Chase-Based Computation of Cores for Existential Rules

Author: Lukas Gerlach

Supervisor: Prof. Dr. Markus Krötzsch

A thesis presented for the degree of
Diplom Informatik

Faculty of Computer Science
Institute of Theoretical Computer Science
Knowledge-Based Systems Group

29.08.2021

Abstract

The chase is a well studied sound and complete algorithm for computing universal models of knowledge bases that consist of an existential rule set and a set of facts. Since universal models can be used to solve generally undecidable reasoning tasks like BCQ entailment, it is no surprise that termination of the chase is undecidable as well. While conditions for termination and non-termination of variants like the skolem- or restricted-chase have been studied extensively, similar conditions for the core chase rarely exist. In practice, the core chase does not seem to be feasible. Still, compared to the other chase variants, the core chase not only yields a universal model but even a core, which intuitively is the smallest universal model that exists (up to isomorphism). Thus, the core chase terminates if and only if the given knowledge base has a finite universal model. In recent work, it has been shown that for rule sets that are "core-stratified", the restricted chase also yields universal models that are cores if it terminates.

In our work, we strengthen the existing result and proof that restricted and core chase termination exactly coincide for core-stratified rule sets. This also implies that we can use sufficient conditions for restricted chase non-termination as sufficient conditions for the non-existence of finite universal models. We also find a new fragment of existential rules for which core chase termination is decidable based on an existing result that shows decidability of restricted chase termination for the same fragment and we conjecture that this even holds for a slightly larger fragment by generalizing the so-called Fairness Theorem, which is a key part of the decidability proof. For non-core-stratified rule sets, we investigate a possible heuristic for core computation and introduce the hybrid chase as a mixture of restricted and core chase as a new chase variant equivalent to the core chase.

Declaration of Authorship

(Personal information and signature omitted in online version.)

I hereby declare that I have written this thesis on my own and that any participation of others has been acknowledged. I have not submitted this thesis partly or as a whole anywhere else for any purpose. I have clearly marked all references to existing work to the best of my knowledge and I have used no other than the cited sources.

Date	Signature (Lukas Gerlach)

Acknowledgements

I want to thank Prof. Dr. Markus Krötzsch for supervising and reviewing this thesis, especially for the very helpful discussions we had and the insights he gave as well as for providing the base work and various ideas for this thesis.

I also want to thank Prof. Sebastian Rudolph for reviewing this thesis as part of the formal assessment.

Additionally, I want to thank Alex Ivliev for the nice chat we had about the practical relevance of this work, which helped to formulate some of the corollaries more clearly.

I thank all further proofreaders, namely Alex Ivliev, Benjamin Schulz, Dana Kuban, and Lars Gerlach, for their feedback on the final draft of this thesis.

Last but not least, I want to thank my family and friends, including my former employer and colleagues at 3m5., for support during the preparation of this thesis and throughout my studies in general.

Contents

Chapter 1

Introduction

Existential rules play an important role for knowledge representation and reasoning [2]. Common fields of application are database dependencies, where existential rules are also known as tuple-generating dependencies [1], data exchange settings [13], and ontologies in general. In our work, we lean towards the latter point of view and study ontologies or knowledge bases, which, broadly speaking, consist of explicit logical facts and (existential) rules that allow the derivation of further facts. We regard the following basic example throughout the thesis that helps to understand many important notions.

Example 1.1. *Consider a knowledge base that consists of the logical fact set* $\{ \mathit{Pizza}(\mathit{order1}), \mathit{WeeklyOrder}(\mathit{order1}, \mathit{order2}) \}$ *and the following existential rules*

$$\mathit{Pizza}(x) \rightarrow \exists z.\mathit{SameDeliverer}(x, z) \wedge \mathit{Pizza}(z) \qquad (\rho_1)$$

$$\mathit{WeeklyOrder}(y, x) \rightarrow \exists z.\, \mathit{WeeklyOrder}(x, z) \qquad (\rho_2)$$

$$\mathit{Pizza}(x) \wedge \mathit{WeeklyOrder}(x, y) \rightarrow \mathit{Pizza}(y) \wedge \mathit{SameDeliverer}(x, y) \qquad (\rho_3)$$

Intuitively, we know that there exists an order1 that is a Pizza and another order2 that follows a week after order1. The rules allow to infer further knowledge according to the following intuitive meaning. By the first rule, if we orders pizza, then at some point we order another pizza from the same delivery service. By the second rule, if we start the habit of placing an order a week after another one, then we do the same every week in the future. By the third rule, if we have a pizza order and another order that results from our weekly

ordering habit, then we know that this order is also a pizza order and comes from the same deliverer as in the week before. When referencing this example, we also just reference the rules by ρ_1, ρ_2, and ρ_3, respectively.

