Extending Acyclicity Notions for Existential Rules

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

Extending Acyclicity Notions for Existential Rules (*long version*)

Jean-François Baget[1] and **Fabien Garreau**[2] and **Marie-Laure Mugnier**[3] and **Swan Rocher**[3]

This report contains a revised version (July 2014) of the paper that will appear in the proceedings of ECAI 2014 and an appendix with proofs that could not be included in the paper for space restriction reasons.

Abstract. Existential rules have been proposed for representing ontological knowledge, specifically in the context of Ontology-Based Query Answering. 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. First, we propose a new tool that allows to extend existing acyclicity conditions ensuring chase termination, while keeping good complexity properties. Second, we consider the extension to existential rules with nonmonotonic negation under stable model semantics and further extend acyclicity results obtained in the positive case.

1 INTRODUCTION

Ontology-Based Query Answering is a new paradigm in data management, which aims to exploit ontological knowledge when accessing data. *Existential rules* have been proposed for representing ontological knowledge, specifically in this context [8, 3]. These rules allow to assert the existence of unknown individuals, an essential feature in an open-domain perspective. They generalize lightweight description logics, such as DL-Lite and $\mathcal{EL}$ [10, 1] and overcome some of their limitations by allowing any predicate arity as well as cyclic structures.

In this paper, we focus on a breadth-first forward chaining algorithm, known as the *chase* in the database literature [24]. The chase was originally used in the context of very general database constraints, called tuple-generating dependencies, which have the same logical form as existential rules [6].

Given a knowledge base composed of data and existential rules, the chase triggers the rules and materializes performed inferences in the data. The "saturated" data can then be queried like a classical database. This allows to benefit from optimizations implemented in current data management systems. However, the chase is not ensured to terminate— which applies to any sound and complete mechanism, since entailment with existential rules is undecidable ([5, 11] on tuple-generating dependencies). Various acyclicity notions ensuring chase termination have been proposed in knowledge representation and database theory.

[1] INRIA, France
[2] University of Angers, France
[3] University of Montpellier, France

Paper contributions. We generalize known acyclicity conditions, first, for plain existential rules, second, for their extension to nonmonotonic negation with stable semantics.

1. Plain existential rules. Acyclicity conditions found in the literature can be classified into two main families: the first one constrains the way existential variables are propagated during the chase, e.g., [15, 25, 18], and the second one constrains dependencies between rules, i.e., the fact that a rule may lead to trigger another rule, e.g., [2, 14, 4, 12]. These conditions are based on different graphs, but all of them can be seen as forbidding "dangerous" cycles in the considered graph. We define a new family of graphs that allows to unify and strictly generalize these acyclicity notions, without increasing worst-case complexity.

2. Extension to nonmonotonic negation. Nonmonotonic negation is a useful feature in ontology modeling. Nonmontonic extensions to existential rules were recently considered in [8] with stratified negation, [17] with well-founded semantics and [23] with stable model semantics. The latter paper focuses on cases where a unique finite model exists; we consider the same rule framework, however without enforcing a unique model. We further extend acyclicity results obtained on positive rules by exploiting negative information as well.

The paper is organized according to these two issues.

2 PRELIMINARIES

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. An *atomset* is a finite set of atoms. If F is an atom or an atomset, we denote by *terms*(F) (resp. *vars*(F)) the set of terms (resp. variables) that occur in F. In the examples illustrating the paper, all the terms are variables (denoted by x, y, z, etc.), unless otherwise specified. Given atomsets A_1 and A_2, a *homomorphism* h from A_1 to A_2 is a substitution of *vars*(A_1) by *terms*(A_2) such that $h(A_1) \subseteq A_2$.

An *existential rule* (and simply a rule hereafter) is of the form $R = \forall \vec{x} \forall \vec{y}(B \rightarrow \exists \vec{z} H)$, where B and H are conjunctions of atoms, with *vars*$(B) = \vec{x} \cup \vec{y}$, and *vars*$(H) = \vec{x} \cup \vec{z}$. B and H are respectively called the *body* and the *head* of R. We also use the notations *body*(R) for B and *head*(R) for H. Variables $\vec{x}$, which appear in both B and H, are called *frontier variables*. Variables $\vec{z}$, which appear only in H, are called *existential variables*. Hereafter, we omit quantifiers in rules as there is no ambiguity. E.g., $p(x, y) \rightarrow p(y, z)$ stands for $\forall x \forall y(p(x, y) \rightarrow \exists z(p(y, z)))$.

An existential rule with an empty body is called a *fact*. A fact is thus an existentially closed conjunction of atoms. A *Boolean conjunctive query* (BCQ) has the same form. A *knowledge base* $\mathcal{K} = (F, \mathcal{R})$ is composed of a finite set of facts (which is seen as a single fact) F and a finite set of existential rules $\mathcal{R}$. The fundamen-

tal problem associated with query answering, called BCQ ENTAILMENT, is the following: given a knowledge base $(F, \mathcal{R})$ and a BCQ Q, is it true that $F, \mathcal{R} \models Q$, where $\models$ denotes the standard logical consequence? This problem is undecidable (which follows from the undecidability of the implication problem on tuple-generating dependencies [5, 11]).

In the following, we see conjunctions of atoms as atomsets. A rule $R : B \rightarrow H$ is *applicable* to an atomset F if there is a homomorphism π from B to F. The *application of R to F w.r.t. π* produces an atomset $\alpha(F, R, \pi) = F \cup \pi(\textit{safe}(H))$, where *safe*$(H)$ is obtained from H by replacing existential variables with fresh variables (see Example 1).

The *chase* procedure starts from the initial set of facts F and performs rule applications in a breadth-first manner. Several chase variants can be found in the literature, mainly *oblivious* (or naive), e.g., [7], *skolem* [25], *restricted* (or standard) [15], and *core* chase [14]. The oblivious chase performs all possible rule applications. The *skolem chase* relies on a skolemisation of the rules (notation *sk*): for each rule R, *sk*(R) is obtained from R by replacing each occurrence of an existential variable y with a functional term $f_y^R(\vec{x})$, where $\vec{x}$ is the set of frontier variables in R. Then, the oblivious chase is run on skolemized rules.

