A Single Approach to Decide Chase Termination on Linear Existential Rules

Michel Leclère, Marie-Laure Mugnier, Michaël Thomazo, Federico Ulliana

A Single Approach to Decide Chase Termination on Linear Existential Rules

Michel Leclère

University of Montpellier, CNRS, Inria, LIRMM, France
leclere@lirmm.fr

Marie-Laure Mugnier

University of Montpellier, CNRS, Inria, LIRMM, France
mugnier@lirmm.fr

Michaël Thomazo

Inria, DI ENS, ENS, CNRS, PSL University, France
michael.thomazo@inria.fr

Federico Ulliana

University of Montpellier, CNRS, Inria, LIRMM, France
ulliana@lirmm

Abstract

Existential rules, long known as tuple-generating dependencies in database theory, have been intensively studied in the last decade as a powerful formalism to represent ontological knowledge in the context of ontology-based query answering. A knowledge base is then composed of an instance that contains incomplete data and a set of existential rules, and answers to queries are logically entailed from the knowledge base. This brought again to light the fundamental chase tool, and its different variants that have been proposed in the literature. It is well-known that the problem of determining, given a chase variant and a set of existential rules, whether the chase will halt on any instance, is undecidable. Hence, a crucial issue is whether it becomes decidable for known subclasses of existential rules. In this work, we consider linear existential rules with atomic head, a simple yet important subclass of existential rules that generalizes inclusion dependencies. We show the decidability of the *all-instance* chase termination problem on these rules for three main chase variants, namely *semi-oblivious*, *restricted* and *core* chase. To obtain these results, we introduce a novel approach based on so-called derivation trees and a single notion of forbidden pattern. Besides the theoretical interest of a unified approach and new proofs for the semi-oblivious and core chase variants, we provide the first positive decidability results concerning the termination of the restricted chase, proving that chase termination on linear existential rules with atomic head is decidable for both versions of the problem: Does *every* chase sequence terminate? Does *some* chase sequence terminate?

2012 ACM Subject Classification Theory of Computation → Logic; Computing Methodologies → Artificial Intelligence → Knowledge Representation and Reasoning;

Keywords and phrases Chase, Tuple Generating Dependencies, Existential rules, Decidability

Digital Object Identifier 10.4230/LIPIcs.ICDT.2019.15

Related Version arXiv:1810.02132

1 Introduction

The chase procedure is a fundamental tool for solving many issues involving tuple-generating dependencies, such as data integration [21], data-exchange [11], query answering using views [16] or query answering on probabilistic databases [24]. In the last decade, tuple-generating dependencies raised a renewed interest under the name of *existential rules* for the problem known as ontology-based query answering. In this context, the aim is to query a knowledge

22nd International Conference on Database Theory (ICDT 2019).
Editors: Pablo Barcelo and Marco Calautti; Article No. 15; pp. 15:1–15:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

base (I, Σ), where I is an instance and Σ is an ontology given as a set of existential rules (see e.g. the survey chapters [5, 23]). In more classical database terms, this problem can be recast as querying an instance I under open-world assumption, provided with a set of constraints Σ, which are tuple-generating dependencies. The chase is a fundamental tool to solve dependency-related problems as it allows one to compute a (possibly infinite) *universal model* of (I, Σ), *i.e.*, a model that can be homomorphically mapped to any other model of (I, Σ). Hence, the answers to a conjunctive query (and more generally to any kind of query closed by homomorphism) over (I, Σ) can be defined by considering solely this universal model.

Several variants of the chase have been introduced, and we focus in this paper on the main ones: semi-oblivious [22] (aka Skolem [22]), restricted [3, 11] (aka standard [25]) and core [9]. Any chase variant starts from an instance and exhaustively performs a sequence of non-redundant rule applications according to a redundancy criterion which characterizes the variant itself. The built sequence is required to be *fair*, i.e., no rule application deemed non-redundant can be indefinitely left out. It is well known that all the above variants produce homomorphically equivalent results but obey increasingly stronger redundancy criteria, hence terminate for increasingly larger subclasses of existential rules.

The question of whether a chase variant terminates on *all instances* for a given set of existential rules is known to be undecidable when there is no restriction on the kind of rules [1, 12]. A number of *sufficient* syntactic conditions for termination have been proposed in the literature for the semi-oblivious chase (see e.g. [25, 15, 27] for syntheses), as well as for the restricted chase [8] (note that the latter paper also defines a sufficient condition for non-termination). However, only few positive results exist regarding the termination of the chase on specific classes of rules. Decidability was shown for the semi-oblivious chase on guarded-based rules (linear rules, and their extension to (weakly-)guarded rules) [4]. Decidability of the core chase termination on guarded rules for a fixed instance was shown in [17].

In this work, we provide new insights on the chase termination problem for *linear* existential rules with atomic head, a simple yet important subclass of guarded existential rules, which generalizes inclusion dependencies [10] and some practical ontological languages [6]. When we are interested in the ontology-based query answering problem, we note that linear rules are first-order rewritable, hence answering conjunctive queries on linear rule knowledge bases can be solved by query rewriting [1, 13, 18]. However, it is well known that the size and the unusual form of the rewritten query may lead to practical efficiency issues. Hence, the materialization of ontological inferences in the data is often an effective alternative to query rewriting, provided that some chase algorithm terminates. Finally, having the choice of how to process a set of linear rules extends the applicability of query answering techniques that combine query rewriting and materialization [1].

Precisely, the question of whether a chase variant terminates on all instances for a set of linear existential rules with atomic head is studied in two fashions:

- does *every* chase sequence terminate?
- does *some* chase sequence terminate?

It is well-known that these two questions have the same answer for the semi-oblivious and the core chase variants, but not for the restricted chase. Indeed, this last one may admit both terminating and non-terminating sequences over the same knowledge base. We show that the termination problem is decidable for linear existential rules with atomic head, no matter which version of the problem and which chase variant we consider.

We study chase termination by exploiting in a novel way a graph structure, namely the *derivation tree*, which was originally introduced to solve the ontology-based (conjunctive) query answering problem for the family of greedy-bounded treewidth sets of existential rules [2, 28], a class that generalizes guarded-based rules and in particular linear rules. We first use derivation trees to show the decidability of the termination problem for the semi-oblivious and restricted chase variants, and then generalize them to *entailment trees* to show the decidability of termination for the core chase. For every chase variant we consider, we adopt the same high-level procedure: starting from a finite set of canonical instances (representative of all possible instances), we build a (set of) tree structures for each canonical instance, while forbidding the occurrence of a specific pattern, that we call *unbounded-path witness*. The structures that we build are finite thanks to this forbidden pattern, and this allows us to decide if the chase terminates on the associated canonical instance. By doing so, we obtain a uniform approach to study the termination of several chase variants, which we believe to be of theoretical interest per se. In particular, some important differences in the chase behaviors become more visible. The derivation tree is moreover a simple structure, which makes the algorithms built on it effectively implementable. Let us also point out that our approach is constructive: if the chase terminates on a given instance, the algorithm that decides termination actually computes the result of the chase (or a superset of it in the case of the core chase), otherwise it pinpoints a forbidden pattern responsible for non-termination.

Besides providing new theoretical tools to study chase termination for linear existential rules with atomic head, we obtain the following results:

- a new proof of the decidability of the semi-oblivious chase termination, building on different objects than the previous proof provided in [4]; we show that our algorithm provides the same complexity upper-bound;
- the decidability of the restricted chase termination, for both versions of the problem, i.e., termination of all chase sequences and termination of some chase sequence; to the best of our knowledge, these are the first positive results on the decidability of the restricted chase termination;
- a new proof of the decidability of the core chase termination, with different objects than previous work reported in [17]; although this latter paper solves the question of the core chase termination given a fixed instance, the results actually allow to infer the decidability of the *all*-instance version of the problem (still on linear rules), by noticing that only a finite number of instances need to be considered (see the next section).

