Capturing Homomorphism-Closed Decidable Queries with Existential Rules

Camille Bourgaux[1] , **David Carral**[2] , **Markus Krötzsch**[3] , **Sebastian Rudolph**[3] , **Michaël Thomazo**[1]

[1] DI ENS, ENS, CNRS, PSL University & Inria, Paris, France
[2] LIRMM, Inria, University of Montpellier, CNRS, Montpellier, France
[3] Technische Universität Dresden, Dresden, Germany
{camille.bourgaux, david.carral, michael.thomazo}@inria.fr,
{markus.kroetzsch, sebastian.rudolph}@tu-dresden.de

Abstract

Existential rules are a very popular ontology-mediated query language for which the chase represents a generic computational approach for query answering. It is straightforward that existential rule queries exhibiting chase termination are decidable and can only recognize properties that are preserved under homomorphisms. In this paper, we show the converse: every decidable query that is closed under homomorphism can be expressed by an existential rule set for which the standard chase universally terminates. Membership in this fragment is not decidable, but we show via a diagonalisation argument that this is unavoidable.

1 Introduction

At the core of contemporary logic-based knowledge representation is the concept of *querying* data sources, often using elaborate query formalisms that allow for taking background knowledge into account. The classical decision problem related to such knowledge-aware querying is *Boolean query entailment*. From an abstract point of view, a Boolean query identifies a class of *databases* $\mathcal{D}$ – those that satisfy the query, i.e., to which the query "matches". This view allows us to define and investigate properties of (abstract) queries independently from the syntax used to specify them. Such properties can be structural (morphisms, closure properties) or computational (decidability, complexity).

A very popular querying formalism are *existential rules*, also referred to as *tuple-generating dependencies*. It is straightforward that the class of databases satisfying some existential rule query is closed under homomorphisms and recursively enumerable. Conversely, it was established that *every* homomorphism-closed query that is recursively enumerable can be expressed using existential rules (Rudolph and Thomazo 2015). That is, plain existential rules already realize their full potential; further syntactic extensions within these boundaries do not enhance expressivity.

For questions related to automated deduction, however, a more restricted requirement than recursive enumerability is of central interest: decidability. Therefore, the crucial question we tackle in this paper is:

Can we characterize an existential rules fragment capable of expressing every *decidable homomorphism-closed query?*

The generic computational paradigm for existential rules, the *chase* (Beeri and Vardi 1984), is based on repetitive, forward-chaining rule application, starting from the database. As this may cause the iterated introduction of new domain elements, this procedure is not guaranteed to terminate – yet, termination is a crucial criterion for decidability. The chase comes in several variants, mainly differing in their (increasingly thorough) mechanisms to prevent unnecessary rule applications: While the *Skolem* chase (Marnette 2009) essentially just avoids duplicate rule applications, the *standard* (Fagin et al. 2005) and the *core* chase (Deutsch, Nash, and Remmel 2008) check for redundancy on a local and global level, respectively.

The class of existential rule sets with terminating[1] Skolem chase has already been weighed and found wanting: it only comprises those queries that are already expressible in plain Datalog – and hence can be evaluated in polynomial time (Marnette 2009; Krötzsch and Rudolph 2011; Zhang, Zhang, and You 2015). For the standard-chase-terminating and the core-chase-terminating existential rules classes, on the other hand, we only know that the former is contained in the latter (Grahne and Onet 2018), but little more than that (Krötzsch, Marx, and Rudolph 2019). In this paper, we clarify the situation significantly by showing the following:

Standard-chase-terminating existential rules capture the class of all decidable homomorphism-closed queries.

Notably, this implies that standard-chase-terminating and core-chase-terminating existential rule queries are equally expressive and no decidable enhancement of this formalism that preserves homomorphism-closedness (e.g. by allowing disjunction in rule heads) can be strictly more expressive.

As a downside, the existential rules fragment thus identified is not even semi-decidable, but we show via a diagonalisation argument that this downside is, in fact, unavoidable.

Additional proofs and details are given in the appendix.

2 Preliminaries

Rules We consider first-order formulas over countably infinite sets Vars of variables and Preds of *predicates*, where each $\mathtt{p} \in \mathsf{Preds}$ has an arity $Ar(\mathtt{p}) \geq 0$. Lists of variables are denoted $\vec{x} = x_1, \ldots, x_k$ and will be treated like sets when order is not relevant. An *atom* is an expression $\mathtt{p}(\vec{x})$ with $\mathtt{p} \in \mathsf{Preds}$ and $|\vec{x}| = Ar(\mathtt{p})$.

[1]We always mean universal termination, i.e., for every database.

The fragment of *disjunctive existential rules* consists of formulae of the form:

$$\forall \vec{x}. \Big(\beta[\vec{x}] \to \bigvee_{i=1}^{k} \exists \vec{y}_i . \eta_i[\vec{x}_i, \vec{y}_i] \Big), \tag{1}$$

where $\beta[\vec{x}]$ and $\eta_i[\vec{x}_i, \vec{y}_i]$ ($i = 1, \ldots, k$) are conjunctions of atoms with variables $\vec{x}$ and $\vec{x}_i \cup \vec{y}_i$, respectively. We call β *body* and $\bigvee_{i=1}^{k} \exists \vec{y}_i . \eta_i$ *head*. Bodies can be empty (we then omit $\to$), but heads must be non-empty. We require that $\vec{x}$ and $\vec{y}_i$ ($i = 1, \ldots, k$) are mutually disjoint and that $\vec{x}_i \subseteq \vec{x}$ for all $i = 1, \ldots, k$. We single out the fragment of *existential rules* by disallowing disjunction, i.e. requiring $k = 1$, and *Datalog rules* by disallowing existential quantifiers. We often omit the universal quantifiers from rules and treat conjunctions of atoms as sets of atoms.

Databases, Interpretations, and Entailment The semantics of formulas is based on logical interpretations, which we define as relational structures over a countably infinite set Nulls of *nulls*. A *schema* $\mathcal{S}$ is a finite set of predicates. An *interpretation* $\mathcal{I}$ over schema $\mathcal{S}$ is a set of expressions $\mathtt{p}(\vec{n})$ with $\mathtt{p} \in \mathcal{S}$ and $\vec{n}$ a list of nulls of length $Ar(\mathtt{p})$. We write $\mathsf{Nulls}(\mathcal{I})$ for the set of nulls in $\mathcal{I}$. A *database* is a finite interpretation. See also the remarks on this notation below.

A *homomorphism* $h : \mathcal{I}_1 \to \mathcal{I}_2$ between interpretations $\mathcal{I}_1$ and $\mathcal{I}_2$ is a mapping h from the nulls in $\mathcal{I}_1$ to the nulls in $\mathcal{I}_2$, such that $\mathtt{p}(\vec{n}) \in \mathcal{I}_1$ implies $\mathtt{p}(h(\vec{n})) \in \mathcal{I}_2$, where $h(\vec{n})$ denotes the list of h-values over $\vec{n}$. We also write $h(\mathcal{I})$ for $\{\mathtt{p}(h(\vec{n})) \mid \mathtt{p}(\vec{n}) \in \mathcal{I}\}$.

A *substitution* σ is a mapping from variables to nulls, which we extend to lists of variables and formulas as usual. A rule ρ as in (1) is *satisfied* by interpretation $\mathcal{I}$ if every substitution $\sigma : \vec{x} \to \mathsf{Nulls}$ with $\sigma(\beta) \subseteq \mathcal{I}$ can be extended to a substitution $\sigma' : \vec{x} \cup \vec{y}_i \to \mathsf{Nulls}$ for some $i \in \{1, \ldots, k\}$ such that $\sigma'(\eta_i) \subseteq \mathcal{I}$. Otherwise, if $\sigma(\beta) \subseteq \mathcal{I}$ but no extension σ' of σ verifies $\sigma'(\eta_i) \subseteq \mathcal{I}$ for some $i \in \{1, \ldots, k\}$, then $\langle \rho, \sigma \rangle$ is *applicable* to $\mathcal{I}$.

$\mathcal{I}$ *satisfies* a set Σ of rules if it satisfies every rule in Σ. An interpretation $\mathcal{J}$ is *satisfied* by an interpretation $\mathcal{I}$ if there is a homomorphism $h : \mathcal{J} \to \mathcal{I}$. $\mathcal{I}$ is a *model* of a rule/rule set/interpretation/database $\mathcal{X}$ if $\mathcal{X}$ is satisfied by $\mathcal{I}$, written $\mathcal{I} \models \mathcal{X}$. As usual, we also write $\mathcal{X} \models \mathcal{Y}$ ("$\mathcal{X}$ entails $\mathcal{Y}$") if every model of $\mathcal{X}$ is a model of $\mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ might be rules, rule sets, databases, or lists of several such elements. Note that the semantics of a database $\mathcal{D}$ in this context corresponds to the semantics of a *Boolean conjunctive query* $\exists \vec{x}. \bigwedge \{\mathtt{p}(x_{n_1}, \ldots, x_{n_\ell}) \mid \mathtt{p}(n_1, \ldots, n_\ell) \in \mathcal{D}\}$ – we will therefore not introduce such queries as a separate notion. Also note that entailment and satisfaction between interpretations/databases coincide.

Abstract Queries, Expressivity, and Decidability An *(abstract) query* $\mathfrak{Q}$ over a schema $\mathcal{S}$ is a set of databases over $\mathcal{S}$ that is *closed under isomorphism*, i.e., such that whenever $\mathcal{D} \in \mathfrak{Q}$ and $\mathcal{D}'$ is obtained from $\mathcal{D}$ by bijective renaming of nulls, then $\mathcal{D}' \in \mathfrak{Q}$. $\mathfrak{Q}$ is further *closed under homomorphisms* if, for all $\mathcal{D} \in \mathfrak{Q}$ and homomorphisms $h : \mathcal{D} \to \mathcal{D}'$, we have $\mathcal{D}' \in \mathfrak{Q}$.

Definition 1. Let $\mathtt{Goal}$ be a nullary predicate. A query $\mathfrak{Q}$ over $\mathcal{S}$ is *expressed by* a set Σ of rules if, for every database $\mathcal{D}$ over $\mathcal{S}$, we have $\mathcal{D} \in \mathfrak{Q}$ if and only if $\Sigma, \mathcal{D} \models \mathtt{Goal}$.

To discuss decidability of queries, we need to conceive databases as Turing machine inputs over a fixed alphabet. A *serialisation* for a schema $\mathcal{S}$ is a word $s \in (\{0, 1, \|\} \cup \mathcal{S})^*$ of the form $e_1 \cdots e_n$ where $n \geq 0$ and $e_i = \mathtt{p}_i \| w_{i1} \| \cdots \| w_{iAr(\mathtt{p}_i)} \|$ for $w_{ij} \in \{0, 1\}^+$ and $\mathtt{p}_i \in \mathcal{S}$. Given s of this form and an injection $\eta : \{0, 1\}^+ \to \mathsf{Nulls}$, let $\eta(s)$ denote the database $\{\mathtt{p}_i(\eta(w_{i1}), \ldots, \eta(w_{iAr(\mathtt{p}_i)})) \mid 1 \leq i \leq n\}$. Then s *corresponds to* a database $\mathcal{D}$ if $\eta(s)$ is isomorphic to $\mathcal{D}$; note that this does not depend on the choice of η.

A query $\mathfrak{Q}$ with schema $\mathcal{S}$ is *decidable* if the set of all serialisations for $\mathcal{S}$ that correspond to some $\mathcal{D} \in \mathfrak{Q}$ is a decidable language.

Remarks on our Notation Many works consider constants to appear in databases (not just nulls), but complexity and expressivity is usually studied for queries that are closed under isomorphisms, a.k.a. *generic* (Abiteboul, Hull, and Vianu 1995, Ch. 16), and nulls are more natural there. One can admit finitely many exceptions (elements that must not be renamed), but such "constants" can be simulated by marking them with dedicated unary predicates.

Specifying logical interpretations as sets of "atoms" that may use nulls is a notational convenience with some side effects: our interpretations cannot contain elements that do not stand in any relation, but they can have an empty domain. Both aspects do not change the notion of logical entailment *on the formulas we consider*.

Universal Models and the Chase Entailment of databases (corresponding to Boolean conjunctive queries) can be decided by considering only a subset of all models. Given sets $\mathfrak{I}$ and $\mathfrak{K}$ of interpretations, $\mathfrak{I}$ is *universal* for $\mathfrak{K}$ if, for all $\mathcal{K} \in \mathfrak{K}$, there is $\mathcal{I} \in \mathfrak{I}$ and a homomorphism $\mathcal{I} \to \mathcal{K}$. Consider a rule set Σ and database $\mathcal{D}$, and let $\mathfrak{M}$ be the set of all models of $\Sigma, \mathcal{D}$. Then $\mathfrak{I}$ is a *universal model set* for Σ and $\mathcal{D}$ if $\mathfrak{I} \subseteq \mathfrak{M}$ and $\mathfrak{I}$ is universal for $\mathfrak{M}$.

Fact 1. *If $\mathfrak{I}$ is a universal model set for Σ and $\mathcal{D}$ then, for every database $\mathcal{C}$, we have $\Sigma, \mathcal{D} \models \mathcal{C}$ iff $\mathcal{I} \models \mathcal{C}$ for all $\mathcal{I} \in \mathfrak{I}$.*

Universal model sets can be computed with the *chase* algorithm. Here, we consider a variation of the *standard* (or *restricted*) chase for rules with disjunction, introduced by (Carral, Dragoste, and Krötzsch 2017).

Definition 2. A *chase tree* for a rule set Σ and database $\mathcal{D}$ is a (finite or infinite) tree where each node is labelled by a database, such that:

1. The root is labelled with $\mathcal{D}$.
2. For every node with label $\mathcal{E}$ that has ℓ children labelled $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$, there is a rule $\rho \in \Sigma$ of the form (1) and a substitution $\sigma : \vec{x} \to \mathsf{Nulls}$ such that (i) $\langle \rho, \sigma \rangle$ is applicable to $\mathcal{E}$, (ii) ρ has $k = \ell$ head disjuncts, and (iii) $\mathcal{C}_i = \mathcal{E} \cup \sigma_i(\eta_i)$ where σ_i extends σ by mapping each variable $y \in \vec{y}_i$ to a fresh null.

3. For each rule $\rho \in \Sigma$ and each substitution σ, there is $i \geq 1$ such that $\langle \rho, \sigma \rangle$ is not applicable to the label of any node of depth $\geq i$.

The *result* that corresponds to a chase tree is the set of all interpretations that can be obtained as the union of all interpretations along a path in the tree.

Condition (3) ensures fair, exhaustive rule application, but different orders of application can lead to different chase trees, which can also have different results. Nevertheless, every result is semantically correct in the following sense:

Fact 2. *Every result of a chase on a rule set Σ and database $\mathcal{D}$ is a universal model set for Σ and $\mathcal{D}$.*

The pair $\langle \Sigma, \mathcal{D} \rangle$ is *chase-terminating* if all its chase trees are finite – by König's Lemma, this is equivalent to all chase results for $\langle \Sigma, \mathcal{D} \rangle$ containing only finite interpretations; this corresponds to *all-strategy termination*. Σ is *chase-terminating* if $\langle \Sigma, \mathcal{D} \rangle$ is chase-terminating for every database $\mathcal{D}$; this corresponds to *universal termination*.

Turing Machines We will use (deterministic) Turing machines (TM), denoted as a tuple $M = \langle Q, \Gamma, \delta \rangle$, with states Q, tape alphabet Γ with blank $\llcorner\!\lrcorner \in \Gamma$, and transition function δ. M has a distinguished initial state $q_S \in Q$, and accepting and rejecting halting states $q_A, q_R \in Q$. For all states $q \in Q \setminus \{q_A, q_R\}$ and tape symbols $a \in \Gamma$, there is exactly one transition $(q, a) \mapsto (r, b, D) \in \delta$. We assume that TM tapes are unbounded to the right but bounded to the left, and that TMs will never attempt to move left on the first position of the tape (this is w.l.o.g., since one can modify any TM to insert a marker at the tape start to recognise this case).

3 On the Expressivity of Disjunctive Rules

In this section, we show how to express a homomorphism-closed, decidable query with a disjunctive rule set. This construction will be the basis for finding a chase-terminating set of (deterministic) existential rules. Throughout this section, $\mathfrak{Q}$ is a fixed but arbitrary homomorphism-closed query over signature $\mathcal{S}$, and $M = \langle Q, \Gamma, \delta \rangle$ is a TM that decides $\mathfrak{Q}$.

To express $\mathfrak{Q}$, we specify five rule sets $\mathcal{R}_1 \subseteq \mathcal{R}_2 \subseteq \mathcal{R}_3 \subseteq \mathcal{R}_4 \subseteq \mathcal{R}_5$ (see Figures 1–6), which will be explained later. We want to show the following result:

Theorem 3. *The set $\mathcal{R}_5$ of disjunctive existential rules expresses the query $\mathfrak{Q}$.*

To show this, we fix an arbitrary database $\mathcal{D}$ over $\mathcal{S}$. Theorem 3 then follows from Fact 1 and the next lemma:

Lemma 4. *There is a universal model set $\mathfrak{M}$ of $\mathcal{R}_5$ and $\mathcal{D}$ such that $\mathcal{D} \in \mathfrak{Q}$ iff $\mathtt{Goal} \in \mathcal{I}$ for every $\mathcal{I} \in \mathfrak{M}$.*

The universal model set $\mathfrak{M}$ is a complicated structure that we describe step by step, by specifying five sets of interpretations – $\mathfrak{I}_1$, $\mathfrak{I}_2$, $\mathfrak{I}_3$, $\mathfrak{I}_4$, and $\mathfrak{I}_5$ –, such that (1) $\mathfrak{I}_i$ is a universal model set of $\mathcal{R}_i, \mathcal{D}$ for each $1 \leq i \leq 5$, (2) $|\mathfrak{I}_1| = |\mathfrak{I}_2| = |\mathfrak{I}_3| = |\mathfrak{I}_4| = |\mathfrak{I}_5|$, and (3) for each $1 \leq i < j \leq 5$ and each $\mathcal{I} \in \mathfrak{I}_i$, there is exactly one $\mathcal{J} \in \mathfrak{I}_j$ with $\mathcal{I} \subseteq \mathcal{J}$. Lemma 4 can then be shown using $\mathfrak{M} = \mathfrak{I}_5$.

Before dwelling into the details of each rule set and universal model, we give an overview of the construction: $\mathcal{R}_1$ constructs all possible linear orders over the nulls in $\mathcal{D}$, as well as all possible completions of $\mathcal{D}$ with facts built using these nulls; $\mathcal{R}_2 \setminus \mathcal{R}_1$ extracts successor relations from the linear orders; $\mathcal{R}_3 \setminus \mathcal{R}_2$ associates to nulls representations of their positions in successor relations; $\mathcal{R}_4 \setminus \mathcal{R}_3$ encodes all initial TM configurations corresponding to some linear order and completion, and $\mathcal{R}_5 \setminus \mathcal{R}_4$ simulates the run of the TM on these configurations.

$$\to \exists y.\mathtt{First}(y) \wedge \mathtt{DbDom}(y) \quad (2)$$
$$\to \exists z.\mathtt{Last}(z) \wedge \mathtt{DbDom}(z) \quad (3)$$
$$\mathtt{p}(\vec{x}) \to \mathtt{In}_\mathtt{p}(\vec{x}) \wedge \textstyle\bigwedge_{x \in \vec{x}} \mathtt{DbDom}(x) \quad (4)$$
$$\mathtt{DbDom}(x) \to \mathtt{Eq}(x, x) \quad (5)$$
$$\mathtt{Eq}(x, y) \to \mathtt{Eq}(y, x) \quad (6)$$
$$\mathtt{NEq}(x, y) \to \mathtt{NEq}(y, x) \quad (7)$$
$$\mathtt{R}(\vec{x}) \wedge \mathtt{Eq}(x_i, y) \to \mathtt{R}(\vec{x}_{x_i \mapsto y}) \quad (8)$$
$$\mathtt{DbDom}(x) \wedge \mathtt{DbDom}(y) \to \mathtt{Eq}(x, y) \vee \mathtt{NEq}(x, y) \quad (9)$$
$$\mathtt{LT}(x, y) \wedge \mathtt{LT}(y, z) \to \mathtt{LT}(x, z) \quad (10)$$
$$\mathtt{First}(x) \wedge \mathtt{NEq}(x, y) \to \mathtt{LT}(x, y) \quad (11)$$
$$\mathtt{NEq}(x, y) \wedge \mathtt{Last}(y) \to \mathtt{LT}(x, y) \quad (12)$$
$$\mathtt{NEq}(x, y) \to \mathtt{LT}(x, y) \vee \mathtt{LT}(y, x) \quad (13)$$
$$\textstyle\bigwedge_{x \in \vec{x}} \mathtt{DbDom}(x) \to \mathtt{In}_\mathtt{p}(\vec{x}) \vee \mathtt{NIn}_\mathtt{p}(\vec{x}) \quad (14)$$

Figure 1: The rule set $\mathcal{R}_1$, where rules (4) and (14) are instantiated for each $\mathtt{p} \in \mathcal{S}$, and rules (8) are instantiated for each $\mathtt{R} \in \{\mathtt{First}, \mathtt{Last}, \mathtt{Eq}, \mathtt{NEq}, \mathtt{LT}\} \cup \{\mathtt{In}_\mathtt{p}, \mathtt{NIn}_\mathtt{p} \mid \mathtt{p} \in \mathcal{S}\}$ and $1 \leq i \leq Ar(\mathtt{R})$, and $\vec{x}_{x_i \mapsto y}$ denotes $\vec{x}$ with x_i replaced by y.

$$\mathtt{First}(x) \to \exists u.\mathtt{Root}(u) \wedge \mathtt{Rep}(x, u) \quad (15)$$
$$\mathtt{Rep}(x, v) \wedge \mathtt{LT}(x, z) \to \exists w.\mathtt{Chi}(v, w) \wedge \mathtt{Rep}(z, w) \quad (16)$$
$$\mathtt{Last}(x) \wedge \mathtt{Rep}(x, u) \to \mathtt{Leaf}(u) \quad (17)$$
$$\mathtt{Rep}(x, u) \wedge \mathtt{Eq}(x, y) \to \mathtt{Rep}(y, u) \quad (18)$$
$$\mathtt{In}_\mathtt{p}(\vec{x}) \wedge \textstyle\bigwedge_{i=1}^{|\vec{x}|} \mathtt{Rep}(x_i, u_i) \to \mathtt{In}'_\mathtt{p}(\vec{u}) \quad (19)$$
$$\mathtt{NIn}_\mathtt{p}(\vec{x}) \wedge \textstyle\bigwedge_{i=1}^{|\vec{x}|} \mathtt{Rep}(x_i, u_i) \to \mathtt{NIn}'_\mathtt{p}(\vec{u}) \quad (20)$$

Figure 2: The rule set $\mathcal{R}_2$ contains $\mathcal{R}_1$ (see Figure 1) and all above rules, where (19) and (20) are instantiated for each $\mathtt{p} \in \mathcal{S}$.

$\mathcal{R}_1$: Linear Order and Database Completion $\mathcal{R}_1$ serves two distinct purposes: (1) predicates $\mathtt{First}, \mathtt{Last}, \mathtt{Eq}$ ("$=$"), $\mathtt{NEq}$ ("$\neq$"), and $\mathtt{LT}$ ("$<$") encode representations of possible linear orders over nulls in $\mathcal{D}$ (collected in predicate $\mathtt{DbDom}$); and (2) predicates $\mathtt{In}_\mathtt{p}$ and $\mathtt{NIn}_\mathtt{p}$ for each $\mathtt{p} \in \mathcal{S}$ explicitly encode positive and negative (absent) facts in $\mathcal{D}$. Both purposes require disjunctive reasoning. Possible models include representations of strict, total linear orders (1) and the exact database completion (2), but also models for collapsed orders and inconsistent completions. The latter is not problematic since we consider homomorphism-closed queries.