Example 1 (Oblivious / Skolem chase) *Let $R = p(x, y) \rightarrow p(x, z)$ and $F = \{p(a, b)\}$, where a and b are constants. The oblivious chase does not halt: it applies R according to $h_0 = \{(x, a), (y, b)\}$, hence adds $p(a, z_0)$; then, it applies R again according to $h_1 = \{(x, a), (y, z_0)\}$, and adds $p(a, z_1)$, and so on. The skolem chase considers the rule $p(x, y) \rightarrow p(x, f_z^R(x))$; it adds $p(a, f_y^R(a))$ then halts.*

Due to space restrictions, we do not detail on the other chase variants. Given a chase variant C, we call *C-finite* the class of set of rules $\mathcal{R}$, such that the C-chase halts on $(F, \mathcal{R})$ for any atomset F. It is well-known that oblivious-finite $\subset$ skolem-finite $\subset$ restricted-finite $\subset$ core-finite (see, e.g., [26]). When $\mathcal{R}$ belongs to a C-finite class, BCQ ENTAILMENT can be solved, for any F and Q, by running the C-chase on $(F, \mathcal{R})$, which produces a saturated set of facts F^*, then checking if $F^* \models Q$.

3 KNOWN ACYCLICITY NOTIONS

Acyclicity notions can be divided into two main families, each of them relying on a different graph. The first family relies on a graph encoding variable sharing between *positions* in predicates, while the second one relies on a graph encoding *dependencies* between rules, i.e., the fact that a rule may lead to trigger another rule (or itself).

3.1 Position-based approach

In the position-based approach, dangerous cycles 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 creating 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 acyclicity notion in this family is that of *weak-acyclicity (wa)* [15], 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) position i in predicate p). Then, for each rule $R : B \rightarrow H$ and each frontier variable x in B occurring in position (p, i), edges with origin (p, i) are built as follows: 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 (see Example 2).

Example 2 (Weak-acyclicity) *Let $R_1 = h(x) \rightarrow p(x, y)$ 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)*[18]) or to positions in atoms instead of predicates (*super-weak-acyclicity (swa)* [25]). Other related notions can be imported from logic programming, e.g., *finite domain (fd)* [9] and *argument-restricted (ar)* [22]. 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.

3.2 Rule dependency-based approach

In the second approach, the aim is to avoid cyclic triggering of rules [2, 14, 3, 4, 12]. We say that a rule R_2 *depends* on a rule R_1 if an application of R_1 may lead to a new application of R_2: there exists an atomset F such that R_1 is applicable to F with homomorphism π and R_2 is applicable to $F' = \alpha(F, R_1, \pi)$ with homomorphism π', which is *new* (π' is not a homomorphism to F) and *useful* (π' cannot be extended to a homomorphism from H_2 to F'). This abstract dependency relation can be computed with a unification operation known as piece-unifier [3]. 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 occur 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 some easily checked additional conditions (atom erasing [4] and usefulness [19, 12]). Following Example 3 illustrates the difference between piece-unification and classical unification.

Example 3 (Rule dependency) *Let R_1 and R_2 from Example 2. Although the atoms $p(u, v) \in B_2$ and $p(x, y) \in H_1$ are unifiable, there is no piece-unifier of B_2 with H_1. Indeed, the most general unifier $\mu = \{(u, x), (v, y)\}$ (or, equivalently, $\{(x, u), (y, v)\}$), 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'). It follows that R_2 does not depend on R_1.*

The *graph of rule dependencies* of a set of rules $\mathcal{R}$, denoted by GRD$(\mathcal{R})$, is a directed graph with set of nodes $\mathcal{R}$ and an edge (R_i, R_j) if R_j depends on R_i. E.g., with the rules in Example 3, the only edge is (R_2, R_1). When the GRD is acyclic (*aGRD*, [2]), any derivation sequence is finite.

3.3 Combining both approches

Both approaches are incomparable: there may be a dangerous cycle on positions but no cycle w.r.t. rule dependencies (Example 2 and 3),

and there may be a cycle w.r.t. rule dependencies whereas rules have no existential variables (e.g., $p(x, y) \rightarrow p(y, x)$). So far, attempts to combine both notions only succeded to combine them in a "modular way", by considering the strongly connected components (s.c.c.) of the GRD [2, 14]; briefly, if a chase variant stops on each subset of rules associated with a s.c.c., then it stops on the whole set of rules. 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 increasing complexity. The new acyclicity notions are those with a gray background in Figure 1.

Finally, let us mention *model-faithful acyclicity (mfa)* [12], which cannot be captured by our approach. Briefly, checking *mfa* involves running the skolem chase until termination or a cyclic functional term is found. The price to pay is high complexity: checking if a set of rules is model-faithful acyclic for any set of facts is 2EXPTIME-complete. Checking *model-summarizing acyclicity (msa)* [12], which approximates mfa, remains EXPTIME-complete. In contrast, checking position-based properties is in PTIME and checking aGRD is co-NP-complete.

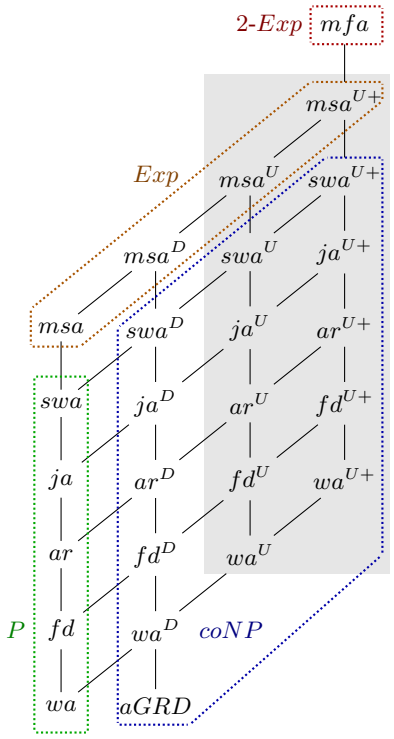

Figure 1. Relations between recognizable acyclicity properties. All inclusions are strict and complete (i.e., if there is no path between two properties then they are incomparable).

It remains to specify for which chase variants the above acyclicity notions ensure termination. Since *mfa* generalizes all properties in Figure 1, and sets of rules satisfying *mfa* are skolem-finite, all these properties ensure C-finiteness, for any chase variant C at least as strong as the skolem chase. We point out that the oblivious chase may not stop on *wa* rules. Actually, the only acyclicity notion in Figure 1 that ensures the termination of the oblivious chase is *aGRD*, since all other notions generalize *wa*.