The paper is organized as follows. After introducing some preliminary notions (Section 2), we define the main components of our framework, namely derivation trees and unbounded-path witnesses (Section 3). We build on these objects to prove the decidability of the semi-oblivious and restricted chase termination (Section 4). Finally, we generalize derivation-trees to entailment trees and use them to prove the decidability of the core chase termination (Section 5). Detailed proofs are provided in [19]. A previous version of this work was presented as an extended abstract in [20].

2 Preliminaries

We consider a logical *vocabulary* composed of a finite set of predicates and an infinite set of constants. An *atom* α has the form $r(t_1, \ldots, t_n)$ where r is a predicate of arity n and the t_i are terms (i.e., variables or constants). We denote by $\mathit{terms}(\alpha)$ (resp. $\mathit{vars}(\alpha)$) the set of terms (resp. variables) in α and extend the notations to a set of atoms. A *ground* atom does not contain any variable. It is convenient to identify (the existential closure of) a conjunction of atoms with the set of these atoms. A set of atoms is *atomic* if it contains a single atom. An *instance* is a set of (non-necessarily ground) atoms, which is finite unless otherwise specified. Abusing terminology, we will often see an instance as its isomorphic model.

Given two sets of atoms S and S', a *homomorphism* from S' to S is a substitution π of $vars(S')$ by $terms(S)$ such that $\pi(S') \subseteq S$. It holds that $S \models S'$ (where $\models$ denotes classical logical entailment) iff there is a homomorphism from S' to S. An *endomorphism* of S is a homomorphism from S to itself. A set of atoms is a *core* if it admits only injective endomorphisms. Any finite set of atoms is logically (or: homomorphically) equivalent to one of its subsets that is a core, and this core is unique up to isomorphism (i.e., bijective variable renaming). Given sets of atoms S and S', we say that S' *folds* onto S if there is a homomorphism π from $S' \setminus S$ to S such that π is the identity on $vars(S)$. The homomorphism π is called a *folding*. In particular, it is well-known that any set of atoms *folds* onto its core.

An *existential rule* (or simply *rule*) is of the form $\sigma = \forall \mathbf{x} \forall \mathbf{y}\ [\mathrm{body}(\mathbf{x}, \mathbf{y}) \rightarrow \exists \mathbf{z}\ \mathrm{head}(\mathbf{x}, \mathbf{z})]$ where $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$ are pairwise disjoint tuples of variables, $\mathrm{body}(\mathbf{x}, \mathbf{y})$ and $\mathrm{head}(\mathbf{x}, \mathbf{z})$ are non-empty conjunctions of atoms respectively on $\mathbf{x} \cup \mathbf{y}$ and $\mathbf{x} \cup \mathbf{z}$, called the *body* and the *head* of the rule, and also denoted by $\mathrm{body}(\sigma)$ and $\mathrm{head}(\sigma)$. The variables of $\mathbf{z}$ are called *existential variables*. The variables of $\mathbf{x}$ form the *frontier* of σ, which is also denoted by $\mathrm{fr}(\sigma)$. For brevity, we will omit universal quantifiers in the examples. A *knowledge base* (KB) is of the form $\mathcal{K} = (I, \Sigma)$, where I is an instance and Σ is a finite set of existential rules.

A rule $\sigma = \mathrm{body}(\sigma) \rightarrow \mathrm{head}(\sigma)$ is *applicable* to an instance I if there is a homomorphism π from $\mathrm{body}(\sigma)$ to I. The pair (σ, π) is called a *trigger* for I. The result of the application of σ according to π on I is the instance $I' = I \cup \pi^s(\mathrm{head}(\sigma))$, where π^s extends π by assigning a distinct fresh variable (also called a *null*) to each existential variable.[1] We also say that I' is obtained by *firing* the trigger (σ, π) on I. By $\pi_{|\mathrm{fr}(\sigma)}$ we denote the restriction of π to the domain $\mathrm{fr}(\sigma)$.

▶ **Definition 1 (Derivation).** *A* Σ-derivation *(or simply* derivation *when* Σ *is clear from the context) from an instance* $I = I_0$ *to an instance* I_n *is a sequence* $I_0, (\sigma_1, \pi_1), I_1 \ldots, (\sigma_n, \pi_n), I_n$*, such that for all* $1 \leq i \leq n$*:* $\sigma_i \in \Sigma$*,* (σ_i, π_i) *is a trigger for* I_{i-1}*,* I_i *is obtained by firing* (σ_i, π_i) *on* I_{i-1}*, and* $I_i \neq I_{i-1}$*. The notion of derivation can be naturally extended to an* infinite *sequence.*

Note that the condition $I_i \neq I_{i-1}$ in the above definition implies that all triggers in a derivation produce distinct atoms. We briefly introduce below the main chase variants and refer to [25] for a detailed presentation.

The *semi-oblivious* chase prevents several applications of the same rule through the same mapping of its frontier. Given a derivation from I_0 to I_i, a trigger (σ, π) for I_i is said to be *active according to the semi-oblivious criterion*, if (1) there is no trigger (σ_j, π_j) in the derivation with $\sigma = \sigma_j$ and $\pi_{|\mathrm{fr}(\sigma)} = \pi_{j|\mathrm{fr}(\sigma_j)}$ and (2) the extension of I_i with (σ, π) remains a derivation, i.e., $\pi(\mathrm{head}(\sigma)) \not\subseteq I_i$. The *restricted* chase performs a rule application only if the added set of atoms is not redundant with respect to the current instance. Given a derivation from I_0 to I_i, a trigger (σ, π) for I_i is said to be *active according to the restricted criterion* if π cannot be extended to a homomorphism from $(\mathrm{body}(\sigma) \cup \mathrm{head}(\sigma))$ to I_i (equivalently, $\pi^s(\mathrm{head}(\sigma))$ does not fold onto I_i). We say that a (possibly infinite) derivation is a *semi-oblivious (resp. restricted) derivation* if it is built by firing only active triggers according to the semi-oblivious (resp. restricted) criterion. A (possibly infinite) derivation is *fair* if no

[1] The "s" superscript stands for *safe*, as newly introduced variables are fresh, hence cannot be confused with already existing variables.

firing of an active trigger is indefinitely delayed; formally: if some I_i in the derivation admits an active trigger (σ, π), then there is $j > i$ such that, either I_j is obtained by firing (σ, π) on I_{j-1}, or (σ, π) is not an active trigger anymore on I_j. A *terminating* derivation is a finite fair derivation (in other words, a finite derivation that does not admit any active trigger on its final instance). A semi-oblivious (resp. restricted) *chase sequence* is a fair semi-oblivious (resp. restricted) derivation. Hence, a chase sequence is terminating if and only if it is finite.

In its original definition [9], the *core* chase proceeds in a breadth-first manner, and, at each step, first fires in parallel all active triggers according to the restricted chase criterion, then computes the core of the result. Alternatively, to bring the definition of the core chase closer to the above definitions of the semi-oblivious and restricted chases, one can define a *core derivation* as a possibly infinite sequence $I_0, (\sigma_1, \pi_1), I_1, \ldots$, alternating instances and triggers, such that each (σ_i, π_i) is an active trigger on I_{i-1} according to the restricted criterion, and I_i is obtained from I_{i-1} by first firing (σ_i, π_i), then computing the core of the result. Then, a core chase sequence is a fair core derivation. An instance admits a terminating core chase sequence in that sense if and only if the core chase as originally defined terminates on that instance.

For the three chase variants, a chase sequence defines a (possibly infinite) *universal model* of the KB, but only the core chase stops if and only if the KB has a *finite* universal model.

It is well-known that, for the semi-oblivious and the core chase, if there is a terminating chase sequence from an instance I then all chase sequences from I are terminating. This is not the case for the restricted chase, since the order in which rules are applied has an impact on termination, as illustrated by Example 2.

▶ **Example 2.** Let $\Sigma = \{\sigma_1, \sigma_2\}$, with $\sigma_1 = p(x, y) \rightarrow \exists z\ p(y, z)$ and $\sigma_2 = p(x, y) \rightarrow p(y, y)$. Let $I = p(a, b)$. The KB (I, Σ) has a finite universal model, for example, $I^* = \{p(a, b), p(b, b)\}$. The semi-oblivious chase does not terminate on I as σ_1 is applied indefinitely, while the core chase terminates after one breadth-first step and returns I^*. The restricted chase has a terminating sequence, for example $I_0 = I, (\sigma_2, \{x \mapsto a, y \mapsto b\}), I_1 = I^*$, but it also has infinite sequences, for instance sequences that apply always σ_1 before σ_2 on a given atom.