1.1 Universal Models and Cores

There are various reasoning tasks over knowledge bases. One of the most prominent ones is *boolean conjunctive query (BCQ) entailment*, which is undecidable [6].

Example 1.2. *For the knowledge base from Example 1.1, we may ask for order2 if there exists a pizza order that was placed one week before order2 and came from the same deliverer using the BCQ*

$$\exists z.\mathit{Pizza}(z) \wedge \mathit{WeeklyOrder}(z, \mathit{order2}) \wedge \mathit{SameDeliverer}(z, \mathit{order2})$$

To check if such a BCQ is entailed by a knowledge base, we need to check if the BCQ is satisfied in every model of the knowledge base, i.e. if the BCQ is satisfied in every set of logical facts that contains the initial facts and satisfies all of the existential rules. For the satisfaction of the rules, we allow the introduction of so-called nulls, which act as a placeholder for the existentially quantified variables. To simplify the entailment check for a BCQ, it suffices to consider a *universal model* of the knowledge base [12]. Such a universal model can be homomorphically mapped into every model and thus generalizes all models. We give an example of such a universal model in the following. Throughout the thesis, when appropriate, we represent fact sets (and models) with predicates of arity at most two as graphs where constants and nulls are nodes, unary predicates are node labels, and binary predicates are edges.

Example 1.3. *A universal model of the knowledge base in Example 1.1 is the set of logical facts in Figure 1.1. We observe that the BCQ from Example 1.2 is entailed in this universal model and hence in any other (universal) model by mapping z to order1.*

For practical purposes, it is beneficial if we can obtain a "smallest" universal model. For example, the universal model in Figure 1.1 is infinite but there

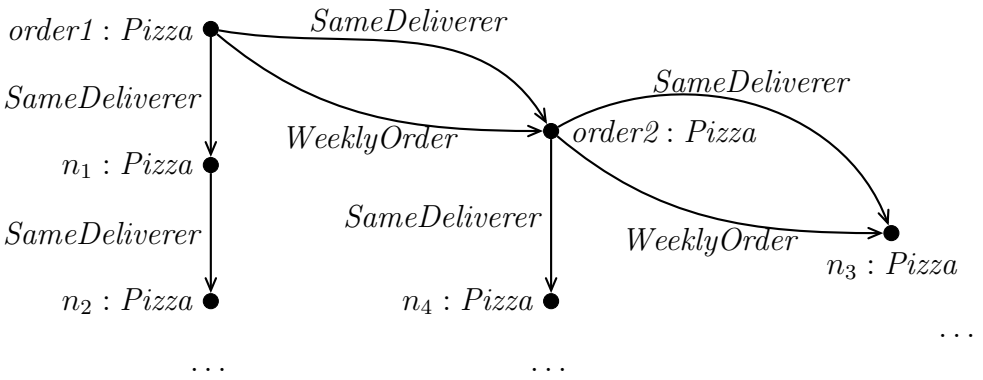


Figure 1.1: Infinite Universal Model of Example 1.1

may still exist a finite one. This is useful not only for BCQ entailment but also in a data exchange setting [14]. To formalize this, we consider *cores* [5, 12, 20]. Intuitively, a universal model that is a core is the smallest universal model up to isomorphism. We call such a model a *universal core model*. However, this intuition is not quite true if we consider infinite models and we give a formal definition later on that captures this case.