We define the interpretation set $\mathfrak{I}_1$ on the set $\Delta = \mathsf{Nulls}(\mathcal{D}) \cup \{u_\alpha, u_\omega\}$, for fresh nulls $u_\alpha, u_\omega \notin \mathsf{Nulls}(\mathcal{D})$. A (partially collapsed) linear order can be represented by

$$\mathtt{Root}(u) \to \exists y_1, y_2.\mathtt{Enc}(u, y_1, y_2) \land \mathtt{S}_0(y_1) \land \mathtt{Nxt}(y_1, y_2) \land \mathtt{S}_1(y_2) \tag{21}$$

$$\mathtt{Enc}(u, y_1, y_\_) \land \mathtt{Chi}(u, v) \to \exists z_1, z_\_.\mathtt{Enc}(v, z_1, z_\_) \land \mathtt{Cpy}_{+1}(y_1, y_\_, z_1, z_\_) \tag{22}$$

$$\mathtt{Cpy}_{+1}(y_1, y_2, z_1, z_\_) \land \mathtt{S}_0(y_1) \land \mathtt{Nxt}(y_1, y_2) \to \mathtt{S}_1(z_1) \land \mathtt{Nxt}(z_1, z_\_) \land \mathtt{S}_1(z_\_) \tag{23}$$

$$\mathtt{Cpy}_{+1}(y_1, y_2, z_1, z_\_) \land \mathtt{S}_1(y_1) \land \mathtt{Nxt}(y_1, y_2) \to \exists z_2.\mathtt{S}_0(z_1) \land \mathtt{Nxt}(z_1, z_2) \land \mathtt{S}_0(z_2) \land \mathtt{Nxt}(z_2, z_\_) \land \mathtt{S}_1(z_\_) \tag{24}$$

$$\mathtt{Cpy}_{+1}(y_1, y_\_, z_1, z_\_) \land \mathtt{S}_0(y_1) \land \mathtt{Nxt}(y_1, y_2) \land \mathtt{Nxt}(y_2, y_3) \to \exists z_2.\mathtt{Cpy}(y_2, y_\_, z_2, z_\_) \land \mathtt{S}_1(z_1) \land \mathtt{Nxt}(z_1, z_2) \tag{25}$$

$$\mathtt{Cpy}_{+1}(y_1, y_\_, z_1, z_\_) \land \mathtt{S}_1(y_1) \land \mathtt{Nxt}(y_1, y_2) \land \mathtt{Nxt}(y_2, y_3) \to \exists z_2.\mathtt{Cpy}_{+1}(y_2, y_\_, z_2, z_\_) \land \mathtt{S}_0(z_1) \land \mathtt{Nxt}(z_1, z_2) \tag{26}$$

$$\mathtt{Cpy}(y_1, y_2, z_1, z_2) \land \mathtt{S}_*(y_1) \land \mathtt{Nxt}(y_1, y_2) \to \mathtt{S}_*(z_1) \land \mathtt{Nxt}(z_1, z_2) \land \mathtt{S}_1(z_2) \tag{27}$$

$$\mathtt{Cpy}(y_1, y_\_, z_1, z_\_) \land \mathtt{S}_*(y_1) \land \mathtt{Nxt}(y_1, y_2) \land \mathtt{Nxt}(y_2, y_3) \to \exists z_2.\mathtt{Cpy}(y_2, y_\_, z_2, z_\_) \land \mathtt{S}_*(z_1) \land \mathtt{Nxt}(z_1, z_2) \tag{28}$$

Figure 3: The rule set $\mathcal{R}_3$ contains $\mathcal{R}_2$ (Figure 2) and all of the above rules, where (27) and (28) are instantiated for each $* \in \{0, 1\}$.

an ordered partition $\vec{T} = T_1, \ldots, T_k$ ($k \geq 1$) of Δ where $u_\alpha \in T_1$ and $u_\omega \in T_k$. Let *Ords* be the set of all such $\vec{T}$, and let $Compls(\vec{T})$ be the set of all interpretations with nulls Δ that are set-minimal among the models of $\mathcal{R}_1$ and $\mathcal{D}$, and that contain the database

$$\{\mathtt{First}(u) \mid u \in T_1\} \cup \{\mathtt{Last}(u) \mid u \in T_k\} \cup$$
$$\{\mathtt{Eq}(t, u) \mid 1 \leq i \leq k; t, u \in T_i\} \cup$$
$$\{\mathtt{LT}(t,u), \mathtt{NEq}(t,u), \mathtt{NEq}(u,t) \mid 1 \leq i < j \leq k; t \in T_i; u \in T_j\}.$$

By minimality, each $\mathcal{I} \in Compls(\vec{T})$ contains exactly one of $\{\mathtt{In}_\mathtt{p}(\vec{t}), \mathtt{NIn}_\mathtt{p}(\vec{t})\}$ for every $\vec{t} \subseteq \Delta$ with $|\vec{t}| = Ar(\mathtt{p})$.

Lemma 5. $\mathfrak{I}_1 = \bigcup_{\vec{T} \in Ords} Compls(\vec{T})$ *is a universal model set of* $\mathcal{R}_1$ *and* $\mathcal{D}$.

Proof Sketch. We can construct a chase tree for $\mathcal{R}_1$ and $\mathcal{D}$ that prioritises the application of rules in the order of their appearance in Figure 1. The result of that chase $\mathfrak{K}$ is a finite universal model set (Fact 2) and we can show that every $\mathcal{K} \in \mathfrak{K}$ is isomorphic to a unique $\mathcal{I} \in \mathfrak{I}_1$. $\square$

Each set $\mathfrak{I}_i$ for some $2 \leq i \leq 5$ is obtained as $\mathfrak{I}_i = \{Int_i(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_{i-1}\}$ for a function Int_i as defined below. Every $\mathcal{I} \in \mathfrak{I}_i$ with $1 \leq i \leq 5$ contains a unique interpretation $Seed(\mathcal{I}) \in \mathfrak{I}_1$, and there is a unique ordered partition $Order(\mathcal{I}) \in Ords$ such that $Seed(\mathcal{I}) \in Compls(Order(\mathcal{I}))$.

$\mathcal{R}_2$: Representative Tree The purpose of $\mathcal{R}_2$ is to extract successor relations from the transitive linear order LT. Given an ordered partition $Order(\mathcal{I}) = T_1, \ldots, T_k$ of some model $\mathcal{I}$ of $\mathcal{R}_1$, $\mathcal{R}_2$ constructs a finite tree structure – defined using predicates Root, Chi ("child"), and Leaf – where each path represents a sub-sequence $T_{z_1}, \ldots, T_{z_p}$ of $T_1, \ldots, T_k$ with $z_1 = 1$ and $z_p = k$. Unavoidably, some paths will skip some T_i, but it suffices for our purposes if one path is complete. The elements of any set T_j are related to the tree nodes that represent T_j by a predicate Rep. Moreover, nodes are related via predicates $\mathtt{In}'_\mathtt{p}$ and $\mathtt{NIn}'_\mathtt{p}$ that reflect the relations for $\mathtt{In}_\mathtt{p}$ and $\mathtt{NIn}_\mathtt{p}$ that hold between the represented elements (in the database completion of the considered model $\mathcal{I}$ of $\mathcal{R}_1$).

Given $\mathcal{I}$ with $|Order(\mathcal{I})| = k$ as above, let $\mathbf{Z}$ be the set of all words $z_1 \cdots z_p \in \{1, \ldots, k\}^*$ such that $z_1 = 1$ and $z_i < z_{i+1}$ for all $i \in \{1, \ldots, p-1\}$. Moreover, let $end(z_1 \cdots z_p) := z_p$. Using fresh nulls $\{u_w \mid w \in \mathbf{Z}\}$, we define $Tree(\mathcal{I})$ to be the interpretation

$$\{\mathtt{Root}(u_1)\} \cup \{\mathtt{Chi}(u_w, u_{wz}) \mid wz \in \mathbf{Z}, 2 \leq z \leq k\} \cup$$
$$\{\mathtt{Leaf}(u_w) \mid w \in \mathbf{Z}; end(w) = k\} \cup$$
$$\{\mathtt{Rep}(x, u_w) \mid w \in \mathbf{Z}; x \in T_{end(w)}\}.$$

Now $Int_2(\mathcal{I})$ is the least interpretation that contains $\mathcal{I}$ and $Tree(\mathcal{I})$, and that further satisfies rules (19) and (20). We can extend Lemma 5 as follows. The required universal model set is obtained by any chase that prioritises rule (18).

Lemma 6. $\mathfrak{I}_2 = \{Int_2(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_1\}$ *is a universal model set of* $\mathcal{R}_2$ *and* $\mathcal{D}$.

$\mathcal{R}_3$: Position Binary Encodings The purpose of $\mathcal{R}_3$ is to associate each node in the tree of $\mathcal{R}_2$ with a binary encoding of its distance from the root (the root starts with "distance" 2 for technical reasons). Encodings start at the least significant bit and always end in 1 (i.e., have no leading 0s). To simplify upcoming steps, encodings take the form of little TM tapes, represented by a Nxt-connected chain of nulls with unary predicates $\mathtt{S}_0$ and $\mathtt{S}_1$ encoding the symbol at each position. Nodes u relate to the first and last null t_s and t_e of their "tape" through facts $\mathtt{Enc}(u, t_s, t_e)$. Facts $\mathtt{Cpy}(a_s, a_e, b_s, b_e)$ are used to create a tape between b_s and b_e that contains a copy of the information on the tape between a_s and a_e. Predicate $\mathtt{Cpy}_{+1}$ is analogous, but creates a representation of the successor of the number that is copied.

Consider a model $\mathcal{I}$ of $\mathcal{R}_2$ and define the set of sequences $\mathbf{Z}$ as before. For $w \in \mathbf{Z}$ of length $|w|$, and $b_1 \cdots b_\ell$ the binary representation of $|w| + 1$, let $EncPos(w)$ be the database

$$\{\mathtt{Enc}(u_w, e^1_w, e^\ell_w)\} \cup \{\mathtt{S}_{b_i}(e^i_w) \mid 1 \leq i \leq \ell\} \cup$$
$$\{\mathtt{Nxt}(e^{i-1}_w, e^i_w) \mid 2 \leq i \leq \ell\}.$$

Let $\mathcal{J} = \mathcal{I} \cup \bigcup_{w \in \mathbf{Z}} EncPos(w)$. We define $Int_3(\mathcal{I})$ as the smallest superset of $\mathcal{J}$ that satisfies all rules in $\mathcal{R}_3$ while including only the nulls in $\mathcal{J}$. $Int_3(\mathcal{I})$ extends $\mathcal{J}$ only by missing Cpy and $\mathtt{Cpy}_{+1}$ relations, which can be inferred by slightly rewritten rules. For example, rule (22) is satisfied when applying the following rule to $\mathcal{J}$:

$$\mathtt{Enc}(u, y_1, y_\_) \land \mathtt{Chi}(u, v) \land \mathtt{Enc}(v, z_1, z_\_)$$
$$\to \mathtt{Cpy}_{+1}(y_1, y_\_, z_1, z_\_).$$

$$\mathtt{Ld_p}(u,t,\vec{v}) \wedge \mathtt{In'_p}(\vec{v}) \to \exists \vec{x}, y.\mathtt{S_p}(t) \wedge \mathtt{Nxt}(t,x_1) \wedge \bigwedge_{i=1}^{|\vec{v}|} \mathtt{LdE}(v_i,x_i,x_{i+1}) \wedge \mathtt{Nxt}(x_{|\vec{v}|+1},y) \wedge \mathtt{Rdy_p}(u,y,\vec{v}) \quad (29)$$

$$\mathtt{LdE}(v,x_s,x_e) \wedge \mathtt{Enc}(v,y_1,y_\lrcorner) \to \exists z_1,z_\lrcorner.\mathtt{S_{||}}(x_s) \wedge \mathtt{Nxt}(x_s,z_1) \wedge \mathtt{Cpy}(y_1,y_\lrcorner,z_1,z_\lrcorner) \wedge \mathtt{Nxt}(z_\lrcorner,x_e) \wedge \mathtt{S_{||}}(x_e) \quad (30)$$

Figure 4: The rule set $\mathcal{R}_4$ contains $\mathcal{R}_3$ (see Figure 3), the rules from Figure 5, and the above rules instantiated for all $\mathtt{p} \in \mathcal{S}$.

$$\mathtt{Leaf}(u) \to \exists t.\mathtt{Ld_1}(u,t,u) \wedge \mathtt{Hd}_{q_S}(t) \quad (31)$$

$$\mathtt{Ld}_\ell(u,t,\vec{v}) \to \mathtt{Ld}_{\mathtt{p}_1^\ell}(u,t,\vec{v}) \quad (32)$$

$$\mathtt{Rdy}_{\mathtt{p}_j^\ell}(u,t,\vec{v}) \to \mathtt{Ld}_{\mathtt{p}_{j+1}^\ell}(u,t,\vec{v}) \quad (33)$$

$$\mathtt{Rdy}_{\mathtt{p}_{\bar{n}}^\ell}(u,t,\vec{v}) \to \mathtt{Rdy}_\ell(u,t,\vec{v}) \quad (34)$$

$$\mathtt{Rdy}_\ell(u,t,\vec{v}) \wedge \bigwedge_{i=k+1}^{\ell} \mathtt{Root}(v_i) \wedge \mathtt{Chi}(w,v_k) \to \mathtt{Ld}_\ell(u,t,v_1,\cdots,v_{k-1},w,u,\cdots,u) \quad (35)$$

$$\mathtt{Rdy}_\ell(u,t,\vec{v}) \wedge \bigwedge_{i=1}^{\ell} \mathtt{Root}(v_i) \to \mathtt{Ld}_{\ell+1}(u,t,u,\cdots,u) \quad (36)$$

$$\mathtt{Rdy}_{\bar{m}}(u,t,\vec{v}) \wedge \bigwedge_{i=1}^{\bar{m}} \mathtt{Root}(v_i) \to \mathtt{S_\lrcorner}(t) \wedge \mathtt{End}(t) \quad (37)$$

$$\mathtt{Ld_p}(u,t,\vec{v}) \wedge \mathtt{NIn'_p}(\vec{v}) \to \mathtt{Rdy_p}(u,t,\vec{v}) \quad (38)$$

Figure 5: Some rules of $\mathcal{R}_4$, to be instantiated for all $1 \leq j \leq \bar{n}-1$, $1 \leq k \leq \ell \leq \bar{m}$, and $\mathtt{p} \in \mathcal{S}$.

All other rules can be rewritten analogously, since every existentially quantified variable is used in unique ways with predicates other than Cpy and $\mathtt{Cpy}_{+1}$.

For every $\mathcal{I} \in \mathfrak{I}_2$, we show that $\mathit{Int}_3(\mathcal{I})$ is isomorphic to a result of the chase on $\mathcal{R}_3$ and $\mathcal{I}$. The next result then follows from Lemma 6.

Lemma 7. *$\mathfrak{I}_3 = \{\mathit{Int}_3(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_2\}$ is a universal model set of $\mathcal{R}_3$ and $\mathcal{D}$.*

$\mathcal{R}_4$: Initial TM Configuration For each leaf in the tree of completions, $\mathcal{R}_4$ creates the representation of an initial TM configuration. The tape is again represented by a Nxt-chain, using further unary predicates $\mathtt{S_{||}}$, $\mathtt{S_\lrcorner}$, and $\mathtt{S_p}$ (for all $\mathtt{p} \in \mathcal{S}$) for additional tape symbols. $\mathtt{Hd}_{q_S}$ marks the TM's starting position and initial state q_S, and End the end of the tape.

Let $\bar{m}$ be the maximal arity of predicates in $\mathcal{S}$. We require that there is some $\bar{n} > 0$ such that $\mathcal{S}$ contains exactly $\bar{n}$ predicates $\mathtt{p}_1^i, \ldots, \mathtt{p}_{\bar{n}}^i$ of arity i, for every $1 \leq i \leq \bar{m}$. This is without loss of generality, except for the exclusion of nullary predicates. Our results do not depend on this restriction, but it helps to simplify the presentation of our main ideas.

To serialise the data as a tape, we iterate over all predicate arities $\ell = 1, \ldots, \bar{m}$ and over all lists $\vec{v}$ of tree nodes with length ℓ. In this process, $\mathtt{Ld}_\ell(u,t,\vec{v})$ expresses that, while encoding the leaf u, after constructing the tape until position t, we continue serialising ℓ-ary predicate data for arguments $\vec{v}$. Analogously, $\mathtt{Rdy}_\ell(u,t,\vec{v})$ means that this was completed at tape position t. Similar predicates $\mathtt{Ld_p}$ and $\mathtt{Rdy_p}$ are used to consider a specific predicate $\mathtt{p} \in \mathcal{S}$ during this process. The rules in Figure 5 start the serialisation (31), proceed over all predicates (32)–(34), iterate over parameter vectors (35) and arities (36), and finally end the tape (37).

Absent facts do not need to be serialised (38), while present facts can be treated by copying the encodings for each of their parameters (29) and (30). In the latter, LdE states that a specific argument is serialised between two given tape positions.

The resulting TM tapes serialise facts $\mathtt{In'_p}(\vec{u})$ as introduced by $\mathcal{R}_2$, i.e., where $\vec{u}$ are nodes in the representative tree. Given a model $\mathcal{I} \in \mathfrak{I}_3$ with some $\mathtt{Leaf}(u_w) \in \mathcal{I}$, let $\mathit{branch}(u_w)$ be the set of all nodes $u_{w'}$ on the branch of u_w, i.e. all nulls $u_{w'}$ where w' is a prefix of w. Elements of $\mathit{branch}(u_w)$ are totally ordered by setting $u_{w_1} \prec u_{w_2}$ if $|w_1| > |w_2|$. Predicates are totally ordered by setting $\mathtt{p}_i^a \prec \mathtt{p}_j^b$ if either $a < b$, or both $a = b$ and $i < j$. We can then order facts as $\mathtt{p}(\vec{u}) \prec \mathtt{q}(\vec{v})$ if $\vec{u},\vec{v} \subseteq \mathit{branch}(u_w)$ and $\langle \mathtt{p}, \vec{u}\rangle$ is lexicographically before $\langle \mathtt{q}, \vec{v}\rangle$.

Now let $\mathit{branchDb}(\mathcal{I},u_w) = \{\mathtt{p}(\vec{u}) \mid \mathtt{In'_p}(\vec{u}) \in \mathcal{I}, \vec{u} \subseteq \mathit{branch}(u_w)\}$ denote the set of all facts on the branch with leaf u_w, and let $\mathit{branchTape}(\mathcal{I},u_w)$ denote the TM tape serialisation (as defined in Section 2) of $\mathit{branchDb}(\mathcal{I},u_w)$ according to the total order $\prec$ and representing each node u_w by the binary representation of $|w|+1$ as before. Given $S = \mathit{branchTape}(\mathcal{I},u_w)$, let $\mathit{startConf}(\mathcal{I},u_w)$ be the following interpretation:

$$\{\mathtt{Ld_1}(u_w,t_w^1,u_w), \mathtt{Hd}_{q_S}(t_w^1), \mathtt{End}(t_w^{|S|+1})\} \cup$$
$$\{\mathtt{Nxt}(t_w^{j-1},t_w^j) \mid 2 \leq j \leq |S|+1\} \cup$$
$$\{\mathtt{S}_a(t_w^j) \mid 1 \leq j \leq |S|, a = S[j]\} \cup \{\mathtt{S_\lrcorner}(t_w^{|S|+1})\}.$$

Let $\mathcal{J}$ be the extension of $\mathcal{I}$ with $\mathit{startConf}(\mathcal{I},u_w)$ for every $\mathtt{Leaf}(u_w) \in \mathcal{I}$. We define $\mathit{Int}_4(\mathcal{I})$ to be the smallest superset of $\mathcal{J}$ that satisfies all rules in $\mathcal{R}_4$ while including only the nulls in $\mathcal{J}$. As in the case of Int_3, the missing relations can easily be inferred using the original rules or, for (29) and (30), with simple rewritings thereof.

Lemma 8. *$\mathfrak{I}_4 = \{\mathit{Int}_4(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_3\}$ is a universal model set of $\mathcal{R}_4$ and $\mathcal{D}$.*

$\mathcal{R}_5$: TM Run The purpose of $\mathcal{R}_5$ is to simulate the run of the deterministic TM $\langle Q,\Gamma,\delta\rangle$ on each of the initial tapes created by $\mathcal{R}_4$. We continue to use predicate Nxt for neighbouring tape cells (augmented with its transitive closure $\mathtt{Nxt}^+$), $\mathtt{S}_b$ to encode tape symbols $b \in \Gamma$, and $\mathtt{Hd}_q$ to encode head position and current state $q \in Q$. Predicate Stp connects tape cells in each configuration to the corresponding tape cells in the next configuration (provided the TM performs another step). The rules in Figure 6 are a standard TM encoding, with the slight exception of rule (43), which adds a new blank tape cell in each step (even if not used by the TM). Our rules use the assumptions on TMs in Section 2.

Consider some $\mathcal{I} \in \mathfrak{I}_4$. It is easy and only mildly laborious to define interpretations $\mathit{Run}(u_w)$ that represent all successor configurations of the starting configuration $\mathit{startConf}(\mathcal{I},u_w)$, appropriately connected with Stp and

$$\mathtt{Hd}_{q_A}(x) \to \mathtt{Goal} \tag{39}$$

$$\mathtt{Nxt}(x,y) \to \mathtt{Nxt}^+(x,y) \tag{40}$$

$$\mathtt{Nxt}^+(x,y) \wedge \mathtt{Nxt}^+(y,z) \to \mathtt{Nxt}^+(x,z) \tag{41}$$

$$\mathtt{Nxt}(x,y) \wedge \mathtt{Stp}(x,z) \wedge \mathtt{Stp}(y,w) \to \mathtt{Nxt}(z,w) \tag{42}$$

$$\mathtt{End}(x) \wedge \mathtt{Stp}(x,z) \to \exists v.\mathtt{Nxt}(z,v) \wedge \mathtt{S}_{\llcorner}(v) \wedge \mathtt{End}(v) \tag{43}$$

$$\mathtt{Hd}_q(x) \wedge \mathtt{S}_a(x) \to \exists z.\mathtt{Stp}(x,z) \wedge \mathtt{S}_b(z) \tag{44}$$

$$\mathtt{Hd}_q(x) \wedge \mathtt{Nxt}^+(x,y) \wedge \mathtt{S}_c(y) \to \exists z.\mathtt{Stp}(y,z) \wedge \mathtt{S}_c(z) \tag{45}$$

$$\mathtt{Hd}_q(x) \wedge \mathtt{Nxt}^+(y,x) \wedge \mathtt{S}_c(y) \to \exists z.\mathtt{Stp}(y,z) \wedge \mathtt{S}_c(z) \tag{46}$$

$$\mathtt{Hd}_q(x) \wedge \mathtt{S}_a(x) \wedge \mathtt{Stp}(x,z) \wedge \mathtt{Nxt}(z,w) \to \mathtt{Hd}_r(w) \tag{47}$$

$$\mathtt{Hd}_q(x) \wedge \mathtt{S}_a(x) \wedge \mathtt{Stp}(x,z) \wedge \mathtt{Nxt}(w,z) \to \mathtt{Hd}_r(w) \tag{48}$$

Figure 6: The rule set $\mathcal{R}_5$ contains $\mathcal{R}_4$ (see Figure 4) and the above rules, where we instantiate rules (44)–(46) for all transitions $(q,a) \mapsto (r,b,X) \in \delta$ and $c \in \Gamma$; rule (47) for all $(q,a) \mapsto (r,b,+1) \in \delta$; and rule (48) for all $(q,a) \mapsto (r,b,-1) \in \delta$.

the transitive closure $\mathtt{Nxt}^+$. Moreover, let $\mathcal{J}$ be the extension of $\mathcal{I}$ with all $\mathtt{Nxt}^+$ required to satisfy (40) and (41) (note that $\mathtt{Nxt}$ also occurs in encodings from $\mathcal{R}_3$). We define $\mathit{Int}_5(\mathcal{I})$ as the union of $\mathcal{J}$ with the interpretations $\mathit{Run}(u_w)$ for all u_w with $\mathtt{Leaf}(u_w) \in \mathcal{I}$.