We study the following problems for the semi-oblivious, restricted and core chase variants:

- *(All-instance) all-sequence termination:* Given a set of rules Σ, is it true that, for any instance, all chase sequences are terminating?
- *(All-instance) one-sequence termination:* Given a set of rules Σ, is it true that, for any instance, there is a terminating chase sequence?

Note that, according to the terminology of [14], these problems can be recast as deciding whether, for a chase variant, a given set of rules belongs to the class $\mathrm{CT}_{\forall\forall}$ or $\mathrm{CT}_{\forall\exists}$, respectively.

An existential rule is called *linear* if its body is atomic, see e.g., [6]. As often done, we moreover restrict linear rules to atomic heads (and still use the name linear rules). Linear rules generalize *inclusion dependencies* [10] by allowing several occurrences of the same variable in an atom. They also generalize positive inclusions in the description logic DL-Lite$_\mathcal{R}$ (the formal basis of the web ontological language OWL 2 QL) [7], which can be seen as inclusion dependencies restricted to unary and binary predicates.

Note that the restriction of existential rules to rules with an atomic head is often made in the literature, considering that any existential rule with a complex head can be decomposed into several rules with an atomic head, by introducing a fresh predicate for each rule. However, while this translation preserves the termination of the semi-oblivious chase, it is not the case for the restricted and the core chases. Hence, considering linear rules with a complex head would require to extend the techniques developed in this paper.

To simplify the presentation, we assume in the following that each rule frontier is of size at least one. This assumption is made without loss of generality. [2]

We first point out that the termination problem on linear rules can be recast by considering solely atomic instances (as already remarked in several contexts).

▶ **Proposition 3.** *Let Σ be a set of linear rules. The semi-oblivious (resp. restricted, core) chase terminates on all instances if and only if it terminates on all atomic instances.*

We will furthermore rely on the following notion of the type of an atom.

▶ **Definition 4 (Type of an atom).** *The* type of an atom $\alpha = r(t_1, \ldots, t_n)$*, denoted by* $type(\alpha)$*, is the pair* $(r, \mathcal{P})$ *where* $\mathcal{P}$ *is the partition of* $\{1, \ldots, n\}$ *induced by term equality (i.e.,* i *and* j *are in the same class of* $\mathcal{P}$ *iff* $t_i = t_j$*).*

There are finitely (more specifically, exponentially) many types for a given vocabulary.

When two atoms α and α' have the same type, there is a *natural mapping* from α to α', denoted by $\varphi_{\alpha\to\alpha'}$, and defined as follows: it is a bijective mapping from $terms(\alpha)$ to $terms(\alpha')$, that maps the i-th term of α to the i-th term of α'. Note that $\varphi_{\alpha\to\alpha'}$ may not be an isomorphism, as constants from α may not be mapped to themselves. However, if (σ, π) is a trigger for $\{\alpha\}$, then $(\sigma, \varphi_{\alpha\to\alpha'} \circ \pi)$ is a trigger for $\{\alpha'\}$, as there are no constants in the considered rules.

Together with Proposition 3, this implies that one can check all-instance all-sequence termination by checking all-sequence termination on a finite set of instances, called *canonical instances*: for each type, we take exactly one canonical instance that has this type.

We will consider different kinds of tree structures, which have in common to be *trees of bags.*

▶ **Definition 5 (Tree of bags).** *A* tree of bags *is a rooted tree whose nodes, called bags,*[3] *are labeled by an atom. For any bag* B*, we define the following notations:*

- $atom(B)$ *is the label of* B*;*
- $terms(B) = terms(atom(B))$ *is the set of terms of* B*;*
- $terms(B)$ *is divided into two sets of terms, those* generated *in* B*, denoted by* $generated(B)$*, and those shared with its parent, denoted by* $shared(B)$*; precisely,* $terms(B) = shared(B) \cup generated(B)$*,* $shared(B) \cap generated(B) = \emptyset$*, and if* B *is the root of* $\mathcal{T}$*, then* $generated(B) = terms(B)$ *(hence* $shared(B) = \emptyset$*), otherwise* B *has a parent* B_p *and* $generated(B) = terms(B) \setminus terms(B_p)$ *(hence,* $shared(B) = terms(B_p) \cap terms(B)$*).*

Last, we denote by $atoms(\mathcal{T})$ *the set of atoms that label the bags in* $\mathcal{T}$*.*

Finally, we recall some classical mathematical notions. A *subsequence* S' of a sequence S is a sequence that can be obtained from S by deleting some (or no) elements without changing the order of the remaining elements. The *arity* of a tree is the maximal number of children for a node. A *prefix* T' of a tree T is a tree that can be obtained from T by repeatedly deleting some (or no) leaves of T.

[2] For instance, it can always be ensured by adding a position to all predicates, which is filled by the same fresh constant in the initial instance and by a new frontier variable in each rule. E.g. let c_0 be the new constant, then an instance atom of the form $p(a, b)$ would become $p(c_0, a, b)$ and a rule of the form $p(x, y) \to \exists z_1 \exists z_2\ q(z_1, z_2)$ would become $p(u, x, y) \to \exists z_1 \exists z_2\ q(u, z_1, z_2)$, where u is a variable. This translation does not change the behavior of any chase variant.

[3] The trees of bags that we consider are tree decompositions [26], hence the term "bag", which is classical in this setting.

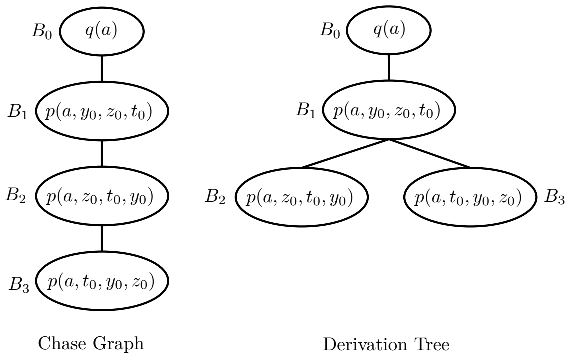

Figure 1 Chase Graph and Derivation Tree of Example 7

3 Derivation Trees

A classical tool to reason about the chase is the so-called *chase graph* (see e.g., [6]), which is the directed graph consisting of all atoms that appear in the considered derivation, and with an arrow from a node n_1 to a node n_2 iff n_2 is created by a rule application on n_1 and possibly other atoms. [4] In the specific case of KBs of the form (I, Σ), where I is an atomic instance and Σ a set of linear rules, the chase graph is a tree. We recall below its definition in this specific case, in order to emphasize its differences with another tree, called *derivation tree*, on which we will actually rely.

▶ **Definition 6 (Chase Graph for Linear Rules).** *Let $I_0 = \{\alpha\}$ be an atomic instance, Σ be a set of linear rules, and $S = I_0, (\sigma_1, \pi_1), I_1, \dots, (\sigma_n, \pi_n), I_n$ be a semi-oblivious Σ-derivation. The* chase graph *(also called* chase tree*) assigned to S is a tree of bags built as follows:*

- *the set of bags is in bijection with I_n via the labeling function atom();*
- *the set of edges is in bijection with the set of triggers in S and is built as follows: for each trigger (σ_i, π_i) in S, there is an edge (B, B') with $\text{atom}(B) = \pi_i(\text{body}(\sigma_i))$ and $\text{atom}(B') = \pi_i^s(\text{head}(\sigma_i))$.*

▶ **Example 7.** Let $I_0 = q(a)$ and $\Sigma = \{\sigma_1, \sigma_2\}$ where $\sigma_1 = q(x) \to \exists y \exists z \exists t\ p(x, y, z, t)$ and $\sigma_2 = p(x, y, z, t) \to p(x, z, t, y)$. Let $S = I_0, (\sigma_1, \pi_1), I_1, (\sigma_2, \pi_2), I_3, (\sigma_2, \pi_3), I_3$ with $\pi_1 = \{x \mapsto a\}$, $\pi_1^s(\text{head}(\sigma_1)) = p(a, y_0, z_0, t_0)$, $\pi_2 = \{x \mapsto a, y \mapsto y_0, z \mapsto z_0, t \mapsto t_0\}$ and $\pi_3 = \{x \mapsto a, y \mapsto z_0, z \mapsto t_0, t \mapsto y_0\}$. The chase graph associated with S is a path of four nodes as pictured in Figure 1.