Example 1.4. *For the knowledge base in Example 1.1, we find an infinite universal core model in Figure 1.2. In comparison to Figure 1.1, the nodes occurring in the "chains" of only SameDeliverer relations can be mapped to nodes in the "chain" of both WeeklyOrder and SameDeliverer relations. By that, we obtain an endomorphism into a subset of the fact set in Figure 1.1. Although both universal models are infinite, we can say intuitively that the universal core model in Figure 1.2 is still "smaller".*

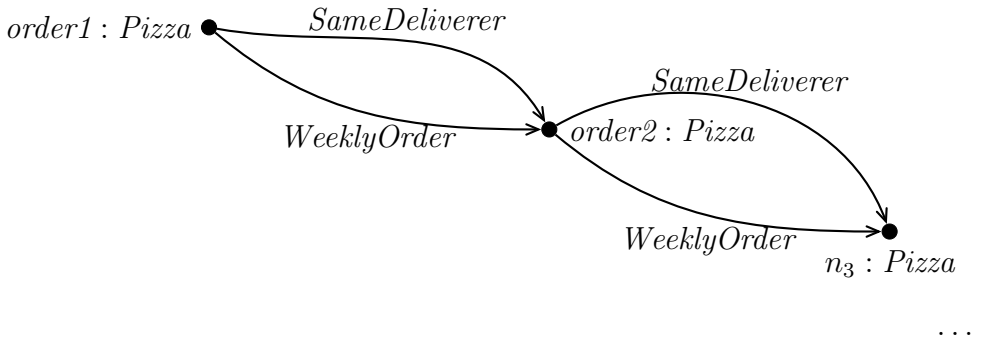


Figure 1.2: Infinite Universal Core Model of Example 1.1

1.2 The Chase

The chase is a sound and complete algorithm for computing a universal model of a given knowledge base [12, 18, 24]. Since this would allow for a decision procedure for BCQ entailment, it is not surprising that it is undecidable if the chase terminates on a given knowledge base. Still, some proofs regarding chase termination are rather recent [15, 18]. Different variants of this algorithm exist of which some terminate on more knowledge bases than others while being computationally more complex in theory and in practice [18, 22]. In our work, we consider the skolem chase, the restricted chase (also known as the standard chase), and the core chase. This order goes from "easiest" to "hardest" computationally but also from terminating on less knowledge bases to terminating on more knowledge bases. We also formalize termination later on.

The core chase, as its name suggests, computes a universal core model of a knowledge base. It is particularly interesting that the core chase terminates on a knowledge base if and only if a finite universal model of this knowledge base exists. Still, the core chase is not feasible in practice since this chase variant computes cores of intermediate fact sets during the chase computation, which is computationally intensive especially if the fact sets grow larger. To the best of our knowledge, the core chase is also not implemented in most current reasoning tools [8]. An exception to this is DEMo, which is a system for data exchange [25].

Recent work shows that for particular sets of existential rules, even the restricted chase yields universal core models without the need of an explicit core computation [21]. This is the case for rule sets that are *core-stratified*. While the restricted chase is still more involved than the skolem chase, it is more efficient in practice than the core chase.

1.3 Our Contributions

We study two main topics in this work.

First, we study core-stratified rule sets in more detail. We show as a main result that restricted and core chase termination exactly coincide for such rule

sets. This insight is obtained by strengthening a result from recent work where this is already suggested [21]. Within the proof, we utilize the *transfinite chase* as a formal framework for (partially) restricting application orders of rules. By the strengthened result, for core-stratified rule sets, we know that we can obtain an application order of rules for the restricted chase that terminates iff a finite universal model exists. Since a specific decision problem for restricted chase termination is known to be decidable for a fragment of existential rules, namely single-head guarded existential rules [16], we aim to expand this result to core chase termination. In this context, we also investigate if notions similar to core-stratification can be used to generalize the result for the restricted chase itself [16]. We also briefly explore if a notion similar to core-stratification can be found for the skolem chase instead of the restricted chase.

Second, we study if we can improve upon the core chase for rule sets that are not core-stratified. We briefly examine if we can use the notion of *pieces* [2] to transform certain rule sets into equivalent rule sets that are core-stratified. For the cases where we still do not obtain core-stratified rule sets, we aim to construct a refined version of the core chase that makes use of "partially core-stratified" rule sets. In particular, we investigate a heuristic for core computation within a chase sequence based on so-called *extended alternative matches* by formally introducing the *eam chase* while we also point out limitations of this approach. Furthermore, we can use a mixture of the restricted chase and the eam/core chase such that the intermediate core computation only needs to be done in some cases. Formally, we base this idea on the transfinite chase and call the adjusted variant the *hybrid chase*. In particular, we can combine the hybrid and eam chase to obtain a promising new chase variant as the main contribution of this second part.