Lemma 9. *$\mathfrak{I}_5 = \{\mathit{Int}_5(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_4\}$ is a universal model set of $\mathcal{R}_5$ and $\mathcal{D}$.*

Proving Lemma 4 To complete the proof of Lemma 4, we set $\mathfrak{M} = \mathfrak{I}_5$. For $\mathcal{I} \in \mathfrak{M}$, let $\mathit{Db}(\mathcal{I}) = \{\mathtt{p}(\vec{t}) \mid \mathtt{In}_{\mathtt{p}}(\vec{t}) \in \mathcal{I}\}$ denote the completed database created by $\mathcal{R}_1$. Due to rule (4) in Figure 1, there is a homomorphism $\mathcal{D} \to \mathit{Db}(\mathcal{I})$. Moreover, the representation tree constructed for $\mathcal{I}$ by $\mathcal{R}_2$ has a branch that is maximal, i.e., has $|\mathit{Order}(\mathcal{I})|$ nodes; this branch has a leaf u_w with $|w| = |\mathit{Order}(\mathcal{I})|$. We obtain a homomorphism $\mathit{Db}(\mathcal{I}) \to \mathit{branchDb}(\mathcal{I}, u_w)$. Lemma 4 now follows from Lemma 9 and Lemmas 11 and 10 below.

Lemma 10. *If $\mathcal{D} \in \mathfrak{Q}$, then $\mathtt{Goal} \in \mathcal{I}$ for each $\mathcal{I} \in \mathfrak{M}$.*

Proof. As shown above, there is a homomorphism $\mathcal{D} \to \mathit{branchDb}(\mathcal{I}, u_w)$ for the node u_w where $|w| = |\mathit{Order}(\mathcal{I})|$. Since $\mathfrak{Q}$ is closed under homomorphisms, $\mathcal{D} \in \mathfrak{Q}$ implies $\mathit{branchDb}(\mathcal{I}, u_w) \in \mathfrak{Q}$. By the correctness of our TM simulation, we obtain $\mathtt{Goal} \in \mathcal{I}$. □

Lemma 11. *If $\mathcal{D} \notin \mathfrak{Q}$, then $\mathtt{Goal} \notin \mathcal{I}$ for some $\mathcal{I} \in \mathfrak{M}$.*

Proof. Consider some $\mathcal{I} \in \mathfrak{M}$ such that $\mathit{Db}(\mathcal{I}) = \mathcal{D}$ and $\mathtt{NEq}(t,u) \in \mathcal{I}$ for each $t, u \in \mathsf{Nulls}(\mathcal{D})$ with $t \neq u$. Let u_w denote the leaf node with $|w| = |\mathit{Order}(\mathcal{I})|$ as before. Then $\mathit{branchDb}(\mathcal{I}, u_w)$ is isomorphic to $\mathit{Db}(\mathcal{I}) = \mathcal{D}$. By the correctness of our TM simulation, $\mathtt{Goal}$ is not derived from this maximal branch. Moreover, for all other leaf nodes u_v with $\mathtt{Leaf}(u_v) \in \mathcal{I}$, there is a homomorphism $\mathit{branchDb}(\mathcal{I}, u_v) \to \mathit{branchDb}(\mathcal{I}, u_w)$. Since $\mathfrak{Q}$ is closed under homomorphisms, the TM does not accept any such $\mathit{branchDb}(\mathcal{I}, u_v)$, so $\mathtt{Goal} \notin \mathcal{I}$. □

4 Ensuring Chase Termination

While the rules in Section 3 are semantically correct, the disjunctive chase may not terminate on them. Many known fragments of existential rules can guarantee chase termination, including for expressive cases where termination might be exponential (Carral et al. 2019), but they are not applicable to our case, since the runtime of TMs that decide a query can in general not be bounded by any elementary function. Indeed, we rely on the TM to stop "naturally", by virtue of being a decider. Nevertheless, our rules lead to infinite chase trees, e.g., if the disjunctive guessing of $\mathtt{LT}$ leads to a cycle, which enables rule (16) to create an infinite path in the representation tree. We will now show that this can be avoided:

Theorem 12. *Every homomorphism-closed decidable query is expressed by a set of disjunctive rules that is chase-terminating for all databases over the schema of the query.*

To show this, we refine and generalise the "emergency brake" technique of Krötzsch, Marx, and Rudolph (2019), and re-formulate it as a general rule set transformation. This not only yields a generic method that is of independent interest, but it also allows us to address potential termination problems in our prior modelling.

Definition 3. Consider a rule set Σ and a nullary predicate $\mathtt{Halt}$ that does not occur in Σ. For every predicate $\mathtt{p}$ in Σ, let $\hat{\mathtt{p}}$ be a fresh predicate of the same arity, and, for any formula ψ, let $\hat{\psi}$ be ψ with all predicates $\mathtt{p}$ replaced by $\hat{\mathtt{p}}$. Now the set $\mathit{brake}(\Sigma, \mathtt{Halt})$ consists of the following rules:

$$\to \exists v.\mathtt{Brake}(v) \tag{49}$$

$$\mathtt{Halt} \wedge \mathtt{Brake}(x) \to \mathtt{Real}(x) \tag{50}$$

$$\hat{\mathtt{p}}(\vec{x}) \wedge \textstyle\bigwedge_{x \in \vec{x}} \mathtt{Real}(x) \to \mathtt{p}(\vec{x}) \qquad \text{for all } \mathtt{p} \text{ in } \Sigma \tag{51}$$

For every rule $\rho : \beta[\vec{x}] \to \bigvee_{i=1}^{k} \exists \vec{y}_i.\eta_i[\vec{x}_i, \vec{y}_i]$:

$$\beta[\vec{x}] \wedge \mathtt{Brake}(v) \to \textstyle\bigvee_{i=1}^{k} \big(\mathtt{B}^i_\rho(\vec{x}_i) \wedge \hat{\eta}_i[\vec{x}_i, \vec{y}_i \mapsto v] \wedge \bigwedge_{x \in \vec{x}_i} \mathtt{Real}(x)\big) \tag{52}$$

$$\mathtt{B}^i_\rho(\vec{x}_i) \to \exists \vec{y}_i.\hat{\eta}_i[\vec{x}_i, \vec{y}_i] \wedge \textstyle\bigwedge_{y \in \vec{y}_i} \mathtt{Real}(y) \tag{53}$$

where $\hat{\eta}_i[\vec{x}_i, \vec{y}_i \mapsto v]$ is $\hat{\eta}_i$ with each variable $y \in \vec{y}_i$ replaced by v, and $\mathtt{Brake}$, $\mathtt{Real}$, and all $\mathtt{B}^i_\rho$ are fresh predicates with arities as indicated.

Note that $\mathit{brake}(\Sigma, \mathtt{Halt})$ does not define rules to derive $\mathtt{Halt}$, and indeed the transformation largely preserves the models of Σ in the following sense:

Lemma 13. *Consider a rule set Σ and database $\mathcal{D}$ over predicates that occur in Σ. For every model $\mathcal{I}$ of $\mathit{brake}(\Sigma, \mathtt{Halt})$ and $\mathcal{D}$, the set $\mathcal{I}^- = \{\mathtt{p}(\vec{n}) \mid \mathtt{p}(\vec{n}) \in \mathcal{I}, \mathtt{p}$ occurs in $\Sigma\}$ is a model of Σ and $\mathcal{D}$, and every model $\mathcal{J}$ of Σ and $\mathcal{D}$ is of this form.*

Proof. Consider a rule $\rho \in \Sigma$ as in Definition 3, and let σ be a substitution such that $\sigma(\beta) \subseteq \mathcal{I}^-$. Then we can apply rules (52), (53), and finally (51) to derive $\sigma'(\eta_i) \subseteq \mathcal{I}^-$ for a suitable extension σ' of σ. Hence $\mathcal{I}^- \models \Sigma$.

Conversely, let $\mathcal{J} \models \Sigma$. A model $\mathcal{I}$ of $\mathit{brake}(\Sigma, \mathtt{Halt})$ can be found by adding, for each matching body $\sigma(\beta) \subseteq \mathcal{J}$

of rule ρ, an atom $\sigma(\mathtt{B}^i_\rho(\vec{x}_i))$ for some i such that $\sigma'(\eta_i) \subseteq \mathcal{J}$ for an extension σ' of σ. To obtain the required model $\mathcal{I}$ of $\mathit{brake}(\Sigma, \mathtt{Halt})$, it remains to add facts $\mathtt{Brake}(b)$ for a fresh null b, $\sigma(\hat{\eta}_i[\vec{x}_i, \vec{y}_i \mapsto b])$ as in (52) for every $\sigma(\mathtt{B}^i_\rho(\vec{x}_i)) \in \mathcal{I}$, and $\mathtt{Real}(n)$ for every $\mathtt{p}(\vec{n}) \in \mathcal{J}$ and $n \in \vec{n}$. □

For $\mathit{brake}(\Sigma, \mathtt{Halt})$ to be useful, we need to add rules that can "pull the brake" by deriving $\mathtt{Halt}$. Doing so stops the chase in the following sense:

Lemma 14. *Consider a rule set Σ, a database $\mathcal{D}$ over predicates that occur in Σ, and a set Π of rules of the form $\beta \to \mathtt{Halt}$ where β only uses predicates in Σ. If $\mathcal{I}$ is the label of a node in a chase tree for $\Sigma \cup \Pi$ and $\mathcal{D}$ such that $\mathtt{Halt} \in \mathcal{I}$, then the tree starting at the node of $\mathcal{I}$ is finite.*

Proof. Since $\mathtt{Halt} \in \mathcal{I}$, there is a substitution σ such that $\langle \rho_{(50)}, \sigma \rangle$ is applicable (for $\rho_{(50)}$ in (50)). By fairness, $\mathtt{Real}(\sigma(x))$ will be derived at some depth of the tree. From this depth on, no rule of form (53) is applicable: given $\mathtt{Real}(\sigma(x))$, the head of rules of form (52) already satisfies the head of the rule (53) that could be applied to a newly derived atom for $\mathtt{B}^i_\rho$. Rules other than (53) do not contain existential quantifiers thus can only be applied a finite number of times before the chase on this part of the tree terminates. □

If $\mathtt{Halt}$ is derived, the semantic correspondence of Lemma 13 is weakened, but suffices to preserve entailments:

Lemma 15. *Consider Σ, $\mathcal{D}$, and Π as in Lemma 14. For every model $\mathcal{I}$ of $\mathit{brake}(\Sigma, \mathtt{Halt}) \cup \Pi$ and $\mathcal{D}$, $\mathcal{I}^-$ (as in Lemma 13) is a model of Σ and $\mathcal{D}$.*

Proof. This is immediate from Lemma 13 and the fact that every model of $\mathit{brake}(\Sigma, \mathtt{Halt}) \cup \Pi$ and $\mathcal{D}$ is also a model of $\mathit{brake}(\Sigma, \mathtt{Halt})$ and $\mathcal{D}$. □

Having established the key properties of the emergency brake construction, we can now apply it to show Theorem 12. Given the rule set $\mathcal{R}_5$ as defined for a query $\mathfrak{Q}$ in Section 3, let $\mathcal{R}_6$ denote the extension of $\mathit{brake}(\mathcal{R}_5, \mathtt{Halt})$ with the following rules:

$$\mathtt{In}_\mathtt{p}(\vec{x}) \wedge \mathtt{NIn}_\mathtt{p}(\vec{x}) \to \mathtt{Halt} \tag{54}$$
$$\mathtt{LT}(x, x) \to \mathtt{Halt} \tag{55}$$
$$\mathtt{Last}(x) \wedge \mathtt{LT}(x, y) \to \mathtt{Halt} \tag{56}$$
$$\mathtt{LT}(x, y) \wedge \mathtt{First}(y) \to \mathtt{Halt} \tag{57}$$

Lemma 16. *$\mathcal{R}_6$ expresses the query $\mathfrak{Q}$.*

Proof. For a database $\mathcal{D}$ over $\mathcal{S}$, let $\mathfrak{M}$ be the universal model set constructed in Section 3. If $\mathcal{D} \in \mathfrak{Q}$, then $\mathcal{R}_5, \mathcal{D} \models \mathtt{Goal}$ by Theorem 3. Then $\mathcal{R}_6, \mathcal{D} \models \mathtt{Goal}$ since any model of $\mathcal{R}_6$ and $\mathcal{D}$ must contain $\mathtt{Goal}$ by Lemma 15.

Conversely, if $\mathcal{D} \notin \mathfrak{Q}$, then there is $\mathcal{U} \in \mathfrak{M}$ with $\mathtt{Goal} \notin \mathcal{U}$. By Lemma 13, there is a model $\mathcal{I}$ of $\mathit{brake}(\mathcal{R}_5, \mathtt{Halt})$ with $\mathcal{I}^- = \mathcal{U}$, and hence $\mathtt{Goal} \notin \mathcal{I}$. By construction of $\mathfrak{M}$, none of the rules (54)–(57) applies to $\mathcal{U}$, and hence $\mathcal{I}$ is also a model of $\mathcal{R}_6$, i.e., $\mathcal{R}_6, \mathcal{D} \not\models \mathtt{Goal}$. □

Lemma 17. *$\mathcal{R}_6$ is chase-terminating for all databases over the schema $\mathcal{S}$ of the query $\mathfrak{Q}$.*

Proof. Consider a chase over $\mathcal{R}_6$ and input database $\mathcal{D}$. Chase branches where $\mathtt{Halt}$ is eventually part of a node label terminate by Lemma 14. Let b denote any branch of the chase where $\mathtt{Halt}$ is not derived, and let $\mathcal{I}$ be the union of all node labels on that branch. We want to show that $\mathcal{I}$ (and hence b) is finite.

By Lemma 13, $\mathcal{I} \models \mathcal{R}_1$. Moreover, since $\mathtt{Halt} \notin \mathcal{I}$, rules (54)–(57) are not applicable to $\mathcal{I}$. Both properties together suffice to show that the set $\mathcal{I}_{\mathcal{R}_1} = \{\mathtt{p}(\vec{n}) \in \mathcal{I} \mid \mathtt{p}$ is a predicate in $\mathcal{R}_1\}$ is an element of $\mathfrak{I}_1$ defined in Section 3.

Since the predicates in rule bodies of $\mathcal{R}_1$ do not occur in any rule head in $\mathcal{R}_5 \setminus \mathcal{R}_1$, we can assume without loss of generality (and without affecting chase termination), that the corresponding rules of $\mathit{brake}(\mathcal{R}_1, \mathtt{Halt}) \subseteq \mathcal{R}_6$ have been applied first. This shows that $\mathcal{I}$ is equal to the result of a chase with non-disjunctive rules $\mathcal{R}_6 \setminus \mathit{brake}(\mathcal{R}_1, \mathtt{Halt})$ on a database $\mathcal{I}_{\mathcal{R}_1} \in \mathfrak{I}_1$. The claim follows by noting that any such chase must terminate: this was shown in Section 3, where we described a deterministic process of defining the elements in the universal model set $\mathfrak{I}_5$ from those in $\mathfrak{I}_1$. Each steps in this construction is fully determined and introduces isomorphic sets of nulls irrespectively of the order of rule applications. The only exception are application of rules (15), (16), and (18). For example, given facts $\mathtt{First}(n_1)$, $\mathtt{First}(n_2)$, and $\mathtt{Eq}(n_1, n_2)$, the standard model of Section 3 contains one fact $\mathtt{Root}(u_1)$ with $\mathtt{Rep}(n_1, u_1)$ and $\mathtt{Rep}(n_2, u_1)$, which can be obtained using (15) (on n_1) and (18). If we apply (15) to both n_1 and n_2 before applying (18), we obtain two distinct $\mathtt{Root}(u_1)$ and $\mathtt{Root}(u_1')$. Similar variations can occur with other tree nodes if (16) is applied before (18). If is easy to see that this does not endanger termination, but merely leads to several isomorphic paths in the representation tree. □

Together, Lemmas 16 and 17 show Theorem 12.

5 Removing Disjunctions

Our main result is that any decidable homomorphism-closed query is expressible by a chase-terminating existential rule set. To conclude the proof of this statement, we remove the disjunction from the rule set $\mathcal{R}_6$ of Section 4. We present this as a general technique of expressing disjunctive Datalog using existential rules, which is also of independent interest.

For a rule set Σ, the *input schema* $\mathcal{S}_{\mathsf{in}}(\Sigma)$ is the set of all predicates in Σ that do not occur in any rule head. We focus on rule sets that can be split into a disjunctive part and an existential part, such that it is admissible to completely apply the disjunctive rules first, and the existential ones afterwards:

Definition 4. A *split* of a set Σ of disjunctive existential rules consists of a set Σ_1 of disjunctive Datalog rules and a set Σ_2 of existential rules, such that $\Sigma = \Sigma_1 \cup \Sigma_2$ and:

For every database $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, and for every chase result $\mathfrak{M}$ over $\langle \Sigma, \mathcal{D} \rangle$, there is a chase result $\mathfrak{M}_1$ over $\langle \Sigma_1, \mathcal{D} \rangle$, such that $\mathfrak{M} = \{\mathcal{C}_\mathcal{I} \mid \mathcal{I} \in \mathfrak{M}_1\}$ where each $\mathcal{C}_\mathcal{I}$ is the (unique) interpretation resulting from some chase over $\langle \Sigma_2, \mathcal{I} \rangle$.

$$\to \exists w.\mathtt{Init}(w) \land \mathtt{Done}(w) \land \mathtt{Empty}(w) \tag{58}$$

$$\mathtt{Done}(w) \land \mathtt{Init}(w) \land \mathtt{p}(\vec{x}) \to \exists w'.\mathtt{Ins}_{\mathtt{p}}(\vec{x}, w, w') \land \mathtt{Subs}(w', w') \land \mathtt{Init}(w') \tag{59}$$

$$\mathtt{Done}(w) \land \textstyle\bigwedge_{\mathtt{p}(\vec{x})\in\beta} \mathtt{Ins}_{\mathtt{p}}(\vec{x}, w, w) \to \exists w_1.\mathtt{Ins}_{\mathtt{p}_1}(\vec{x_1}, w, w_1) \land \mathtt{Subs}(w_1, w_1) \tag{60}$$

$$\mathtt{Done}(w) \land \textstyle\bigwedge_{\mathtt{p}(\vec{x})\in\beta} \mathtt{Ins}_{\mathtt{p}}(\vec{x}, w, w) \to \exists w_2.\mathtt{Ins}_{\mathtt{p}_2}(\vec{x_2}, w, w_2) \land \mathtt{Subs}(w_2, w_2) \tag{61}$$

$$\mathtt{Ins}_{\mathtt{p}}(\vec{x}, w_0, w_1) \land \mathtt{Subs}(w_1, w_2) \to \mathtt{Ins}_{\mathtt{p}}(\vec{x}, w_2, w_2) \land \mathtt{p}'(\vec{x}, w_2) \land \mathtt{Subs}(w_0, w_2) \tag{62}$$

$$\mathtt{Empty}(w) \land \mathtt{Subs}(w, w') \to \mathtt{Done}(w') \tag{63}$$

Figure 7: The rule set Σ_1', where we instantiate (59) and (62) for all $\mathtt{p} \in \mathsf{Preds}$ in $\Sigma_1 \cup \Sigma_2$, and (60) and (61) for all $\beta \to \mathtt{p}_1(\vec{x_1}) \lor \mathtt{p}_2(\vec{x_2}) \in \Sigma_1$.

$$\mathtt{Done}(w) \land \textstyle\bigwedge_{\mathtt{p}(\vec{x})\in\beta} \mathtt{p}'(\vec{x}, w) \to \exists \vec{z}. \textstyle\bigwedge_{\mathtt{q}(\vec{y})\in\eta} \mathtt{q}'(\vec{y}, w) \tag{64}$$

Figure 8: The rule set Σ_2', where we instantiate (64) for all $\beta \to \exists \vec{z}.\eta \in \Sigma_2$.

$$\mathtt{Goal}'(w) \to \mathtt{Acc}(w) \tag{65}$$

$$\begin{aligned} &\mathtt{Ins}_{\mathtt{p}_1}(\vec{x}_1, w, w_1) \land \mathtt{Acc}(w_1) \land \\ &\mathtt{Ins}_{\mathtt{p}_2}(\vec{x}_2, w, w_2) \land \mathtt{Acc}(w_2) \land \\ &\textstyle\bigwedge_{\mathtt{p}(\vec{x})\in\beta} \mathtt{Ins}_{\mathtt{p}}(\vec{x}, w, w) \to \mathtt{Acc}(w) \end{aligned} \tag{66}$$

$$\mathtt{Init}(w) \land \mathtt{Acc}(w) \to \mathtt{Goal} \tag{67}$$

Figure 9: The rule set Σ_3', where (66) is instantiated for all rules $\beta \to \mathtt{p}_1(\vec{x_1}) \lor \mathtt{p}_2(\vec{x_2}) \in \Sigma_1$

Lemma 18. *Consider a rule set Σ with split $\langle \Sigma_1, \Sigma_2 \rangle$. There is a set Σ' of existential rules, such that, for every database $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, we have:*

1. *$\mathcal{D}, \Sigma \models \mathtt{Goal}$ iff $\mathcal{D}, \Sigma' \models \mathtt{Goal}$, and*
2. *if $\langle \Sigma_2, \mathcal{D}_2 \rangle$ is chase-terminating for every database $\mathcal{D}_2$ over $\mathcal{S}_{\mathsf{in}}(\Sigma_2)$, then $\langle \Sigma', \mathcal{D} \rangle$ is also chase-terminating.*

To construct this set Σ', we assume w.l.o.g. that all disjunctive rules have exactly two disjuncts in the head. We define Σ' as the union of sets Σ_1', Σ_2' and Σ_3' as shown in Figures 7, 8 and 9, which we explain below.

Σ_1' uses a technique for modelling sets with chase-terminating existential rules (Krötzsch, Marx, and Rudolph 2019, Fig. 2). We adapt this to sets of ground atoms, called *worlds* and denoted by variables w in the figures. Facts $\mathtt{Ins}_{\mathtt{p}}(\vec{t}, w, w')$ express that world w' is obtained by adding $\mathtt{p}(\vec{t})$ to world w. In particular, $\mathtt{Ins}_{\mathtt{p}}(\vec{t}, w, w)$ states that $\mathtt{p}(\vec{t})$ is in w, and we define $\mathsf{world}(w) = \{\mathtt{p}(\vec{a}) \mid \mathtt{Ins}_{\mathtt{p}}(\vec{a}, w, w) \in \mathcal{I}\}$ for any interpretation $\mathcal{I}$. Worlds are created by adding database facts (59) or by applying rules to existing worlds (60)–(61). Worlds containing only database facts are marked with $\mathtt{Init}$. Predicate $\mathtt{Subs}$ defines the subset relation on worlds. Rules (62)–(63) copy all prior facts to a new world before marking it $\mathtt{Done}$.