To check termination of a chase variant on a given KB (with an atomic instance), the general idea is to build a tree of bags associated with the chase on this KB in such a way that the occurrence of some forbidden pattern indicates that a path of unbounded length can be developed, hence the chase does not terminate. The forbidden pattern is composed of two distinct nodes such that one is an ancestor of the other and, intuitively speaking, these nodes "can be extended in similar ways", which leads to an arbitrarily long path that repeats the pattern.

Two atoms with the same type admit the same rule triggers, however, within a derivation, the same rule applications cannot necessarily be performed on both of them because of the

[4] Note that the chase graph in [9] is a different notion.

presence of other atoms (this is already true for datalog rules, since the same atom is never produced twice). Hence, on the one hand we will specialize the notion of type, into that of a *sharing type*, and, on the other hand, adopt another tree structure, called a *derivation tree*, in which two nodes with the same sharing type have the required similar behavior.

▶ **Definition 8 (Sharing Type).** *Given a tree of bags, the* sharing type *of a bag B is a pair $(type(atom(B)), P)$ where P is the set of positions in* $\mathsf{atom}(B)$ *in which a term of* $\mathsf{shared}(B)$ *occurs. We denote the fact that two bags B and B' have the same sharing type by $B \equiv_{st} B'$.*

We can now specify the forbidden pattern that we will consider: it is a pair of two distinct nodes with the same sharing type, such that one is an ancestor of the other.

▶ **Definition 9 (Unbounded-Path Witness).** *An* unbounded-path witness *(UPW) in a derivation tree is a pair of distinct bags (B, B') such that B and B' have the same sharing type and B is an ancestor of B'.*

As explained below on Example 7, the chase graph is not the appropriate tool to use this forbidden pattern as a witness of chase non-termination.

Example 7 (cont'd). B_1, B_2 and B_3 in the chase graph of Figure 1 have the same classical type, $t = (p, \{\{1\}, \{2\}, \{3\}, \{4\}\})$. The sharing type of B_1 is $(t, \{1\})$, while B_2 and B_3 have the same sharing type $(t, \{1, 2, 3, 4\})$. B_2 and B_3 fulfill the condition of the forbidden pattern, however it is easily checked that any derivation that extends this derivation is finite.

Derivation trees were introduced as a tool to define the *greedy bounded treewidth set (gbts)* family of existential rules [2, 28]. A derivation tree is associated with a derivation, however it does not have the same structure as the chase graph. The fundamental reason is that, when a rule σ is applied to an atom α via a homomorphism π, the newly created bag is not necessarily attached in the tree as a child of the bag labeled by α. Instead, it is attached as a child of the *highest* bag in the tree labeled by an atom that contains $\pi(\mathrm{fr}(\sigma))$, the image by π of the frontier of σ (note that $\pi(\mathrm{fr}(\sigma))$ remains the set of terms shared between the new bag and its parent). This highest bag is unique. It is also the first created bag that contains $\pi(\mathrm{fr}(\sigma))$.

In the following definition, a derivation tree is not associated with *any* derivation, but with a semi-oblivious derivation, which has the advantage of yielding trees with *bounded arity.* This is appropriate to study the termination of the semi-oblivious chase, and later the restricted chase, as a restricted derivation is a specific semi-oblivious derivation.

▶ **Definition 10 (Derivation Tree).** *Let $I_0 = \{\alpha\}$ be an atomic instance, Σ be a set of linear rules, and $S = I_0, (\sigma_1, \pi_1), I_1, \ldots, (\sigma_n, \pi_n), I_n$ be a semi-oblivious Σ-derivation. The* derivation tree *assigned to S is a tree $\mathcal{T}$ of bags built as follows:*

- *the root of the tree, B_0, is such that* $\mathsf{atom}(B_0) = \alpha$;
- *for each trigger (σ_i, π_i), $0 < i \leq n$, let B_i be the bag such that* $\mathsf{atom}(B_i) = \pi_i^s(\mathrm{head}(\sigma_i))$. *Let j be the smallest integer such that $\pi_i(\mathrm{fr}(\sigma_i)) \subseteq \mathsf{terms}(B_j)$: B_i is added as a child to B_j.*

By extension, we say that a derivation tree $\mathcal{T}$ is associated with α and Σ *if there exists a semi-oblivious Σ-derivation S from α such that $\mathcal{T}$ is assigned to S.*

Example 7 (cont'd). The derivation tree associated with S is represented in Figure 1. Bags have the same sharing types in the chase tree and in the derivation tree. However, we can see here that they are not linked in the same way: B_3 was a child of B_2 in the chase tree, it becomes a child of B_1 in the derivation tree. Hence, the forbidden pattern cannot be found anymore in the tree.

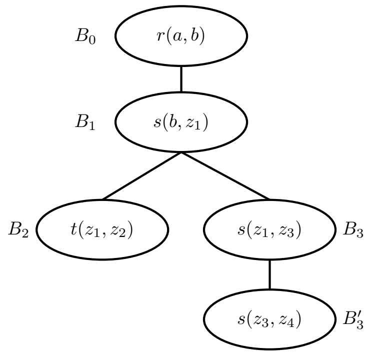

Figure 2 Copy Operation and Derivation Trees

Note that every non-root bag B shares at least one term with its parent (since the rule frontiers are not empty), furthermore this term is *generated* in its parent (otherwise B would have been added at a higher level in the tree).

4 Semi-Oblivious and Restricted Chase Termination

We now use derivation trees and sharing types to characterize the termination of the semi-oblivious chase. The fundamental property of derivation trees that we exploit is that, when two nodes have the same sharing type, the considered (semi-oblivious) derivation can always be extended so that these nodes have the same number of children, and in turn these children have the same sharing type. We first specify the notion of *bag copy*.

▶ **Definition 11 (Bag Copy).** *Let $\mathcal{T}$ and $\mathcal{T}'$ be (possibly equal) trees of bags. Let B be a bag of $\mathcal{T}$ and B' be a bag of $\mathcal{T}'$ such that $B \equiv_{st} B'$. Let B_c be a child of B. A* copy *of B_c* under *B' is a bag B'_c such that $\mathrm{atom}(B'_c) = \varphi^s(\mathrm{atom}(B_c))$, where φ^s is a substitution of $\mathrm{terms}(B_c)$ defined as follows:*

- *if $t \in \mathrm{shared}(B_c)$, then $\varphi^s(t) = \varphi_{\mathrm{atom}(B)\to\mathrm{atom}(B')}(t)$, where $\varphi_{\mathrm{atom}(B)\to\mathrm{atom}(B')}$ is the natural mapping from $\mathrm{atom}(B)$ to $\mathrm{atom}(B')$;*
- *if $t \in \mathrm{generated}(B_c)$, then $\varphi^s(t)$ is a fresh new variable.*

Let $\mathcal{T}_e$ be obtained from a derivation tree $\mathcal{T}$ by adding a copy of a bag: strictly speaking, $\mathcal{T}_e$ may not be a derivation tree in the sense that there may be no derivation to which it can be assigned, as illustrated by Example 12. Intuitively, some rule applications that would allow to produce the copy may be missing. However, there is some derivation tree of which $\mathcal{T}_e$ is a *prefix* (intuitively, one can add bags to $\mathcal{T}_e$ to obtain a derivation tree). That is why the following proposition considers more generally prefixes of derivation trees.

