(F_i)$. Note that this produced atomset is usually defined as the infinite union of the $\sigma_i(F_i)$. Both definitions are equivalent when the criterion C is *local*. But the usual definition would produce too big an atomset with a non-local chase such as the core chase : an atom generated at step i and removed at step j would still be present in the infinite union. We say that a (possibly infinite) derivation obtained by the C-chase is *complete* when any further rule application on that derivation would produce the same atomset. A complete derivation obtained by any C-chase produces a *universal model* (i.e., most general) of $(F, \mathcal{R})$: for any atomset Q, we have $F, \mathcal{R} \models Q$ iff $(F, \mathcal{R})^C \models Q$.

We say that the C-chase *halts* on $(F, \mathcal{R})$ when the C-chase generates a finite complete $\mathcal{R}$-derivation from F to F_k. Then $(F, \mathcal{R})^C = \sigma_k(F_k)$ is a finite universal model. We say that $\mathcal{R}$ is *universally C-terminating* when the C-chase halts on $(F, \mathcal{R})$ for any atomset F. We call *C-finite* the class of universally C-terminating sets of rules. It is well known that the chase variants do not behave in the same way w.r.t. termination. The following examples highlight these different behaviors.

Example 2 (Oblivious / Skolem chase) *Let $R = p(x, y) \rightarrow p(x, z)$ and $F = \{p(a, b)\}$. The oblivious chase does not halt : it adds $p(a, z_0)$, $p(a, z_1)$, etc. The skolem chase considers the rule $p(x, y) \rightarrow p(x, f^R_z(x))$; it adds $p(a, f^R_y(a))$ then halts.*

Example 3 (Skolem / Restricted chase) *Let $R : p(x) \rightarrow r(x, y), r(y, y), p(y)$ and $F = \{p(a)\}$. The skolem chase does not halt : at Step 1, it maps x to a and adds $r(a, f^R_y(a))$, $r(f^R_y(a), f^R_y(a))$ and $p(f^R_y(a))$; at step 2, it maps x to $f^R_y(a)$ and adds $r(f^R_y(a), f^R_y(f^R_y(a)))$, etc. The restricted chase performs a single rule application, which adds $r(a, y_0)$, $r(y_0, y_0)$ and $p(y_0)$; indeed, the rule application that maps x to y_0 yields only redundant atoms w.r.t. $r(y_0, y_0)$ and $p(y_0)$.*

Example 4 (Restricted / Core chase) *Let $F = s(a)$, $R_1 = s(x) \rightarrow p(x, x_1), p(x, x_2), r(x_2, x_2)$, $R_2 = p(x, y) \rightarrow q(y)$ and $R_3 = q(x) \rightarrow r(x, y), q(y)$. Note that R_1 creates redundancy and R_3 could be applied indefinitely if it were the only rule. R_1 is the first applied rule, which creates new variables, called x_1 and x_2 for simplicity. The restricted chase does not halt : R_3 is not applied on x_2 because it is already satisfied at this point, but it is applied on x_1, which creates an infinite chain. The core chase applies R_1, computes the core of the result, which removes $p(a, x_1)$, then halts.*

It is natural to consider the oblivious chase as the weakest form of chase and necessary to consider the core chase as the strongest form of chase (since the core is the minimal representative of its equivalence class). We say that a criterion C is *stronger* than C' and write $C \geq C'$ when C'-finite $\subseteq$ C-finite. We say that C is *strictly stronger* than C' (and write $C > C'$) when $C \geq C'$ and $C' \not\geq C$.

It is well-known that core > restricted > skolem > oblivious. An immediate remark is that core-finite corresponds to *finite expansion sets (fes)* defined in (Baget and Mugnier 2002). To sum up, the following inclusions hold between C-finite classes : oblivious-finite $\subset$ skolem-finite = frontier-finite $\subset$ restricted-finite $\subset$ core-finite = fes.

Known Acyclicity Notions

We can only give a brief overview of known acylicity notions, which should however allow to place our contribution within the existing landscape. A comprehensive taxonomy can be found in (Cuenca Grau et al. 2013).

Acyclicity notions ensuring that some chase variant terminates can be divided into two main families, each of them relying on a different graph : a "position-based" approach which basically relies on a graph encoding variable sharing between positions in predicates and a "rule dependency approach" which relies on a graph encoding dependencies between rules, i.e., the fact that a rule may lead to trigger another rule (or itself).

Position-based approach In the position-based approach, cycles identified as dangerous are those passing through positions that may contain existential variables ; intuitively, such a cycle means that the creation of an existential variable in a given position may lead to create another existential variable in the same position, hence an infinite number of existential variables. Acyclicity is then defined by the absence of dangerous cycles. The simplest notion of acyclicity in this family is that of *weak acyclicity (wa)* (Fagin et al. 2003) (Fagin et al. 2005), which has been widely used in databases. It relies on a directed graph whose nodes are the positions in predicates (we denote by (p, i) the position i in predicate p). Then, for each rule $R : B \rightarrow H$ and each variable x in B occurring in position (p, i), edges with origin (p, i) are built as follows : if x is a frontier variable, there is an edge from (p, i) to each position of x in H ; furthermore, for each existential variable y in H occurring in position (q, j), there is a special edge from (p, i) to (q, j). A set of rules is weakly acyclic if its associated graph has no cycle passing through a special edge.

Example 5 (Weak-acyclicity) *Let $R_1 = h(x) \rightarrow p(x, y)$, where y is an existential variable, and $R_2 = p(u, v), q(v) \rightarrow h(v)$. The position graph of $\{R_1, R_2\}$ contains a special edge from $(h, 1)$ to $(p, 2)$ due to R_1 and an edge from $(p, 2)$ to $(h, 1)$ due to R_2, thus $\{R_1, R_2\}$ is not wa.*

Weak-acyclicity has been generalized, mainly by shifting the focus from positions to existential variables (*joint-acyclicity (ja)*(Krötzsch and Rudolph 2011)) or to positions in atoms instead of predicates (*super-weak-acyclicity (swa)* (Marnette 2009)). Other related notions can be imported from logic programming, e.g., *finite domain*

(fd) (Calimeri et al. 2008) and *argument-restricted (ar)* (Lierler and Lifschitz 2009). See the first column in Figure 1, which shows the inclusions between the corresponding classes of rules (all these inclusions are known to be strict).

Rule Dependency In the second approach, the aim is to avoid cyclic triggering of rules (Baget 2004; Baget et al. 2009; Deutsch, Nash, and Remmel 2008; Cuenca Grau et al. 2012). We say that a rule R_2 *depends* on a rule R_1 if there exists an atomset F such that R_1 is applicable to F according to a homomorphism π and R_2 is applicable to $F' = \alpha(F, R_1, \pi)$ according to a new useful homomorphism. This abstract dependency relation can be effectively computed with a unification operation known as piece-unifier (Baget et al. 2009). Piece-unification takes existential variables into account, hence is more complex than the usual unification between atoms. A *piece-unifier* of a rule body B_2 with a rule head H_1 is a substitution μ of $vars(B'_2) \cup vars(H'_1)$, where $B'_2 \subseteq B_2$ and $H'_1 \subseteq H_1$, such that *(1)* $\mu(B'_2) = \mu(H'_1)$ and *(2)* existential variables in H'_1 are not unified with separating variables of B'_2, i.e., variables that occur both in B'_2 and in $(B_2 \setminus B'_2)$; in other words, if a variable x occuring in B'_2 is unified with an existential variable y in H'_1, then all atoms in which x occurs also belong to B'_2. It holds that R_2 depends on R_1 iff there is a piece-unifier of B_2 with H_1 satisfying easy to check additional conditions (atom erasing (Baget et al. 2011) and usefulness (Cuenca Grau et al. 2013)).