Proposition 19. *Σ_1' is chase-terminating and for every $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, the (unique) interpretation $\mathcal{I}$ resulting from some chase over Σ_1' and $\mathcal{D}$ is such that:*

- *if $\mathtt{p}(\vec{a}) \in \mathcal{D}$ and $\mathtt{Done}(w) \in \mathcal{I}$, there exists w' such that $\mathtt{Ins}_{\mathtt{p}}(\vec{a}, w, w') \in \mathcal{I}$ and $\mathsf{world}(w') = \mathsf{world}(w) \cup \{\mathtt{p}(\vec{a})\}$;*
- *if $\rho \in \Sigma_1$ is applicable to $\mathsf{world}(w)$, creating $\mathtt{p}_1(\vec{a})$ or $\mathtt{p}_2(\vec{b})$, there exists w_1 and w_2 such that $\{\mathtt{Ins}_{\mathtt{p}_1}(\vec{a}, w, w_1), \mathtt{Ins}_{\mathtt{p}_2}(\vec{b}, w, w_2)\} \subseteq \mathcal{I}$, $\mathsf{world}(w_1) = \mathsf{world}(w) \cup \{\mathtt{p}_1(\vec{a})\}$ and $\mathsf{world}(w_2) = \mathsf{world}(w) \cup \{\mathtt{p}_2(\vec{b})\}$.*

Note that we cannot distinguish worlds that are not containing all database facts, and that some worlds may contain more facts than needed to satisfy all disjunctive heads.

Σ_2' now simulates the application of rules from Σ_2 in any of the worlds. Computations relative to different worlds are independent from each other. Finally, Σ_3' aggregates results from all worlds: a world is accepting ($\mathtt{Acc}$) if either $\mathtt{Goal}$ was derived locally (65) or it has two successor worlds for a disjunctive rule that are both accepting (66). $\mathtt{Goal}$ is a consequence if any initial world is accepting (67). This finishes the construction of Σ' as the main ingredient for proving Lemma 18.

Finally, we can apply Lemma 18 to $\mathcal{R}_6 = \mathit{brake}(\mathcal{R}_5, \mathtt{Halt}) \cup \Pi$ from Section 4, where Π denotes rules (54)–(57). Intuitively, a possible split is $\mathit{brake}(\mathcal{R}_1, \mathtt{Halt})$ and $\mathit{brake}(\mathcal{R}_5 \setminus \mathcal{R}_1, \mathtt{Halt}) \cup \Pi$. Formally, however, $\mathit{brake}(\mathcal{R}_1, \mathtt{Halt})$ is not disjunctive Datalog due to existential rules (2), (3), and (49). However, our result easily extends to such rules with empty body: we can just add them to Σ_1' and treat their inferences like facts from the initial database. The other properties of Definition 4 are easy to verify. The fact that both rule sets have some common rules, such as (50), is no concern. Finally, it remains to argue termination for Σ_2 as required for item (2) in Lemma 18. This is slightly stronger than Lemma 17 since we must also consider databases that use some inferred predicates of $\mathcal{R}_1$. However, the proof of Lemma 17 and the emergency brake technique in general served the main purpose of safeguarding against problematic structures among the inferred predicates of $\mathcal{R}_1$, and it is not hard to see that this already showed what we require here. Combining all of our insights, we finally obtain:

Theorem 20. *Chase-terminating existential rules capture the class of all decidable homomorphism-closed queries.*

6 Limitations of Semi-Decidable Languages

A *query language* $\mathfrak{F}$ over a schema $\mathcal{S}$ is a function from a set L to $2^{\mathfrak{D}_\mathcal{S}}$, where $\mathfrak{D}_\mathcal{S}$ is the set of all databases over schema $\mathcal{S}$. We say that $\mathfrak{F}$ is *semi-decidable* if membership to L is semi-decidable, and that its *query answering problem is decidable* if there exists a TM $M_\mathfrak{F}$ that takes as input some

$(l, \mathcal{D}) \in (L \times \mathfrak{D}_{\mathcal{S}})$ and decides whether $\mathcal{D} \in \mathfrak{F}(l)$.

The set of chase-terminating existential rule sets is a query language that is not semi-decidable (Grahne and Onet 2018) and for which the query answering problem is decidable (by running the chase). In fact, we show that one cannot find a semi-decidable query language with similar properties.

Theorem 21. *There are no semi-decidable query languages that (i) express all decidable, homomorphism-closed queries and (ii) for which query answering is decidable.*

To show this result, we define a set $\mathcal{M}$ of TMs (cf. Definition 5), show that $\mathcal{M}$ can be enumerated up to equivalence if there is a semi-decidable language that satisfies (i) and (ii) above (cf. Lemma 22), and finally prove that the consequence of this implication does not hold (cf. Lemma 23).

Definition 5. Consider the set $\mathcal{M}$ of all TMs M such that: (i) The TM M halts on all inputs. (ii) If M accepts some word w, then w corresponds to a database over schema $\{\mathtt{ed}\}$. (iii) Consider some words w and v that correspond to some $\mathcal{D}$ and $\mathcal{E}$ in $\mathfrak{D}_{\{\mathtt{ed}\}}$, respectively. If M accepts w and there is a homomorphism $h : \mathcal{D} \to \mathcal{E}$, then M accepts v.

Intuitively, $\mathcal{M}$ is the set of all deciders that solve homomorphism-closed queries over databases in $\mathfrak{D}_{\{\mathtt{ed}\}}$.

Lemma 22. *If there is a semi-decidable query language $\mathfrak{F}$ that satisfies (i) and (ii) in Theorem 21, then $\mathcal{M}$ is enumerable up to equivalence.*

Proof. If there is a language such as $\mathfrak{F}$, then there is an enumerator P for L that prints out a sequence $l_1, l_2, \ldots$ and a decider $M_{\mathfrak{F}}$ that can be used to check if $\mathcal{D} \in \mathfrak{F}(l)$ for each $(l, \mathcal{D}) \in (L \times \mathfrak{D}_{\{\mathtt{ed}\}})$. For each $i \geq 1$, let M_i be the TM that, on input w, performs the following computation: if w corresponds to a database $\mathcal{D} \in \mathfrak{D}_{\{\mathtt{ed}\}}$ and $M_{\mathfrak{F}}$ accepts $(l_i, \mathcal{D})$, then *accept*; otherwise, *reject*. By modifying P we can define an enumerator that prints out the sequence $M_1, M_2, \ldots$, which contains $\mathcal{M}$ up to equivalence. □

Lemma 23. *The set $\mathcal{M}$ is not enumerable up to equivalence.*

Proof Sketch. Assume that there is an enumerator that outputs a sequence $M_1, M_2, \ldots$ that includes $\mathcal{M}$ up to equivalence. We obtain a contradiction by defining a sequence $\mathcal{D}_1, \mathcal{D}_2, \ldots$ of databases and a TM $M_d \in \mathcal{M}$ that diagonalises over $M_1, M_2, \ldots$ and $\mathcal{D}_1, \mathcal{D}_2, \ldots$ Namely, for each $i \geq 1$, let $\mathcal{D}_i = \{\mathtt{ed}(u_1, u_2), \ldots, \mathtt{ed}(u_{p_{i+1}}, u_1)\}$ where p_{i+1} is the $(i+1)$-th prime. Moreover, M_d is the TM that, on input w, performs the computation: (1) *Reject* if w does not correspond to some $\mathcal{D} \in \mathfrak{D}_{\{\mathtt{ed}\}}$. (2) *Reject* if $\mathcal{D}$ can be homembedded into a path over $\mathtt{ed}$. (3) *Accept* if $\mathtt{ed}(u, u) \in \mathcal{D}$ for some null u. (4) If there is some $i \geq 1$ such that there are less nulls in $\mathcal{D}_i$ than in $\mathcal{D}$, the TM M_i accepts some serialisation that corresponds to $\mathcal{D}_i$, and there is a homomorphism $h : \mathcal{D} \to \mathcal{D}_i$; then *reject*. Otherwise, *accept*. □

7 Discussion and Conclusion

In this paper, we have established a characterization of all decidable homomorphism-closed Boolean queries. We showed that these are exactly the chase-terminating existential rule queries, that is, queries that can be expressed by a set of (non-disjunctive) existential rules for which the standard chase universally terminates irrespective of the order of rule applications (as long as it is fair).

By its nature, our result immediately shows that various extensions of our framework do not increase its expressivity:

Theorem 24. *Chase-terminating existential rule queries have the same expressivity as*

1. *existential rule queries with guaranteed existence of some finite chase tree (for every database),*
2. *existential rule queries for which the chase terminates according to some fair strategy (such as datalog-first),*
3. *core-chase-terminating existential rule queries,*
4. *disjunctive chase-terminating existential rule queries.*

Proof. (3) Standard-chase termination implies core-chase termination. On the other hand, core chase termination implies decidability and thus our result applies. (1) and (2) Standard-chase termination implies these weaker form of guarantees, which themselve imply core chase termination. (4) Obviously, every (non-disjunctive) existential rule set is a special case of a disjunctive one and for this special case, disjunctive chase termination coincides with termination of the (non-disjunctive) standard chase. On the other hand, disjunctive existential rule queries are also closed under homomorphisms, and disjunctive universal chase termination obviously implies decidability. So our result applies. □

However, the applicability of our result does not stop at (syntactic) extensions of our framework, as it applies to arbitrary query languages and querying formalisms of different types. In particular we would like to stress the relationship to the very comprehensive existential rules fragment of *bounded treewidth sets (bts) of rules* (Baget et al. 2011a) that is *not* chase-terminating and encompasses a plethora of well-known existential rule fragments with decidable query entailment, including guarded (Calì, Gottlob, and Kifer 2008), frontier-guarded (Baget et al. 2011a), and glut-guarded existential rules (Krötzsch and Rudolph 2011), as well as greedy bts (Baget et al. 2011b):

Theorem 25. *Let Σ be a bounded-treewidth set of rules and Q a conjunctive query. There is a chase-terminating set Σ_Q of existential rules such that $\mathcal{D}, \Sigma \models Q$ iff $\mathcal{D}, \Sigma_Q \models \mathtt{Goal}$.*

While possibly surprising, this is a straightforward consequence of decidability of conjunctive query entailment from bts and of homomorphism-closedness of existential rule queries in general. Note, however, that every Q would give rise to a different Σ_Q. In fact, asking for a "uniform" chase-terminating existential rules set Σ' satisfying $\mathcal{D}, \Sigma' \models Q$ iff $\mathcal{D}, \Sigma \models Q$ would change the game (Zhang, Zhang, and You 2015). Such a set will not exist in all cases.

While our result addresses many of the open questions regarding *expressivity* of the terminating chase (Krötzsch, Marx, and Rudolph 2019) an important avenue for future work is to investigate potential differences when it comes to the corresponding computational *complexities*. We deem it likely that not all of the discussed chase variants give rise to worst-case optimal computations.

Acknowledgements This work is partly supported by DFG in project number 389792660 (TRR 248, Center for Perspicuous Systems), by BMBF in the Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI), by the Center for Advancing Electronics Dresden (cfaed), by the ERC Consolidator Grant DeciGUT (project number 771779), and by the ANR project CQFD (ANR-18-CE23-0003).

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison-Wesley.

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

Baget, J.; Mugnier, M.; Rudolph, S.; and Thomazo, M. 2011b. Walking the complexity lines for generalized guarded existential rules. In Walsh, T., ed., *Proc. 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI'11)*, 712–717. AAAI Press/IJCAI.

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

Calì, A.; Gottlob, G.; and Kifer, M. 2008. Taming the infinite chase: Query answering under expressive relational constraints. In Brewka, G., and Lang, J., eds., *Proc. 11th Int. Conf. on Knowledge Representation and Reasoning (KR'08)*, 70–80. AAAI Press.

Carral, D.; Dragoste, I.; Krötzsch, M.; and Lewe, C. 2019. Chasing sets: How to use existential rules for expressive reasoning. In Kraus, S., ed., *Proc. 28th Int. Joint Conf. on Artificial Intelligence, IJCAI'19*, 1624–1631. ijcai.org.

Carral, D.; Dragoste, I.; and Krötzsch, M. 2017. Restricted chase (non)termination for existential rules with disjunctions. In Sierra, C., ed., *Proc. 26th Int. Joint Conf. on Artificial Intelligence, IJCAI'17*, 922–928. ijcai.org.

Deutsch, A.; Nash, A.; and Remmel, J. B. 2008. The chase revisited. In Lenzerini, M., and Lembo, D., eds., *Proc. 27th Symposium on Principles of Database Systems (PODS'08)*, 149–158. ACM.

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

Grahne, G., and Onet, A. 2018. Anatomy of the chase. *Fundam. Informaticae* 157(3):221–270.

Krötzsch, M., and Rudolph, S. 2011. Extending decidable existential rules by joining acyclicity and guardedness. In Walsh, T., ed., *Proc. 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI'11)*, 963–968. AAAI Press/IJCAI.

Krötzsch, M.; Marx, M.; and Rudolph, S. 2019. The power of the terminating chase. In Barceló, P., and Calautti, M., eds., *Proc. 22nd Int. Conf. on Database Theory, ICDT'19*, volume 127 of *LIPIcs*, 3:1–3:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Marnette, B. 2009. Generalized schema-mappings: from termination to tractability. In Paredaens, J., and Su, J., eds., *Proc. 28th Symposium on Principles of Database Systems (PODS'09)*, 13–22. ACM.

Rudolph, S., and Thomazo, M. 2015. Characterization of the expressivity of existential rule queries. In Yang, Q., and Wooldridge, M., eds., *Proc. 24th Int. Joint Conf. on Artificial Intelligence (IJCAI'15)*, 3193–3199. AAAI Press.

Zhang, H.; Zhang, Y.; and You, J. 2015. Existential rule languages with finite chase: Complexity and expressiveness. In Bonet, B., and Koenig, S., eds., *Proc. 29th AAAI Conf. on Artificial Intelligence (AAAI'15)*. AAAI Press.

A Proofs for Section 3

In this section, we prove the claims made in Section 3. In particular, besides proving Lemmas 5, 6, 7, 8, 9, 10, and 11, we show Lemma 26 from which the next claims (given in the order they occur in the section) follow.

1. $|\mathfrak{I}_1| = |\mathfrak{I}_2| = |\mathfrak{I}_3| = |\mathfrak{I}_4| = |\mathfrak{I}_5|$.
2. For each $1 \leq i < j \leq 5$ and each $\mathcal{I} \in \mathfrak{I}_i$, there is exactly one $\mathcal{J} \in \mathfrak{I}_j$ with $\mathcal{I} \subseteq \mathcal{J}$.
3. Every $\mathcal{I} \in \mathfrak{I}_i$ with $1 \leq i \leq 5$ contains a unique interpretation $\mathit{Seed}(\mathcal{I}) \in \mathfrak{I}_1$, and there is a unique ordered partition $\mathit{Order}(\mathcal{I}) \in \mathit{Ords}$ such that $\mathit{Seed}(\mathcal{I}) \in \mathit{Compls}(\mathit{Order}(\mathcal{I}))$.

Recall that $\Delta = \mathsf{Nulls}(\mathcal{D}) \cup \{u_\alpha, u_\omega\}$. We will call a *$\mathcal{D}$-order* a database over schema $\{\mathtt{First}, \mathtt{Last}, \mathtt{LT}, \mathtt{Eq}, \mathtt{NEq}\}$ and nulls from Δ, and a *$\mathcal{D}$-completion* a database over schema $\{\mathtt{In}_\mathtt{p}, \mathtt{NIn}_\mathtt{p} \mid \mathtt{p} \in \mathcal{S}\}$ and nulls from Δ. We say that a $\mathcal{D}$-completion is *complete* if (i) it includes $\{\mathtt{In}_\mathtt{p}(\vec{t}) \mid \mathtt{p}(\vec{t}) \in \mathcal{D}\}$ and (ii) for every $\mathtt{p} \in \mathcal{S}$ and tuple $\vec{t} \subseteq \Delta$ of matching arity, it contains exactly one of $\{\mathtt{In}_\mathtt{p}(\vec{t}), \mathtt{NIn}_\mathtt{p}(\vec{t})\}$.

Lemma 26. *There is a one-to-one correspondence between the interpretations in $\mathfrak{I}_i$ ($1 \leq i \leq 5$) and the pairs of the form $(\vec{T}, \mathcal{F}_\mathcal{D})$, where $\vec{T} = T_1, \ldots, T_k \in \mathit{Ords}$ is an ordered partition of Δ with $u_\alpha \in T_1$ and $u_\omega \in T_k$, and $\mathcal{F}_\mathcal{D}$ is a complete $\mathcal{D}$-completion. More precisely:*

- *An interpretation $\mathcal{I} \in \mathfrak{I}_1$ corresponds to a pair $(\vec{T}, \mathcal{F}_\mathcal{D})$ iff $\mathcal{I}$ is the union of the following databases:*
  - $\mathcal{D} \cup \{\mathtt{DbDom}(t) \mid t \in \Delta\}$,
  - *the $\mathcal{D}$-order corresponding to $\vec{T}$:* $\{\mathtt{First}(t) \mid t \in T_1\} \cup \{\mathtt{Last}(t) \mid t \in T_k\} \cup \{\mathtt{Eq}(t, u) \mid 1 \leq i \leq k; t, u \in T_i\} \cup \{\mathtt{LT}(t, u), \mathtt{NEq}(t, u), \mathtt{NEq}(u, t) \mid 1 \leq i < j \leq k, t \in T_i, u \in T_j\}$*, and*
  - $\mathcal{F}_\mathcal{D}$.
- *An interpretation $\mathcal{I} \in \mathfrak{I}_i$ ($2 \leq i \leq 5$) corresponds to $(\vec{T}, \mathcal{F}_\mathcal{D})$ iff it includes some $\mathcal{I}_1 \in \mathfrak{I}_1$ that corresponds to $(\vec{T}, \mathcal{F}_\mathcal{D})$.*

In particular, it follows that every $\mathcal{I} \in \mathfrak{I}_i$ with $1 \leq i \leq 5$ contains a unique interpretation $\mathit{Seed}(\mathcal{I}) \in \mathfrak{I}_1$, and there is a unique ordered partition $\mathit{Order}(\mathcal{I}) \in \mathit{Ords}$ such that $\mathit{Seed}(\mathcal{I}) \in \mathit{Compls}(\mathit{Order}(\mathcal{I}))$.

Proof. The correspondence between interpretations in $\mathfrak{I}_1$ and pairs of the form $(\vec{T}, \mathcal{F}_\mathcal{D})$ is well defined because $\mathfrak{I}_1 = \bigcup_{\vec{T} \in \mathit{Ords}} \mathit{Compls}(\vec{T})$ where $\mathit{Compls}(\vec{T})$ is the set of all minimal models of $\mathcal{R}_1$ and $\mathcal{D}$ that contain the $\mathcal{D}$-order corresponding to $\vec{T}$. In particular, every $\mathcal{I} \in \mathfrak{I}_1$ must include a complete $\mathcal{D}$-completion to satisfy (14) and (4), and by minimality $\mathcal{I}$ does not include any set of the form $\{\mathtt{In}_\mathtt{p}(\vec{t}), \mathtt{NIn}_\mathtt{p}(\vec{t})\}$. Moreover, by minimality, $\mathcal{I}$ does not contain any other atom over predicates $\{\mathtt{First}, \mathtt{Last}, \mathtt{LT}, \mathtt{Eq}, \mathtt{NEq}\}$.

Each $\mathfrak{I}_i$ ($2 \leq i \leq 5$) is obtained from $\mathfrak{I}_{i-1}$ by applying the function Int_i to each of its elements and by construction of Int_i, $\mathcal{I} \subseteq \mathit{Int}_i(\mathcal{I})$ and $\mathit{Int}_i(\mathcal{I}) \setminus \mathcal{I}$ does not contain any atoms on predicates from $\mathcal{S} \cup \{\mathtt{DbDom}, \mathtt{First}, \mathtt{Last}, \mathtt{LT}, \mathtt{Eq}, \mathtt{NEq}\} \cup \{\mathtt{In}_\mathtt{p}, \mathtt{NIn}_\mathtt{p} \mid \mathtt{p} \in \mathcal{S}\}$. Hence the one-to-one correspondence between elements of $\mathfrak{I}_1$ and pairs of the form $(\vec{T}, \mathcal{F}_\mathcal{D})$ is preserved when applying Int_i to elements of $\mathfrak{I}_{i-1}$ to obtain the elements of $\mathfrak{I}_i$ for each $2 \leq i \leq 5$. ☐

Lemma 5. $\mathfrak{I}_1 = \bigcup_{\vec{T} \in \mathit{Ords}} \mathit{Compls}(\vec{T})$ *is a universal model set of $\mathcal{R}_1$ and $\mathcal{D}$.*

Proof. Let $\mathfrak{K}$ be the result of some chase tree for $\mathcal{R}_1$ and $\mathcal{D}$ that prioritises the application of rules in the order of their appearance in Figure 1. By Fact 2, $\mathfrak{K}$ is a universal model set for $\mathcal{R}_1$ and $\mathcal{D}$ and we show that each $\mathcal{K} \in \mathfrak{K}$ is isomorphic to a unique $\mathcal{I} \in \mathfrak{I}_1$.