▶ **Example 12.** Let $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$ with:

$\sigma_1 : r(x,y) \to \exists z\ s(y,z) \quad \sigma_2 : s(x,y) \to \exists z\ t(y,z) \quad \sigma_3 : t(x,y) \to \exists z\ s(x,z)$

Figure 2 pictures the result of copying a bag in a derivation tree $\mathcal{T}$ associated with a Σ-derivation: $\mathcal{T}$ has bags B_0, B_1, B_2 and B_3; then the bag B_3 is copied under itself yielding B'_3 (this copy is possible as B_1 and B_3 have the same sharing type and B_3 is a child of B_1). The obtained tree structure is not a derivation tree, as $s(z_3,z_4)$ cannot be generated by applying a rule on the instance associated with $\mathcal{T}$, however it could be completed to yield a derivation tree.

▶ **Proposition 13.** *Let $\mathcal{T}$ be a prefix of a derivation tree, B and B' be two bags of $\mathcal{T}$ such that $B \equiv_{st} B'$, and B_c be a child of B. Let B'_c be a copy of B_c under B'. Then: (a) it holds that $B_c \equiv_{st} B'_c$, and* (b) *the tree obtained from $\mathcal{T}$ by adding B'_c under B', if no copy of B_c already exists under B', is a prefix of a derivation tree.*

The size of a derivation tree without UPW is bounded, since its arity is bounded and its depth is bounded by the number of sharing types. It remains to show that a derivation tree that contains a UPW can be extended to an arbitrarily large derivation tree. We recall that a similar property would not hold for the chase tree, as witnessed by Example 7.

▶ **Proposition 14.** *There exists an arbitrary large derivation tree associated with α and Σ if and only if there exists a derivation tree associated with α and Σ that contains an unbounded-path witness.*

The previous proposition yields a characterization of the existence of an infinite semi-oblivious derivation. At this point, one may notice that an infinite semi-oblivious derivation is not necessarily fair. However, from this infinite derivation one can always build a fair derivation by inserting missing triggers. Obviously, this operation has no effect on the termination of the semi-oblivious chase. More precaution will be required for the restricted chase.

One obtains an algorithm to decide termination of the semi-oblivious chase for a given set of rules: for each canonical instance, build a semi-oblivious derivation and the associated derivation tree by applying rules until a UPW is created (in which case the answer is no) or all possible rule applications have been performed; if no instance has returned a negative answer, the answer is yes.

▶ **Corollary 15.** *The all-sequence termination problem for the semi-oblivious chase on linear rules is decidable.*

This algorithm can be modified to run in polynomial space (which is optimal [4]), by guessing a canonical instance and a UPW of its derivation tree.

▶ **Proposition 16.** *The all-sequence termination problem for the semi-oblivious chase on linear rules is in* PSpace.

We now consider the restricted chase. To this aim, we call *restricted derivation tree* associated with α and Σ a derivation tree associated with a restricted Σ-derivation from α. We first point out that Proposition 13 is not true anymore for a restricted derivation tree, as the order in which rules are applied matters.

▶ **Example 17.** Consider a restricted tree that contains bags B and B' with the same sharing type, labeled by atoms $q(t, u)$ and $q(v, w)$ respectively, where the second term is generated. Consider the following rules (the same as in Example 2):

$\sigma_1 : q(x, y) \to \exists z\ q(y, z) \qquad \sigma_2 : q(x, y) \to q(y, y)$

Assume B has a child B_c labeled by $q(u, z_0)$ obtained by an application of σ_1, and B' has a child B'_1 labeled by $q(w, w)$ obtained by an application of σ_2. It is not possible to extend this tree by copying B_c under B'. Indeed, the corresponding application of σ_1 does not comply with the restricted chase criterion: it would produce an atom of the form $q(w, z_1)$ that folds onto $q(w, w)$.

We thus prove a weaker proposition by considering that B' is a leaf in the restricted derivation tree.

▶ **Proposition 18.** *Let $\mathcal{T}$ be a prefix of a restricted derivation tree, B and B' be two bags of $\mathcal{T}$ such that $B \equiv_{st} B'$ and B'* is a leaf. *Let B_c be a child of B and B'_c be a copy of B_c under B'. Then:* (a) *$B_c \equiv_{st} B'_c$ and* (b) *the tree obtained from $\mathcal{T}$ by adding the copy B'_c of B_c under B' is a prefix of a restricted derivation tree.*

The previous proposition allows us to obtain a variant of Proposition 14 adapted to the restricted chase: [5]

▶ **Proposition 19.** *There exists an arbitrary large restricted derivation tree associated with α and Σ if and only if there exists a restricted derivation tree associated with α and Σ that contains an unbounded-path witness.*

It is less obvious than in the case of the semi-oblivious chase that the existence of an infinite derivation entails the existence of an infinite *fair* derivation. However, this property still holds:

▶ **Proposition 20.** *For linear rules, every (infinite) non-terminating restricted derivation is a subsequence of a fair restricted derivation (i.e., a restricted chase sequence).*

Proof: Let S be an infinite unfair restricted derivation. In particular, there exists at least one infinite branch in the associated derivation tree. Let us consider the following derivation: when the node B_k of depth k on this branch has been generated, complete the corresponding subsequence by trying to apply (i.e., while respecting the restricted criterion) all currently applicable triggers that add a bag a depth at most $k-1$. These additional rule applications cannot prevent the creation of any bag that is below B_k in the derivation tree. Indeed, let α_c be an atom possibly created by a rule application, whose bag would be attached as a child of a bag B (in the subtree of B_k); since α_c and $atom(B)$ share a variable that is generated in B, hence only occurs in the subtree of B, the only possibility for α_c to fold into the current instance, is to be mapped to an atom *in the subtree of B*. By construction, any possible rule application will be performed or inhibited at some point, which implies that the derivation that we build in this fashion is fair. □

We point out that the previous property is not true anymore if linear rules are extended to non-atomic heads, as illustrated by the next example (in which only binary atoms are used).

▶ **Example 21.** Let $\Sigma = \{\sigma_1 : r(x,y) \rightarrow r(x,x);\ \sigma_2 : r(x,y) \rightarrow s(x,x);\ \sigma_3 : r(x,y) \rightarrow \exists z\ r(x,z) \wedge s(z,x) \wedge s(y,z)\}$. Starting from the instance $r(a,b)$, one builds an infinite restricted derivation that repeatedly applies σ_3, ignoring the two other rules. In order to turn this derivation into a fair derivation, one should at some point perform the applications of σ_1 and σ_2 to the initial instance, which produce the atoms $r(a,a)$ and $s(a,a)$. However, as soon as these atoms are produced, all triggers involving σ_3 are deactivated. Actually, there is no infinite fair restricted Σ-derivation from $r(a,b)$.

Similarly to Proposition 14 for the semi-oblivious chase, Proposition 19 provides an algorithm to decide termination of the restricted chase. The difference is that it is not sufficient to build a single derivation for a given canonical instance; instead, all possible restricted derivations from this instance have to be built (note that the associated restricted

[5] Actually, a weak version of Proposition 13 where B' is a leaf would be sufficient to prove Proposition 14. However, the interest of Proposition 13 is to pinpoint an important difference between the semi-oblivious and restricted chase behaviors.

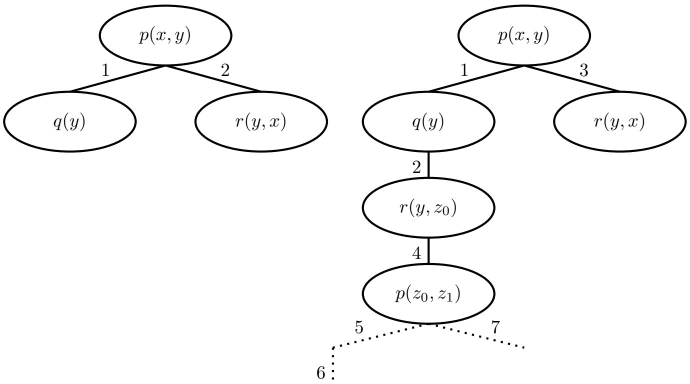

Figure 3 Finite versus Infinite Derivation Tree for Example 25

derivation trees are finite for the same reasons as before, and there is obviously a finite number of them). Hence, we obtain:

▶ **Corollary 22.** *The all-sequence termination problem for the restricted chase on linear rules is decidable.*

A rough analysis of the proposed algorithm provides a co-N2ExpTime upper-bound for the complexity of the problem, by guessing a derivation that is of length at most double exponential, and checking whether there is a UPW in the corresponding derivation tree.