Example 6 (Rule dependency) *Consider the rules from Example 5. There is no piece-unifier of B_2 with H_1. The substitution $\mu = \{(u, x), (v, y)\}$, with $B'_2 = p(u, v)$ and $H'_1 = H_1$, is not a piece-unifier because v is unified with an existential variable, whereas it is a separating variable of B'_2 (thus, $q(v)$ should be included in B'_2, which is impossible). Thus R_2 does not depend on R_1.*

The *graph of rule dependencies* of a set of rules $\mathcal{R}$, denoted by GRD($\mathcal{R}$), encodes the dependencies between rules in $\mathcal{R}$. It is a directed graph with set of nodes $\mathcal{R}$ and an edge (R_i, R_j) if R_j depends on R_i (intuition : "R_i may lead to trigger R_j in a new way"). E.g., considering the rules in Example 6, the only edge is (R_2, R_1).

When the GRD is acyclic (*aGRD*, (Baget 2004)), any derivation sequence is necessarily finite. This notion is incomparable with those based on positions.

We point out here that the *oblivious* chase may not stop on *wa* rules. Thus, the only acyclicity notion in Figure 1 that ensures the termination of the oblivious chase is *aGRD* since all other notions generalize *wa*.

Combining both approches Both approaches have their weaknesses : there may be a dangerous cycle on positions but no cycle w.r.t. rule dependencies (see the preceeding examples), and there may be a cycle w.r.t. rule dependencies whereas rules contain no existential variables (e.g. $p(x, y) \rightarrow p(y, x), q(x)$). Attempts to combine both notions only succeded to combine them in a "modular way" : if the rules in each strongly connected component (s.c.c.) of the GRD belong to a *fes* class, then the set of rules is *fes* (Baget 2004; Deutsch, Nash, and Remmel 2008). More specifically, it is easy to check that if for a given C-chase, each s.c.c. is C-finite, then the C-chase stops.

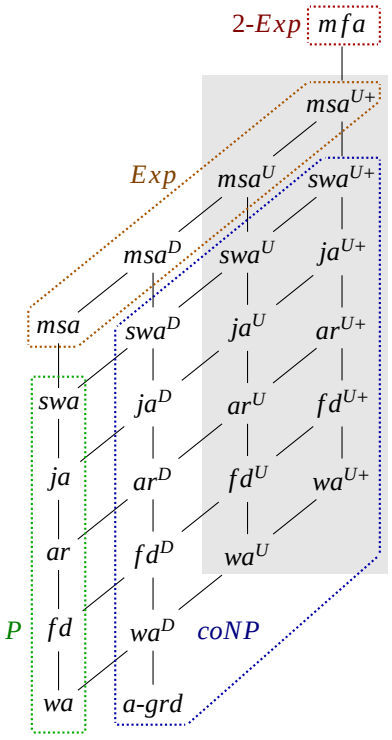

FIGURE 1 – Relations between recognizable acyclicity properties. All inclusions are strict and complete (i.e., if there is no path between two properties then they are incomparable).

In this paper, we propose an "integrated" way of combining both approaches, which relies on a single graph. This allows to unify preceding results and to generalize them without complexity increasing (the new acyclicity notions are those with a gray background in Figure 1).

Finally, let us mention *model-faithful acyclicity (mfa)* (Cuenca Grau et al. 2012), which generalizes the previous acyclicity notions and cannot be captured by our approach. Briefly, mfa involves running the skolem chase until termination or a cyclic functional term is found. The price to pay for the generality of this property is high complexity : checking if a set of rules is universally mfa (i.e., for any set of facts) is 2EXPTIME-complete. Checking *model-summarizing acyclicity (msa)*, which approximates mfa, remains EXPTIME-complete. In contrast, checking position-based properties is in PTIME and checking agrd is also co-NP-complete. Sets of rules satisfying mfa are skolem-finite (Cuenca Grau et al. 2012), thus all properties studied in this paper ensure C-finiteness, when $C \geq$ skolem.

Extending Acyclicity Notions

In this section, we combine rule dependency and propagation of existential variables into a single graph. W.l.o.g. we assume that distinct rules do not share any variable. Gi-

ven an atom $a = p(t_1, \ldots, t_k)$, the i^{th} position in a is denoted by $\langle a, i\rangle$, with $pred(\langle a, i\rangle) = p$ and $term(\langle a, i\rangle) = t_i$. If A is an atomset such that $a \in A$, we say that $\langle a, i\rangle$ is in A. If $term(\langle a, i\rangle)$ is an existential (resp. frontier) variable, $\langle a, i\rangle$ is called an *existential* (resp. *frontier*) position. In the following, we use "position graph" as a generic name to denote a graph whose nodes are positions in *atoms*.

We first define the notion of a basic position graph, which takes each rule in isolation. Then, by adding edges to this graph, we define three position graphs with increasing expressivity, i.e., allowing to check termination for increasingly larger classes of rules.

Definition 1 ((Basic) Position Graph (PG)) *The* position graph *of a rule $R : B \to H$ is the directed graph $PG(R)$ defined as follows :*

- *there is a node for each $\langle a, i\rangle$ in B or in H ;*
- *for all frontier positions $\langle b, i\rangle \in B$ and all $\langle h, j\rangle \in H$, there is an edge from $\langle b, i\rangle$ to $\langle h, j\rangle$ if $term(\langle b, i\rangle) = term(\langle h, j\rangle)$ or if $\langle h, j\rangle$ is existential.*

Given a set of rules $\mathcal{R}$, the basic position graph *of $\mathcal{R}$, denoted by $PG(\mathcal{R})$, is the disjoint union of $PG(R_i)$, for all $R_i \in \mathcal{R}$.*

An existential position $\langle a, i\rangle$ is said to be *infinite* if there is an atomset F such that running the chase on F produces an unbounded number of instantiations of $term(\langle a, i\rangle)$. To detect infinite positions, we encode how variables may be "propagated" among rules by adding edges to $PG(\mathcal{R})$, called *transition edges*, which go from positions in rule heads to positions in rule bodies. The set of transition edges has to be *correct* : if an existential position $\langle a, i\rangle$ is infinite, there must be a cycle going through $\langle a, i\rangle$ in the graph.

We now define three position graphs by adding transition edges to $PG(\mathcal{R})$, namely $PG^F(\mathcal{R})$, $PG^D(\mathcal{R})$ and $PG^U(\mathcal{R})$. All three graphs have correct sets of edges. Intuitively, $PG^F(\mathcal{R})$ corresponds to the case where all rules are supposed to depend on all rules ; its set of cycles is in bijection with the set of cycles in the predicate position graph defining weak-acyclicity. $PG^D(\mathcal{R})$ encodes actual paths of rule dependencies. Finally, $PG^U(\mathcal{R})$ adds information about the piece-unifiers themselves. This provides an accurate encoding of variable propagation from an atom position to another.