We use the one-to-one correspondence between the interpretations in $\mathfrak{I}_1$ and the pairs of the form $(\vec{T}, \mathcal{F}_\mathcal{D})$ described in Lemma 26. Since $\mathfrak{K}$ is a result of the chase on $\mathcal{R}_1$ and $\mathcal{D}$, the domain of every $\mathcal{K} \in \mathfrak{K}$ contains all nulls in $\mathsf{Nulls}(\mathcal{D})$ and exactly two fresh nulls introduced to satisfy rules (2) and (3), respectively. We name them u_α and u_ω respectively, so that Δ is the domain of $\mathcal{K}$, and define a one-to-one correspondence between the interpretations in $\mathfrak{K}$ and the pairs of the form $(\vec{T}, \mathcal{F}_\mathcal{D})$ as in Lemma 26. This can be done because of the following points:

- Every $\mathcal{K} \in \mathfrak{K}$ includes $\mathcal{D}$ and does not contain any other atom on predicates from $\mathcal{S}$ because such predicates do not occur in rule heads.
- Every $\mathcal{K} \in \mathfrak{K}$ includes $\{\texttt{DbDom}(t) \mid t \in \Delta\}$ and does not contain any other atom on predicate $\texttt{DbDom}$ (since there are no more nulls).
- Every $\mathcal{K} \in \mathfrak{K}$ includes a $\mathcal{D}$-order corresponding to some $\vec{T} \in \mathit{Ords}$ and does not contain any other atom on predicates from $\{\texttt{First}, \texttt{Last}, \texttt{LT}, \texttt{Eq}, \texttt{NEq}\}$:
  - Let $\sim$ be the relation over Δ defined by $t \sim t'$ iff $\texttt{Eq}(t, t') \in \mathcal{K}$. This is an equivalence relation because (i) $\{\texttt{DbDom}(t) \mid t \in \Delta\} \subseteq \mathcal{K}$ and $\mathcal{K}$ satisfies (5) (reflexivity); (ii) $\mathcal{K}$ satisfies (6) (symmetry); and (iii) $\mathcal{K}$ satisfies (8) instantiated for $\texttt{R} = \texttt{Eq}$ (transitivity). The equivalence classes form a partition $T_1, \ldots, T_k$ of Δ such that $\{\texttt{Eq}(t, u) \mid 1 \leq i \leq k; t, u \in T_i\} \subseteq \mathcal{K}$.
  - For every $u, t \in \Delta$, $\mathcal{K}$ contains at least one of $\{\texttt{Eq}(u, t), \texttt{NEq}(u, t)\}$ because it satisfies (9). Moreover, $\mathcal{K}$ cannot contain both $\texttt{Eq}(u, t)$ and $\texttt{NEq}(u, t)$ because the rules (5), (6) and (8) have a higher priority than (9). Hence $\{\texttt{NEq}(t, u), \texttt{NEq}(u, t) \mid 1 \leq i < j \leq k, t \in T_i, u \in T_j\} \subseteq \mathcal{K}$.
  - $\mathcal{K}$ includes $\{\texttt{First}(u) \mid u \sim u_\alpha\}$ and $\{\texttt{Last}(u) \mid u \sim u_\omega\}$ by definition of u_α, u_ω and because it satisfies (8). Assuming w.l.o.g. that T_1 is the equivalence class of u_α and T_k that of u_ω: $\{\texttt{First}(t) \mid t \in T_1\} \cup \{\texttt{Last}(t) \mid t \in T_k\} \subseteq \mathcal{K}$.
  - $\mathcal{K}$ does not include any other atom over $\texttt{First}$ or $\texttt{Last}$ because rules (2) and (3) are applied only once.
  - For every $u, t \in \Delta$ such that $\texttt{NEq}(u, t) \in \mathcal{K}$, $\mathcal{K}$ contains at least one of $\{\texttt{LT}(u, t), \texttt{LT}(t, u)\}$ because it satisfies (13).
  - For every $u, t \in \Delta$, if $\texttt{First}(u) \in \mathcal{K}$ (i.e. $u \sim u_\alpha$) and $\texttt{First}(t) \notin \mathcal{K}$ (i.e. $t \not\sim u_\alpha$) then $\texttt{LT}(u, t) \in \mathcal{K}$ because of (11).
  - For every $u, t \in \Delta$, if $\texttt{Last}(u) \in \mathcal{K}$ (i.e. $u \sim u_\omega$) and $\texttt{Last}(t) \notin \mathcal{K}$ (i.e. $t \not\sim u_\omega$) then $\texttt{LT}(t, u) \in \mathcal{K}$ because of (12).
  - If $\texttt{LT}(u, t) \in \mathcal{K}$ then $\texttt{LT}(u', t') \in \mathcal{K}$ for all $u' \sim u$ and $t \sim t'$ because $\mathcal{K}$ satisfies (8).
  - $\mathcal{K}$ contains the transitive closure of $\texttt{LT}$ because it satisfies (10).
  - Since the rules (11), (12), (5), (6), (8), and (10) have a higher priority than (13), then $\mathcal{K}$ cannot contain both $\texttt{LT}(u, t)$ and $\texttt{LT}(t, u)$.
  - Hence $T_1, \ldots, T_k$ can be ordered in a way such that $\{\texttt{LT}(t, u) \mid 1 \leq i < j \leq k, t \in T_i, u \in T_j\} \subseteq \mathcal{K}$ and the ordered partition corresponds to some $\vec{T} \in \mathit{Ords}$ as required.
- Every $\mathcal{K} \in \mathfrak{K}$ contains a complete $\mathcal{D}$-completion $\mathcal{F}_\mathcal{D}$ and does not contain any other atom on predicates from $\{\texttt{In}_\texttt{p}, \texttt{NIn}_\texttt{p} \mid \texttt{p} \in \mathcal{S}\}$:
  - $\mathcal{K}$ includes $\{\texttt{In}_\texttt{p}(\vec{t}) \mid \texttt{p}(\vec{t}) \in \mathcal{D}\}$ because it satisfies (4).
  - For every $\texttt{p} \in \mathcal{S}$ and tuple $\vec{t} \subseteq \Delta$ of matching arity, $\mathcal{K}$ contains at least one of $\{\texttt{In}_\texttt{p}(\vec{t}), \texttt{NIn}_\texttt{p}(\vec{t})\}$ because it satisfies (14). We show that it does not contain both. The rules of the form (14) have the lowest priority and applying such a rule can only trigger the application of some rules of the form (8), which themselves cannot trigger the application of any rules but (8). Hence, (i) the application of these rules in the chase tree happens after the construction of the $\mathcal{D}$-order in $\mathcal{K}$, and (ii) when a rule of the form (14) is applied and adds some $\texttt{In}_\texttt{p}(\vec{t})$ (resp. $\texttt{NIn}_\texttt{p}(\vec{t})$), the exhaustive application of rules (8) adds all $\texttt{In}_\texttt{p}(\vec{t'})$ (resp. $\texttt{NIn}_\texttt{p}(\vec{t'})$) such that $\vec{t'} \sim \vec{t}$, where $\vec{t'} \sim \vec{t}$ iff $t_i \sim t'_i$ for every i. Suppose for a contradiction that there exists $\vec{u}$ such that $\texttt{In}_\texttt{p}(\vec{u}) \in \mathcal{K}$ and $\texttt{NIn}_\texttt{p}(\vec{u}) \in \mathcal{K}$. Since the rules (14) and (8) are the only ones that have predicates from $\{\texttt{In}_\texttt{p}, \texttt{NIn}_\texttt{p} \mid \texttt{p} \in \mathcal{S}\}$ in the head, it follows that there exist $\texttt{In}_\texttt{p}(\vec{u}_1) \in \mathcal{K}$ and $\texttt{NIn}_\texttt{p}(\vec{u}_2) \in \mathcal{K}$ that have both been added by some application of (14) and are

such that $\vec{u}_1 \sim \vec{u}$ and $\vec{u}_2 \sim \vec{u}$. Assume that $\mathtt{In}_{\mathtt{p}}(\vec{u}_1)$ was added first. Then by (ii) $\mathtt{In}_{\mathtt{p}}(\vec{u}_2)$ has been added before any new application of (14), and the rule $\bigwedge_{x \in \vec{x}} \mathtt{DbDom}(x) \rightarrow \mathtt{In}_{\mathtt{p}}(\vec{x}) \vee \mathtt{NIn}_{\mathtt{p}}(\vec{x})$ with the substitution that maps $\vec{x}$ to $\vec{u}_2$ was already satisfied, contradicting the introduction of $\mathtt{NIn}_{\mathtt{p}}(\vec{u}_2)$.

- For every $(\vec{T}, \mathcal{F}_{\mathcal{D}})$, there exists $\mathcal{K} \in \mathfrak{K}$ that corresponds to $(\vec{T}, \mathcal{F}_{\mathcal{D}})$. The existence of $\mathcal{K}$ is witnessed by the path of the chase tree such that for every vertex labelled with $\mathcal{E}$ where a disjunctive rule (9), (13), or (14) is applied, the path chooses the child labelled with $\mathcal{C}_i = \mathcal{E} \cup \sigma_i(\eta_i)$ where $\sigma_i(\eta_i)$ is included in $\mathcal{F}_{\mathcal{D}}$ or in the $\mathcal{D}$-order corresponding to $\vec{T}$.

Hence, we can define a one-to-one correspondence between interpretations in $\mathfrak{K}$ and interpretations in $\mathfrak{I}_1$ using the corresponding pairs of the form $(\vec{T}, \mathcal{F}_{\mathcal{D}})$. By construction, every $\mathcal{K} \in \mathfrak{K}$ and $\mathcal{I} \in \mathfrak{I}_1$ that are in correspondence are isomorphic (note that since we choose to name the two fresh nulls u_α and u_ω, they are actually identical). □

Lemma 27. *Given some $\mathcal{I} \in \mathfrak{I}_1$, $Int_2(\mathcal{I})$ is isomorphic to the unique interpretation in a result of the chase over $\mathcal{R}_2$ and $\mathcal{I}$.*

Proof. We will let $Order(\mathcal{F}) = T_1, \ldots, T_k$. The interpretation $Int_2(\mathcal{I})$ is equal to:

$$\begin{aligned}
&\mathcal{I} \cup \{\mathtt{Root}(u_1)\} \cup \{\mathtt{Chi}(u_w, u_{wz}) \mid wz \in \mathbf{Z}, 2 \leq z \leq k\} \cup \\
&\{\mathtt{Leaf}(u_w) \mid w \in \mathbf{Z}; end(w) = k\} \cup \{\mathtt{Rep}(x, u_w) \mid w \in \mathbf{Z}; x \in T_{end(w)}\} \cup \\
&\{\mathtt{In}'_{\mathtt{p}}(u_{w_1}, \ldots, u_{w_{Ar(\mathtt{p})}}) \mid \mathtt{In}_{\mathtt{p}}(t_1, \ldots, t_{Ar(\mathtt{p})}) \in \mathcal{I}; w_i \in \mathbf{Z}; t_i \in T_{end(w_i)}; 1 \leq i \leq Ar(\mathtt{p})\} \cup \\
&\{\mathtt{NIn}'_{\mathtt{p}}(u_{w_1}, \ldots, u_{w_{Ar(\mathtt{p})}}) \mid \mathtt{NIn}_{\mathtt{p}}(t_1, \ldots, t_{Ar(\mathtt{p})}) \in \mathcal{I}; w_i \in \mathbf{Z}; t_i \in T_{end(w_i)}; 1 \leq i \leq Ar(\mathtt{p})\}.
\end{aligned}$$

We describe a chase over $\mathcal{R}_2$ and $\mathcal{I}$ that prioritises rule (18) step by step (note that if we do not prioritise (18), we could get a result which is not isomorphic to $Int_2(\mathcal{I})$). Instead of constructing an isomorphism by giving for each fresh null introduced during the chase its counterpart in $Int_2(\mathcal{I})$, we directly set the the names of the fresh nulls so that the chase result coincides with $Int_2(\mathcal{I})$. Note that no rule in Figure 1 is applicable to $\mathcal{I}$ and that applying a rule in Figure 2 cannot make a rule in Figure 1 applicable.

- Apply $\langle \rho, \sigma \rangle$ with $\rho =$ (15) and σ the substitution that maps x to some $t \in T_1$ (recall that $\{\mathtt{First}(t) \mid t \in T_1\} \subseteq \mathcal{I}$). This introduces a fresh null that we call u_1, and adds $\mathtt{Root}(u_1)$ and $\mathtt{Rep}(t, u_1)$.
- Apply exhaustively (18): add $\mathtt{Rep}(t, u_1)$ for all $t \in T_1$.
- For $2 \leq i \leq k$:
  - Choose one $t_1 \in T_1$ and one $t_i \in T_i$ and apply $\langle \rho, \sigma \rangle$ with $\rho =$ (16) and σ the substitution that maps x to t_1, v to u_1, and z to t_i (recall that the atoms on predicate $\mathtt{LT}$ in $\mathcal{I}$ are $\{\mathtt{LT}(t, u) \mid 1 \leq j < j' \leq k, t \in T_j, u \in T'_j\}$). This introduces a fresh null that we call $u_{1\,i}$ and adds $\mathtt{Chi}(u_1, u_{1\,i})$, $\mathtt{Rep}(t_i, u_{1\,i})$ (note that this is the first fact of the form $\mathtt{Rep}(t_i, x)$ with $t_i \in T_i$ that we introduce).
  - Apply exhaustively (18): add $\mathtt{Rep}(t, u_{1\,i})$ for all $t \in T_i$.
- Continue applying (16) and (18) exhaustively as done in the previous step, introducing all nulls of the form u_w with $w \in \mathbf{Z}$, and adding facts $\{\mathtt{Chi}(u_w, u_{wz}) \mid wz \in \mathbf{Z}, 2 \leq z \leq k\}$ and $\{\mathtt{Rep}(x, u_w) \mid w \in \mathbf{Z}; x \in T_{end(w)}\}$.
- Apply exhaustively (17) and add $\{\mathtt{Leaf}(u_w) \mid w \in \mathbf{Z}; end(w) = k\}$ (recall that $\{\mathtt{Last}(t) \mid t \in T_k\} \subseteq \mathcal{I}$).
- Apply exhaustively (19): for all $\mathtt{In}_{\mathtt{p}}(t_1, \ldots, t_{Ar(\mathtt{p})}) \in \mathcal{I}$, add all $\mathtt{In}'_{\mathtt{p}}(u_{w_1}, \ldots, u_{w_{Ar(\mathtt{p})}})$ where $t_i \in T_{end(w_i)}$.
- Apply exhaustively (20): for all $\mathtt{NIn}_{\mathtt{p}}(t_1, \ldots, t_{Ar(\mathtt{p})}) \in \mathcal{I}$, add all $\mathtt{NIn}'_{\mathtt{p}}(u_{w_1}, \ldots, u_{w_{Ar(\mathtt{p})}})$ where $t_i \in T_{end(w_i)}$. □

Lemma 6. $\mathfrak{I}_2 = \{Int_2(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_1\}$ *is a universal model set of* $\mathcal{R}_2$ *and* $\mathcal{D}$.

Proof. By Lemmas 5 and 27, $\mathfrak{I}_2$ only contains models of $\mathcal{R}_2$ and $\mathcal{D}$. We show that for every model $\mathcal{M}$ of $\mathcal{R}_2$ and $\mathcal{D}$, there exists $\mathcal{I}_2 \in \mathfrak{I}_2$ and a homomorphism $h_2 : \mathcal{I}_2 \to \mathcal{M}$.

1. Let $\mathcal{M}$ be a model of $\mathcal{R}_2$ and $\mathcal{D}$.
2. By Lemma 5: since $\mathcal{M}$ is a model of $\mathcal{R}_2 \supseteq \mathcal{R}_1$ and $\mathcal{D}$, there exist $\mathcal{I}_1 \in \mathfrak{I}_1$ and a homomorphism $h_1 : \mathcal{I}_1 \to \mathcal{M}$.
3. By Lemma 27: since $\mathcal{I}_1 \in \mathfrak{I}_1$, there exists a chase over $\mathcal{R}_2$ and $\mathcal{I}_1$ such that the unique interpretation $\mathcal{J}$ in its result is such that there exists an isomorphism $g_2 : Int_2(\mathcal{I}_1) \to \mathcal{J}$.
4. By definition of $\mathcal{J}$, and since no rule from $\mathcal{R}_1$ is applied in the chase of over $\mathcal{R}_2$ and $\mathcal{I}_1$ (see proof of Lemma 27) and $\mathcal{R}_2 \setminus \mathcal{R}_1$ contains no disjunctive rules, it follows that $\mathcal{J}$ is a universal model of $\mathcal{R}_2 \setminus \mathcal{R}_1$ and $\mathcal{I}_1$.
5. By (1), (2) and (4), there exists a homomorphism $g_1 : \mathcal{J} \to \mathcal{M}$ that extends h_1.
6. Let $\mathcal{I}_2 = Int_2(\mathcal{I}_1) \in \mathfrak{I}_2$. By (3) and (5), there is an homomorphism $h_2 = g_2 \circ g_1 : \mathcal{I}_2 \to \mathcal{M}$. $\square$

In the next proofs, we will use the following notation. Given a chase tree node labelled $\mathcal{C}_i = \mathcal{E} \cup \sigma_i(\eta_i)$ and obtained by applying some $\langle \rho, \sigma \rangle$ to its parent node labelled by $\mathcal{E}$, then for each variable y of ρ not in the domain of σ, we denote by $u_{\sigma,\rho,y}$ the fresh null such that $\sigma_i(y) = u_{\sigma,\rho,y}$ where σ_i is as introduced in Definition 2.

Lemma 28. *Given some* $\mathcal{I} \in \mathfrak{I}_2$*,* $Int_3(\mathcal{I})$ *is well-defined and* $Int_3(\mathcal{I})$ *is isomorphic to the unique interpretation in a result of the chase over* $\mathcal{R}_3$ *and* $\mathcal{I}$*.*

Proof. We will let $Order(\mathcal{I}) = T_1, \ldots, T_k$. The interpretation $Int_3(\mathcal{I})$ is equal to the union of the following databases:

1. $\mathcal{I}$.
2. For each $w \in \mathbf{Z}$ of length $|w|$, with $b_1 \cdots b_\ell$ the binary representation of $|w| + 1$, the database $EncPos(w)$:
$$\{\mathtt{Enc}(u_w, e_w^1, e_w^\ell)\} \cup \{\mathtt{S}_{b_i}(e_w^i) \mid 1 \leq i \leq \ell\} \cup \{\mathtt{Nxt}(e_w^{i-1}, e_w^i) \mid 2 \leq i \leq \ell\}.$$
3. For each $w \in \mathbf{Z}$ of length $|w| < k$, and $z \in \{1, \ldots, k\}$ such that $wz \in \mathbf{Z}$, with $b_1 \cdots b_\ell$ the binary representation of $|w| + 2$ and $b'_1 \cdots b'_{\ell'}$ the binary representation of $|w| + 1$, the database $CpyPlusOne(w, wz)$ defined as follows:
$$\{\mathtt{Cpy}_{+1}(e_w^i, e_w^{\ell'}, e_{wz}^j, e_{wz}^\ell) \mid$$
$$\text{the number represented by } b_j \cdots b_\ell \text{ is equal to the number represented by } b'_i \cdots b'_{\ell'} \text{ plus one}\}$$
$$\cup \{\mathtt{Cpy}(e_w^i, e_w^{\ell'}, e_{wz}^i, e_{wz}^\ell) \mid b_i \cdots b_\ell = b'_i \cdots b'_{\ell'}\}.$$

We describe a (Datalog-first) chase over $\mathcal{R}_3$ and $\mathcal{I}$ step by step. We directly set the the names of the fresh nulls so that the chase result coincides with $Int_3(\mathcal{I})$. Note that no rule in Figures 1 or 2 is applicable to $\mathcal{I}$ and that applying a rule in Figure 3 cannot make a rule in Figures 1 or 2 applicable.

- Apply $\langle \rho, \sigma \rangle$ with $\rho = (21)$ and $\sigma(u) = u_1$. This introduces two fresh nulls, $u_{\sigma,\rho,y_1} = e_1^1$ and $u_{\sigma,\rho,y_2} = e_1^2$ and adds $\mathtt{Enc}(u_1, e_1^1, e_1^2)$, $\mathtt{S}_0(e_1^1)$, $\mathtt{Nxt}(e_1^1, e_1^2)$ and $\mathtt{S}_1(e_1^2)$.

We show by induction on p that for every $w \in \mathbf{Z}$ of length $|w| = p$ and $z \in \{1, \ldots, k\}$ such that $wz \in \mathbf{Z}$, our chase sequence adds $EncPos(wz)$, and $CpyPlusOne(w, wz)$.

Case $p = 1$. In this case, $w = 1$ and $z \in \{2, \ldots, k\}$.

- For $2 \leq i \leq k$:

– Apply $\langle\rho,\sigma\rangle$ with $\rho = (22)$ and $\sigma(u) = u_1$, $\sigma(y_1) = e_1^1$, $\sigma(y_\_) = e_1^2$, $\sigma(v) = u_{1\,i}$. This introduces two fresh nulls, $u_{\sigma,\rho,z_1} = e_{1\,i}^1$ and $u_{\sigma,\rho,z_\_} = e_{1\,i}^2$ and adds $\mathtt{Enc}(u_{1\,i}, e_{1\,i}^1, e_{1\,i}^2)$ and $\mathtt{Cpy}_{+1}(e_1^1, e_1^2, e_{1\,i}^1, e_{1\,i}^2)$.

– Apply (23): add $\mathtt{S}_1(e_{1\,i}^1)$, $\mathtt{Nxt}(e_{1\,i}^1, e_{1\,i}^2)$, $\mathtt{S}_1(e_{1\,i}^2)$.

Induction step: Assume that our chase sequence already added all required facts for every $w' \in \mathbf{Z}$ of length $|w'| < p$, and $z' \in \{1,\dots,k\}$ such that $w'z' \in \mathbf{Z}$, and consider $w \in \mathbf{Z}$ of length $|w| = p$, and $z \in \{1,\dots,k\}$ such that $wz \in \mathbf{Z}$. Let $b_1 \cdots b_\ell$ the binary representation of $p+2$ and $b'_1 \cdots b'_{\ell'}$ the binary representation of $p+1$.

- By induction hypothesis, $\mathtt{Enc}(u_w, e_w^1, e_w^{\ell'})$ was added by the chase sequence. Apply $\langle\rho,\sigma\rangle$ with $\rho = (22)$ and $\sigma(u) = u_w$, $\sigma(y_1) = e_w^1$, $\sigma(y_\_) = e_w^{\ell'}$, $\sigma(v) = u_{wz}$. This introduces two fresh nulls, $u_{\sigma,\rho,z_1} = e_{wz}^1$ and $u_{\sigma,\rho,z_\_} = e_{wz}^\ell$ and adds $\mathtt{Enc}(u_{wz}, e_{wz}^1, e_{wz}^\ell)$ and $\mathtt{Cpy}_{+1}(e_w^1, e_w^{\ell'}, e_{wz}^1, e_{wz}^\ell)$.
- By induction hypothesis, the chase sequence added $\mathtt{S}_{b'_i}(e_w^i)$, $1 \le i \le \ell'$, and $\mathtt{Nxt}(e_w^{i-1}, e_w^i)$, $2 \le i \le \ell'$. We distinguish two cases, depending on the value of b'_1.
  (1) In the case where $b'_1 = 0$, we have $\ell = \ell'$ and $b_1 \cdots b_\ell = 1b'_2 \cdots b'_\ell$. Moreover, note that $\ell > 2$ (otherwise $b'_1 b'_2 = 01$ and $p+1 = 2$).
  – Apply $\langle\rho,\sigma\rangle$ with $\rho = (25)$ and $\sigma(y_1) = e_w^1$, $\sigma(y_\_) = e_w^\ell$, $\sigma(y_2) = e_w^2$, $\sigma(y_3) = e_w^3$, $\sigma(z_1) = e_{wz}^1$, $\sigma(z_\_) = e_{wz}^\ell$. This introduces a fresh null, $u_{\sigma,\rho,z_2} = e_{wz}^2$, and adds the atoms $\mathtt{Cpy}(e_w^2, e_w^\ell, e_{wz}^2, e_{wz}^\ell)$, $\mathtt{S}_1(e_{wz}^1)$, and $\mathtt{Nxt}(e_{wz}^1, e_{wz}^2)$.
  – While $\mathtt{Cpy}(e_w^{\ell-1}, e_w^\ell, e_{wz}^{\ell-1}, e_{wz}^\ell)$ has not been introduced, apply (28), introducing fresh nulls e_{wz}^i for $2 < i < \ell$ and adding atoms $\mathtt{Cpy}(e_w^i, e_w^\ell, e_{wz}^i, e_{wz}^\ell)$, $\mathtt{S}_{b_{i-1}}(e_{wz}^{i-1})$, and $\mathtt{Nxt}(e_{wz}^{i-1}, e_{wz}^i)$.
  – Apply (27) and add $\mathtt{S}_{b_{\ell-1}}(e_{wz}^{\ell-1})$, $\mathtt{Nxt}(e_{wz}^{\ell-1}, e_{wz}^\ell)$, $\mathtt{S}_1(e_{wz}^\ell)$.
  (2) In the case where $b'_1 = 1$, we have $b_1 = 0$ and the number represented by $b_2 \cdots b_\ell$ is equal to the number represented by $b'_2 \cdots b'_{\ell'}$ plus one.
  – While no atom of the form $\mathtt{Cpy}(e_w^j, e_w^{\ell'}, e_{wz}^j, e_{wz}^\ell)$ or $\mathtt{Cpy}_{+1}(e_w^{\ell'-1}, e_w^{\ell'}, e_{wz}^{\ell'-1}, e_{wz}^\ell)$ have been introduced, and for i starting from 2 and growing by one at each cycle, apply the applicable $\langle\rho,\sigma\rangle$ among:
    * $\rho = (25)$ and $\sigma(y_1) = e_w^{i-1}$, $\sigma(y_\_) = e_w^{\ell'}$, $\sigma(y_2) = e_w^i$, $\sigma(y_3) = e_w^{i+1}$, $\sigma(z_1) = e_{wz}^{i-1}$, $\sigma(z_\_) = e_{wz}^\ell$. This introduces a fresh null, $u_{\sigma,\rho,z_2} = e_{wz}^i$, and adds the atoms $\mathtt{Cpy}(e_w^i, e_w^{\ell'}, e_{wz}^i, e_{wz}^\ell)$, $\mathtt{S}_1(e_{wz}^{i-1}) = \mathtt{S}_{b_{i-1}}(e_{wz}^{i-1})$, and $\mathtt{Nxt}(e_{wz}^{i-1}, e_{wz}^i)$.
    * $\rho = (26)$ and σ as above. This introduces a fresh null, $u_{\sigma,\rho,z_2} = e_{wz}^i$, and adds the atoms $\mathtt{Cpy}_{+1}(e_w^i, e_w^{\ell'}, e_{wz}^i, e_{wz}^\ell)$, $\mathtt{S}_0(e_{wz}^{i-1}) = \mathtt{S}_{b_{i-1}}(e_{wz}^{i-1})$, and $\mathtt{Nxt}(e_{wz}^{i-1}, e_{wz}^i)$.
  – If an atom of the form $\mathtt{Cpy}(e_w^j, e_w^{\ell'}, e_{wz}^j, e_{wz}^\ell)$ is introduced at some point (i.e. the applicable $\langle\rho,\sigma\rangle$ in the previous step is such that $\rho = (25)$), we use a sequence of chase steps similar to the case $b'_1 = 0$, and obtain the required atoms.
  – Otherwise, $\mathtt{Cpy}_{+1}(e_w^{\ell'-1}, e_w^{\ell'}, e_{wz}^{\ell'-1}, e_{wz}^\ell)$ is introduced at the end of the loop. Apply the applicable $\langle\rho,\sigma\rangle$ among:
    * $\rho = (23)$ and $\sigma(y_1) = e_w^{\ell'-1}$, $\sigma(y_2) = e_w^{\ell'}$, $\sigma(z_1) = e_{wz}^{\ell'-1}$, $\sigma(z_\_) = e_{wz}^\ell$ with $\ell' = \ell$. This adds $\mathtt{S}_1(e_{wz}^{\ell-1})$, $\mathtt{Nxt}(e_{wz}^{\ell-1}, e_{wz}^\ell)$, and $\mathtt{S}_1(e_{wz}^\ell)$.
    * $\rho = (24)$ and σ is as above but $\ell = \ell'+1$ so that $\sigma(z_1) = e_{wz}^{\ell'-1} = e_{wz}^{\ell-2}$. This introduces a new null $u_{\sigma,\rho,z_2} = e_{wz}^{\ell-1}$ and adds $\mathtt{S}_0(e_{wz}^{\ell-2})$, $\mathtt{Nxt}(e_{wz}^{\ell-2}, e_{wz}^{\ell-1})$, $\mathtt{S}_0(e_{wz}^{\ell-1})$, $\mathtt{Nxt}(e_{wz}^{\ell-1}, e_{wz}^\ell)$ and $\mathtt{S}_1(e_{wz}^\ell)$. □