We now prove the decidability of the one-sequence termination problem. Note that a (restricted) derivation tree $\mathcal{T}$ that contains a UPW (B, B') is a witness of the existence of an infinite (restricted fair) derivation, but does not prove that *every* (restricted fair) derivation that extends $\mathcal{T}$ is infinite.

▶ **Proposition 23.** *There exists a finite fair restricted derivation associated with α and Σ if and only if there exists one whose associated derivation tree does not contain any UPW.*

Proof:[Proof sketch] We show that the derivation tree of the smallest finite fair restricted derivation cannot contain a UPW. This is done by contradiction: we assume there is a UPW (B, B') in the associated derivation tree of the shortest finite fair restricted derivation. We create a new derivation that is equal to the original one up to the creation of B, then copy the subtree below B' under B, and applies "similar rules" so as to keep fairness, which yields a strictly shortest fair restricted derivation. The last step requires some care. ☐

An algorithm to decide the existence of a finite restricted chase sequence for any instance is as follows: for any canonical instance, build all restricted derivations until either *(i)* there is a UPW in their associated derivation tree or *(ii)* there is no active trigger anymore. Output Yes if and only if for all instances, there is a least one restricted derivation that halts because of the second condition. Such a derivation is of size at most double exponential, and we obtain a N2ExpTime upper bound for the complexity of the one-sequence termination problem.

▶ **Corollary 24.** *The one-sequence termination problem for the restricted chase on linear rules is decidable.*

Importantly, the previous algorithms are naturally able to consider solely some type of restricted derivations, i.e., build only derivation trees associated with such derivations, which is of theoretical but also of practical interest. Indeed, implementations of the restricted chase often proceed by building *breadth-first* derivations (which are intrinsically fair when they are infinite), or variants of these. As witnessed by the next example, the termination of all breadth-first fair derivations is a strictly weaker requirement than the termination of all fair sequences, in the sense that the restricted chase terminates on more sets of rules.

▶ **Example 25.** Consider the following set of rules:
$\sigma_1 = p(x,y) \to q(y)$ $\quad$ $\sigma_2 = p(x,y) \to r(y,x)$
$\sigma_3 = q(y) \to \exists z\ r(y,z)$ $\quad$ $\sigma_4 = r(x,y) \to \exists z\ p(y,z)$
All breadth-first (fair) restricted derivations terminate, whatever the initial instance is. Remark that every application of σ_1 is followed by an application of σ_2 in the same breadth-first step, which prevents the application of σ_3. However, there is a fair restricted derivation that does not terminate (and this is even true for any instance). Indeed, an application of σ_2 can always be delayed, so that it comes too late to prevent the application of σ_3. See Figure 3: on the left, a finite derivation tree associated with a breadth-first derivation from instance $p(x,y)$; on the right, an infinite derivation tree associated with a (non-breadth-first) fair infinite derivation from the same instance. The numbers on edges give the order in which bags are created.

We conclude this section by noting that the previous Example 25 may give the (wrong) intuition that, given a set of rules, it is sufficient to consider breadth-first fair derivations to decide if there exists a terminating sequence. The following example shows that it is not the case: here, no breadth-first chase sequence is terminating, while there exists a terminating chase sequence for the given instance.

▶ **Example 26.** Let $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$ with $\sigma_1 = p(x,y) \to \exists z\ p(y,z)$, $\sigma_2 = p(x,y) \to h(y)$, and $\sigma_3 = h(x) \to\ p(x,x)$. In this case, for every instance, there is a terminating restricted chase sequence, where the application of σ_2 and σ_3 prevents the indefinite application of σ_1. However, starting from $I = \{p(a,b)\}$, by applying rules in a breadth-first fashion one obtains a non-terminating restricted chase sequence, since σ_1 and σ_2 are always applied in parallel from the same atom, before applying σ_3.

As for the all-sequence termination problem, the algorithm may restrict the derivations of interest to specific kinds.

5 Core Chase Termination

We now consider the termination of the core chase on linear rules. Keeping the same approach, we prove that the finiteness of the core chase is equivalent to the existence of a finite tree of bags whose set of atoms is a minimal universal model. We call this a *(finite) complete core.* To bound the size of a complete core, we show that it cannot contain an unbounded-path witness. However, we cannot rely on derivation trees: indeed, there are linear sets of rules for which no derivation tree forms a complete core, as shown in Example 27. We will thus introduce a more general tree structure, namely *entailment trees.*

▶ **Example 27.** Let us consider the following rules:

$$s(x) \to \exists y \exists z\ p(y,z,x) \qquad p(y,z,x) \to \exists v\ q(y,v,x) \qquad q(y,v,x) \to p(y,v,x)$$

Let $I = \{s(a)\}$. The first rule applications yield a derivation tree $\mathcal{T}$ which is a path of bags B_0, B_1, B_2, B_3 respectively labeled by the following atoms:
$s(a), p(y_0,z_0,a), q(y_0,v_0,a)$ and $p(y_0,v_0,a)$. $\mathcal{T}$ is represented on the left of Figure 4. Let A be this set of atoms. First, note that A is not a core: indeed it is equivalent to its strict

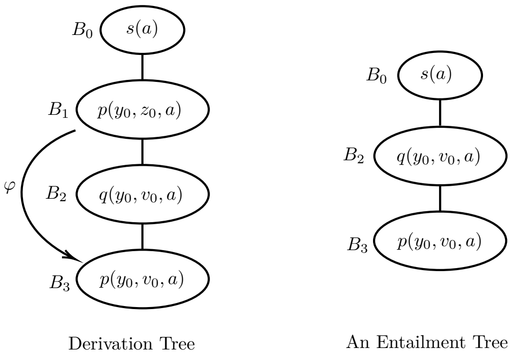

Figure 4 Derivation tree and entailment tree for Example 27

subset A' defined by $\{B_0, B_2, B_3\}$ with a homomorphism π that maps $atom(B_1)$ to $atom(B_3)$. Trivially, A' is a core since it does not contain two atoms with the same predicate. Second, note that any further rule application on $\mathcal{T}$ is redundant, i.e., generates a set of atoms equivalent to A (and A'). Hence, A' is a complete core. However, there is no derivation tree that corresponds to A'. There is even no *prefix* of a derivation tree that corresponds to it (which ruins the alternative idea of building a prefix of a derivation tree that would be associated with a complete core). In particular, note that $\{B_0, B_1, B_2\}$ is indeed a core, but it is not complete.

The following notion of *twins* is useful to define entailment trees (more precisely, to ensure that they have bounded arity).

▶ **Definition 28.** *Two bags B and B' of a tree of bags are* twins *if they have the same sharing type, the same parent B_p and if the natural mapping $\varphi_{\text{atom}(B)\to\text{atom}(B')}$ is the identity on the terms of $\text{atom}(B_p)$.*

In the following definition of entailment tree, we use the notation $\alpha_1 \to \alpha_2$, where α_i is an atom, to denote the rule $\forall X(\alpha_1 \to \exists Y\ \alpha_2)$ with $X = vars(\alpha_1)$ and $Y = vars(\alpha_2) \setminus X$.

▶ **Definition 29 (Entailment Tree).** *An* entailment tree *associated with α and Σ is a tree of bags $\mathcal{T}$ such that:*

1. *B_r, the root of $\mathcal{T}$, is such that $\Sigma \models \alpha \to atom(B_r)$ and $\Sigma \models atom(B_r) \to \alpha$;*
2. *For any bag B_c child of a node B, the following holds:* (i) *$terms(B_c) \cap generated(B) \neq \emptyset$* (ii) *The terms in $generated(B_c)$ are variables that do not occur outside the subtree of $\mathcal{T}$ rooted in B_c* (iii) *$\Sigma \models atom(B) \to atom(B_c)$.*
3. *There is no pair of twins.*

If α is a ground atom, then B_r is simply labeled by α. Otherwise, it may happen that α does not belong to the result of the core chase (e.g., $\alpha = p(x, y)$, with x and y variables, and $\Sigma = \{p(x, y) \to p(x, x)\}$), hence Point 1.