Definition 2 (PG^X) *Let $\mathcal{R}$ be a set of rules. The three following position graphs are obtained from $PG(\mathcal{R})$ by adding a (transition) edge from each k^{th} position $\langle h, k\rangle$ in a rule head H_i to each k^{th} position $\langle b, k\rangle$ in a rule body B_j, with the* same predicate*, provided that some condition is satisfied :*

- full PG, *denoted by $PG^F(\mathcal{R})$: no additional condition ;*
- dependency PG, *denoted by $PG^D(\mathcal{R})$: if R_j depends directly or indirectly on R_i, i.e., if there is a path from R_i to R_j in $GRD(\mathcal{R})$;*
- PG with unifiers, *denoted by $PG^U(\mathcal{R})$: if there is a piece-unifier μ of B_j with the head of an agglomerated rule R_i^j such that $\mu(term([b, k])) = \mu(term([h, k]))$, where R_i^j is formally defined below (Definition 3)*

An agglomerated rule associated with (R_i, R_j) gathers information about selected piece-unifiers along (some) paths from R_i to (some) predecessors of R_j.

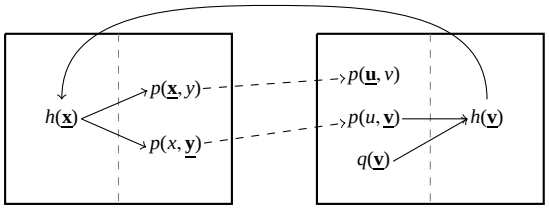

FIGURE 2 – $PG^F(\mathcal{R})$ and $PG^D(\mathcal{R})$ from Example 7. Position $\langle a, i\rangle$ is represented by underlining the i-th term in a. Dashed edges do not belong to $PG^D(\mathcal{R})$.

Definition 3 (Agglomerated Rule) *Given R_i and R_j rules from $\mathcal{R}$, an agglomerated rule associated with (R_i, Rj) has the following form :*

$$R_i^j = B_i \cup_{t \in T \subseteq terms(H_i)} fr(t) \to H_i$$

where fr is a new unary predicate that does not appear in $\mathcal{R}$, and the atoms $fr(t)$ are built as follows. Let $\mathcal{P}$ be a non-empty set of paths from R_i to direct predecessors of R_j in $GRD(\mathcal{R})$. Let $P = (R_1, \ldots, R_n)$ be a path in $\mathcal{P}$. One can associate a rule R^P with P by building a sequence $R_1 = R_1^p, \ldots, R_n^p = R^P$ such that $\forall 1 \leq l < n$, there is a piece-unifier μ_l of B_{l+1} with the head of R_l^p, where the body of R_{l+1}^p is $B_l^p \cup \{fr(t) \mid t$ is a term of H_l^p unified in $\mu_l\}$, and the head of R_{l+1}^p is H_1. Note that for all l, $H_l^p = H_1$, however, for $l \neq 1$, R_l^p may have less existential variables than R_l due to the added atoms. The agglomerated rule R_i^j built from $\{R^P | P \in \mathcal{P}\}$ is $R_i^j = \bigcup_{P \in \mathcal{P}} R^P$.

Proposition 1 (Inclusions between PG^X) *Let $\mathcal{R}$ be a set of rules. $PG^U(\mathcal{R}) \subseteq PG^D(\mathcal{R}) \subseteq PG^F(\mathcal{R})$. Furthermore, $PG^D(\mathcal{R}) = PG^F(\mathcal{R})$ if the transitive closure of $GRD(\mathcal{R})$ is a complete graph.*

Example 7 (PG^F and PG^D) *Let $\mathcal{R} = \{R_1, R_2\}$ from Example 5. Figure 2 pictures $PG^F(\mathcal{R})$ and $PG^D(\mathcal{R})$. The dashed edges belong to $PG^F(\mathcal{R})$ but not to $PG^D(\mathcal{R})$. Indeed, R_2 does not depend on R_1. $PG^F(\mathcal{R})$ has a cycle while $PG^D(\mathcal{R})$ has not.*

Example 8 (PG^D and PG^U) *Let $\mathcal{R} = \{R_1, R_2\}$, with $R_1 = t(x, y) \to p(z, y), q(y)$ and $R_2 = p(u, v), q(u) \to t(v, w)$. In Figure 3, the dashed edges belong to $PG^D(\mathcal{R})$ but not to $PG^U(\mathcal{R})$. Indeed, the only piece-unifier of B_2 with H_1 unifies u and y. Hence, the cycle in $PG^D(\mathcal{R})$ disappears in $PG^U(\mathcal{R})$.*

We now study how acyclicity properties can be expressed on position graphs. The idea is to associate, with an acyclicity property, a function that assigns to each position a subset of positions reachable from this position, according to some propagation constraints ; then, the property is fulfilled if no existential position can be reached from itself. More precisely, a *marking function Y* assigns to each node $\langle a, i\rangle$ in a position graph PG^X, a subset of its (direct or indirect) successors, called its *marking*. A *marked cycle* for $\langle a, i\rangle$ (w.r.t.

X and Y) is a cycle C in PG^X such that $\langle a, i\rangle \in C$ and for all $\langle a', i'\rangle \in C$, $\langle a', i'\rangle$ belongs to the marking of $\langle a, i\rangle$. Obviously, the less situations there are in which the marking may "propagate" in a position graph, the stronger the acyclicity property is.

Definition 4 (Acyclicity property) *Let Y be a marking function and PG^X be a position graph. The* acyclicity property *associated with Y in PG^X, denoted by Y^X, is satisfied if there is no marked cycle for an existential position in PG^X. If Y^X is satisfied, we also say that $PG^X(\mathcal{R})$* satisfies *Y.*

For instance, the marking function associated with weak-acyclicity assigns to each node the set of its successors in $PG^F(\mathcal{R})$, without any additional constraint. The next proposition states that such marking functions can be defined for each class of rules between *wa* and *swa* (first column in Figure 1), in such a way that the associated acyclicity property in PG^F characterizes this class.

Proposition 2 *A set of rules $\mathcal{R}$ is wa (resp. fd, ar, ja, swa) iff $PG^F(\mathcal{R})$ satisfies the acyclicity property associated with wa- (resp. fd-, ar-, ja-, swa-) marking.*

As already mentioned, all these classes can be safely extended by combining them with the GRD. To formalize this, we recall the notion $Y^<$ from (Cuenca Grau et al. 2013) : given an acyclicity property Y, a set of rules $\mathcal{R}$ is said to satisfy $Y^<$ if each s.c.c. of $GRD(\mathcal{R})$ satisfies Y, except for those composed of a single rule with no loop. [1] Whether $\mathcal{R}$ satisfies $Y^<$ can be checked on $PG^D(\mathcal{R})$:

Proposition 3 *Let $\mathcal{R}$ be a set of rules, and Y be an acyclicity property. $\mathcal{R}$ satisfies $Y^<$ iff $PG^D(\mathcal{R})$ satisfies Y, i.e., $Y^< = Y^D$.*

For the sake of brevity, if Y_1 and Y_2 are two acyclicity properties, we write $Y_1 \subseteq Y_2$ if any set of rules satisfying Y_1 also satisfies Y_2. The following results are straightforward.

Proposition 4 *Let Y_1, Y_2 be two acyclicity properties. If $Y_1 \subseteq Y_2$, then $Y_1^D \subseteq Y_2^D$.*

Proposition 5 *Let Y be an acyclicity property. If a-grd $\not\subseteq Y$ then $Y \subset Y^D$.*

Hence, any class of rules satisfying a property Y^D strictly includes both *a-grd* and the class characterized by Y ; (e.g., Figure 1, from Column 1 to Column 2). More generally, strict inclusion in the first column leads to strict inclusion in the second one :

Proposition 6 *Let Y_1, Y_2 be two acyclicity properties such that $Y_1 \subset Y_2$, wa $\subseteq Y_1$ and $Y_2 \not\subseteq Y_1^D$. Then $Y_1^D \subset Y_2^D$.*

The next theorem states that PG^U is strictly more powerful than PG^D ; moreover, the "jump" from Y^D to Y^U is at least as large as from Y to Y^D.