Lemma 7. *$\mathfrak{I}_3 = \{Int_3(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_2\}$ is a universal model set of $\mathcal{R}_3$ and $\mathcal{D}$.*

Proof. Similar to the proof of Lemma 6, using Lemmas 6 and 28. □

Lemma 29. *Given some $\mathcal{I} \in \mathfrak{I}_3$, $Int_4(\mathcal{I})$ is well-defined and $Int_4(\mathcal{I})$ is isomorphic to the unique interpretation in a result of the chase over $\mathcal{R}_4$ and $\mathcal{I}$.*

Proof. Before giving the extension of $Int_4(\mathcal{I})$, we introduce some notation. Consider some fact of the form $\texttt{Leaf}(u_w) \in \mathcal{I}$ and let $S = branchTape(\mathcal{I}, u_w)$. By construction, $S = e_1 \cdots e_{|branchDb(\mathcal{I},u_w)|}$ where each e_i corresponds to the serialisation of some $\texttt{p}_g^x(u_{w_1}, \ldots, u_{w_x}) \in branchDb(\mathcal{I}, u_w)$ and is of the form $\texttt{p}_g^x \| b_1^1 \cdots b_{\ell_1}^1 \| \cdots \| b_1^x \cdots b_{\ell_x}^x \|$ where $b_1^i \cdots b_{\ell_i}^i$ is the binary representation of $|w_i| + 1$. For every $\texttt{p}_g^x \in \mathcal{S}$ and $u_{w_1}, \ldots, u_{w_x} \subseteq branch(u_w)$, let $j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x})$ and $j_S^e(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) - 1$ be the indexes where the serialisation of $\texttt{p}_g^x(u_{w_1}, \ldots, u_{w_x})$ starts and ends respectively if $\texttt{p}_g^x(u_{w_1}, \ldots, u_{w_x}) \in branchDb(\mathcal{I}, u_w)$, and otherwise $j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) = j_S^e(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x})$ be the index where starts the serialisation of the first fact of $branchDb(\mathcal{I}, u_w)$ that follows $\texttt{p}_g^x(u_{w_1}, \ldots, u_{w_x})$ according to $\prec$ (and $|S| + 1$ if there is no such fact). (Recall that $\prec$ is defined before Lemma 8.) It is easy to verify that:

- $j_S^s(\texttt{p}_1^1, u_w) = 1$.
- If $\texttt{In}'_{\texttt{p}_g^x}(u_{w_1}, \ldots, u_{w_x}) \notin \mathcal{I}$, then $j_S^e(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) = j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x})$.
- If $\texttt{In}'_{\texttt{p}_g^x}(u_{w_1}, \ldots, u_{w_x}) \in \mathcal{I}$, then $j_S^e(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) = j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) + \Sigma_{i=1}^x \ell_i + x + 2$, where ℓ_i is the length of the binary encoding of $|w_i| + 1$.
- If $g < \bar{n}$, then $j_S^e(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) = j_S^s(\texttt{p}_{g+1}^x, u_{w_1}, \ldots, u_{w_x})$.
- If $u_{w_1}, \ldots, u_{w_x} \neq u_1, \ldots, u_1$, then $j_S^e(\texttt{p}_{\bar{n}}^x, u_{w_1}, \ldots, u_{w_x}) = j_S^s(\texttt{p}_1^x, u_{w'_1}, \ldots, u_{w'_x})$, where $u_{w'_1}, \ldots, u_{w'_x}$ comes right after $u_{w_1}, \ldots, u_{w_x}$ in the lexicographic order according to $\prec$.
- If $x < \bar{m}$, then $j_S^e(\texttt{p}_{\bar{n}}^x, u_1, \ldots, u_1) = j_S^s(\texttt{p}_1^{x+1}, u_w, \ldots, u_w)$.

Note that $j_S^e(\texttt{p}_{\bar{n}}^{\bar{m}}, u_1, \ldots, u_1) = |S| + 1$ and that for every $\texttt{p}_g^x(u_{w_1}, \ldots, u_{w_x}) \in branchDb(\mathcal{I}, u_w)$, if $b_1^i \cdots b_{\ell_i}^i$ is the binary encoding of $|w_i| + 1$ ($1 \leq i \leq x$), then :

- $S[j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x})] = \texttt{p}_g^x$,
- $S[j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) + 1] = \|$,
- and for $1 \leq i \leq x$:
  - $S[j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) + i + \Sigma_{j=1}^{i-1} \ell_j + 1] = b_1^i$,
  - $\ldots$,
  - $S[j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) + i + \Sigma_{j=1}^{i-1} \ell_j + \ell_i] = b_{\ell_i}^i$,
  - $S[j_S^s(\texttt{p}_g^x, u_{w_1}, \ldots, u_{w_x}) + i + \Sigma_{j=1}^{i} \ell_j + 1] = \|$.

Finally, let $J_S(u_{w'})$ be the set of indexes of S where starts the sequence $\| b'_1 \cdots b'_{\ell'} \|$ where $b'_1 \cdots b'_{\ell'}$ is the binary representation of $|w'| + 1$. The interpretation $Int_4(\mathcal{I})$ is equal to the union of the following databases:

1. $\mathcal{I}$;
2. For every $\texttt{Leaf}(u_w) \in \mathcal{I}$ with serialisation $S = branchTape(\mathcal{I}, u_w)$, the database $startConf(\mathcal{I}, u_w)$:

$$\{\texttt{Ld}_1(u_w, t_w^1, u_w), \texttt{Hd}_{q_S}(t_w^1), \texttt{End}(t_w^{|S|+1}), \texttt{S}_{\_}(t_w^{|S|+1})\} \cup$$
$$\{\texttt{Nxt}(t_w^{j-1}, t_w^j) \mid 2 \leq j \leq |S| + 1\} \cup \{\texttt{S}_a(t_w^j) \mid 1 \leq j \leq |S|, a = S[j]\}.$$

3. For every $\mathtt{Leaf}(u_w) \in \mathcal{I}$ with serialisation $S = \mathit{branchTape}(\mathcal{I}, u_w)$, the database $\mathit{loadS}(\mathcal{I}, u_w)$ defined as follows:

$$\begin{aligned}
&\{\mathtt{Ld}_x(u_w, t_w^j, u_{w_1}, \dots, u_{w_x}) \mid 1 \leq x \leq \bar{m}; u_{w_i} \in \mathit{branch}(u_w); j = j_S^s(\mathtt{p}_1^x, u_{w_1}, \dots, u_{w_x})\} \\
&\cup\{\mathtt{Rdy}_x(u_w, t_w^j, u_{w_1}, \dots, u_{w_x}) \mid 1 \leq x \leq \bar{m}; u_{w_i} \in \mathit{branch}(u_w); j = j_S^e(\mathtt{p}_{\bar{n}}^x, u_{w_1}, \dots, u_{w_x})\} \\
&\cup\{\mathtt{Ld}_{\mathtt{p}_g^x}(u_w, t_w^j, u_{w_1}, \dots, u_{w_x}) \mid 1 \leq g \leq \bar{n}; 1 \leq x \leq \bar{m}; u_{w_i} \in \mathit{branch}(u_w); j = j_S^s(\mathtt{p}_g^x, u_{w_1}, \dots, u_{w_x})\} \\
&\cup\{\mathtt{Rdy}_{\mathtt{p}_g^x}(u_w, t_w^j, u_{w_1}, \dots, u_{w_x}) \mid 1 \leq g \leq \bar{n}; 1 \leq x \leq \bar{m}; u_{w_i} \in \mathit{branch}(u_w); j = j_S^e(\mathtt{p}_g^x, u_{w_1}, \dots, u_{w_x})\} \\
&\cup\{\mathtt{LdE}(u_{w'}, t_w^{j+1}, t_w^{j+\ell'+2}) \mid u_{w'} \in \mathit{branch}(u_w); b'_1 \cdots b'_{\ell'} \text{ bin. rep. of } |w'|+1; j \in J_S(u_{w'})\} \\
&\cup\{\mathtt{Cpy}(e_{w'}^r, e_{w'}^{\ell'}, t_w^{j+r+1}, t_w^{j+\ell'+1}) \mid u_{w'} \in \mathit{branch}(u_w); b'_1 \cdots b'_{\ell'} \text{ bin. rep. of } |w'|+1; \\
&\qquad 1 \leq r \leq \ell'-1; j-1 \in J_S(u_{w'})\}
\end{aligned}$$

We describe a (Datalog-first) chase over $\mathcal{R}_4$ and $\mathcal{I}$ step by step. We directly set the the names of the fresh nulls so that the chase result coincides with $\mathit{Int}_4(\mathcal{I})$. Note that no rule in Figures 1, 2, or 3 is applicable to $\mathcal{I}$ but that applying rule (30) may trigger the application of the "copy" rules (27) and (28) in Figure3.

For every $\mathtt{Leaf}(u_w) \in \mathcal{I}$ with serialisation $S = \mathit{branchTape}(\mathcal{I}, u_w)$:

- Apply $\langle \rho, \sigma \rangle$ with $\rho = (31)$ and σ that maps u to u_w. This introduces a fresh null that we call t_w^1 and adds atoms $\mathtt{Ld}_1(u_w, t_w^1, u_w)$ and $\mathtt{Hd}_{q_S}(t_w^1)$. Note that $j_S^s(\mathtt{p}_1^1, u_w) = 1$ so that we have indeed added $\mathtt{Ld}_1(u_w, t_w^{j_S^s(\mathtt{p}_1^1, u_w)}, u_w)$.
- Let $\mathit{branch}(u_w) = \{u_{w_1}, \dots, u_{w_p}\}$ with $u_w = u_{w_1} \prec \cdots \prec u_{w_p} = u_1$ (i.e. $|w| = |w_1| > \cdots > |w_p| = 1$). For $1 \leq i \leq p$:
  - Apply (32) to $\mathtt{Ld}_1(u_w, t_w^{j_S^s(\mathtt{p}_1^1, u_{w_i})}, u_{w_i})$ and add $\mathtt{Ld}_{\mathtt{p}_1^1}(u_w, t_w^{j_S^s(\mathtt{p}_1^1, u_{w_i})}, u_{w_i})$.
  - For $1 \leq g \leq \bar{n}$ and $j = j_S^s(\mathtt{p}_g^1, u_{w_i})$:
    * If $\mathtt{NIn}'_{\mathtt{p}_g^1}(u_{w_i}) \in \mathcal{I}$, apply (38) and add $\mathtt{Rdy}_{\mathtt{p}_g^1}(u_w, t_w^j, u_{w_i})$. Note that $j = j_S^s(\mathtt{p}_g^1, u_{w_i}) = j_S^e(\mathtt{p}_g^1, u_{w_i})$.
    * Else, $\mathtt{In}'_{\mathtt{p}_g^1}(u_{w_i}) \in \mathcal{I}$: apply $\langle \rho, \sigma \rangle$ with ρ the instantiation of (29) for $\mathtt{p}_g^1$, and $\sigma(u) = u_w$, $\sigma(t) = t_w^j$ and $\sigma(v) = u_{w_i}$. This introduces three fresh nulls $u_{\rho,\sigma,x_1} = t_w^{j+1}$, $u_{\rho,\sigma,x_2} = t_w^{j+\ell_i+2}$ and $u_{\rho,\sigma,y} = t_w^{j+\ell_i+3}$, where $b_1^i \cdots b_{\ell_i}^i$ is the binary representation of $|w_i|+1$. This adds atoms $\mathtt{S}_{\mathtt{p}_g^1}(t_w^j)$, $\mathtt{Nxt}(t_w^j, t_w^{j+1})$, $\mathtt{LdE}(u_{w_i}, t_w^{j+1}, t_w^{j+\ell_i+2})$, $\mathtt{Nxt}(t_w^{j+\ell_i+2}, t_w^{j+\ell_i+3})$ and $\mathtt{Rdy}_{\mathtt{p}_g^1}(u_w, t_w^{j+\ell_i+3}, u_{w_i})$. Note that $j + \ell_i + 3 = j_S^s(\mathtt{p}_g^1, u_{w_i}) + \ell_i + 3 = j_S^e(\mathtt{p}_g^1, u_{w_i})$.
    * If $g < \bar{n}$: Apply (33) and add $\mathtt{Ld}_{\mathtt{p}_{g+1}^1}(u_w, t_w^{j'}, u_{w_i})$ with $j' = j$ if $\mathtt{NIn}'_{\mathtt{p}_g^1}(u_{w_i}) \in \mathcal{I}$, $j' = j + \ell_i + 3$ otherwise. Note that $j' = j_S^e(\mathtt{p}_g^1, u_{w_i}) = j_S^s(\mathtt{p}_{g+1}^1, u_{w_i})$, independently of whether $\mathtt{In}'_{\mathtt{p}_g^1}(u_{w_i})$ is in $\mathcal{I}$.
  - Apply (34) and add $\mathtt{Rdy}_1(u_w, t_w^{j_S^e(\mathtt{p}_{\bar{n}}^1, u_{w_i})}, u_{w_i})$.
  - If $i < p$ (i.e. $u_{w_i} \neq u_1$): Apply the instantiation of (35) $\mathtt{Rdy}_1(u, t, v) \wedge \mathtt{Chi}(w, v) \to \mathtt{Ld}_1(u, t, w)$ and add $\mathtt{Ld}_1(u_w, t_w^{j'}, u_{w_{i+1}})$ with $j' = j_S^e(\mathtt{p}_{\bar{n}}^1, u_{w_i}) = j_S^s(\mathtt{p}_1^1, u_{w_{i+1}})$. Recall that $u_{w_{i+1}}$ comes right after u_{w_i} in the lexicographic order according to $\prec$.
- At the end of the above loop, we obtain an atom $\mathtt{Rdy}_1(u_w, t_w^{j_S^e(\mathtt{p}_{\bar{n}}^1, u_1)}, u_1)$. Apply the instantiation of (36) $\mathtt{Rdy}_1(u, t, v) \wedge \mathtt{Root}(v) \to \mathtt{Ld}_2(u, t, u, u)$ and add $\mathtt{Ld}_2(u_w, t^{j_S^e(\mathtt{p}_{\bar{n}}^1, u_1)}, u_w, u_w)$. Note that $j_S^e(\mathtt{p}_{\bar{n}}^1, u_1) = j_S^s(\mathtt{p}_1^2, u_w, u_w)$.
- Repeat the process, so that we obtain the atoms of the following form for $2 \leq x \leq \bar{m}$ and $u_{w_1}, \dots, u_{w_x} \in \mathit{branch}(u_w)$:

– $\mathtt{Ld}_x(u_w, t_w^j, u_{w_1}, \ldots, u_{w_x})$ where $j = j_S^s(\mathtt{p}_1^x, u_{w_1}, \ldots, u_{w_x})$;
– $\mathtt{Rdy}_x(u_w, t_w^j, u_{w_1}, \ldots, u_{w_x})$ where $j = j_S^e(\mathtt{p}_{\bar{n}}^x, u_{w_1}, \ldots, u_{w_x})$;
– for $1 \leq g \leq \bar{n}$:
  * $\mathtt{Ld}_{\mathtt{p}_g^x}(u_w, t_w^j, u_{w_1}, \ldots, u_{w_x})$ where $j = j_S^s(\mathtt{p}_g^x, u_{w_1}, \ldots, u_{w_x})$;
  * $\mathtt{Rdy}_{\mathtt{p}_g^x}(u_w, t_w^j, u_{w_1}, \ldots, u_{w_x})$ where $j = j_S^e(\mathtt{p}_g^x, u_{w_1}, \ldots, u_{w_x})$;
  * for $\mathtt{In}'_{\mathtt{p}_g^x}(u_{w_1}, \ldots, u_{w_x}) \in \mathcal{I}$ and $j = j_S^s(\mathtt{p}_g^x, u_{w_1}, \ldots, u_{w_x})$:
    · $\mathtt{S}_{\mathtt{p}_g^x}(t_w^j)$,
    · $\mathtt{Nxt}(t_w^j, t_w^{j+1}), \ldots, \mathtt{Nxt}(t_w^{j+\Sigma_{l=1}^x \ell_l + x+1}, t_w^{j+\Sigma_{l=1}^x \ell_l + x+2})$,
    · $\mathtt{LdE}(u_{w_1}, t_w^{j+1}, t_w^{j+\ell_1+2}), \ldots, \mathtt{LdE}(u_{w_x}, t_w^{j+\Sigma_{l=1}^{x-1} \ell_l + x}, t_w^{j+\Sigma_{l=1}^x \ell_l + x+1})$.

• We have obtained an atom $\mathtt{Rdy}_{\bar{m}}(u_w, t_w^j, u_1, \ldots, u_1)$ with $j = j_S^e(\bar{m}, u_1, \ldots, u_1) = |S| + 1$. Apply (37) and add $\mathtt{S}_{\text{\_}}(t_w^{|S|+1})$ and $\mathtt{End}(t_w^{|S|+1})$.
• For all $u_{w_i} \in \mathit{branch}(u_w)$ and atoms of the form $\mathtt{LdE}(u_{w_i}, t_w^{j+1}, t_w^{j+\ell_i+2})$:
  – Apply $\langle \rho, \sigma \rangle$ with $\rho =$ (30) and $\sigma(v) = u_{w_i}$, $\sigma(x_s) = t_w^{j+1}$, $\sigma(x_e) = t_w^{j+\ell_i+2}$, $\sigma(y_1) = e_{w_i}^1$ and $\sigma(y_{\text{\_}}) = e_{w_i}^{\ell_i}$ where $b_1^i \cdots b_{\ell_i}^i$ is the binary representation of $|w_i| + 1$. This introduces two fresh nulls $u_{\rho,\sigma,z_1} = t_w^{j+2}$ and $u_{\rho,\sigma,z_{\text{\_}}} = t_w^{j+\ell_i+1}$ and adds the atoms: $\mathtt{S}_{\|}(t_w^{j+1})$, $\mathtt{Nxt}(t_w^{j+1}, t_w^{j+2})$, $\mathtt{Cpy}(e_{w_i}^1, e_{w_i}^{\ell_i}, t_w^{j+2}, t_w^{j+\ell_i+1})$, $\mathtt{Nxt}(t_w^{j+\ell_i+1}, t_w^{j+\ell_i+2})$ and $\mathtt{S}_{\|}(t_w^{j+\ell_i+2})$.
  – Apply exhaustively rules (27) and (28). This process introduces nulls $t^{j+3}, \ldots, t^{j+\ell_i}$ and adds the following atoms (cf. proof of Lemma 28):
    * $\mathtt{S}_{b_1^i}(t_w^{j+2}), \ldots, \mathtt{S}_{b_{\ell_i}^i}(t_w^{j+\ell_i+1})$,
    * $\mathtt{Nxt}(t_w^{j+2}, t_w^{j+3}), \ldots, \mathtt{Nxt}(t_w^{j+\ell_i}, t_w^{j+\ell_i+1})$
    * $\mathtt{Cpy}(e_{w_i}^2, e_{w_i}^{\ell_i}, t_w^{j+3}, t_w^{j+\ell_i+1}), \ldots, \mathtt{Cpy}(e_{w_i}^{\ell_i-1}, e_{w_i}^{\ell_i}, t_w^{j+\ell_i}, t_w^{j+\ell_i+1})$.

At this point, for every null u_w such that $\mathtt{Leaf}(u_w) \in \mathcal{I}$, we have added $\mathit{startConf}(\mathcal{I}, u_w)$ and $\mathit{loadS}(\mathcal{I}, u_w)$ and no rule is applicable. □

Lemma 8. *$\mathfrak{I}_4 = \{\mathit{Int}_4(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_3\}$ is a universal model set of $\mathcal{R}_4$ and $\mathcal{D}$.*

Proof. Similar to the proof of Lemma 7, using Lemmas 7 and 29. □

Before proving Lemma 9, we introduce some notation about TM runs. A *configuration* for a TM $M = \langle Q, \Gamma, \delta \rangle$ is a finite sequence $\vec{C} = C_1, \ldots, C_k$ where $C_i = \{q, a\}$ for some $i \leq k$, $q \in Q$, and $a \in \Gamma$, while for each $j \neq i$, $C_j = \{a_j\}$ for some $a_j \in \Gamma$. The *successor configuration* of $\vec{C}$ under M, denoted $M(\vec{C})$, is the configuration $\vec{C'} = C'_1, \ldots, C'_{k+1}$ satisfying, assuming $\delta(q, a) = (q', a', l)$,

$$C'_j \cap \Gamma = \begin{cases} \{a'\} & \text{if } j = i, \\ \{\text{␣}\} & \text{if } j = k+1, \\ C_j \cap \Gamma & \text{otherwise,} \end{cases}$$

while $C'_j \cap Q = \{q'\}$ if $j = \max(1, i + l)$, otherwise $\emptyset$. For a word $\vec{\gamma} = \gamma_1, \ldots, \gamma_k$, let $\mathit{StConf}(M, \vec{\gamma})$ denote the sequence $\{q_S, \gamma_1\}$, $\{\gamma_2\}$, $\ldots$, $\{\gamma_k\}$, $\{\text{␣}\}$. Let $\mathit{Comp}(M, \vec{\gamma}) = \vec{C}_1, \vec{C}_2, \ldots$ be the (possibly finite) configuration sequence satisfying $\vec{C}_1 = \mathit{StConf}(M, \vec{\gamma})$ as well as $\vec{C}_{i+1} = M(\vec{C}_i)$, whenever defined (otherwise – and only then – the sequence ends at i). Note that M halts on $\vec{\gamma}$ if $\mathit{Comp}(M, \vec{\gamma})$ is finite, and accepts $\vec{\gamma}$ if the last configuration in $\mathit{Comp}(M, \vec{\gamma})$ features q_A.