We structure the thesis as follows:

- Chapter 2: We formally introduce basic notions like existential rules, universal core models, the chase, and decision problems for chase termination, which are well known from other work.
- Chapter 3: We recall important notions and results for when the restricted chase yields universal core models [21] to lay the basis for most of our investigations.

- Chapter 4: We investigate core-stratified rule sets in more detail by relating ideas presented in Chapter 3 to chase termination as outlined above. We show cases where restricted and core chase termination coincide utilizing the transfinite chase and we derive many implications. In particular, we investigate what changes if we consider only (single-head) guarded existential rules.
- Chapter 5: We focus on a practical procedure of computing cores for rule sets that are not core-stratified. We introduce the *eam chase* as a heuristic for the core chase computation based on extended alternative matches and point out limitations of this approach. As a main result of this chapter, we present the *hybrid chase* as a mixture of the restricted chase and the eam/core chase based on the idea of a transfinite chase. The eam chase and hybrid chase and in particular the combination of the two terminates iff a finite universal (core) model exists and yields such a finite universal core model in this case.

Chapter 2

Preliminaries

In this chapter, we introduce basic notions that we use throughout the thesis. Most prominently, this includes existential rules as a fragment of first-order logic as well as the chase as a basic reasoning algorithm for this fragment.

2.1 Existential Rules and Universal Models

As building blocks for existential rules [2], we define $\mathsf{V}, \mathsf{C}, \mathsf{N}, \mathsf{P}$ to be countably infinite, pairwise disjoint sets of *variables*, *constants*, *nulls*, and *predicates*, respectively. Every predicate is associated with its *arity* by the function $ar : \mathsf{P} \to \mathbb{N}$. Elements of $\mathsf{V} \cup \mathsf{C} \cup \mathsf{N}$ are called *terms*. A term t is *ground* if $t \in \mathsf{C} \cup \mathsf{N}$. An *atom* is an expression of the form $P(t_1, \ldots, t_n)$, where P is a predicate with arity n and $t_1, \ldots, t_n$ are terms. The atom $P(t_1, \ldots, t_n)$ is *ground* if $t_1, \ldots, t_n$ are ground. We also refer to ground atoms as *facts*.

For existential rules, we consider *conjunctions of atoms* $a_1 \wedge \cdots \wedge a_m$ that we also treat as sets $\{ a_1, \ldots, a_m \}$ when suitable. We also denote a conjunction of atoms ϕ with $\phi(\vec{x})$ to stress that ϕ features exactly the variables in $\vec{x}$.

Definition 2.1 (Existential Rule)**.** An *(existential) rule* ρ is an expression of the form

$$\rho = \forall \vec{x}, \vec{y}.[\mathit{Body}(\vec{x}, \vec{y}) \to \exists \vec{z}.\mathit{Head}(\vec{x}, \vec{z})]$$

where *Body* and *Head* are conjunctions of atoms such that all terms are variables or constants and $\vec{x}, \vec{y}$, and $\vec{z}$ are pairwise disjoint lists of variables.

We omit universal quantifiers in rules in the following. Note that a rule does not feature variables other than those in $\vec{x}, \vec{y}$, and $\vec{z}$. We refer to *Body* and *Head* as $body(\rho)$ and $head(\rho)$, respectively. We call the list of variables $\vec{x}$ that occur in both $body(\rho)$ and $head(\rho)$ the *frontier* of ρ and denote it by $frontier(\rho)$. A rule is *Datalog* if it does not feature existentially quantified variables. A rule ρ is a *single-head* rule if $head(\rho)$ consists of exactly one atom. A rule ρ is *guarded* if there is an atom in $body(\rho)$ that contains all universally quantified variables in ρ. A rule ρ is *linear* if $body(\rho)$ consist of exactly one atom. Note that every linear rule is guarded. Analogously, we say that a rule set is *single-head*, *guarded*, or *linear* if every rule within the rule set is *single-head*, *guarded*, or *linear*, respectively.

Definition 2.2 (Homomorphism)**.** For two sets of atoms A and A', a function $h : \mathsf{V} \cup \mathsf{C} \cup \mathsf{N} \to \mathsf{V} \cup \mathsf{C} \cup \mathsf{N}$ is a *homomorphism* from A to A', if

- $h(c) = c$ for all $c \in \mathsf{C}$ and
- $P(h(t_1), \ldots, h(t_n)) \in A'$ for all $P(t_1, \ldots, t_n) \in A$.