Theorem 1 *Let Y be an acyclicity property. If $Y \subset Y^D$ then $Y^D \subset Y^U$. Furthermore, there is an injective mapping from the sets of rules satisfying Y^D but not Y, to the sets of rules satisfying Y^U but not Y^D.*

Proof: Assume $Y \subset Y^D$ and $\mathcal{R}$ satisfies Y^D but not Y. $\mathcal{R}$ can be rewritten into $\mathcal{R}'$ by applying the following steps. First, for each rule $R_i = B_i[\vec{X}, \vec{Y}] \rightarrow H_i[\vec{Y}, \vec{Z}] \in \mathcal{R}$, let $R_{i,1} = B_i[\vec{X}, \vec{Y}] \rightarrow p_i(\vec{X}, \vec{Y})$ where p_i is a fresh predicate ; and $R_{i,2} = p_i(\vec{X}, \vec{Y}) \rightarrow H_i[\vec{Y}, \vec{Z}]$. Then, for each rule $R_{i,1}$, let $R'_{i,1}$ be the rule $(B'_{i,1} \rightarrow H_{i,1})$ with $B'_{i,1} = B_{i,1} \cup \{p'_{j,i}(x_{j,i}) : \forall R_j \in \mathcal{R}\}$, where $p'_{j,i}$ are fresh predicates and $x_{j,i}$ fresh variables. Now, for each rule $R_{i,2}$, let $R'_{i,2}$ be the rule $(B_{i,2} \rightarrow H'_{i,2})$ with $H'_{i,2} = H_{i,2} \cup \{p'_{i,j}(z_{i,j}) : \forall R_j \in \mathcal{R}\}$, where $z_{i,j}$ are fresh existential variables. Let $\mathcal{R}' = \bigcup_{R_i \in \mathcal{R}} \{R'_{i,1}, R'_{i,2}\}$. This construction ensures that each $R'_{i,2}$ depends on $R'_{i,1}$, and each $R'_{i,1}$ depends on each $R'_{j,2}$, thus, there is a *transition* edge from each $R'_{i,1}$ to $R'_{i,2}$ and from each $R'_{j,2}$ to each $R'_{i,1}$. Hence, $PG^D(\mathcal{R}')$ contains exactly one cycle for each cycle in $PG^F(\mathcal{R})$. Furthermore, $PG^D(\mathcal{R}')$ contains at least one marked cycle w.r.t. Y, and then $\mathcal{R}'$ does not satisfy Y^D. Now, each cycle in $PG^U(\mathcal{R}')$ is also a cycle in $PG^D(\mathcal{R})$, and, since $PG^D(\mathcal{R})$ satisfies Y, $PG^U(\mathcal{R}')$ also does. Hence, $\mathcal{R}'$ does not belong to Y^D but to Y^U. □

We also check that strict inclusions in the second column in Figure 1 lead to strict inclusions in the third column.

Theorem 2 *Let Y_1 and Y_2 be two acyclicity properties. If $Y_1^D \subset Y_2^D$ then $Y_1^U \subset Y_2^U$.*

Proof: Let $\mathcal{R}$ be a set of rules such that $\mathcal{R}$ satisfies Y_2^D but does not satisfy Y_1^D. We rewrite $\mathcal{R}$ into $\mathcal{R}'$ by applying the following steps. For each pair of rules $R_i, R_j \in \mathcal{R}$ such that there is a dependency path from R_i to R_j,for each variable x in the frontier of R_j and each variable y in the head of R_i, if x and y occur both in a given predicate position, we add to the body of R_j a new atom $p_{i,j,x,y}(x)$ and to the head of R_i a new atom $p_{i,j,x,y}(y)$, where $p_{i,j,x,y}$ denotes a fresh predicate. This construction allows each term from the head of R_i to propagate to each term from the body of R_j, if they share some predicate position in $\mathcal{R}$. Thus, any cycle in $PG^D(\mathcal{R})$ is also in $PG^U(\mathcal{R}')$, without any change in the behavior w.r.t. the acyclicity properties. Hence $\mathcal{R}'$ satisfies Y_2^U but does not satisfy Y_1^U. □

The next result states that Y^U is a sufficient condition for chase termination :

Theorem 3 *Let Y be an acyclicity property ensuring the halting of some chase variant C. Then, the C-chase halts for any set of rules $\mathcal{R}$ that satisfies Y^U (hence Y^D).*

1. This particular case is to cover *aGRD*, in which each s.c.c. is an isolated node.

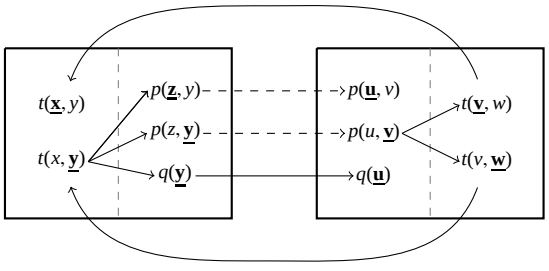

FIGURE 3 – $PG^D(\mathcal{R})$ and $PG^U(\mathcal{R})$ from Example 8. Dashed edges do not belong to $PG^U(\mathcal{R})$.

Example 9 *Consider again the set of rules $\mathcal{R}$ from Example 8. Figure 3 pictures the associated position graphs $PG^D(\mathcal{R})$ and $PG^U(\mathcal{R})$. $\mathcal{R}$ is not* aGRD, *nor* wa, *nor* wa^D *since $PG^D(\mathcal{R})$ contains a (marked) cycle that goes through the existential position $\langle t(v,w),2\rangle$. However, $\mathcal{R}$ is obviously* wa^U *since $PG^U(\mathcal{R})$ is acyclic. Hence, the skolem chase and stronger chase variants halt for $\mathcal{R}$ and any set of facts.*

Finally, we remind that classes from *wa* to *swa* can be recognized in PTIME, and checking *a-grd* is coNP-complete. Hence, as stated by the next result, the expressiveness gain is without increasing worst-case complexity.

Theorem 4 (Complexity) *Let Y be an acyclicity property, and $\mathcal{R}$ be a set of rules. If checking that $\mathcal{R}$ satisfies Y is in coNP, then checking that $\mathcal{R}$ satisfies Y^D or Y^U is coNP-complete.*

Further Refinements

Still without complexity increasing, we can further extend Y^U into Y^{U^+} by a finer analysis of marked cycles and unifiers. We define the notion of *incompatible* sequence of unifiers, which ensures that a given sequence of rule applications is impossible. Briefly, a marked cycle for which all sequences of unifiers are incompatible can be ignored. Beside the gain for positive rules, this refinement will allow one to take better advantage of negation.

We first point out that the notion of piece-unifier is not appropriate to our purpose. We have to relax it, as illustrated by the next example. We call *unifier*, of a rule body B_2 with a rule head H_1, a substitution μ of $vars(B_2') \cup vars(H_1')$, where $B_2' \subseteq B_2$ and $H_1' \subseteq H_1$, such that $\mu(B_2') = \mu(H_1')$ (thus, it satisfies Condition (1) of a piece-unifier).