Lemma 30. *Given some $\mathcal{I} \in \mathfrak{I}_4$, $\mathit{Int}_5(\mathcal{I})$ is isomorphic to the unique interpretation in a result of the chase over $\mathcal{R}_5$ and $\mathcal{I}$.*

Proof. The interpretation $\mathit{Int}_5(\mathcal{I})$ is equal to the union of the following interpretations:

1. $\mathcal{I}$;
2. For each $w \in \mathbf{Z}$ with $b_1 \cdots b_\ell$ the binary representation of $|w|+1$: $\{\mathtt{Nxt}^+(e^i_w, e^j_w) \mid 1 \leq i < j \leq \ell\}$.
3. For each $u_w \in \mathsf{Nulls}(\mathcal{I})$ with $S = \mathit{branchTape}(\mathcal{I}, u_w)$, $\mathit{Comp}(M, S) = \vec{C}_1, \ldots, \vec{C}_\ell$, and $\vec{C}_i = C_{i,1}, \ldots, C_{i,|S|+i}$ for each $1 \leq i \leq \ell$, the database $\mathit{Run}(u_w)$ defined as follows:

$$\begin{aligned}
&\{\mathtt{Nxt}^+(t^i_w, t^j_w) \mid 1 \leq i < j \leq |S|+1\} \cup \\
&\{\mathtt{End}(t^{i,|S|+i}_w) \mid 2 \leq i \leq \ell\} \cup \\
&\{\mathtt{Nxt}(t^{i,j-1}_w, t^{i,j}_w) \mid 2 \leq i \leq \ell, 2 \leq j \leq |S|+i\} \cup \\
&\{\mathtt{Nxt}^+(t^{i,j}_w, t^{i,j'}_w) \mid 1 \leq i \leq \ell, 1 \leq j < j' \leq |S|+i\} \cup \\
&\{\mathtt{Stp}(t^{i-1,j}_w, t^{i,j}_w) \mid 3 \leq i \leq \ell, 1 \leq j \leq |S|+i-1\} \cup \\
&\{\mathtt{S}_a(t^{i,j}_w) \mid 2 \leq i \leq \ell, 1 \leq j \leq |S|+i, a \in C_{i,j} \cap \Gamma\} \cup \\
&\{\mathtt{Hd}_q(t^{i,j}_w) \mid 2 \leq i \leq \ell, 1 \leq j \leq |S|+i, q \in C_{i,j} \cap Q\} \cup \\
&\{\mathtt{Goal} \mid 1 \leq j \leq |S|+\ell, q_A \in C_{\ell,j}\};
\end{aligned}$$

We describe a (Datalog-first) chase over $\mathcal{R}_5$ and $\mathcal{I}$ step by step. We directly set the the names of the fresh nulls so that the chase result coincides with $\mathit{Int}_5(\mathcal{I})$. Note that no rule in Figures 1, 2, 3, 4, or 5 is applicable to $\mathcal{I}$ and that applying a rule in Figure 6 cannot make a rule in Figures 1, 2, 3, 4, or 5 applicable.

We first apply exhaustively (40) and (41) which compute a transitive closure on the atoms of the form $\mathtt{Nxt}(e^i_w, e^j_w)$ and $\mathtt{Nxt}(t^i_w, t^j_w)$ in $\mathcal{I}$. This adds the atoms $\{\mathtt{Nxt}^+(e^i_w, e^j_w) \mid 1 \leq i < j \leq \ell\}$ described in point (2) and $\{\mathtt{Nxt}^+(t^i_w, t^j_w) \mid 1 \leq i < j \leq |S|+1\}$ of the first line in point (3).

For every null u_w such that $\mathtt{Leaf}(u_w) \in \mathcal{I}$, $S = \mathit{branchTape}(\mathcal{I}, u_w)$ and $\mathit{Comp}(M, S) = \vec{C}_1, \ldots, \vec{C}_\ell$, we show by induction that for every $2 \leq i \leq \ell$ the following atoms belong to the chase result:

- $\mathtt{End}(t^{i,|S|+i}_w)$,
- $\mathtt{Nxt}(t^{i,j-1}_w, t^{i,j}_w)$ for $2 \leq j \leq |S|+i$,
- $\mathtt{Nxt}^+(t^{i,j}_w, t^{i,j'}_w)$ for $1 \leq j < j' \leq |S|+i$,
- $\mathtt{Stp}(t^{i-1,j}_w, t^{i,j}_w)$ for $1 \leq j \leq |S|+i-1$,
- $\mathtt{S}_a(t^{i,j}_w)$ for $1 \leq j \leq |S|+i, a \in C_{i,j} \cap \Gamma$,
- $\mathtt{Hd}_q(t^{i,j}_w)$ for $1 \leq j \leq |S|+i, q \in C_{i,j} \cap Q$.

Base case $i = 2$. Recall that for every $\mathtt{Leaf}(u_w) \in \mathcal{I}$ with serialisation $S = \mathit{branchTape}(\mathcal{I}, u_w)$, $\mathcal{I}$ includes:

$$\{\mathtt{Ld}_1(u_w, t^1_w, u_w), \mathtt{Hd}_{q_S}(t^1_w), \mathtt{End}(t^{|S|+1}_w), \mathtt{S}_{\llcorner}(t^{|S|+1}_w)\} \cup \{\mathtt{Nxt}(t^{j-1}_w, t^j_w) \mid 2 \leq j \leq |S|+1\} \cup$$
$$\{\mathtt{S}_a(t^j_w) \mid 1 \leq j \leq |S|, a = S[j]\}.$$

Moreover $\vec{C}_1 = \mathit{StConf}(M, S) = \{q_S, S[1]\}, \{S[2]\}, \ldots, \{S[|S|]\}, \{\llcorner\}$. Let $(q_S, S[1]) \mapsto (r, b, +1) \in \delta$ (recall that we assume that the TM will never attempt to move left on the first position of the tape). By definition, we have $C_{2,1} \cap \Gamma = \{b\}$, $C_{2,j} \cap \Gamma = C_{1,j} \cap \Gamma = \{S[j]\}$ for every $1 < j \leq |S|+1$, and $C_{2,2} \cap Q = \{r\}$.

- For every $1 < j \leq |S|+1$, let $a \in C_{1,j} \cap \Gamma = C_{2,j} \cap \Gamma$.
  - Apply $\langle \rho, \sigma \rangle$ with ρ the instantiation of (45) for transition $(q_S, S[1]) \mapsto (r, b, +1)$ and tape symbol $S[j]$, and $\sigma(x) = t^1_w$, $\sigma(y) = t^j_w$. This introduces a fresh null $u_{\rho,\sigma,z} = t^{2,j}_w$ and adds $\mathtt{Stp}(t^j_w, t^{2,j}_w)$ and $\mathtt{S}_{S[j]}(t^{2,j}_w)$.

- Apply $\langle \rho, \sigma \rangle$ with ρ the instantiation of (44) for $(q_S, S[1]) \mapsto (r, b, +X)$ and $\sigma(x) = t_w^1$. This introduces a fresh null $u_{\rho,\sigma,z} = t_w^{2,1}$ and adds $\texttt{Stp}(t_w^1, t_w^{2,1})$ and $\texttt{S}_b(t_w^{2,1})$.
- Apply $\langle \rho, \sigma \rangle$ with $\rho =$ (43) with $\sigma(x) = t_w^{|S|+1}$ and $\sigma(z) = t_w^{2,|S|+1}$. This introduces a fresh null $u_{\rho,\sigma,v} = t_w^{2,|S|+2}$ and adds $\texttt{Nxt}(t_w^{2,|S|+1}, t_w^{2,|S|+2})$, $\texttt{S}_{\_}(t_w^{2,|S|+2})$ and $\texttt{End}(t_w^{2,|S|+2})$.
- Apply exhaustively (42), adding all atoms $\texttt{Nxt}(t_w^{2,j-1}, t_w^{2,j})$ for $2 \leq j \leq |S| + 2$.
- Apply exhaustively (40) and (41), adding all atoms $\texttt{Nxt}^+(t_w^{2,j}, t_w^{2,j'})$ for $1 \leq j < j' \leq |S| + 2$.
- Apply (47) and add $\texttt{Hd}_r(t_w^{2,2})$.

Induction step: Assume the property is true for some $2 \leq i < \ell$ and let j_0 be the unique C_{i,j_0} such that $C_{i,j_0} \cap Q \neq \emptyset$, $\{q_0\} = C_{i,j_0} \cap Q$, $\{a_0\} = C_{i,j_0} \cap \Gamma$, and $(q_0, a_0) \mapsto (r, b, +X) \in \delta$. By definition, we have $C_{i+1,j_0} \cap \Gamma = \{b\}$, $C_{i+1,j} \cap \Gamma = C_{i,j} \cap \Gamma$ for every $j \neq j_0$, and $C_{i+1,j_0+X} \cap Q = \{r\}$.

- For every $j > j_0$ (resp. $j < j_0$), let $a \in C_{i,j} \cap \Gamma = C_{i+1,j} \cap \Gamma$.
  - Apply $\langle \rho, \sigma \rangle$ with ρ the instantiation of (45) (resp. (46)) for transition $(q_0, a_0) \mapsto (r, b, +X)$ and tape symbol a, and $\sigma(x) = t_w^{i,j_0}$, $\sigma(y) = t_w^{i,j}$. This introduces a fresh null $u_{\rho,\sigma,z} = t_w^{i+1,j}$ and adds $\texttt{Stp}(t_w^{i,j}, t_w^{i+1,j})$ and $\texttt{S}_a(t_w^{i+1,j})$.
- Apply $\langle \rho, \sigma \rangle$ with ρ the instantiation of (44) for $(q_0, a_0) \mapsto (r, b, +X)$ and $\sigma(x) = t_w^{i,j_0}$. This introduces a fresh null $u_{\rho,\sigma,z} = t_w^{i+1,j_0}$ and adds $\texttt{Stp}(t_w^{i,j_0}, t_w^{i+1,j_0})$ and $\texttt{S}_b(t_w^{i+1,j_0})$.
- Apply $\langle \rho, \sigma \rangle$ with $\rho =$ (43) with $\sigma(x) = t_w^{i,|S|+i}$ and $\sigma(z) = t_w^{i+1,|S|+i}$. This introduces a fresh null $u_{\rho,\sigma,v} = t_w^{i+1,|S|+i+1}$ and adds $\texttt{Nxt}(t_w^{i+1,|S|+i}, t_w^{i+1,|S|+i+1})$, $\texttt{S}_{\_}(t_w^{i+1,|S|+i+1})$ and $\texttt{End}(t_w^{i+1,|S|+i+1})$.
- Apply exhaustively (42), adding all atoms $\texttt{Nxt}(t_w^{i+1,j-1}, t_w^{i+1,j})$ for $2 \leq j \leq |S| + i + 1$.
- Apply exhaustively (40) and (41), adding all atoms $\texttt{Nxt}^+(t_w^{i+1,j}, t_w^{i+1,j'})$ for $1 \leq j < j' \leq |S| + i + 1$.
- If $X = +1$, apply (47) and add $\texttt{Hd}_r(t_w^{i+1,j_0+X})$.
- If $X = -1$, then $j_0 > 1$ (since the TM never attempts to move left on the first position of the tape): apply (48) and add $\texttt{Hd}_r(t_w^{i+1,j_0+X})$.

We have shown that all atoms of $\mathit{Run}(u_w)$ but $\texttt{Goal}$ belong to the chase result. Finally, for every null u_w such that $\texttt{Leaf}(u_w) \in \mathcal{I}$, $S = \mathit{branchTape}(\mathcal{I}, u_w)$ and $\mathit{Comp}(M, S) = \vec{C}_1, \ldots, \vec{C}_\ell$, if there exists j such that $\texttt{Hd}_{q_A}(t_w^{\ell,j}) \in \mathit{Run}(u_w)$, apply (39) and add $\texttt{Goal}$. □

Lemma 9. *$\mathfrak{I}_5 = \{\mathit{Int}_5(\mathcal{I}) \mid \mathcal{I} \in \mathfrak{I}_4\}$ is a universal model set of $\mathcal{R}_5$ and $\mathcal{D}$.*

Proof. Similar to the proof of Lemma 8, using Lemmas 8 and 30. □

Lemma 10. *If $\mathcal{D} \in \mathfrak{Q}$, then* $\texttt{Goal} \in \mathcal{I}$ *for each $\mathcal{I} \in \mathfrak{M}$.*

Proof. Let $\mathcal{I} \in \mathfrak{M}$. We have shown that there is a homomorphism $\mathcal{D} \to \mathit{branchDb}(\mathcal{I}, u_w)$ for the node u_w where $|w| = |\mathit{Order}(\mathcal{I})|$. Since $\mathfrak{Q}$ is closed under homomorphisms, $\mathcal{D} \in \mathfrak{Q}$ implies $\mathit{branchDb}(\mathcal{I}, u_w) \in \mathfrak{Q}$.

Let $\mathit{Comp}(M, S) = \vec{C}_1, \ldots, \vec{C}_\ell$ be the computation of M on the serialisation $S = \mathit{branchTape}(\mathcal{I}, u_w)$ of the database $\mathit{branchDb}(\mathcal{I}, u_w)$. Since $\mathit{branchDb}(\mathcal{I}, u_w) \in \mathfrak{Q}$ and M decides $\mathfrak{Q}$, $\vec{C}_1, \ldots, \vec{C}_\ell$ is such that $\vec{C}_\ell$ features q_A. By construction of $\mathit{Run}(u_w)$ (see proof of Lemma 30), it follows that $\texttt{Goal} \in \mathit{Run}(u_w)$. Hence $\texttt{Goal} \in \mathcal{I}$. □

Lemma 11. *If $\mathcal{D} \notin \mathfrak{Q}$, then* $\texttt{Goal} \notin \mathcal{I}$ *for some $\mathcal{I} \in \mathfrak{M}$.*

Proof. Consider some $\mathcal{I} \in \mathfrak{M}$ such that $Db(\mathcal{I}) = \mathcal{D}$ and $\mathtt{NEq}(t,u) \in \mathcal{I}$ for each $t,u \in \mathsf{Nulls}(\mathcal{D})$ with $t \neq u$ (such a $\mathcal{I}$ exists by Lemma 26). Let u_w denote the leaf node with $|w| = |Order(\mathcal{I})|$. Then $branchDb(\mathcal{I},u_w)$ is isomorphic to $Db(\mathcal{I}) = \mathcal{D}$ and $branchDb(\mathcal{I},u_w) \notin \mathfrak{Q}$.

Let $Comp(M,S) = \vec{C}_1,\dots,\vec{C}_\ell$ be the computation of M on the serialisation $S = branchTape(\mathcal{I},u_w)$ of the database $branchDb(\mathcal{I},u_w)$. Since $branchDb(\mathcal{I},u_w) \notin \mathfrak{Q}$ and M decides $\mathfrak{Q}$, $\vec{C}_1,\dots,\vec{C}_\ell$ is such that $\vec{C}_\ell$ does not feature q_A. By construction of $Run(u_w)$ (see proof of Lemma 30), it follows that $\mathtt{Goal} \notin Run(u_w)$.

Moreover, for all other leaf nodes u_v with $\mathtt{Leaf}(u_v) \in \mathcal{I}$, there is a homomorphism $branchDb(\mathcal{I},u_v) \to branchDb(\mathcal{I},u_w)$. Since $\mathfrak{Q}$ is closed under homomorphisms, M does not accept any such $branchDb(\mathcal{I},u_v)$, and $\mathtt{Goal} \notin Run(u_v)$. By construction of $\mathfrak{M}$, $\mathtt{Goal}$ occurs in $\mathcal{I}$ iff it occur in some $Run(u_v)$, so $\mathtt{Goal} \notin \mathcal{I}$. □

B Proofs for Section 5

We fix for this whole appendix part a split (Σ_1,Σ_2) of Σ fulfilling the condition of Item 2 of Lemma 18, and $\Sigma'_1,\Sigma'_2,\Sigma'_3$ built from Σ_1,Σ_2 as described in the body of the paper.

Proof of Proposition 19

Definition 6 (World Structure)**.** Let $\mathcal{I}$ be an interpretation of Σ'_1. The world structure of $\mathcal{I}$ is the graph (V,E) where:

- $V = \{w \mid \mathtt{Done}(w) \in \mathcal{I}\}$
- $E = \{(w,w') \mid w \neq w' \wedge \exists \mathtt{p} \exists \vec{x}\, \mathtt{Ins}_\mathtt{p}(\vec{x},w,w') \in \mathcal{I}\}$

Lemma 31. *For every $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, the world structure of a model $\mathcal{I}$ of $\mathcal{D}$ and Σ'_1 generated by a chase sequence is a finite tree, such that:*

- *its root is the unique element w such that $\mathtt{Empty}(w) \in \mathcal{I}$;*
- *if $(w,w') \in E$, then $\mathsf{world}(w') = \mathsf{world}(w) \cup \{\mathtt{p}(\vec{x})\}$ for some atom $\mathtt{p}(\vec{x})$, and $\mathtt{p}(\vec{x}) \notin \mathsf{world}(w)$;*
- *if $\mathtt{Init}(w) \in \mathcal{I}$ and $\mathtt{p}(\vec{a}) \in \mathcal{D} \setminus \mathsf{world}(w)$, there exists w' such that $\mathtt{Ins}_\mathtt{p}(\vec{a},w,w') \in \mathcal{I}$.*

Proof. Let us first notice that Rule (58) is applicable exactly once, as its frontier is empty. The null it creates, say $w_\emptyset$, is then the only one such that $\mathtt{Empty}(w_\emptyset) \in \mathcal{I}$. Moreover, there is no edge incoming in $w_\emptyset$ in the world structure, as the only rules Rules (59), (60) and (61) that introduce an atom of the shape $\mathtt{Ins}_\mathtt{p}(\vec{x},w,w')$ with $w \neq w'$ are such that w' is existentially quantified. For the same reason, note that there is at most one edge incoming in any vertex in the world structure.

Next notice that no new null is created unless Rules (59), (60) or (61) are applied, and they all must be applied by mapping w to an element of the world structure, as $\mathtt{Done}(w)$ is in the body of each of these rules. Hence the world structure is connected, and is thus a tree of root $w_\emptyset$.

Now let us notice that in any interpretation $\mathcal{I}$ generated by a sequence of rule applications, if $\mathtt{Done}(w) \in \mathcal{I}$, then $\mathsf{world}_\mathcal{I}(w) = \mathsf{world}_{\mathcal{I}'}(w)$ for any $\mathcal{I}'$ obtained by extending the sequence of rule applications that generated $\mathcal{I}$. Indeed, the only way to derive $\mathtt{Done}(w)$ is to apply Rule (63), which requires the atoms $\mathtt{Empty}(w_\emptyset)$ and $\mathtt{Subs}(w_\emptyset,w)$, as $w_\emptyset$ is the only element for which $\mathtt{Empty}$ holds. The only way to create an atom of the shape $\mathtt{Subs}(w_0,w_2)$ with $w_0 \neq w_2$ is by applying an instantiation of Rule (62), which can be applied if w_0 is a parent of w_1 in the world structure, and w_1 is such that $\mathtt{Subs}(w_1,w_2)$ holds. Hence $\mathtt{Done}(w)$ is entailed only when Rule (62) as been applied by mapping w_0 to every ancestor of w and w_2 to w, which has a effect to ensure that $\mathsf{world}_\mathcal{I}(w)$ contains all the atoms possibly present in $\mathsf{world}_{\mathcal{I}'}(w)$.

If $(w,w') \in E$, then w' has been created by the application of Rule (59), (60) or (61). In all cases, $\mathtt{Done}(w)$ must hold at the time of the rule application. If it is by Rule (59) and substitution σ, then $\mathtt{p}(\sigma(\vec{x}))$ cannot belong to $\mathsf{world}(w)$, as this would make Rule (59) not applicable with σ. By the sequence of rule applications described before, $\mathsf{world}(w') = \mathsf{world}(w) \cup \{\mathtt{p}(\sigma(\vec{x}))\}$. Rules (60) and (61) are treated in a similar way.

Both the depth and the arity of the world structure is thus upper bounded by the number of atoms using a predicate appearing in $\Sigma_1 \cup \Sigma_2$ and nulls from $\mathcal{D}$.

For the last item, let us notice that if $\mathtt{Init}(w) \in \mathcal{I}$, and $\mathtt{p}(\vec{a}) \in \mathcal{D} \setminus \mathsf{world}(w)$, then Rule (59) instantiated for $\mathtt{p}$ is applicable by mapping $\vec{x}$ to $\vec{a}$. Applying that rule will create a fresh null w' which will fulfill the conditions stated in the lemma. □

Lemma 32. *Let $\mathcal{D}$ be a database over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, and let $\mathcal{I}$ be a model of $\mathcal{D}$ and Σ_1' generated by a chase sequence. Let $v \in \mathsf{Nulls}(\mathcal{I})$ such that $\mathtt{Done}(v) \in \mathcal{I}$. If a disjunctive rule ρ is applicable to $\mathsf{world}(v)$ by σ, then there exist in $\mathcal{I}$ two elements w_1 and w_2 such that $\mathtt{Ins}_{\mathtt{p}_1}(\vec{u}_1, v, w_1)$ and $\mathtt{Ins}_{\mathtt{p}_2}(\vec{u}_2, v, w_2)$ such that $\mathtt{p}_1(\vec{u}_1)$ and $\mathtt{p}_2(\vec{u}_2)$ are the two atoms created by the application of $\langle \rho, \sigma \rangle$.*

Proof. Let us first notice that σ, extended by mapping w to v, is a substitution that maps the bodies of Rules (60) and (61) to $\mathcal{I}$, by definition of $\mathsf{world}(v)$ and since $\mathtt{Done}(v) \in \mathcal{I}$. As $\mathcal{I}$ is a model of Σ_1', the head of these two rules should be also mappable in $\mathcal{I}$, hence there exists w_1 and w_2 fulfilling the conditions of the lemma. □

Note that Proposition 19 is a direct consequence of Lemmas 31 and 32.

Proof of Lemma 18

Definition 7 (Saturated World of a Null in an Interpretation)**.** Let w be a null in an interpretation $\mathcal{I}$ such that $\mathtt{Subs}(w, w) \in \mathcal{I}$. The saturated world of w is defined by $\mathsf{saturatedWorld}_{\mathcal{I}}(w) = \{\mathtt{p}(\vec{x}) \mid \mathtt{p}'(\vec{x}, w) \in \mathcal{I}\}$.

Lemma 33. *If there is a chase tree w.r.t. $(\Sigma_1' \cup \Sigma_2')$ and $\mathcal{D}$ whose unique leaf $\mathcal{I}$ is such that there exists $w^* \in \mathsf{Nulls}(\mathcal{I})$ s.t. $\mathtt{Goal} \in \mathsf{saturatedWorld}_{\mathcal{I}}(w^*)$ and $\mathtt{Done}(w^*) \in \mathcal{I}$, then there exists a chase tree w.r.t Σ_2 and $\mathsf{world}(w^*)$ whose unique leaf contains $\mathtt{Goal}$.*

Proof. Let us consider a chase tree T' w.r.t. $(\Sigma_1' \cup \Sigma_2')$ and $\mathcal{D}$, and let w^* be such that $\mathtt{Done}(w^*)$ is in the result $\mathcal{I}$ of T'. We show that there exists a chase tree T w.r.t Σ_2 and $\mathsf{world}(w^*)$ such that $\mathsf{saturatedWorld}_{\mathcal{I}}(w^*)$ is mapped to the label of the (unique) leaf of T by a homomorphism ψ.

For every label $\mathcal{J}$ of a node in T' such that $\mathtt{Done}(w^*) \in \mathcal{J}$, we show that there exists a prefix of a chase tree w.r.t. Σ_2 and $\mathsf{world}(w^*)$ such that $\mathsf{saturatedWorld}_{\mathcal{J}}(w^*)$ is mapped to the label of the leaf of that prefix by a homomorphism ψ. We do this by induction on the number of rule applications $\langle \rho, h \rangle$ with ρ of the form Rule (64) and $h(w) = w^*$ that are done in T' before the node labelled by $\mathcal{J}$.