4 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. Given an atom $a = p(t_1, \ldots, t_k)$, the i^{th} position in a is denoted by $[a, i]$, with *pred*$([a, i]) = p$ and *term*$([a, i]) = t_i$. If A is an atomset such that $a \in A$, we say that $[a, i]$ is in A. If *term*$([a, i])$ is an existential (resp. frontier) variable, $[a, i]$ 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 $R : B \rightarrow H$ is the directed graph $PG(R)$ defined as follows:*

- *there is a node for each $[a, i]$ in B or in H;*
- *for all frontier positions $[b, i] \in B$ and all $[h, j] \in H$, there is an edge from $[b, i]$ to $[h, j]$ if term$([b, i]) =$ term$([h, j])$ or if $[h, j]$ 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 $[a, i]$ 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*$([a, i])$. 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 $[a, i]$ is infinite, there must be a cycle going through $[a, i]$ 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 $[h, k]$ in a rule head H_i to each k^{th} position $[b, k]$ 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.

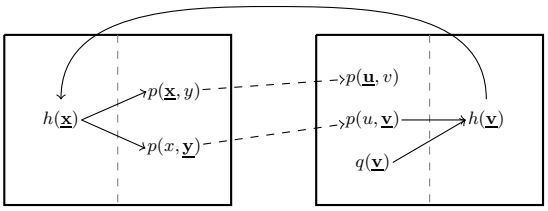

Figure 2. $PG^F(\mathcal{R})$ and $PG^D(\mathcal{R})$ from Example 4. Position $[a, i]$ is represented by underlining the i-th term in a. Dashed edges do not belong to $PG^D(\mathcal{R})$.

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:*

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

where fr is a new unary predicate that does not appear in $\mathcal{R}$, and the atoms $\textit{fr}(t)$ 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 $B_l^p \cup \{\textit{fr}(t) \mid t$ 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 $R_i^j = \bigcup_{P \in \mathcal{P}} R^P$.

The following inclusions follow from the definitions:

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 4 (PG^F and PG^D) *Let $\mathcal{R} = \{R_1, R_2\}$ from Example 2. 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 5 (PG^D and PG^U) *Let $\mathcal{R} = \{R_1, R_2\}$, with $R_1 = t(x, y) \rightarrow p(z, y), q(y)$ and $R_2 = p(u, v), q(u) \rightarrow t(v, w)$. 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 $PG^U(\mathcal{R})$.*

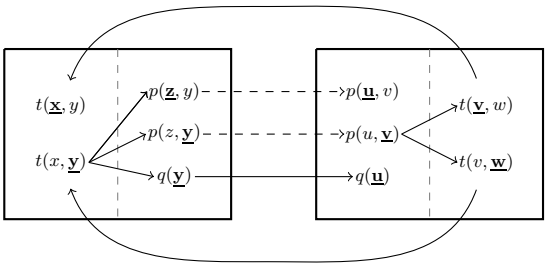

Figure 3. $PG^D(\mathcal{R})$ and $PG^U(\mathcal{R})$ from Example 5. Dashed edges do not belong to $PG^U(\mathcal{R})$.

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 $[a, i]$ in a position graph PG^X, a subset of its (direct or indirect) successors, called its *marking*. A *marked cycle* for $[a, i]$ (w.r.t. X and Y) is a cycle C in PG^X such that $[a, i] \in C$ and for all $[a', i'] \in C$, $[a', i']$ belongs to the marking of $[a, i]$. 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 [12]: 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.[4] 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 $aGRD \not\subseteq Y$ then $Y \subset Y^D$.*

Hence, any class of rules satisfying a property Y^D strictly includes both $aGRD$ 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.

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

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).*

Example 6 *Consider again the set of rules $\mathcal{R}$ from Example 5. 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 $[t(v, w), 2]$. 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 $aGRD$ is co-NP-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 co-NP, then checking that $\mathcal{R}$ satisfies Y^D or Y^U is co-NP-complete.*

5 FURTHER REFINEMENTS

In this section, we show how to further extend Y^U into Y^{U^+} by a finer analysis of marked cycles and unifiers. This extension can be performed without increasing complexity. 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 7 *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 arbitrarily 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 $[a, i]$ in B'_2, such that $\mu($term$([a, i]))$ is an existential variable z in H'_1, $PG^U(\mathcal{R})$ contains a path, from a position in which z occurs, to $[a, i]$, 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 7, 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 $[a, i]$ *in PG^U are all marked cycles C for $[a, i]$ w.r.t. Y, such that there is a compatible sequence of unifiers induced by C. Property Y^{U+} is satisfied if, for each existential position $[a, i]$, there is no compatible cycle for $[a, i]$ 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.

6 EXTENSION TO NONMONOTONIC NEGATION

We now add nonmonotonic negation, which we denote by **not**. A *nonmonotonic existential* (NME) rule R is of the form $\forall \vec{x} \forall \vec{y}(B^+ \wedge \mathbf{not} B_1^- \wedge \ldots \wedge \mathbf{not} B_k^- \rightarrow \exists \vec{z} H)$, where B^+, $B^- = \{B_1^- \ldots B_k^-\}$ and H are atomsets, respectively called the *positive* body, the *negative* body and the head of R; furthermore, *vars*$(B^-) \subseteq$ *vars*(B^+). R is *applicable* to F if there is a homomorphism h from B^+ to F such that $h(B^-) \cap F = \emptyset$. In this section, we rely on a skolemization of the knowledge base. Then, the application of R to F w.r.t. h produces $h(sk(H))$. R is *self-blocking* if $B^- \cap (B^+ \cup H) \neq \emptyset$, i.e., R is never applicable.

Since skolemized NME rules can be seen as normal logic programs, we can rely on the standard definition of stable models [16], which we omit here since it is not needed to understand the sequel. Indeed, our acyclicity criteria essentially ensure that there is a finite number of skolemized rule applications. Although the usual definition of stable models relies on grounding (i.e., instantiating) skolemized rules, stable models of $(F, \mathcal{R})$ can be computed by a skolem chase-like procedure, as performed by Answer Set Programming solvers that instantiate rules on the fly [21, 13].