Example 10 *Let $\mathcal{R} = \{R_1, R_2, R_3, R_4\}$ with :*
$R_1 : p(x_1,y_1) \rightarrow q(y_1,z_1)$
$R_2 : q(x_2,y_2) \rightarrow r(x_2,y_2)$
$R_3 : r(x_3,y_3) \wedge s(x_3,y_3) \rightarrow p(x_3,y_3)$
$R_4 : q(x_4,y_4) \rightarrow s(x_4,y_4)$
There is a dependency cycle (R_1,R_2,R_3,R_1) and a corresponding cycle in PG^U. We want to know if such a sequence of rule applications is possible. We build the following new rule, which is a composition of R_1 and R_2 (formally defined later) : $R_1 \diamond_\mu R_2 : p(x_1,y_1) \rightarrow q(y_1,z_1) \wedge r(y_1,z_1)$
There is no piece-unifier of R_3 with $R_1 \diamond_\mu R_2$, since y_3 would be a separating variable mapped to the existential variable z_1. This actually means that R_3 is not applicable right after *$R_1 \diamond_\mu R_2$. However, the atom needed to apply $s(x_3,y_3)$ can be brought by a sequence of rule applications (R_1,R_4). We thus relax the notion of piece-unifier to take into account arbitrary long sequences of rule applications.*

Definition 5 (Compatible unifier) *Let R_1 and R_2 be rules. A unifier μ of B_2 with H_1 is* compatible *if, for each position $\langle a,i\rangle$ in B_2', such that $\mu(term(\langle a,i\rangle))$ is an existential variable z in H_1', $PG^U(\mathcal{R})$ contains a path, from a position in which z occurs, to $\langle a,i\rangle$, that does not go through another existential position. Otherwise, μ is* incompatible.

Note that a piece-unifier is necessarily compatible.

Proposition 7 *Let R_1 and R_2 be rules, and let μ be a unifier of B_2 with H_1. If μ is incompatible, then no application of R_2 can use an atom in $\mu(H_1)$.*

We define the rule corresponding to the composition of R_1 and R_2 according to a compatible unifier, then use this notion to define a compatible sequence of unifiers.

Definition 6 (Unified rule, Compatible sequence of unifiers)

- *Let R_1 and R_2 be rules such that there is a compatible unifier μ of B_2 with H_1. The associated* unified rule *$R_\mu = R_1 \diamond_\mu R_2$ is defined by $H_\mu = \mu(H_1) \cup \mu(H_2)$, and $B_\mu = \mu(B_1) \cup (\mu(B_2) \setminus \mu(H_1))$.*
- *Let $(R_1,\ldots,R_{k+1})$ be a sequence of rules. A sequence $s = (R_1\ \mu_1\ R_2 \ldots\ \mu_k\ R_{k+1})$, where, for $1 \leq i \leq k$, μ_i is a unifier of B_{i+1} with H_i, is a* compatible sequence *of unifiers if :* (1) *μ_1 is a compatible unifier of B_2 with H_1, and* (2) *if $k > 0$, the sequence obtained from s by replacing $(R_1\ \mu_1\ R_2)$ with $R_1 \diamond_{\mu_1} R_2$ is a* compatible *sequence of unifiers.*

E.g., in Example 10, the sequence $(R_1\ \mu_1\ R_2\ \mu_2\ R_3\ \mu_3\ R_1)$, with the obvious μ_i, is compatible. We can now improve all previous acyclicity properties (see the fourth column in Figure 1).

Definition 7 (Compatible cycles) *Let Y be an acyclicity property, and PG^U be a position graph with unifiers. The* compatible cycles for $\langle a,i\rangle$ *in PG^U are all marked cycles C for $\langle a,i\rangle$ wrt Y, such that there is a compatible sequence of unifiers induced by C. Property Y^{U+} is satisfied if, for each existential position $\langle a,i\rangle$, there is no compatible cycle for $\langle a,i\rangle$ in PG^U.*

Results similar to Theorem 1 and Theorem 2 are obtained for Y^{U^+} w.r.t. Y^U, namely :

- For any acyclicity property Y, $Y^U \subset Y^{U+}$.
- For any acyclicity properties Y_1 and Y_2, if $Y_1^U \subset Y_2^U$, then $Y_1^{U+} \subset Y_2^{U+}$.

Moreover, Theorem 3 can be extended to Y^{U^+} : let Y be an acyclicity property ensuring the halting of some chase variant C ; then the C-chase halts for any set of rules $\mathcal{R}$ that satisfies Y^{U^+} (hence Y^U). Finally, the complexity result from Theorem 4 still holds for this improvement.

Handling Nonmonotonic Negation

We now add nonmonotonic negation, which we denote by **not**. A *nonmonotonic existential rule* (NME rule) R is of the form $(B^+, \mathbf{not}B_1^-, \ldots, \mathbf{not}B_k^- \rightarrow H)$, where B^+, B_i^- and H are atomsets, respectively called the *positive* body, the *negative* bodies and the head of R. Note that we generalize the usual notion of negative body by allowing to negate conjunctions of atoms. Moreover, the rule head may contain several atoms. However, we impose a safeness condition : $\forall 1 \leq i \leq k$, $vars(B_i^-) \subseteq vars(B^+)$. The formula assigned to R is $\Phi^{not}(R) = \forall\vec{x}\forall\vec{y}(\phi(B^+) \wedge \mathbf{not}\phi(B_1^-),\ldots,\mathbf{not}\phi(B_k^-) \rightarrow \exists\vec{z}\phi(H)$. We write $pos(R)$ the existential rule obtained from R by removing its negative bodies, and $pos(\mathcal{R})$ the set of all $pos(R)$ rules, for $R \in \mathcal{R}$.

About our Stable Model Semantics Answer Set Programming (Gelfond 2007) introduced stable model semantics for propositional logic, and was naturally extended to grounded programs (*i.e.,* sets of NME rules without variables). In this framework, the semantics can be provided through the Gelfond-Lifschitz reduct operator that allows to compute a saturation (*i.e.,* a chase) using only grounded NME rules. This semantics can be easily extended to rules with no existential variable in the head, or to skolemized NME rules, as done, for instance, in (Magka, Krötzsch, and Horrocks 2013). The choice of the chase/saturation mechanism is here irrelevant, since no such mechanism can produce any redundancy.

The problem comes when considering existential variables in the head of rules. Several semantics have been proposed in that case, for instance circumscription in (Ferraris, Lee, and Lifschitz 2011), or justified stable models in (You, Zhang, and Zhang 2013). We have chosen not to adopt circumscription since it translates NME rules to second-order expressions, and thus would not have allowed to build upon results obtained in the existential rule formalism. In the same way, we have not considered justified stable models, whose semantics does not correspond to stable models on grounded rules, as shown by the following example :

Example 11 *Let* $\Pi_1 = \{\emptyset \rightarrow p(a); p(a), \mathbf{not}\, q(a) \rightarrow t(a).\}$ *be a set of ground NME rules. Then* $\{p(a); q(a)\}$ *is a justified stable model, but not a stable model. Let* $\Pi_2 = \{\emptyset \rightarrow p(a); p(a), \mathbf{not}\, q(b) \rightarrow t(a)\}$ *. Then* $\{p(a); t(a)\}$ *is a stable model but not a justified stable model.*

Let us now recast the Gelfond-Lifschitz reduct-based semantics in terms of the skolem-chase. Essentially (we will be more precise in the next section), a stable model M is a possibly infinite atomset produced by a skolem-chase that respects some particular conditions :

- all rule applications are sound, *i.e.,* none of its negative bodies can be found in the stable model produced (the rule is not blocked) ;
- the derivation is complete, *i.e.,* any rule applicable and not blocked is applied in the derivation.