A homomorphism h from A to A' is *strong* if additionally $P(h(t_1), \ldots, h(t_n)) \notin A'$ for all $P(t_1, \ldots, t_n) \notin A$.

We implicitly lift a homomorphism $h : \mathsf{V} \cup \mathsf{C} \cup \mathsf{N} \to \mathsf{V} \cup \mathsf{C} \cup \mathsf{N}$ from A to A' to an atom mapping $h : A \to A'$ such that, for an atom a, $h(a)$ is the atom that results from replacing all terms in a according to h. Similarly, we implicitly lift homomorphisms to sets of atoms. In particular, we usually identify a homomorphism h from A to A' by its atom mapping $h : A \to A'$.

A *substitution* is a function $\theta : \mathsf{V} \cup \mathsf{C} \cup \mathsf{N} \to \mathsf{V} \cup \mathsf{C} \cup \mathsf{N}$ with $\theta(t) = t$ for all $t \in \mathsf{C} \cup \mathsf{N}$, i.e. θ is only allowed to remap variables. A fact set F entails a rule ρ, written $F \models \rho$, if for every substitution θ that is a homomorphism from $body(\rho)$ to F, there exists a substitution θ' that is a homomorphism from $head(\rho)$ to F such that $\theta(x) = \theta'(x)$ for every variable $x \in frontier(\rho)$. Analogously, a fact set F entails a rule set R, written $F \models R$, if $F \models \rho$ for all $\rho \in R$.

An *instance* is a finite fact set that does not feature nulls. A *knowledge base* is a pair of a rule set and an instance. We already introduced such a knowledge base (informally) in Example 1.1.

Definition 2.3 (Model)**.** A set of facts M is a *model* for a knowledge base $\mathcal{K} = \langle R, I \rangle$ if $I \subseteq M$ and $M \models R$.

A model U of a knowledge base $\mathcal{K}$ is *universal* [12, 18] if for every model M of $\mathcal{K}$, we find a homomorphism $h : U \to M$. For the knowledge base in Example 1.1, we know that Example 1.3 is a universal model. This follows since this model can be found using the chase, which we describe in Section 2.3. A fact set C is a *core* if every endomorphism $h : C \to C$ is strong and injective. Other definitions for capturing infinite cores are discussed in the literature [5]. Since we base our considerations on many notions that are introduced in a recent work by Krötzsch [21], we use the same definition as it is used in that paper. Note that, for a finite fact set C', we can say that C' is a core if every endomorphism is surjective, i.e. if every endomorphism is an automorphism. This is indeed not always true for infinite fact sets as we observe in the following example.

Example 2.4. *Consider the fact set from Figure 2.1 that only features nulls and no constants.*

There exists a (strong and injective) endomorphism h that maps $n_i \mapsto n_{i+1}$ for all $i > 0$ that is not an automorphism.

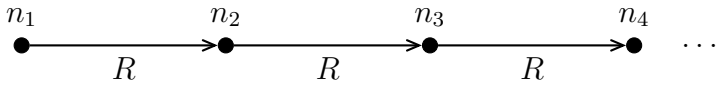


Figure 2.1: Core where some Endomorphism is not an Automorphism

Furthermore, for a fact set F, a *core of* F is a fact set $F' \subseteq F$ such that there exists a homomorphism $h : F \to F'$ and F' is a core. If F is finite, then it has a unique core (up to bijective renaming of nulls) that we denote with $\mathit{core}(F)$ [20].

For two fact sets F and F', checking if $F = \mathit{core}(F')$ is captured by the decision problem *CoreIdentification* [14].

Proposition 2.5 ([14])**.** *CoreIdentification is* DP*-complete.*

Note that DP is the class of problems that form an intersection of a problem in NP and a problem in coNP. Intuitively, for CoreIdentification, we need

to check that there is a homomorphism from F' to F (NP) and that F is a core, i.e. that every endomorphism in F is strong and injective (coNP).