We check that, when the skolem chase halts on the positive part of NME rules (i.e., obtained by ignoring the negative body), the stable computation based on the skolem chase halts. We can thus rely on preceding acyclicity conditions, which already generalize known acyclicity conditions applicable to skolemized NME rules (for instance *finite-domain* and *argument-restricted*, which were defined for normal logic programs). We can also extend them by exploiting negation.

First, we consider the natural extensions of a unified rule (Def. 6) and of rule dependency: to define $R_\mu = R_1 \diamond_\mu R_2$, we add that $B_\mu^- = \mu(B_1^-) \cup \mu(B_2^-)$; besides, R_2 depends on R_1 if there is a piece-unifier μ of H_2 with B_1 such that $R_1 \diamond_\mu R_2$ is not self-blocking; if $R_1 \diamond_\mu R_2$ is self-blocking, we say that μ is self-blocking. Note that this extended dependency is equivalent to the *positive reliance* from [23]. In this latter paper, positive reliance is used to define an acyclicity condition: a set of NME rules is said to be *R-acyclic* if no cycle of positive reliance involves a rule with an existential variable. Consider now PG^D with extended dependency: then, R-acyclicity is stronger than aGRD (since cycles are allowed on rules without existential variables) but weaker than wa^D (since all s.c.c. are necessarily wa).

By considering extended dependency, we can extend the results obtained with PG^D and PG^U (note that for PG^U we only encode non-self-blocking unifiers). We can further extend Y^{U+} classes by considering *self-blocking compatible sequences* of unifiers. Let C be a compatible cycle for $[a, i]$ in PG^U, and C_μ be the set of all compatible sequences of unifiers induced by C. A sequence $\mu_1 \ldots \mu_k \in \mathcal{C}_\mu$ is said to be self-blocking if the rule $R_1 \diamond_{\mu_1} R_2 \ldots R_k \diamond_{\mu_k} R_1$ is self-blocking. When all sequences in C_μ are self-blocking, C is said to be self-blocking.

Example 8 *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 8 *If, for each existential position $[a, i]$, all compatible cycles for $[a, i]$ in PG^U are self-blocking, then the stable computation based on the skolem chase halts.*

Finally, we point out that these improvements do not increase worst-case complexity of the acyclicity test.

7 CONCLUSION

We have proposed a tool that allows to unify and generalize most existing acyclicity conditions for existential rules, without increasing worst-case complexity. This tool can be further refined to deal with nonmonotonic (skolemized) existential rules, which, to the best of our knowledge, extends all known acyclicity conditions for this kind of rules.

Further work includes the implementation of the tool[5] and experiments on real-world ontologies, as well as the study of chase variants that would allow to process existential rules with stable negation without skolemization.

Acknowledgements. This work was partially supported by French *Agence Nationale de la Recherche* (ANR), under project grants ASPIQ (ANR-12-BS02-0003), Pagoda (ANR-12-JS02-0007) and Qualinca (ANR-12-CORD-0012). We thank Michael Thomazo for pointing out a flaw in the original definitions of $PG^D(\mathcal{R})$ and $PG^U(\mathcal{R})$.

REFERENCES

[1] F. Baader, S. Brandt, and C. Lutz, 'Pushing the EL envelope', in *IJCAI'05*, pp. 364–369, (2005).
[2] J.-F. Baget, 'Improving the forward chaining algorithm for conceptual graphs rules', in *KR'04*, pp. 407–414. AAAI Press, (2004).
[3] J.-F. Baget, M. Leclère, M.-L. Mugnier, and E. Salvat, 'Extending decidable cases for rules with existential variables', in *IJCAI'09*, pp. 677–682, (2009).
[4] J.-F. Baget, M. Leclère, M.-L. Mugnier, and E. Salvat, 'On rules with existential variables: Walking the decidability line', *Artificial Intelligence*, **175**(9-10), 1620–1654, (2011).

[5] It will be developed as an extension of KIABORA, an analyzer of existential rule bases [20].

[5] C. Beeri and M. Vardi, 'The implication problem for data dependencies', in *ICALP'81*, volume 115 of *LNCS*, pp. 73–85, (1981).
[6] C. Beeri and M.Y. Vardi, 'A proof procedure for data dependencies', *Journal of the ACM*, **31**(4), 718–741, (1984).
[7] A. Calì, G. Gottlob, and M. Kifer, 'Taming the infinite chase: Query answering under expressive relational constraints', in *KR'08*, pp. 70–80, (2008).
[8] A. Calì, G. Gottlob, and T. Lukasiewicz, 'A general datalog-based framework for tractable query answering over ontologies', in *PODS'09*, pp. 77–86, (2009).
[9] F. Calimeri, S. Cozza, G. Ianni, and N. Leone, 'Computable functions in asp: Theory and implementation', in *Logic Programming*, 407–424, (2008).
[10] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati, 'Tractable reasoning and efficient query answering in description logics: The DL-Lite family', *J. Autom. Reasoning*, **39**(3), 385–429, (2007).
[11] A. K. Chandra, H. R. Lewis, and J. A. Makowsky, 'Embedded implicational dependencies and their inference problem', in *STOC'81*, pp. 342–354. ACM, (1981).
[12] B. Cuenca Grau, I. Horrocks, M. Krötzsch, C. Kupke, D. Magka, B. Motik, and Z. Wang, 'Acyclicity notions for existential rules and their application to query answering in ontologies', *J. Art. Intell. Res.*, **47**, 741–808, (2013).
[13] M. Dao-Tran, T. Eiter, M. Fink, G. Weidinger, and A. Weinzierl, 'Omiga: an open minded grounding on-the-fly answer set solver', in *Logics in Artificial Intelligence*, 480–483, (2012).
[14] A. Deutsch, A. Nash, and J.B. Remmel, 'The chase revisited', in *PODS'08*, pp. 149–158, (2008).
[15] 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).
[16] M. Gelfond and V. Lifschitz, 'The stable model semantics for logic programming', in *ICLP/SLP*, pp. 1070–1080, (1988).
[17] A. Hernich, C. Kupke, T. Lukasiewicz, and G. Gottlob, 'Well-founded semantics for extended datalog and ontological reasoning', in *PODS*, pp. 225–236, (2013).
[18] M. Krötzsch and S. Rudolph, 'Extending decidable existential rules by joining acyclicity and guardedness', in *IJCAI'11*, pp. 963–968, (2011).
[19] B. Lamare, 'Optimisation de la notion de dépendance', Internship report, ENS Cachan and LIRMM/ INRIA, (Sept. 2012).
[20] M. Leclère, M.-L. Mugnier, and S. Rocher, 'Kiabora: An analyzer of existential rule bases', in *RR*, pp. 241–246, (2013).
[21] C. Lefèvre and P. Nicolas, 'A first order forward chaining approach for answer set computing', in *LPNMR*, 196–208, (2009).
[22] Y. Lierler and V. Lifschitz, 'One more decidable class of finitely ground programs', in *Logic Programming*, 489–493, (2009).
[23] D. Magka, M. Krötzsch, and I. Horrocks, 'Computing stable models for nonmonotonic existential rules', in *IJCAI*, (2013).
[24] D. Maier, A. O. Mendelzon, and Y. Sagiv, 'Testing implications of data dependencies', *ACM Trans. Database Syst.*, **4**(4), 455–469, (1979).
[25] B. Marnette, 'Generalized schema-mappings: from termination to tractability', in *PODS*, pp. 13–22, (2009).
[26] A. Onet, 'The chase procedure and its applications in data exchange', in *Data Exchange, Information, and Streams*, pp. 1–37, (2013).