In the next subsection, we formally define the notion of a stable model, while replacing the skolem-chase with any C-chase. We thus obtain a family of semantics parameterized by the considered chase, and define different notions of C-stable models.

On the Chase and Stable Models We define a notion of stable model directly on nonmonotonic existential rules and provide a derivation algorithm inspired from the notion of computation in (Liu et al. 2010) and Answer Set Programming solvers that instantiate rules on the fly (Lefèvre and Nicolas 2009; Dao-Tran et al. 2012) instead of grounding rules before applying them. The difference with our framework is that they consider normal logic programs, which are a generalization of skolemized NME rules.

A natural question is then to understand if the choice of a chase mechanism has an impact, not only on the termination, but also on the semantics. Thus, we consider the chase as a parameter. Intuitively, a C-stable set A is produced by a C-chase that, according to (Gelfond 2007), must satisfy the NME rules (we say that it is *sound, i.e.,* that no negative body appearing in the chase is in A) and the *rationality principle* (the sound chase does not generate anything that cannot be believed, and it must be complete : any rule application not present in the chase would be unsound).

To define *C-stable sets*, we first need to introduce additional notions. A *NME $\mathcal{R}$-derivation from F* is a $pos(\mathcal{R})$-derivation from $\mathcal{R}$. This derivation $D = (F_0 = \sigma_0(F), \ldots, \sigma_k(F_k), \ldots)$ produces a possibly infinite atomset A. Let R be a NME rule such that $pos(R)$ was applied at some step i in D, *i.e.,* $F_{i+1} = \alpha(\sigma_i(F_i), pos(R), \pi_i)$. We say that this application is *blocked* if one of the $\pi_i(B_q^-)$ (for any negative body B_q^- in R) can be found in A. This can happen in two ways. Either $\pi_i(B_q^-)$ can already be found in $\sigma_i(F_i)$ or it appears later in the derivation. In both cases, there is a $\sigma_j(F_j)$ (with $j \geq i$) that contains the atomset $\pi_i(B_q^-)$, *as transformed by the sequence of simplifications from F_i to F_j, i.e.,* there exists F_j with $j \geq i$ s.t. the atomset $\sigma_{i\rightarrow j}(\pi_i(B_q^-)) = \sigma_j(\ldots(\sigma_{i+1}(\pi_i(B_q^-)))\ldots)$ is included in $\sigma_j(F_j)$. We say that a derivation D is *sound* when no rule application is blocked in A. A sound derivation is said to be *complete* when adding any other rule application to the derivation would either make it unsound, or would not change the produced atomset. The derivation is a C-chase when the σ_i used at each step is determined by the criterion C.

Definition 8 (C-stable sets) *Let F be a finite atomset, and $\mathcal{R}$ be a set of NME rules. We say that a (possibly infinite) atomset A is C-stable for $(F, \mathcal{R})$ if there is a complete sound nonmonotonic C-chase from F that produces A.*

Proposition 8 *If $\mathcal{R}$ is a set of existential rules, then there is a unique C-stable set, which is equivalent to the universal model $(F, \mathcal{R})^C$. If $\{F\} \cup \mathcal{R}$ is a set of skolemized NME rules (with F being seen as a rule with empty body), then its skolem-stable sets are in bijection with its stable models.*

Sketch of proof : First part of the claim stems from the fact that existential rules generate a unique branch that corresponds to a derivation. When that branch is complete, it corresponds to a chase. Second part of the claim comes from the fact that our definitions mimic the behavior of the sound and complete algorithm implemented in (Lefèvre and Nicolas 2009). □

***C*-chase Tree** The problem with the fixpoint Definition 8 is that it does not provide an effective algorithm : at each step of the derivation, we need to know the set produced by that derivation. The algorithm used in the solver ASPéRIX (Lefèvre and Nicolas 2009) is here generalized to a procedure that generates the (possibly infinite) *C-derivation tree* of $(F, \mathcal{R})$. All nodes of that tree are labeled by three fields. The field IN contains the atomset that was inferred in the current branch. The field OUT contains the set of forbidden atomsets, *i.e.,* that must not be inferred. Finally, the field MBT ("must be true") contains the atomset that has yet to be proven. A node is called *unsound* when a forbidden atomset has been inferred, or has to be proven, *i.e.,* when $\text{OUT} \cap (\text{IN} \cup \text{MBT}) \neq \emptyset$. At the initial step, the root of the C-derivation tree is a positive node labeled $(\sigma_0(F), \emptyset, \emptyset)$.

Then, let us chose a node N that is not unsound and has no child. Assume there is a rule $R = B^+, \mathbf{not} B_1^-, \ldots, \mathbf{not} B_k^- \rightarrow H$ in $\mathcal{R}$ such that there is a homomorphism π from B^+ to $\text{IN}(N)$. Then we will (possibly) add $k+1$ children under N, namely $N^+, N_1^-, \ldots, N_k^-$. These children are added if the rule application is not blocked, and produces new atoms. Intuitively, the positive child N^+ encodes the effective application of the rule, while the k negative children N_i^- encode the k different possibilities of blocking the rule (with each of the negative bodies). Let us consider the sequence of positive nodes from the root of the tree to N^+. It encodes a $pos(\mathcal{R})$-derivation from F. On that derivation, the C-chase generates a sequence $\sigma_0(F), \ldots, \sigma_p(F_p), S = \sigma(\alpha(\sigma_p(F_p), pos(R), \pi))$. S produces something new when $S \not\subseteq \sigma_p(F_p)$. We now have to fill the fields of the obtained children : let (IN, OUT, MBT) be the label of a node N. Then $label(N^+) = (S, \text{OUT} \cup \{\pi_i(B_1^-), \ldots, \pi_i(B_k^-)\}, \text{MBT})$ and $label(N_i^-) = (\text{IN}, \text{OUT}, \text{MBT} \cup \pi_i(B_i^-))$.

We say that a (possibly infinite) branch in the C-derivation tree is *unsound* when it contains an unsound node. A sound branch is said to be *complete* when its associated derivation is complete. Finally, a sound and complete branch is *stable* when for every node N in the branch such that $B^- \in \text{MBT}(N)$, there exists a descendant N' of N such that $B^- \in \text{IN}(N')$. We say that a branch is *unprovable* if there exists a node N in the branch and an atomset $B^- \in \text{MBT}(N)$ such that no complete branch containing N is stable. We call a *C-chase tree* any C-derivation tree for which all branches are either unsound, unprovable or complete.

Proposition 9 *An atomset A is a C-stable set for $(F, \mathcal{R})$ iff a C-chase tree of $(F, \mathcal{R})$ contains a stable branch whose associated derivation produces A.*

On the applicability of the chase variants In the positive case, all chase variants produce equivalent universal models (up to skolemization). Moreover, running a chase on equivalent knowledge bases produce equivalent results. Do these semantic properties still hold with nonmonotonic existential rules ? The answer is no in general.

The next example shows that the chase variants presented in this paper, core chase excepted, may produce non-equivalent results from equivalent knowledge bases.