Later on, we are mostly interested in universal models that are cores, which we call *universal core models*. We can think of a universal core model as the smallest possible universal model up to isomorphism. Technically, for infinite universal core models, there may exist multiple universal core models that are not isomorphic [11] but this idea suffices as an intuition. Because of this (intuitive) property, universal core models are desirable in practice for reasoning tasks over existential rules, most prominently in data exchange settings [13, 14] but also for BCQ entailment which we briefly discuss in Section 2.2. It is worthy to note that the existence of an infinite universal core model of a knowledge base $\mathcal{K}$ indicates that no finite universal model of $\mathcal{K}$ can exist.

Lemma 2.6. *Consider a knowledge base $\mathcal{K}$. There cannot exist both a finite universal model and an infinite universal core model of $\mathcal{K}$.*

Proof. Suppose for a contradiction that some knowledge base $\mathcal{K}$ has a finite universal model U and an infinite universal core model U'. Since U' is universal, it can be homomorphically mapped into every other model. In particular, there exists a homomorphism $h : U' \to U$. Note that h is not injective since $|U| < |U'|$. Vice versa, there also exists a homomorphism $h' : U \to U'$. Let $h'' \coloneqq h' \circ h$ be an endomorphism on U'. Then h'' is not injective since h is not injective. This contradicts the assumption that U' is a core. □

For Example 1.1, we can obtain a universal core model in Example 1.4 by remapping nulls according to endomorphisms in the universal model that we already have in Example 1.3. By Lemma 2.6, we can already infer that no finite universal model exists for the knowledge base in Example 1.1.

2.2 Boolean Conjunctive Query Entailment

Important reasoning tasks over existential rules are *conjunctive query (CQ) answering* and *boolean conjunctive query (BCQ) entailment* [1]. We introduce these reasoning tasks in more detail since we also use BCQ entailment in proofs later in the thesis.

Definition 2.7. A *conjunctive query* is an expression of the form $\sigma := \exists \vec{z}.\phi(\vec{x}, \vec{z})$ where $\vec{x}$ and $\vec{z}$ are lists of variables and ϕ is a conjunction of atoms such that all terms are variables or constants.

If the list of variables $\vec{x}$ is empty, σ is called *boolean conjunctive query (BCQ)*. In the context of a fact set F, a mapping f from the variables $\vec{x}$ to constants or nulls is an *answer* to σ if there exists a substitution θ that is a homomorphism from ϕ to F with $\theta(x) = f(x)$ for each $x \in \vec{x}$. Analogously, if σ is a BCQ, then σ is *entailed* by F if there exists a substitution θ that is a homomorphism from ϕ to F. For the BCQ σ in Example 1.2, we find that σ is entailed by the fact sets in Examples 1.3 and 1.4.

The reasoning tasks CQ answering and BCQ entailment can be formulated as decision problems CQ and BCQ. CQ is defined as the set that contains all tuples $\langle R, I, \sigma, f\rangle$ where R is a rule set, I is an instance, and σ is a CQ such that f is an answer to σ in the context of every model of $\langle R, I\rangle$. BCQ is defined as the set that contains all tuples $\langle R, I, \sigma\rangle$ where R is a rule set, I is an instance, and σ is a BCQ such that σ is entailed by every model of $\langle R, I\rangle$. Note that this is the case iff $\langle R, I\rangle$ entails σ under first-order logic semantics and iff σ is entailed by some universal model of $\langle R, I\rangle$.

Intuitively, CQ can be reduced to BCQ by replacing variables in the query σ according to the potential answer f. The formal reduction is more involved since f may map variables to nulls but the idea stays the same. Therefore, we only consider BCQ in the rest of the thesis. We use BCQ later on to show some undecidability results using reductions since BCQ itself is known to be undecidable [6].

Proposition 2.8. *BCQ is undecidable.*

Proof Sketch. The halting problem of turing machines can be reduced to BCQ by encoding the turing machine as existential rules and the input as an instance. The BCQ then asks if a halting configuration is reached. □

Note that BCQ is even undecidable if we consider a fixed instance. However, when using only guarded rule sets, BCQ becomes decidable [3].

2.3 The Chase

The chase is a sound and complete algorithm for the computation of universal models of knowledge bases [12, 18, 24]. Thus, the chase is useful for tackling reasoning tasks like BCQ. In practice, it is beneficial if the produced universal models are as small as possible, i.e. cores ideally. Different variants of the chase algorithm exist, some of which are known to yield universal core models. The basic idea of all variants is to use rules to derive new facts from existing facts. We formalize this idea using *triggers*.