First note that an entailment tree is independent from any derivation (in particular, fairness is not an issue). The main difference with a derivation tree is that it employs a more general parent-child relationship, that relies on entailment rather than on rule application, hence the name entailment tree. Intuitively, with respect to a derivation tree, one is allowed to move a bag B higher in the tree, provided that it contains at least one term generated in its new parent B_p; then, the terms of B that are not shared with B_p are freshly renamed. Finally, since the problem of whether an atom is entailed by a linear existential rule knowledge base is decidable (precisely PSPACE-complete [5], even in the case of atomic instances), one can actually generate all non-twin children of a bag. As a bag can only have a bounded number of non-twin children, we are guaranteed to keep a tree of bounded arity.

Derivation trees are entailment trees, but not necessarily conversely. A crucial distinction between these two structures is the following statement, which does not hold for derivation trees, as illustrated by Example 27.

▶ **Proposition 30.** *If the core chase associated with α and Σ is finite, then there exists an entailment tree $\mathcal{T}$ such that the set of atoms associated with $\mathcal{T}$ is a complete core.*

Example 27 (cont'd). The tree defined by the path of bags B_0, B_2, B_3 is an entailment tree, represented on the right of Figure 4, which defines a complete core.

Differently from the semi-oblivious case, we cannot conclude that the chase does not terminate as soon as a UPW is built, because the associated atoms may later be mapped to other atoms, which would remove the UPW. Instead, starting from the initial bag, we recursively add bags that do not generate a UPW (for instance, we can recursively add all such non-twin children to a leaf). Once the process terminates (the non-twin condition and the absence of UPW ensure that it does), we check that the obtained set of atoms C is complete (i.e., is a model of the KB): for that, it suffices to perform each possible rule application on C and check if the resulting set of atoms is equivalent to C (i.e., maps by homomorphism to C). See Algorithm 1. The set C may not be a core, but it is complete iff it contains a complete core.

We now focus on the key properties of entailment trees associated with complete cores. We first introduce the notion of *redundant bags*, which captures some cases of bags that cannot appear in a finite core. As witnessed by Example 27, this is not a characterization: B_1 is not redundant (according to Definition 31 below), but cannot belong to a complete core.

▶ **Definition 31 (Redundancy).** *Given an entailment tree, a bag B_c child of B is* redundant *if there exists an atom β (that may not belong to the tree) with* (i) *$\Sigma \models atom(B) \rightarrow \beta$;* (ii) *there is a homomorphism from $atom(B_c)$ to β that is the identity on $shared(B_c)$;* (iii) *$|terms(\beta) \setminus terms(B)| < |terms(B_c) \setminus terms(B)|$.*

Note that B_c may be redundant even if the "cause" for redundancy, i.e., β, is not in the tree yet. The role of this notion in the proofs is as follows: we show that if a complete entailment tree contains a UPW then it contains a redundant bag, and that a complete core cannot contain a redundant bag, hence a UPW. To prove this, we rely on Proposition 32 below, which is the counterpart for entailment trees of Proposition 13: performing a bag copy from an entailment tree results in an entailment tree (the notion of prefix is not needed, since a prefix of an entailment tree is an entailment tree) and keeps the properties of the copied bag.

▶ **Proposition 32.** *Let B be a bag of an entailment tree $\mathcal{T}$ and B' be a bag of an entailment tree $\mathcal{T}'$ such that $B \equiv_{st} B'$. Let B_c be a child of B. Let B'_c be a copy of B_c under B'. Then: (a) it holds that $B_c \equiv_{st} B'_c$,* (b) *the tree obtained from $\mathcal{T}$ by adding B'_c under B', if no copy of B_c already exists under B', is an entailment tree, and* (c) *B'_c is redundant if and only if B_c is redundant.*

In light of this, the copy of a bag can be naturally extended to the copy of the whole subtree rooted in a bag, which is a crucial element in the proof of Proposition 33 below.

▶ **Proposition 33.** *A complete core contains neither* (i) *a redundant bag, nor* (ii) *an unbounded-path witness.*

From Propositions 30 and 33, we conclude that Algorithm 1 decides the all-sequence termination problem for the core chase.

▶ **Corollary 34.** *The all-sequence termination problem for the core chase on linear rules is decidable.*

Algorithm 1: Deciding core chase termination

Input : A set of linear rules
Output : `true` if and only if the core chase terminates on all instances

for *each canonical atom* α **do**
  Let $\mathcal{T}$ be the entailment tree restricted to a single root bag labeled by α;
  while *a bag B can be added to $\mathcal{T}$ without creating twins nor a UPW* **do**
    add B to $\mathcal{T}$
  for (σ, π) *such that σ is applicable to atoms$(\mathcal{T})$ by π* **do**
    if *atoms*$(\mathcal{T}) \not\models$ *atoms*$(\mathcal{T}) \cup \pi^s(\text{head}(\sigma))$; `// homomorphism check`
    then
      return `false`
return `true`

A rough complexity analysis of this algorithm yields a 2ExpTime upper bound for the termination problem. Indeed, the exponential number of (sharing) types yields a bound on the number of canonical instances to be checked, the arity of the tree, as well as the length of a path without UPW in the tree, and each edge can be generated with a call to a PSpace oracle.

Finally, note that for predicates of arity at most two, it would still be possible to rely on derivation trees (instead of entailment trees), provided that the initial instance is ground. Indeed, in this specific case, the core of a (finite) derivation tree is a prefix of this tree (note that Example 27 uses predicates of arity three). Then, we can incrementally build a prefix of derivation tree until no bag can be added without creating a UPW, and, as in the higher arity case, check if the built tree is complete.

6 Concluding Remarks

We have shown the decidability of all-instance chase termination over atomic-head linear rules for three main chase variants (semi-oblivious, restricted, core) following a novel approach based on derivation trees, and their generalization to entailment trees, and a single notion of forbidden pattern. As far as we know, these are the first decidability results for the restricted chase, on both versions of the termination problem (i.e., *all-sequence* and *one-sequence* termination). The simplicity of the structures and algorithms makes it realistic to implement them.

We leave for future work the study of the precise complexity of the termination problems. A straightforward analysis of the complexity of the algorithms that decide the termination of the restricted and core chases yields upper bounds, however we believe that a finer analysis of the properties of sharing types would provide tighter upper bounds. It is an open question whether our results can be extended to more complex classes of existential rules, i.e., linear rules with a complex head, which is relevant for the termination of the restricted and core chases, and more expressive classes from the guarded family. Concerning the extension to complex-head rules, the difficulty is that an infinite restricted derivation may not be transformable into a fair restricted derivation. Concerning the extension to more expressive classes, derivation trees were precisely defined to represent derivations with guarded rules and their extensions (i.e., greedy bounded treewidth sets), hence they seem to be a promising tool to study chase termination on that family, at least for the one-instance version of the problem, i.e., given a knowledge base. To consider the all-instance termination problem, a preliminary issue would be whether a finite set of canonical instances can be defined.

Acknowledgements

We thank the reviewers for their insightful comments, as well as Antoine Amarilli for Example 21, which is simpler than a previous example and shows that the fairness issue already occurs with binary precidates.

References

1 Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat. On rules with existential variables: Walking the decidability line. *Artif. Intell.*, 175(9-10):1620–1654, 2011. URL: https://doi.org/10.1016/j.artint.2011.03.002, doi:10.1016/j.artint.2011.03.002.

2 Jean-François Baget, Marie-Laure Mugnier, Sebastian Rudolph, and Michaël Thomazo. Walking the complexity lines for generalized guarded existential rules. In Toby Walsh, editor, *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 712–717. IJCAI/AAAI, 2011. URL: https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-126, doi:10.5591/978-1-57735-516-8/IJCAI11-126.