Example 12 *Let $F = \{p(a,y), t(y)\}$ and $F' = \{p(a,y'), p(a,y), t(y)\}$ be two equivalent atomsets. Let $R : p(u,v), \mathbf{not}\, t(v) \rightarrow r(u)$. For any C-chase other than core chase, there is a single C-stable set for $(F, \{R\})$ which is F (or* sk(F)*) and a single C-stable set for $(F', \{R\})$ which is $F' \cup \{r(a)\}$ (or $sk(F') \cup \{r(a)\}$). These sets are not equivalent.*

Of course, if we consider that the initial knowledge base is already skolemized (including F seen as a rule), this trouble does not occur with the skolem-chase since there are no redundancies in facts and no redundancy can be created by a rule application. This problem does not arise with core chase either. Thus the only two candidates for processing NME rules are the core chase and the skolem chase (if we assume *a priori* skolemisation, which is already a semantic shift).

The choice between both mechanisms is important since, as shown by the next example, they may produce different results even when they both produce a *unique* C-stable set. It follows that skolemizing existential rules is not an innocuous transformation in presence of nonmontonic negation.

Example 13 *We consider $F = i(a)$, $R_1 = i(x) \rightarrow p(x,y)$, $R_2 = i(x) \rightarrow q(x,y)$, $R_3 = q(x,y) \rightarrow p(x,y), t(y)$ and $R_4 = p(u,v), \mathbf{not}\, t(v) \rightarrow r(u)$. The core chase produces at first step $p(a,y_0)$ and $q(a,y_1)$, then $p(a,y_1)$ and $t(y_1)$ and removes the redundant atom $p(a,y_0)$, hence R_4 is not applicable. The unique core-stable set is $\{i(a), q(a,y_1), p(a,y_1), t(y_1)\}$. With the skolem chase, the produced atoms are $p(a, f^{R_1}(a))$ and $q(a, f^{R_2}(a))$, then $p(a, f^{R_2}(a))$ and $t(f^{R_2}(a))$. R_4 is applied with $p(u,v)$ mapped to $p(a, f^{R_1}(a))$, which produces $r(a)$. These atoms yield a unique skolem-stable set. These stable sets are not equivalent.*

Termination of the Chase Tree

On the finiteness of C-chase trees We say that the *C-chase-tree halts* on $(F, \mathcal{R})$ when there exists a finite C-chase tree of $(F, \mathcal{R})$ (in that case, a breadth-first strategy for the rule applications will generate it). We can thus define *C-stable-finite* as the class of sets of nonmonotonic existential rules $\mathcal{R}$ for which the C-chase-tree halts on any $(F, \mathcal{R})$. Our first intuition was to assert "if $pos(\mathcal{R}) \in C$-finite, then $\mathcal{R} \in C$-stable-finite". However, this property is not true in general, as shown by the following example :

Example 14 *Let $\mathcal{R} = \{R_1, R_2\}$ where $R_1 = h(x) \rightarrow p(x,y), h(y)$ and $R_2 = p(x,y), \mathbf{not}\, h(x) \rightarrow p(x,x)$. See that $pos(\mathcal{R}) \in$ core-finite (as soon as R_1 is applied, R_2 is also applied and the loop $p(x,x)$ makes any other rule application redundant); however the only core-stable set of $(\{h(a)\}, \mathcal{R})$ is infinite (because all applications of R_2 are blocked).*

The following property shows that the desired property is true for *local* chases.

Proposition 10 *Let $\mathcal{R}$ be a set of NME rules and C be a local chase. If $pos(\mathcal{R}) \in C$-finite, then $\mathcal{R} \in C$-stable-finite.*

We have previously argued that the only two interesting chase variants w.r.t. the desired semantic properties are skolem and core. However, the core-finiteness of the positive part of a set of NME rules does not ensure the core-stable-finiteness of these rules. We should point out now that if $C \geq C'$, then C'-stable-finiteness implies C-stable-finiteness. We can thus ensure core-stable-finiteness when C-finiteness of the positive part of rules is ensured for a local C-chase.

Proposition 11 *Let $\mathcal{R}$ be a set of NME rules and C be a local chase. If $pos(\mathcal{R}) \in C$-finite, then $\mathcal{R} \in$ core-stable-finite.*

We can rely upon all acyclicity results in this paper to ensure that the core-chase tree halts.

Improving finiteness results with negative bodies We now explain how negation can be exploited to enhance preceding acyclicity notions. We first define the notion of *self-blocking rule*, which is a rule that will never be applied in any derivation. A rule $B^+, \mathbf{not}\, B_1^-, \ldots, \mathbf{not}\, B_k^-$ is self-blocking if there is a negative body B_i^- such that $B_i^- \subseteq (B^+ \cup H)$. Such a rule will never be applied in a sound way, so will never produce any atom. It follows that :

Proposition 12 *Let $\mathcal{R}'$ be the non-self-blocking rules of $\mathcal{R}$. If pos($\mathcal{R}'$) $\in$ C-finite and C is local, then $\mathcal{R} \in$ C-stable-finite.*

This idea can be further extended. We have seen for existential rules that if R' depends on R, then there is a unifier μ of *body*(R') with *head*(R), and we can build a rule $R'' = R \diamond_\mu R'$ that captures the sequence of applications encoded by the unifier. We extend Def. 6 to take into account negative bodies : if B^- is a negative body of R or R', then $\mu(B^-)$ is a negative body of R''. We also extend the notion of dependency in a natural way, and say that a unifier μ of *head*(R) with *body*(R') is self-blocking when $R \diamond_\mu R'$ is self-blocking, and R' *depends* on R when there exists a unifier of *head*(R) with *body*(R') that is not self-blocking. This extended notion of dependency exactly corresponds to the *positive reliance* in (Magka, Krötzsch, and Horrocks 2013).

Example 15 *Let $R = q(x), \mathbf{not}\, p(x) \rightarrow r(x,y)$ and $R' = r(x,y) \rightarrow p(x), q(y)$. Their associated positive rules are not core-finite. There is a single unifier μ of R' with R, and $R \diamond_\mu R' : q(x), \mathbf{not}\, p(x) \rightarrow r(x,y), p(x), q(y)$ is self-blocking. Then the skolem-chase-tree halts on $(F, \{R, R'\})$ for any F.*

Results obtained from positive rules can thus be generalized by considering this extended notion of dependency (for PG^U we only encode non self-blocking unifiers). Note that it does not change the complexity of the acyclicity tests.

We can further generalize this and check if a unifier sequence is self-blocking, thus extend the Y^{U+} classes to take into account negative bodies. Let us consider a compatible cycle C going through $\langle a, i \rangle$ that has not been proven safe. Let C_μ be the set of all compatible unifier sequences induced by C. We say that a sequence $\mu_1 \dots \mu_k \in C_\mu$ is self-blocking when the rule $R_1 \diamond_{\mu_1} R_2 \dots R_k \diamond_{\mu_k} R_{k+1}$ obtained by combining these unifiers is self-blocking. When all sequences in C_μ are self-blocking, we say that C is also self-blocking. This test comes again at no additional computational cost.