Definition 2.9 (Trigger)**.** A *trigger* is a pair $\lambda \coloneqq \langle \rho, \theta \rangle$ of a rule ρ and a substitution θ that maps the existential variables in ρ to themselves. In the context of a fact set F, λ is

- *active* if $\theta(\mathit{body}(\rho)) \subseteq F$ and
- *obsolete* if there exists a substitution θ' with $\theta(x) = \theta'(x)$ for all variables $x \in \mathit{frontier}(\rho)$ such that $\theta'(\mathit{head}(\rho)) \subseteq F$.

The *application* of a trigger $\lambda \coloneqq \langle \rho, \theta \rangle$ on a fact set F is defined as $\lambda(F) \coloneqq F \cup \theta'(\mathit{head}(\rho))$ where θ' is a substitution with $\theta(x) = \theta'(x)$ for all variables $x \in \mathit{frontier}(\rho)$ that maps the existential variables in ρ to fresh nulls. Note that θ' is a homomorphism from $\theta(\mathit{head}(\rho))$ to $\lambda(F)$. Also note that the facts that are newly introduced by a trigger do not depend on the fact set, which the trigger is applied upon. Therefore, we also denote the facts that are newly introduced by a trigger λ with $F_\lambda \coloneqq \theta'(\mathit{head}(\rho))$. If the substitution for a trigger application that features a rule ρ is obvious or not relevant, we also say that we apply ρ to indicate that we apply some trigger that features ρ.

Based on triggers, our goal is to define three different chase variants: the *skolem chase* (*sk*), the *restricted chase* (*res*), and the *core chase* (*core*). At first, we give definitions for trigger applicability for the skolem chase and the restricted chase. The core chase uses the same applicability condition as the restricted chase but uses a slightly different idea for the actual chase as we present later on.

Definition 2.10. In the context of a fact set F, a trigger $\lambda = \langle \rho, \theta \rangle$ is

- *sk-applicable* if λ is active w.r.t. F and no fact in F was obtained by a trigger $\lambda' = \langle \rho, \theta' \rangle$ with $\theta'(x) = \theta(x)$ for all variables $x \in \mathit{frontier}(\rho)$ or

- *res-applicable* if λ is active and not obsolete w.r.t. F, respectively.

For a fact set F and a rule ρ, we formulate the decision problem *RuleApplicability* by asking if there exists a substitution θ such that $\langle \rho, \theta \rangle$ is *res*-applicable to F. The complexity of this decision problem is as follows [18].

Proposition 2.11 ([18, Theorem 3.1])**.** *RuleApplicability is Σ_2^{P}-complete w.r.t. the size of the corresponding rule and fact set and is in* P *w.r.t. the size of the fact set if we fix a rule.*

The complexity of RuleApplicability is of interest later on in some proofs as well as for the investigation of core chase performance in Chapter 5.

Based on the applicability notions for triggers, we formally introduce *sk-/res-* and *core-chase sequences* that capture the actual chase idea. When the concrete type of chase sequence is not important, we also just use the term *chase sequence*. We only define *sk-* and *res*-chase sequences first since *core-chase sequences* use a slightly different idea.