Appendix

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 the wa-marking (resp. fd-, ar-, ja-, swa-marking).*

To prove Proposition 2 we rely on some intermediary results. The next proposition is immediate.

Proposition 9 *For each edge (p_i, q_j) in the predicate position graph of a set of rules $\mathcal{R}$, there is the following non-empty set of edges in $PG^F(\mathcal{R})$: $E_{p_i,q_j} = \{([a,i],[a',j]) \mid$ pred$([a,i]) = p$ and pred$([a',j]) = q\}$.*

Furthermore, these sets of edges form a partition of all edges in $PG^F(\mathcal{R})$.

We now define marking functions, whose associated acyclicity property corresponds to wa, fd, ar, ja or swa when it is applied on $PG^F(\mathcal{R})$. The following three conditions, defined for a marking $M([a,i])$, make it easy to compare known acyclicity properties.

- **(P1)** $\Gamma^+([a,i]) \subseteq M([a,i])$, [6]
- **(P2)** for all $[a',i'] \in M([a,i])$ such that $[a',i']$ occurs in some rule head: $\Gamma^+([a',i']) \subseteq M([a,i])$,
- **(P3)** for all variable v in a rule body, such that for all position $[a',i']$ with *term*$([a',i']) = v$, there is $[a'',i'] \in M([a,i])$ with *pred*$([a',i']) =$ *pred*$([a'',i'])$ and *term*$([a'',i']) = v$: $\Gamma^+(v) \subseteq M([a,i])$, where $\Gamma^+(v)$ is the union of all $\Gamma^+(p)$, where p is an atom position in which v occurs.

(P1) ensures that the marking of a given node includes its successors ; **(P2)** ensures that the marking includes the successors of all marked nodes from a rule head ; and **(P3)** ensures that for each frontier variable of a rule such that all predicate positions where it occurs are marked, the marking includes its successors.

Definition 8 (Weak-acyclicity marking) *A marking M is a wa-marking wrt X if for any $[a,i] \in PG^X$, $M([a,i])$ is the minimal set such that:*

- **(P1)** *holds,*
- *for all $[a',i'] \in M([a,i]), \Gamma^+([a',i']) \subseteq M([a,i])$*

Observation: The latter condition implies **(P2)** and **(P3)**.

Proposition 10 *A set of rules $\mathcal{R}$ is wa iff $PG^F(\mathcal{R})$ satisfies the acyclicity property associated with the wa-marking.*

Proof: If $\mathcal{R}$ is not wa, then there is some cycle in the graph of predicate positions going through a *special edge*. Let p_i be the predicate position where this edge ends, and z be the existential variable which occurs in p_i. Let M be the wa-marking of any existential position $[a,i]$ with *pred*$([a,i]) = p$ and *term*$([a,i]) = z$.

(P1) ensures that the successors of $[a,i]$ are marked; then, the propagation function will perform a classic breadth-first traversal of the graph. By Proposition 9, to each cycle in the graph of predicate positions of $\mathcal{R}$ corresponds a set of cycles in $PG^F(\mathcal{R})$. Since p_i belongs to a cycle, $[a,i]$ will obviously be marked by the propagation function. Hence, $PG^F(\mathcal{R})$ does not satisy the associated acyclicity property of the wa-marking.

Conversely, if $\mathcal{R}$ is wa, there is no cycle going through a *special edge* in the graph of predicate positions of $\mathcal{R}$. By Proposition 9, to

[6] For any node v, $\Gamma^+(v)$ denotes the set of (direct) successors of v.

each cycle in $PG^F(\mathcal{R})$ corresponds a cycle in the graph of predicate positions of $\mathcal{R}$, hence no cycle in $PG^F(\mathcal{R})$ goes through an existential position. □

We do not recall here the original definitions of fd, ar, ja, swa. The reader is referred to the papers cited in Section 4 or to [12], where these notions are reformulated with a common vocabulary.

Definition 9 (Finite domain marking) *A marking M is a fd-marking wrt X if for any $[a,i] \in PG^X$, $M([a,i])$ is the minimal set such that:*

- **(P1)** *and* **(P3)** *hold,*
- *for all $[a',i'] \in M([a,i]), \Gamma^+([a',i']) \setminus \{[a,i]\} \subseteq M([a,i])$.*

Observation: The latter condition implies **(P2)**.

Proposition 11 *A set of rules $\mathcal{R}$ is fd iff $PG^F(\mathcal{R})$ satisfies the acyclicity property associated with the fd-marking.*

Proof: Let $\mathcal{R}$ be a set of rules that is fd, then for each existential position p_i there exists a position p_j for each variable of the frontier in the graph of predicate positions such that p_j does not belong to a cycle. Given $PG^F(\mathcal{R})$ we can see that Condition **(P3)** ensures that $\mathcal{R}$ is fd. □

Definition 10 (Argument restricted marking) *A marking M is an ar-marking wrt X if for any $[a,i] \in PG^X$, $M([a,i])$ is the minimal set such that:*

- **(P1)**, **(P2)** *and* **(P3)** *hold,*
- *for each existential position $[a',i']$, $\Gamma^+([a',i']) \subseteq M([a,i])$,*

Observation: If M is an ar-marking, then for all existential positions $[a,i]$,$[a',i'] \in PG^X$, $M([a,i]) = M([a',i'])$.