3 Catriel Beeri and Moshe Y. Vardi. The implication problem for data dependencies. In Shimon Even and Oded Kariv, editors, *Automata, Languages and Programming, 8th Colloquium, Acre (Akko), Israel, July 13-17, 1981, Proceedings*, volume 115 of *Lecture Notes in Computer Science*, pages 73–85. Springer, 1981. URL: https://doi.org/10.1007/3-540-10843-2_7, doi:10.1007/3-540-10843-2_7.

4 Marco Calautti, Georg Gottlob, and Andreas Pieris. Chase termination for guarded existential rules. In Tova Milo and Diego Calvanese, editors, *Proceedings of the 34th ACM Symposium on Principles of Database Systems, PODS 2015, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 91–103. ACM, 2015. URL: http://doi.acm.org/10.1145/2745754.2745773, doi:10.1145/2745754.2745773.

5 Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. Datalog extensions for tractable query answering over ontologies. In Roberto De Virgilio, Fausto Giunchiglia, and Letizia Tanca, editors, *Semantic Web Information Management - A Model-Based Perspective*, pages 249–279. Springer, 2009. URL: https://doi.org/10.1007/978-3-642-04329-1_12, doi:10.1007/978-3-642-04329-1_12.

6 Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general datalog-based framework for tractable query answering over ontologies. *J. Web Sem.*, 14:57–83, 2012. URL: https://doi.org/10.1016/j.websem.2012.03.001, doi:10.1016/j.websem.2012.03.001.

7 Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007. URL: https://doi.org/10.1007/s10817-007-9078-x, doi:10.1007/s10817-007-9078-x.

8 David Carral, Irina Dragoste, and Markus Krötzsch. Detecting chase (non)termination for existential rules with disjunctions. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 922–928. ijcai.org, 2017. URL: `https://doi.org/10.24963/ijcai.2017/128`, `doi:10.24963/ijcai.2017/128`.

9 Alin Deutsch, Alan Nash, and Jeffrey B. Remmel. The chase revisited. In Maurizio Lenzerini and Domenico Lembo, editors, *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008, June 9-11, 2008, Vancouver, BC, Canada*, pages 149–158. ACM, 2008. URL: `http://doi.acm.org/10.1145/1376916.1376938`, `doi:10.1145/1376916.1376938`.

10 Ronald Fagin. A normal form for relational databases that is based on domians and keys. *ACM Trans. Database Syst.*, 6(3):387–415, 1981. URL: `http://doi.acm.org/10.1145/319587.319592`, `doi:10.1145/319587.319592`.

11 Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005. URL: `https://doi.org/10.1016/j.tcs.2004.10.033`, `doi:10.1016/j.tcs.2004.10.033`.

12 Tomasz Gogacz and Jerzy Marcinkowski. All-instances termination of chase is undecidable. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 293–304. Springer, 2014. URL: `https://doi.org/10.1007/978-3-662-43951-7_25`, `doi:10.1007/978-3-662-43951-7\_25`.

13 Georg Gottlob, Giorgio Orsi, and Andreas Pieris. Query rewriting and optimization for ontological databases. *ACM Trans. Database Syst.*, 39(3):25:1–25:46, 2014. URL: `https://doi.org/10.1145/2638546`, `doi:10.1145/2638546`.

14 Gösta Grahne and Adrian Onet. Anatomy of the chase. *Fundam. Inform.*, 157(3):221–270, 2018. URL: `https://doi.org/10.3233/FI-2018-1627`, `doi:10.3233/FI-2018-1627`.

15 Bernardo Cuenca Grau, Ian Horrocks, Markus Krötzsch, Clemens Kupke, Despoina Magka, Boris Motik, and Zhe Wang. Acyclicity notions for existential rules and their application to query answering in ontologies. *J. Artif. Intell. Res.*, 47:741–808, 2013. URL: `https://doi.org/10.1613/jair.3949`, `doi:10.1613/jair.3949`.

16 Alon Y. Halevy. Answering queries using views: A survey. *VLDB J.*, 10(4):270–294, 2001. URL: `https://doi.org/10.1007/s007780100054`, `doi:10.1007/s007780100054`.

17 André Hernich. Computing universal models under guarded TGDs. In Alin Deutsch, editor, *15th International Conference on Database Theory, ICDT '12, Berlin, Germany, March 26-29, 2012*, pages 222–235. ACM, 2012. URL: `http://doi.acm.org/10.1145/2274576.2274600`, `doi:10.1145/2274576.2274600`.

18 Mélanie König, Michel Leclère, Marie-Laure Mugnier, and Michaël Thomazo. Sound, complete and minimal ucq-rewriting for existential rules. *Semantic Web*, 6(5):451–475, 2015. URL: `https://doi.org/10.3233/SW-140153`, `doi:10.3233/SW-140153`.

19 Michel Leclère, Marie-Laure Mugnier, Michaël Thomazo, and Federico Ulliana. A single approach to decide chase termination on linear existential rules. *CoRR*, abs/1602.05828, 2018. URL: `https://arxiv.org/abs/1810.02132`, `arXiv:1810.02132`.

20 Michel Leclère, Marie-Laure Mugnier, Michaël Thomazo, and Federico Ulliana. A single approach to decide chase termination on linear existential rules. In Magdalena Ortiz and Thomas Schneider, editors, *Proceedings of the 31st International Workshop on Description Logics co-located with 16th International Conference on Principles of Knowledge Representation and Reasoning (KR 2018), Tempe, Arizona, US, October 27th - to - 29th, 2018.*, volume 2211 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018. URL: `http://ceur-ws.org/Vol-2211/paper-45.pdf`.

21 Maurizio Lenzerini. Data integration: A theoretical perspective. In Lucian Popa, Serge Abiteboul, and Phokion G. Kolaitis, editors, *Proceedings of the Twenty-first ACM SIGACT-*

SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA, pages 233–246. ACM, 2002. URL: `http://doi.acm.org/10.1145/543613.543644`, `doi:10.1145/543613.543644`.

22 Bruno Marnette. Generalized schema-mappings: from termination to tractability. In Jan Paredaens and Jianwen Su, editors, *Proceedings of the Twenty-Eigth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009, June 19 - July 1, 2009, Providence, Rhode Island, USA*, pages 13–22. ACM, 2009. URL: `http://doi.acm.org/10.1145/1559795.1559799`, `doi:10.1145/1559795.1559799`.

23 Marie-Laure Mugnier and Michaël Thomazo. An introduction to ontology-based query answering with existential rules. In *Reasoning Web. Reasoning on the Web in the Big Data Era - 10th International Summer School 2014, Athens, Greece, September 8-13, 2014. Proceedings*, pages 245–278, 2014. URL: `https://doi.org/10.1007/978-3-319-10587-1_6`, `doi:10.1007/978-3-319-10587-1\_6`.

24 Dan Olteanu, Jiewen Huang, and Christoph Koch. SPROUT: lazy vs. eager query plans for tuple-independent probabilistic databases. In Yannis E. Ioannidis, Dik Lun Lee, and Raymond T. Ng, editors, *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*, pages 640–651. IEEE Computer Society, 2009. URL: `https://doi.org/10.1109/ICDE.2009.123`, `doi:10.1109/ICDE.2009.123`.

25 Adrian Onet. *The chase procedure and its applications.* PhD thesis, Concordia University, Canada, 2012. URL: `https://pdfs.semanticscholar.org/6b1b/327a989d3d8e2488f645488063f391391b89.pdf`.

26 Neil Robertson and Paul D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986. URL: `https://doi.org/10.1016/0196-6774(86)90023-4`, `doi:10.1016/0196-6774(86)90023-4`.

27 Swan Rocher. *Querying Existential Rule Knowledge Bases: Decidability and Complexity. (Interrogation de Bases de Connaissances avec Règles Existentielles : Décidabilité et Complexité).* PhD thesis, University of Montpellier, France, 2016. URL: `https://tel.archives-ouvertes.fr/tel-01483770`.

28 Michaël Thomazo. *Conjunctive Query Answering Under Existential Rules - Decidability, Complexity, and Algorithms.* PhD thesis, Montpellier 2 University, France, 2013. URL: `https://tel.archives-ouvertes.fr/tel-00925722`.