Example 16 *Let $R_1 = q(x_1), \mathbf{not} p(x_1) \rightarrow r(x_1, y_1)$, $R_2 = r(x_2, y_2) \rightarrow s(x_2, y_2)$, $R_3 = s(x_3, y_3) \rightarrow p(x_3), q(y_3)$. $PG^{U+}(\{R_1, R_2, R_3\})$ has a unique cycle, with a unique induced compatible unifier sequence. The rule $R_1 \diamond R_2 \diamond R_3 = q(x_1), \mathbf{not} p(x_1) \rightarrow r(x_1, y_1), s(x_1, y_1), p(x_1), q(y_1)$ is self-blocking, hence $R_1 \diamond R_2 \diamond R_3 \diamond R_1$ also is. Thus, there is no "dangerous" cycle.*

Proposition 13 *If, for each existential position $\langle a, i \rangle$, all compatible cycles for $\langle a, i \rangle$ in PG^U are self-blocking, then the stable computation based on the skolem chase halts.*

Conclusion

We have revisited chase termination with several results. First, a new tool that allows to unify and extend most existing acyclicity conditions, while keeping good computational properties. Second, a chase-like mechanism for nonmonotonic existential rules under stable model semantics, as well the extension of acyclicity conditions to take negation into account. This latter contribution extends the notion of negative reliance of (Magka, Krötzsch, and Horrocks 2013) ; and does not rely upon stratification (and thus does not enforce the existence of a single stable model).

This work will be pursued on the theoretical side by a complexity study of ENTAILMENT for the new acyclic classes and by a deeper study of logical foundations for NME rules, since it remains to relate our core-stable sets to an existing first-order semantics for general NME rules.

Acknowledgements

We thank the reviewers for their comments. This work is part of the ASPIQ and Pagoda projects and was partly funded by the french *Agence Nationale de la Recherche* (ANR) grants ANR-12-BS02-0003 and ANR-12-JS02-0007.

References

[Baader, Brandt, and Lutz 2005] Baader, F. ; Brandt, S. ; and Lutz, C. 2005. Pushing the el envelope. In *IJCAI'05*, 364–369.

[Baget and Mugnier 2002] Baget, J.-F., and Mugnier, M.-L. 2002. The Complexity of Rules and Constraints. *J. Artif. Intell. Res. (JAIR)* 16 :425–465.

[Baget et al. 2009] Baget, J.-F. ; Leclère, M. ; Mugnier, M.-L. ; and Salvat, E. 2009. Extending decidable cases for rules with existential variables. In *IJCAI'09*, 677–682.

[Baget et al. 2011] Baget, J.-F. ; Leclère, M. ; Mugnier, M.-L. ; and Salvat, E. 2011. On rules with existential variables : Walking the decidability line. *Artificial Intelligence* 175(9-10) :1620–1654.

[Baget 2004] Baget, J.-F. 2004. Improving the forward chaining algorithm for conceptual graphs rules. In *KR'04*, 407–414. AAAI Press.

[Beeri and Vardi 1981] Beeri, C., and Vardi, M. 1981. The implication problem for data dependencies. In *ICALP'81*, volume 115 of *LNCS*, 73–85.

[Calì, Gottlob, and Kifer 2008] Calì, A. ; Gottlob, G. ; and Kifer, M. 2008. Taming the infinite chase : Query answering under expressive relational constraints. In *KR'08*, 70–80.

[Calì, Gottlob, and Lukasiewicz 2009a] Calì, A. ; Gottlob, G. ; and Lukasiewicz, T. 2009a. A general datalog-based framework for tractable query answering over ontologies. In *PODS'09*, 77–86.

[Calı, Gottlob, and Lukasiewicz 2009b] Calı, A. ; Gottlob, G. ; and Lukasiewicz, T. 2009b. Tractable query answering over ontologies with datalog±. In *Proceedings of the DL Home 22nd International Workshop on Description Logics (DL 2009)*.

[Calimeri et al. 2008] Calimeri, F. ; Cozza, S. ; Ianni, G. ; and Leone, N. 2008. Computable functions in asp : Theory and implementation. In *Logic Programming*. Springer. 407–424.

[Calvanese et al. 2007] Calvanese, D. ; Giacomo, G. D. ; Lembo, D. ; Lenzerini, M. ; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics : The DL-Lite family. *J. Autom. Reasoning* 39(3) :385–429.

[Chandra, Lewis, and Makowsky 1981] Chandra, A. K. ; Lewis, H. R. ; and Makowsky, J. A. 1981. Embedded implicational dependencies and their inference problem. In *STOC'81*, 342–354. ACM.

[Cuenca Grau et al. 2012] Cuenca Grau, B. ; Horrocks, I. ; Krötzsch, M. ; Kupke, C. ; Magka, D. ; Motik, B. ; and Wang, Z. 2012. Acyclicity conditions and their application to query answering in description logics. In *KR*.

[Cuenca Grau et al. 2013] Cuenca Grau, B. ; Horrocks, I. ; Krötzsch, M. ; Kupke, C. ; Magka, D. ; Motik, B. ; and Wang, Z. 2013. Acyclicity notions for existential rules and their application to query answering in ontologies. *Journal of Artificial Intelligence Research* 47 :741–808.

[Dao-Tran et al. 2012] Dao-Tran, M. ; Eiter, T. ; Fink, M. ; Weidinger, G. ; and Weinzierl, A. 2012. Omiga : an open minded grounding on-the-fly answer set solver. In *Logics in Artificial Intelligence*. Springer. 480–483.

[Deutsch, Nash, and Remmel 2008] Deutsch, A. ; Nash, A. ; and Remmel, J. 2008. The chase revisited. In *PODS'08*, 149–158.

[Fagin et al. 2003] Fagin, R. ; Kolaitis, P. G. ; Miller, R. J. ; and Popa, L. 2003. Data exchange : Semantics and query answering. In *ICDT'03*, 207–224.

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

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

[Gelfond 2007] Gelfond, M. 2007. *In Handbook of Knowledge Representation*. Elsevier Science. chapter Answer Sets.

[Gottlob et al. 2012] Gottlob, G. ; Hernich, A. ; Kupke, C. ; and Lukasiewicz, T. 2012. Equality-friendly well-founded semantics and applications to description logics. In *Description Logics*.

[Krötzsch and Rudolph 2011] Krötzsch, M., and Rudolph, S. 2011. Extending decidable existential rules by joining acyclicity and guardedness. In *IJCAI'11*, 963–968.

[Lefèvre and Nicolas 2009] Lefèvre, C., and Nicolas, P. 2009. A first order forward chaining approach for answer set computing. In *Logic Programming and Nonmonotonic Reasoning*. Springer. 196–208.

[Lierler and Lifschitz 2009] Lierler, Y., and Lifschitz, V. 2009. One more decidable class of finitely ground programs. In *Logic Programming*. Springer. 489–493.

[Liu et al. 2010] Liu, L. ; Pontelli, E. ; Son, T. C. ; and Truszczyński, M. 2010. Logic programs with abstract constraint atoms : The role of computations. *Artificial Intelligence* 174(3–4) :295 – 315.

[Magka, Krötzsch, and Horrocks 2013] Magka, D. ; Krötzsch, M. ; and Horrocks, I. 2013. Computing stable models for nonmonotonic existential rules. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013). AAAI Press*.

[Marnette 2009] Marnette, B. 2009. Generalized schema-mappings : from termination to tractability. In *PODS*, 13–22.

[Mugnier 2011] Mugnier, M.-L. 2011. Ontological Query Answering with Existential Rules. In *RR'11*, 2–23.

[You, Zhang, and Zhang 2013] You, J.-H. ; Zhang, H. ; and Zhang, Y. 2013. Disjunctive logic programs with existential quantification in rule heads. *Theory and Practice of Logic Programming* 13 :563–578.