Proposition 12 *A set of rules $\mathcal{R}$ is ar iff $PG^F(\mathcal{R})$ satisfies the acyclicity property associated with the ar-marking.*

Proof: Let $\mathcal{R}$ be a set of rules that is ar, then there exists a ranking on terms (i.e., arguments) such that for each rule the rank of an existential variable needs to be stricly higher than the rank of each frontier variable in the body and the rank of a frontier variable in the head has to be higher or equal to the rank of this frontier variable in the body. The marking process is equivalent to the ranking, in fact each time a node is marked, the rank of a term is incremented. If we have a cyclic $ar\text{-}marking$, it means that there exists at least one term rank that does not satisfy the property of argument-restricted. We can see the marking process as a method to compute an argument ranking. □

Definition 11 (Joint acyclicity marking) *A marking M is a ja-marking wrt X if for any $[a,i] \in PG^X$, $M([a,i])$ is the minimal set such that* **(P1)**, **(P2)** *and* **(P3)** *hold.*

Proposition 13 *A set of rules $\mathcal{R}$ is ja iff $PG^F(\mathcal{R})$ satisfies the acyclicity property associated with the ja-marking.*

Proof: The definition of the ja propagation function is the same as in [18]. Indeed the "Move" set of a position is defined in the same way as the marking. Furthermore, by Proposition 9, for any predicate position p_i in the graph of joint-acyclicity, there is a cycle going through p_i iff for any existential atom position $[a,i]$ such that *pred*$([a,i]) = p$, we have $[a,i] \in M([a,i])$.

□

Definition 12 (Super-weak-acyclicity marking) *A marking M is a swa-marking wrt X if for any $[a,i] \in PG^X$, $M([a,i])$ is the minimal set such that :*

- **(P1)** *and* **(P3)** *hold,*
- *for all $[a',i'] \in M([a,i])$ occuring in a rule head, $\{[a'',i'] \in \Gamma^+([a',i']) : a'$ and a'' unify$\} \subseteq M([a,i])$.*

Proposition 14 *A set of rules $\mathcal{R}$ is swa iff $PG^F(\mathcal{R})$ satisfies the acyclicity property associated with the swa-marking.*

Proof: In the original paper of [25], the definition of swa was slightly different from this marking, but it has been shown in [12], that swa can be equivalently expressed by a "Move" set similar to ja. As for the ja-marking, the definition of the swa-marking corresponds to the definition of its "Move" set. □

Proof of Proposition 2: Follows from Propositions 10, 11, 12, 13, and 14.

Proposition 6 *Let Y_1 and 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 $Y_1^D \subset Y_2^D$.*
Proof: Let $\mathcal{R}$ be a set of rules such that $\mathcal{R}$ satisfies Y_2 and neither Y_1 nor $aGRD$. $\mathcal{R}$ can be rewritten into $\mathcal{R}'$ by replacing each rule $R_i = (B_i, H_i) \in \mathcal{R}$ with a new rule $R'_i = (B_i \cup \{p(x)\}, H_i \cup \{p(x)\})$ where p is a fresh predicate and x a fresh variable. Each rule can now be unified with each rule, but the only created cycles are those which contain only atoms $p(x)$, hence none of those cycles go through existential positions. Since $wa \subseteq Y_1$ (and so $wa \subseteq Y_2$), the added cycles do not change the behavior of $\mathcal{R}$ w.r.t. Y_1 and Y_2. Hence, $\mathcal{R}'$ is a set of rules satisfying Y_2 and not Y_1, and since $GRD(\mathcal{R}')$ is a complete graph, $PG^D(\mathcal{R}') = PG^F(\mathcal{R}')$. We can conclude that $\mathcal{R}'$ satisfies Y_2^D but not Y_1^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: **(included in the paper)** 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 $\mathcal{R}'$ is not 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. □

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: **(included in the paper)** 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 R_j depends on R_i, 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 will allow each term from the head of R_i to propagate to each term from the body of R_j, if they shared some predicate position in $\mathcal{R}$. Thus, any cycle in $PG^D(\mathcal{R})$ is also in $PG^U(\mathcal{R}')$, without modifying behavior w.r.t. the acyclicity properties. Hence, $\mathcal{R}'$ satisfies Y_2^U but does not satisfy Y_1^U. □

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).*

We will first formalize the notion of a *correct* position graph (this notion being not precisely defined in the core paper). Then, we will prove that PG^F, PG^D and PG^U are correct, which will allow to prove the theorem.

Preliminary definitions Let F be a fact and $\mathcal{R}$ be a set of rules. An *$\mathcal{R}$-derivation* (sequence) (from F to F_k) is a finite sequence $(F_0 = F), (R_1, \pi_1, F_1), \ldots, (R_k, \pi_k, F_k)$ such that for all $0 < i \leq k$, $R_i \in \mathcal{R}$ and π_i is a homomorphism from *body*(R_i) to F_{i-1} such that $F_i = \alpha(F_{i-1}, R_i, \pi_i)$. When only the successive facts are needed, we note $(F_0 = F), F_1, \ldots, F_k$.

Let $S = (F_0 = F), \ldots, F_n$ be a breadth-first $\mathcal{R}$-derivation from F. [7] Let h be an atom in the head of R_i and b be an atom in the body of R_j. We say that (h, π_i) is a *support* of (b, π_j) (in S) if $\pi_i^{safe}(h) = \pi_j(b)$. We also say that an atom $f \in F_0$ is a *support* of (b, π_j) if $\pi_j(b) = f$. In that case, we note (f, init) is a support of (b, π_j). Among all possible supports for (b, π_j), its *first supports* are the (h, π_i) such that i is minimal or $\pi_i = \mathit{init}$. Note that (b, π_j) can have two distinct first supports (h, π_i) and (h', π_i) when the body of R_i contains two distinct atoms h and h' such that $\pi_i^{safe}(h) = \pi_i^{safe}(h')$. By extension, we say that (R_i, π_i) is a *support* of (R_j, p_j) in S when there exist an atom h in the head of R_i and an atom b in the body of R_j such that (h, π_i) is a first support of (b, π_j). In the same way, F_0 is a support of (R_j, π_j) when there exists b in the body of R_j such that $\pi_j(b) \in F_0$. Among all possible supports for (R_j, π_j), its *last support* is the support (R_i, π_i) such that i is maximal.