- If no rule application of the shape $\langle \rho, h \rangle$ with $h(w) = w^*$ is performed in T' before $\mathcal{J}$, then $\mathsf{saturatedWorld}_{\mathcal{J}}(w^*) = \mathsf{world}(w^*)$. We thus define ψ as the identity, and the chase tree prefix consisting of the root labelled by $\mathsf{world}(w^*)$ fulfills the property.
- Assume that the property is true for every $\mathcal{J}$ obtained in T' after at most $i-1$ applications of the shape $\langle \rho', h \rangle$ with ρ' of the form Rule (64) and $h(w) = w^*$. Let $\mathcal{J}_i$ be obtained in T' after i such applications and let $\langle \rho', h \rangle$ be the last one, with ρ' being the instantiation of Rule (64) for $\rho \in \Sigma_2$. Let $\mathcal{J}_{i-1}$ be the label of the node in T' on which $\langle \rho', h \rangle$ is applied. By induction assumption, there exist ψ and a prefix of a chase tree for Σ_2 and $\mathcal{D}$, resulting in S_{i-1}, such that $\psi(\mathsf{saturatedWorld}_{\mathcal{J}_{i-1}}(w^*)) \subseteq S_{i-1}$. Hence $\langle \rho, \psi \circ h_{|\mathsf{Terms}(B_\rho)} \rangle$, where B_ρ is the body of ρ, is applicable on S_{i-1}, creating a new leaf, labeled by S_i, and we extend ψ by mapping every null created by the instantiation of $z_i \in \vec{z}$ in the application of $\langle \rho', h \rangle$ to the null created by the instantiation of $z_i \in \vec{z}$ in the application of $\langle \rho, \psi \circ h_{|\mathsf{Terms}(B_\rho)} \rangle$. We obtain $\psi(\mathsf{saturatedWorld}_{\mathcal{J}_i}(w^*)) \subseteq S_i$. □

Lemma 34. *Any restricted chase w.r.t $\Sigma_1' \cup \Sigma_2' \cup \Sigma_3'$ can be transformed into an equivalent restricted chase in which rules from Σ_1' are applied before rules from Σ_2', which are applied before rules from Σ_3'.*

Proof. Notice that head predicates of Σ'_3 do not appear in do not appear in Σ'_j with $j < 3$. Hence rules of Σ'_3 can be applied last. Note that head predicates of Σ'_2 do not appear as body predicates of Σ'_1. Moroever, a rule of Σ'_2 is applicable by mapping w to w^* only if $\texttt{Done}(w^*)$ as been derived. Further applications of rules of Σ'_1 cannot add an atom of the shape $\mathtt{p}'(\vec{x}, w^*)$, and thus applying them before do not prevent the application of a rule of Σ'_2. □

Proposition 35. *For every database $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, if $\Sigma_1 \cup \Sigma_2, \mathcal{D} \models \texttt{Goal}$, then $\Sigma'_1 \cup \Sigma'_2 \cup \Sigma'_3, \mathcal{D} \models \texttt{Goal}$.*

Proof. Let us consider a finite a chase tree T proving that $\Sigma_1 \cup \Sigma_2, \mathcal{D} \models \texttt{Goal}$ and such that rules of Σ_1 are applied before rules of Σ_2 (this is possible by definition of Σ_1 and Σ_2 that form a split of Σ). We build a chase tree proving that $\Sigma'_1 \cup \Sigma'_2 \cup \Sigma'_3, \mathcal{D} \models \texttt{Goal}$ (we actually describe a sequence of rule applications, as $\Sigma'_1 \cup \Sigma'_2 \cup \Sigma'_3$ contains only deterministic rules).

- Apply Rule (58), creating a fresh null $w_\emptyset$ and facts $\texttt{Init}(w_\emptyset)$, $\texttt{Done}(w_\emptyset)$ and $\texttt{Empty}(w_\emptyset)$.
- Until creation of $w_\mathcal{D}$, perform the following:
  - let $w_{D'}$ be the last introduced null;
  - let $\mathtt{p}(\vec{a}) \in \mathcal{D} \setminus D'$:
    - apply Rule (59), instantiated for $\mathtt{p}$, by mapping $\vec{x}$ to $\vec{a}$ and w to $w_{D'}$, creating a fresh null $w_{D' \cup \{\mathtt{p}(\vec{a})\}}$;
    - apply Rule (62) as many times as necessary until $\texttt{Subs}(w_\emptyset, w_{D' \cup \{\mathtt{p}(\vec{a})\}})$ is created, which makes Rule (63) become applicable by mapping w' to $w_{D' \cup \{\mathtt{p}(\vec{a})\}}$. Then apply Rule (63) and add $\texttt{Done}(w_{D' \cup \{\mathtt{p}(\vec{a})\}})$.
- At the end of the above loop, note that we have created null $w_\mathcal{D}$ and added in particular $\texttt{Done}(w_\mathcal{D})$ as well as $\texttt{Ins}_\mathtt{p}(\vec{a}, w_\mathcal{D}, w_\mathcal{D})$ and for every $\mathtt{p}(\vec{a}) \in \mathcal{D}$.
- Let r be the root node of T. Note that r is labelled by $\mathcal{D}$. In what follows, we will introduce some nulls $w_\mathcal{N}$ such that $\mathcal{N}$ is the label of some node n in T and it will remain true that $\texttt{Done}(w_\mathcal{N})$ as well as $\texttt{Ins}_\mathtt{p}(\vec{a}, w_\mathcal{N}, w_\mathcal{N})$ for every $\mathtt{p}(\vec{a}) \in \mathcal{N}$ are introduced between the creation of $w_\mathcal{N}$ and the creation of the next $w_{\mathcal{N}'}$.
- Define $\psi(r) = w_\mathcal{D}$ and let $N = \{r\}$ and $L = \emptyset$. Until $N = \emptyset$, perform the following.
  - Consider the case where n is a node of T labelled by $\mathcal{N}$ such that $\psi(n) = w_\mathcal{N}$ and n has 2 children c_1 and c_2 labelled by $\mathcal{N}_1$ and $\mathcal{N}_2$ respectively, which correspond to the application of $\langle \rho, \sigma \rangle$ with $\rho \in \Sigma_1$ of the form $\bigwedge_{\mathtt{p}(\vec{x}) \in \beta} \mathtt{p}(\vec{x}) \to \mathtt{p}_1(\vec{x_1}) \vee \mathtt{p}_2(\vec{x_2})$. We have $\mathcal{N}_1 = \mathcal{N} \cup \{\mathtt{p}_1(\vec{a}_1)\}$ and $\mathcal{N}_2 = \mathcal{N} \cup \{\mathtt{p}_2(\vec{a}_2)\}$ where $\vec{a}_1 = \sigma(\vec{x}_1)$ and $\vec{a}_2 = \sigma(\vec{x}_2)$. The instantiations of (60) and (61) corresponding to ρ are applicable through σ', which maps w to $w_\mathcal{N}$ and $\vec{x}$ to $\sigma(\vec{x})$.
    - Apply Rule (60), introducing a fresh null that we call $w_{\mathcal{N}_1}$ and adding in particular $\texttt{Ins}_{\mathtt{p}_1}(\vec{a}_1, w_{\mathcal{N}_1}, w_{\mathcal{N}_1})$. Then apply Rule (62) as many times as necessary, then Rule (63) so that $\texttt{Done}(w_{\mathcal{N}_1})$ is created. Define $\psi(c_1) = w_{\mathcal{N}_1}$.
    - If Rule (61) is not applicable, it implies that $\mathtt{p}_1 = \mathtt{p}_2$ and $\vec{a}_1 = \vec{a}_2$, so that $\mathcal{N}_1 = \mathcal{N}_2$. Set $N = (N \cup \{c_1\}) \setminus \{n\}$, and set $\psi(c_2) = w_{\mathcal{N}_1}$.
    - Otherwise apply Rule (61), introducing a fresh null that we call $w_{\mathcal{N}_2}$ and adding in particular $\texttt{Ins}_{\mathtt{p}_2}(\vec{a}_2, w_{\mathcal{N}_2}, w_{\mathcal{N}_2})$.Then apply Rule (62) as many times as necessary, then Rule (63) so that $\texttt{Done}(w_{\mathcal{N}_2})$ is created. Define $\psi(c_2) = w_{\mathcal{N}_2}$ and set $N = (N \cup \{c_1, c_2\}) \setminus \{n\}$.
  - If n is a node of T having only one child, set $L = L \cup \{n\}$ and $N = N \setminus \{n\}$.

At this point, for every $n \in L$, $\psi(n) = w_\mathcal{N}$ and if $\mathcal{I}$ denotes the current set of facts built by our derivation, $\mathsf{saturatedWorld}_\mathcal{I}(\psi(n)) = \mathcal{N}$ by construction.

For all $n \in L$, the next rule applied on n is a rule from Σ_2, hence no more rule of Σ_1 are applied on a descendant of n in T (by assumption on T). All rule applications below n are thus deterministic, and we "copy" that derivation. Let ψ_n be the identity mapping from the label $\mathcal{N}$ of n to $\mathsf{saturatedWorld}_\mathcal{I}(\psi(n))$.

By construction, ψ_n is a homomorphism. While we extend $\mathcal{I}$ by applying new rules, we will extend ψ_n into a homomorphism from the label of any descendant n' of n in T to $\mathsf{saturatedWorld}_{\mathcal{I}}(\psi(n))$.

- For $n' = n$, this is already done.
- Assume that we have built a derivation such that ψ_n is a homomorphism from the label $\mathcal{N}'$ of some descendant n' of n in T to $\mathsf{saturatedWorld}_{\mathcal{I}}(\psi(n))$. Let n'' be the child of n' in T, labelled by $\mathcal{N}''$, and let $\langle \rho, \sigma \rangle$ be the rule application creating n'', where $\rho \in \Sigma_2$. Then $\langle \rho, \psi_n \circ \sigma \rangle$ can be applied to $\mathsf{saturatedWorld}_{\mathcal{I}}(\psi(n))$, and thus Rule (64), instantiated for ρ is applicable in $\mathcal{I}$ by extending $\psi_n \circ \sigma$ by mapping w to $\psi(n)$. We then define $\psi(n'') = \psi(n)$, and extend ψ_n by mapping each null created in the label of n'' by a variable z_i to the null created by variable z_i in the application of Rule (64).

Let T' be the tree structure having $\psi(r) = w_{\mathcal{D}}$ as root and where w_p is parent of w_c if w_c has been created by an application of Rule (60) or Rule (61), that mapped w to w_p. Note that all the w_n in this tree structure are exactly the nulls of $\mathcal{I}$ such that there exists a node n in T with $\psi(n) = w_n$, c_1, c_2 are children of n in T iff $\psi(c_1), \psi(c_2)$ are children of $\psi(n)$ in T'.

Let ℓ be a leaf of T. By assumption, $\mathtt{Goal}$ belongs to the label of ℓ. Let n be the unique node in L such that ℓ is a descendant of n. Since ψ_n is a homomorphism from the label of ℓ to $\mathsf{saturatedWorld}_{\mathcal{I}}(\psi(n))$, it holds that $\mathsf{saturatedWorld}_{\mathcal{I}}(\psi(n))$ contains $\mathtt{Goal}$. By definition of $\mathsf{saturatedWorld}_{\mathcal{I}}$, it follows that $\mathtt{Goal}'(\psi(n)) \in \mathcal{I}$. Since any leaf w_n of T' has an antecedent n by ψ which has a descendant leaf ℓ in T, then $\mathtt{Goal}'(w)$ holds for each leaf w of T'. Extend the derivation as follows:

- Apply Rule (65) to each w leaf of T'.
- If $\mathtt{Acc}(w)$ has already been derived for the children w of a node $\psi(n)$ in T', let ρ be the rule that has been applied by σ on n to create its children. Apply Rule (66) instantiated for ρ by mapping w_1 and w_2 to $\psi(c_1)$ and $\psi(c_2)$, where c_1 and c_2 are the children of n corresponding to the application of ρ. $\mathtt{Acc}(\psi(n))$ is thus derived.
- By induction on the depth, one derives $\mathtt{Acc}(w_{\mathcal{D}})$. As $\mathtt{Init}(w_{\mathcal{D}})$ holds, one can apply Rule (67) and obtain $\mathtt{Goal}$.

We did not apply every possible rule: hence, to build a chase tree, we must ensure fairness (point 3. of the definition of chase tree), by applying all possible remaining rule applications, which concludes the proof. □

To ease that proof, we will actually replace Rule (66) by the following rule:

$$\begin{aligned} &\mathtt{Ins}_{\mathtt{p}_1}(\vec{x}_1, w, w_1) \wedge \mathtt{Acc}(w_1) \wedge \\ &\mathtt{Ins}_{\mathtt{p}_2}(\vec{x}_2, w, w_2) \wedge \mathtt{Acc}(w_2) \wedge \\ &\qquad \textstyle\bigwedge_{\mathtt{p}(\vec{x}) \in \beta} \mathtt{Ins}_{\mathtt{p}}(\vec{x}, w, w) \\ &\qquad\qquad\qquad \rightarrow \mathtt{Acc}_{\rho}(w, w_1, w_2) \wedge \mathtt{Acc}(w) \end{aligned} \tag{68}$$

which has an additional ternary fresh predicate describing the rule $\rho \in \Sigma_1$ (and the worlds generated by the corresponding rule application) allowing to derive $\mathtt{Acc}(w)$. Note that since $\mathtt{Acc}_{\rho}$ not being appearing anywhere else, all the properties shown so far are still valid.

Proposition 36. *For every database $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, if $\Sigma_1' \cup \Sigma_2' \cup \Sigma_3', \mathcal{D} \models \mathtt{Goal}$ then $\Sigma_1 \cup \Sigma_2, \mathcal{D} \models \mathtt{Goal}$.*

Proof. Let $\mathcal{I}$ be a model of $\mathcal{D}$ and $\Sigma_1' \cup \Sigma_2' \cup \Sigma_3'$ obtained through a restricted chase sequence such that $\mathtt{Goal} \in \mathcal{I}$. We build a chase tree T for $\Sigma_1 \cup \Sigma_2$ and $\mathcal{D}'$ for some $\mathcal{D}' \subseteq \mathcal{D}$ such that $\mathtt{Goal}$ is in the label of every leaf of T.

Let $w_{\mathcal{D}'} \in \mathsf{Nulls}(\mathcal{I})$ be such that Rule (67) is applied on it, with $\mathsf{world}(w_{\mathcal{D}'}) = \mathcal{D}'$. Set $W = \{w_{\mathcal{D}'}\}$, and perform the following until $W = \emptyset$.

- Let $w \in W$. If $\mathtt{Goal}'(w) \in \mathcal{I}$, set $W = W \setminus \{w\}$.

- Otherwise, there must be some $\texttt{Acc}_\rho(w, w_1, w_2) \in \mathcal{I}$, as the only way to derive $\texttt{Acc}(w)$ if $\texttt{Goal}'(w)$ does not hold is to apply some instantiation for some ρ of Rule (68) by some mapping σ. Moreover, there must be such an atom for which w_1 and w_2 are different from w. Consider $\langle \rho, \sigma_{\mathsf{Terms}(\rho)} \rangle$: it is applicable, as σ maps the body of ρ to $\mathsf{world}(w)$, and none of the two atoms in the head are present in $\mathsf{world}(w)$ (otherwise, w would be equal to w_1 or to w_2). Performing this rule application adds two children to the node of T labelled by $\mathsf{world}(w)$, of respective labels $\mathsf{world}(w_1)$ and $\mathsf{world}(w_2)$. Set $W = (W \setminus w) \cup \{w_1, w_2\}$.

As the tree structure of $\mathcal{I}$ is finite, and that at each step, either the size of W is decreasing, or the null w considered is replaced by two nulls w_1 and w_2 such that $\mathsf{world}(w_1)$ and $\mathsf{world}(w_2)$ are labels of nodes of strictly greater depth than $\mathsf{world}(w)$ in that tree, the above process terminates. Moreover, for each leaf of the considered prefix of T, it holds that its label and Σ_2 entail $\texttt{Goal}$, by Lemma 33 and the fact that each leaf is labelled by some $\mathsf{world}(w)$ such that $\texttt{Goal}'(w) \in \mathcal{I}$ (so that $\texttt{Goal} \in \mathsf{saturatedWorld}_{\mathcal{I}}(w)$). Hence, one can expand T in such a way that all its leaves are labeled by sets containing $\texttt{Goal}$, which concludes the proof. □

Lemma 37. *For every database $\mathcal{D}$ over $\mathcal{S}_{\mathsf{in}}(\Sigma)$, $\langle \Sigma'_1 \cup \Sigma'_2 \cup \Sigma'_3, \mathcal{D} \rangle$ is chase-terminating.*

Proof. By Lemma 34, chase sequences can be reordered without changing the number of rule applications such that rules from Σ'_1 are applied first, then rules from Σ'_2, then rules from Σ'_3. Lemma 31 implies that rules from Σ'_1 cannot be triggered indefinitely. As the tree structure is finite, if there are infinitely many rule applications performed with a rule from Σ'_2, there must be one world null w^* such that there are infinitely many rules of Σ'_2 that are performed mapping w to w^*. By the construction of Proposition 36, it implies that there exists a database over $\mathcal{S}_{\mathsf{in}}(\Sigma_2)$ for which Σ_2 does not terminate, which is against our assumptions on Σ_2. Finally, Σ'_3 does not contain any existentially quantified variable, hence only finitely many rules can be triggered. □

Lemma 18 is a direct consequence of Propositions 35, 36 and Lemma 37.

C Proofs of Section 6

Lemma 23. *The set $\mathcal{M}$ is not enumerable up to equivalence.*

Proof by Contradiction.

1. Suppose for a contradiction that the lemma does not hold. Then, there is an enumerator P that outputs a sequence of TMs that includes $\mathcal{M}$ up to equivalence. That is, the enumerator P outputs an infinite sequence $M_1, M_2, \ldots$ of TMs such that:
   - For each $i \geq 1$, we have that $M_i \in \mathcal{M}$.
   - For each $M \in \mathcal{M}$, there is some $i \geq 1$ such that M and M_i are equivalent.

   Two TM M and M' are equivalent if, for each word w, we have that M accepts w iff M' accepts w.
2. Consider the sequence $p_1, p_2, \ldots$ of natural numbers such that $p_1 = 1$ and p_i is the smallest prime with $p_i > p_{i-1}$ for each $i \geq 2$. Note that this sequence is infinite by Euclid's theorem.
3. By (2): there is an infinite sequence $\mathcal{D}_1, \mathcal{D}_2, \ldots$ of databases such that $\mathcal{D}_i = \{\texttt{ed}(u_1, u_2), \ldots, \texttt{ed}(u_{p_{i+1}}, u_1)\}$ for each $i \geq 1$.
4. Consider the TM M_d that, on input w, performs the computation:
   (a) Check if w corresponds to a database $\mathcal{D}$ that only contains facts defined over $\texttt{ed}$. If this is not the case, then *reject.*
   (b) If $\mathcal{D}$ can be homomorphically embedded into a database such as $\{\texttt{ed}(u_1, u_2), \ldots, \texttt{ed}(u_{k-1}, u_k)\}$ where k is smaller or equal than the number of nulls in $\mathcal{D}$, then *reject.*
   (c) If $\texttt{ed}(u, u) \in \mathcal{D}$ for some null u, then *accept.*

(d) If there is some $i \geq 1$ such that (i) there are less or the same number of nulls in $\mathcal{D}_i$ than in $\mathcal{D}$, (ii) M_i accepts some serialisation that corresponds to $\mathcal{D}_i$, and (iii) there is a homomorphism $h : \mathcal{D} \rightarrow \mathcal{D}_i$; then *reject*. Otherwise, *accept*.

5. By (4): the TM M_d halts on all inputs. Note the following remarks about instruction (4.d):
   - The TM M_i accepts some serialisation that corresponds to $\mathcal{D}_i$ if M_i accepts any such serialisation. Therefore, for each instantiation of i, the TM M_d only needs to check one (arbitrarily chosen) serialisation of $\mathcal{D}_i$ when executing (4.d).
   - When executing (4.d), the TM M_d only needs to check a finite amount of instantiations of $i \geq 1$ before rejecting with confidence due to condition (4.d.i).
6. After this enumeration, we prove that M_d does satisfy (iii) in Definition 5 by contradiction.
7. By (4.a), (5), and (6): the TM M_d is in $\mathcal{M}$.
8. By (4): M_d diagonalises over $M_1, M_2, \ldots$ and $\mathcal{D}_1, \mathcal{D}_2, \ldots$ That is, for any given $i \geq 1$, M_d accepts $\mathcal{D}_i$ iff M_i rejects $\mathcal{D}_i$.
   - If M_i accepts $\mathcal{D}_i$, then M_d rejects $\mathcal{D}_i$ in (4.d).
   - If M_i rejects $\mathcal{D}_i$, then M_d accepts $\mathcal{D}_i$ in (4.d). Note that, if we assume that M_d rejects $\mathcal{D}_i$ in this case we obtain a contradiction; namely, we can conclude that p evenly divides q for some prime numbers $p, q > 1$ with $p \neq q$.
9. Contradiction by (1), (7), and (8): the enumerator P is incomplete since it fails to print out a TM that is equivalent to M_d, which is in $\mathcal{M}$.

Suppose for a contradiction that there are some words w and v that correspond to some databases $\mathcal{D}$ and $\mathcal{E}$ such that $\mathcal{D}$ and $\mathcal{E}$ only contain facts defined over ed, M_d accepts $\mathcal{D}$, M_d rejects $\mathcal{E}$, and there is some homomorphism $h : \mathcal{D} \rightarrow \mathcal{E}$. We conduct a case-by-case analysis to show that this assumption results in a contradiction:

- Assume that $\mathcal{D}$ is accepted due to instruction (4.c). Then, $\texttt{ed}(u, u) \in \mathcal{D}$ for some null u and hence, $\texttt{ed}(t, t) \in \mathcal{E}$ for some null t since $h : \mathcal{D} \rightarrow \mathcal{E}$. Therefore, M_d accepts $\mathcal{E}$ due to (4.c) ($\lightning$).
- Assume that $\mathcal{D}$ is accepted due instruction (4.d). Two possible cases arise:
  - By definition $\mathcal{E}$ is a database that only contains facts defined over the predicate ed. Hence, $\mathcal{E}$ cannot be rejected due to (4.a).
  - If $\mathcal{E}$ is rejected due to (4.b), then $\mathcal{E}$ can be hom-embedded into a path over ed. Therefore, $\mathcal{D}$ can also be hom-embedded into the same path since $h : \mathcal{D} \rightarrow \mathcal{E}$ and M_d rejects $\mathcal{D}$ due to (4.b) ($\lightning$).
  - Assume that $\mathcal{E}$ is rejected due to instruction (4.d).

1. There is some $i \geq 1$ and a homomorphism g such that (i) the number of nulls in i is smaller than the number of nulls in $\mathcal{E}$, (ii) $g : \mathcal{E} \rightarrow \mathcal{D}_i$, and (iii) the TM M_i accepts the database $\mathcal{D}_i$.
2. By (1): $g \circ h : \mathcal{D} \rightarrow \mathcal{D}_i$.
3. If we assume that p_{i+1} is smaller or equal than the number of nulls in $\mathcal{D}$, then M_d rejects $\mathcal{D}$ due to (4.d) ($\lightning$). Therefore, we conclude that p_{i+1} is strictly greater than the number of nulls in $\mathcal{D}$.
4. If we assume that $\mathcal{D}$ can be homomorphically embedded into a path over ed, then M_d rejects $\mathcal{D}$ due to (4.b). Hence, we assume that this is not the case.
5. We obtain a contradiction from (2), (3), and (4). □