The *support graph* of S has $n + 1$ nodes, F_0 and the (R_i, π_i). We add an edge from $I = (R_i, \pi_i)$ to $J = (R_j, \pi_j)$ when I is a support of J. Such an edge is called a *last support edge* (LS edge) when I is a last support of J. An edge that is not LS is called *non transitive* (NT) if it is not a transitivity edge. A path in which all edges are either LS or NT is called a *triggering path*.

Definition 13 (Triggering derivation sequence) *A $h \to b$* triggering derivation sequence *is a breadth-first derivation sequence $F = F_0, \ldots, F_n$ from F where (h, π_1) is a first support of (b, π_n).*

Definition 14 (Correct position graph) *Let $\mathcal{R}$ be a set of rules. A position graph of $\mathcal{R}$ is said to be* correct *if, whenever there exists a $h \to b$* triggering derivation sequence, *the position graph contains a transition from $[h, i]$ to $[b, i]$ for all $1 \leq i \leq k$, where k is the arity of the predicate of h and b.*

[7] A derivation is breadth-first if, given any fact F_i in the sequence, all rule applications corresponding to homomorphisms to F_i are performed before rule applications on subsequently derived facts that do not correspond to homomorphisms to F_i.

Proposition 15 *PG^F is correct.*

Proof: Follows from the above definitions. □

Lemma 1 *If S is a $h \to b$ triggering derivation sequence, then there is a triggering path from (R_1, π_1) to (R_n, π_n) in the support graph of S.*

Proof: There is an edge from (R_1, π_1) to (R_n, π_n) in the support graph of S. By removing transitivity edges, it remains a path from (R_1, π_1) to (R_n, π_n) for which all edges are either LS or NT. □

Lemma 2 *If there is an edge from (R_i, π_i) to (R_j, π_j) that is either LS or NT in the support graph of S, then R_j depends on R_i.*

Proof: Assume there is a LS edge from (R_i, π_i) to (R_j, π_j) in the support graph. Then the application of R_i according to π_i on F_{i-1} produces F_i on which all atoms required to map B_j are present (or it would not have been a last support). Since it is a support, there is also an atom required to map B_j that appeared in F_{i-1}. It follows that R_j depends upon R_i.

Suppose now that the edge is NT. Consider F_k such that there is a LS edge from (R_k, π_k) to (R_j, π_j). See that there is no path in the support graph from (R_i, π_i) to (R_k, π_k) (otherwise there a would be a path from (R_i, π_i) to (R_j, π_j) and the edge would be a transitive edge). In the same way, there is no q such that there is a path from (R_i, π_i) to (R_j, π_j) that goes through (R_q, π_q) (or the edge from (R_i, π_i) to (R_j, π_j) would be transitive). Thus, we can consider the atomset $F_{k \setminus i}$ that would have been created by the following derivation sequence:

- first apply from F_0 all rule applications of the initial sequence until (R_{i-1}, π_{i-1});
- then apply all possible rule applications of this sequence, from $i + 1$ until k.

We can apply (R_i, π_i) on the atomset $F_{k \setminus i}$ thus obtained (since it contains the atoms of F_{i-1}). Let us now consider the atomset G obtained after this rule application. We must now check that (R_j, π_j) can be applied on G: this stems from the fact that there is no support path from (R_i, π_i) to (R_j, π_j). This last rule application relies upon an atom that is introduced by the application of (R_i, π_i), thus R_j depends on R_i. □

Proposition 16 *PG^D is correct.*

Proof: If there is a $h \to b$ triggering derivation sequence, then (by Lemma 1) we can exhibit a triggering path that corresponds to a path in the GRD (Lemma 2). □

Proposition 17 *PG^U is correct.*

Proof: Consider a $h \to b$ triggering derivation sequence $F = F_0, \ldots, F_n$. We note $H^P = \pi_n(B_n) \cap \pi_1^{safe}(H_1)$ the atoms of F_n that are introduced by the rule application (R_1, π_1) and are used for the rule application (R_n, π_n). Note that this atomset H^P is not empty, since it contains at least the atom produced from h. Now, consider the set of terms $T^P = \mathit{terms}(H^P) \cap \mathit{terms}(\pi_n(B_n) \setminus H^P)$ that separate the atoms of H^P from the other atoms of $\pi_n(B_n)$.

Now, we consider the rule $R^P = B_1 \cup \{fr(t) \mid t \text{ is a variable of } R_1 \text{ and } \pi_1^{safe}(v) \in T^P\} \to H_1$. Consider the atomset $F^P = F_{n-1} \setminus H^P \cup \{fr(t) \mid \text{t is a term of } T^P\}$.

Consider the mapping π_1^P from the variables of the body of R_P to those of F_P, defined as follows: if v is a variable of B_1, then $\pi_1^P(v) = \pi_1(v)$, otherwise v is a variable in an "fr" atom and $\pi_1^P(v) = \pi_n(v)$. This mapping is a homomorphism, thus we can consider the atomset $F^{P'} = \alpha(F^P, R^P, \pi_1^P)$. This application produces a new application of R_n that maps b to the atom produced from h. Indeed, consider the mapping π_n^P from the variables of B_n to those of $F^{P'}$ defined as follows: if t is a variable of B_n such that $\pi_n(t) \in \mathit{terms}(H^P) \setminus T^P$, then $\pi_n^P(t) = \pi_1^{P^{safe}}(t')$, where t' is the variable of H_1 that produced $\pi_n(t)$, otherwise $\pi_n^P(t) = \pi_n(t)$. This mapping is a homomorphism. This homomorphism is new since it maps b to $\pi_n^{P^{safe}}(h)$. Thus, there is a piece-unifier of B_n with the head of R^P that unifies h and b.

It remains now to prove that for each atom $fr(t)$ in the body of R^P there exists a triggering path $P_i = (R'_1, \pi'_1) = (R_1, \pi_1)$ to $(R'_k, \pi'_k) = (R_n, \pi_n)$ in the support graph such that $fr(t)$ appears in the agglomerated rule R_i^A along $R_1, \ldots, R_{n-1}$.

Let t be a variable occuring in some fr atom in R^P. Suppose that $fr(t)$ does not appear in any agglomerated rule corresponding to a triggering path P_i between (R_1, π_1) and (R_n, π_n). Since $\pi_1(t)$ is an existential variable generated by the application of R_1, and there is no unifier on the GRD paths that correspond to these triggering paths that unify t, $\pi_1(t)$ may only occur in atoms that are not used (even transitively) by (R_n, π_n), i.e. $\pi_1(t) \notin T^P$. Therefore, t does not appear in a fr atom R^P, which leads to a contradiction.

Since R^P and $R^A = \bigcup R_i^A$ have the same head and the frontier of R^P is a subset of the frontier of R^A any unifier with R^P is also a unifier with R^A. Thus, there is a unifier of R_n with R^A that unifies h and b, and there are the corresponding correct transition edges in PG^U.

□

Proof: (of Theorem 3)

Let us say that a transition edge from $[a, i]$ in R_1 to $[a', i,]$ in R_2 is *useful* if there is a fact F, and a homomorphism π_1 from B_1 to F, such that there is a homomorphism π_2 from B_2 to some F' obtained from a derivation of $(\alpha(F, R_1, \pi_1), \mathcal{R})$ and $\pi_1^{safe}(a) = \pi_2(a')$. Furthermore, we say that the application of R_2 *uses* edge $([a, i],[a', i])$.

One can see that a useful edge exactly corresponds to a $h \to b$ triggering sequence where $[a, i]$ occurs in h and $[a, i']$ occurs in b. It follows from the correctness of PG^U and PG^D that no useful edge of PG^F is removed.

Now, let Y be an acyclicity proposition ensuring the halting of some chase variant C. Assume there is a set of rules $\mathcal{R}$ that satisfies Y^U but not Y^D and there is F such that the C-chase does not halt on $(F, \mathcal{R})$. Then, there is a rule application in this (infinite) derivation that uses a transition edge $([a, i],[a', i])$ belonging to Y^D but not Y^U. This is impossible because such an edge is useful. The same arguments hold for Y^D w.r.t. Y^F. □

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

We first state a preliminary proposition.

Proposition 18 *If there is a $h \to b$ triggering derivation sequence (with $h \in$ head(R) and $b \in$ body(R')), then there exist a non-empty set of paths $\mathcal{P} = \{P_1, \ldots, P_k\}$ from R in GRD($\mathcal{R}$) such that $\sum_{1 \leq i \leq k} |P_i| \leq |\mathcal{R}| \times |\mathit{terms}(\mathit{head}(R))|$ and a piece-unifier of B' with the head of an agglomerated rule along $\mathcal{P}$ that unifies h and b.*

Proof: The piece-unifier is entirely determined by the terms that are forced into the frontier by an "fr" atom. Hence, we need to consider at most one path for each term in H. Moreover, each (directed) cycle in the GRD (that is of length at most $|\mathcal{R}|$) needs to be traversed at most $|\mathit{terms}(H)|$ times, since going through such a cycle without creating a new frontier variable cannot create any new unifier. Hence, we need to consider only paths of polynomial length. □

Proof: (of Theorem 4) One can guess a cycle in $PG^D(\mathcal{R})$ (or $PG^U(\mathcal{R})$) such that the property Y is not satisfied by this cycle. From the previous property, each edge of the cycle has a polynomial certificate, and checking if a given substitution is a piece-unifier can also be done in polynomial time. Since Y is in co-NP, we have a polynomial certificate that this cycle does not satisfy Y. Membership to co-NP follows.

The completeness part is proved by a simple reduction from the co-problem of rule dependency checking (which is thus a co-NP-complete problem).

Let R_1 and R_2 be two rules. We first define two fresh predicates p and s of arity $|vars(B_1)|$, and two fresh predicates q and r of arity $|vars(H_2)|$.

We build $R_0 = p(\vec{x}) \to s(\vec{x})$, where $\vec{x}$ is a list of all variables in B_1, and $R_3 = r(\vec{x}) \to p(\vec{z}), q(\vec{x})$, where $\vec{z} = (z, z, \ldots, z)$, where z is a variable which does not appear in H_2, and $\vec{x}$ is a list of all variables in H_2. We rewrite R_1 into $R'_1 = (B_1, s(\vec{x}) \to H_1)$, where $\vec{x}$ is a list of all variables in B_1, and R_2 into $R'_2 = (B_2 \to H_2, r(\vec{x}))$, where $\vec{x}$ is a list of all variables in H_2. One can check that $\mathcal{R} = \{R_0, R'_1, R'_2, R_3\}$ contains a cycle going through an existential variable (thus, it is not wa^D) iff R_2 depends on R_1. □

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 $\mu(H_1)$.*

Proof: We first formalize the sentence "no application of R_2 can use an atom in $\mu(H_1)$" by the following sentence: "no application π' of R_2 can map an atom $a \in B_2$ to an atom b produced by a application (R_1, π) such that $b = \pi(b')$, where π and π' are more specific than μ" (given two substitutions s_1 and s_2, s_1 is more specific than s_2 if there is a substitution s such that $s_1 = s \circ s_2$).

Consider the application of R_1 to a fact w.r.t. a homomorphism π, followed by an application of R_2 w.r.t. a homomorphism π', such that for an atom $a \in B_2$, $\pi'(a) = b = \pi(b')$, where π and π' are more specific than μ. Note that this implies that $\mu(a) = \mu(b')$. Assume that b contains a fresh variable z_i produced from an existential variable z of b' in H_1. Let z' be the variable from a such that $\pi'(z') = z_i$. Since the domain of π' is B_2, all atoms from B_2 in which z' occurs at a given position p_j are also mapped by π' to atoms containing z_i in the same position p_j. Since z_i is a fresh variable, these atoms have been produced by sequences of rule applications starting from (R_1, π). Such a sequence of rule applications exists only if there is a path in PG^U from a position of z in H_1 to p_j; moreover, this path cannot go through an existential position, otherwise z_i cannot be propagated. Hence, μ is necessarily compatible.

□

Proposition 8 *If, for each each existential position $[a, i]$, all compatible cycles for $[a, i]$ in PG^U are self-blocking, then the stable computation based on the skolem chase halts.*

Proof: If a cycle is non-compatible or self-blocking, then no sequence of rule applications can use it (where "used" is defined as

in the proof of Theorem 3). Hence, if all compatible cycles are self-blocking, all derivations obtained with skolemized NME rule applications are finite. Hence, the stable computation based on the skolem chase halts.

□