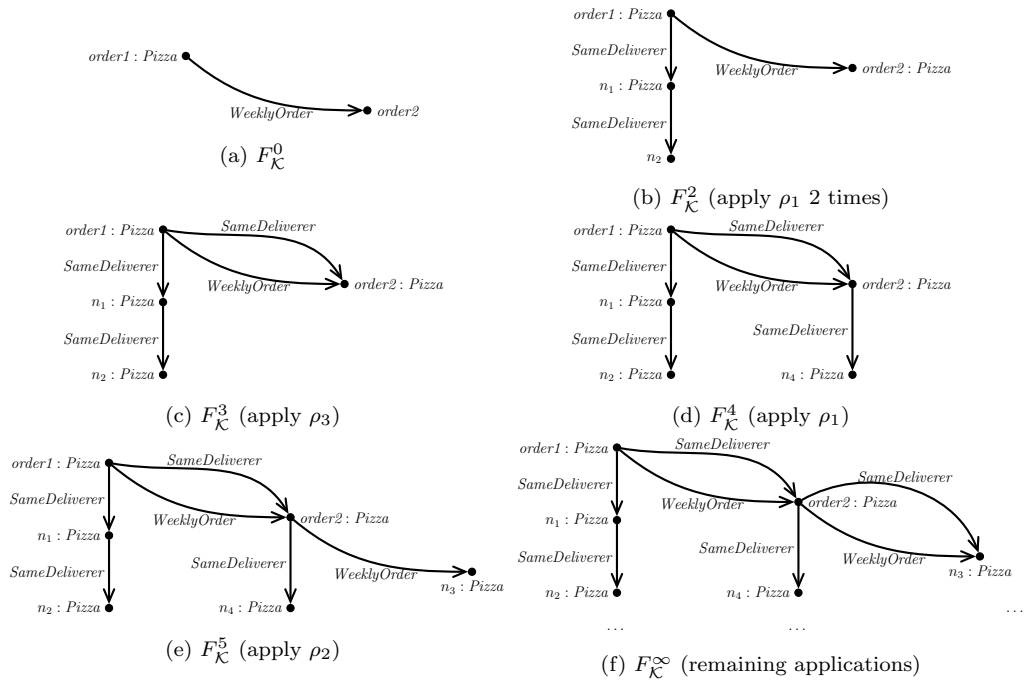
Definition 2.12. Consider a knowledge base $\mathcal{K} \coloneqq \langle R, I \rangle$ and a chase variant $* \in \{ sk, res \}$. A *(fair) $*$-chase sequence* for $\mathcal{K}$ is a sequence of fact sets $F_{\mathcal{K}}^0, F_{\mathcal{K}}^1, \ldots$ such that

1. $F_{\mathcal{K}}^0 = I$,
2. for each $i \geq 0$, $F_{\mathcal{K}}^{i+1} = \lambda(F_{\mathcal{K}}^i)$ for some trigger λ that is $*$-applicable w.r.t. $F_{\mathcal{K}}^i$ or $F_{\mathcal{K}}^{i+1} = F_{\mathcal{K}}^i$ if no such trigger exists, and
3. for every trigger λ that is active w.r.t. $F_{\mathcal{K}}^i$ for some i, there exists an $i' \geq i$ such that λ is not $*$-applicable w.r.t. $F_{\mathcal{K}}^{i'}$.

A sequence of fact sets that fulfills conditions 1 and 2 but not 3 is called *unfair*. Note that we can identify any (un)fair *sk-/res*-chase sequence by the sequence of the used triggers instead of the produced fact sets. The *chase result* of a *sk/res*-chase sequence $F_{\mathcal{K}}^0, F_{\mathcal{K}}^1, \ldots$ is the (possibly infinite) fact set $F_{\mathcal{K}}^\infty \coloneqq \bigcup_{i \geq 0} F_{\mathcal{K}}^i$.

Example 2.13. *We present a res-chase sequence for the knowledge base $\mathcal{K}$ in Example 1.1 in Figure 2.2. At first, we apply ρ_1 two times with the obvious substitutions to get from (a) to (b). For (c), we apply ρ_3 and we use ρ_2, to obtain (d). We can now apply ρ_1 on the newly obtained facts to get*

to (e). Finally, we can keep applying ρ_2 followed by ρ_3. We can also apply ρ_1 infinitely often on the SameDeliverer "chains" that we already started. By that, we obtain an infinite universal model in (f) that is the same as in Example 1.3.



(a) $F^0_{\mathcal{K}}$ (b) $F^2_{\mathcal{K}}$ (apply ρ_1 2 times) (c) $F^3_{\mathcal{K}}$ (apply ρ_3) (d) $F^4_{\mathcal{K}}$ (apply ρ_1) (e) $F^5_{\mathcal{K}}$ (apply ρ_2) (f) $F^\infty_{\mathcal{K}}$ (remaining applications)

Figure 2.2: Restricted Chase for Knowledge Base in Example 1.1

Keep in mind that we can pick different rule application orders for the restricted chase that possibly yield different results.

For the core chase, we aim to obtain a chase sequence that is unique for a given knowledge base [12]. For this sake, triggers are applied in parallel as follows. For a rule set R and a fact set F, we define $R(F) \coloneqq F \cup \bigcup_{\lambda \in \Lambda^F_R} \lambda(F)$ where Λ^F_R is the set of all triggers featuring rules in R that are *res*-applicable w.r.t. F.

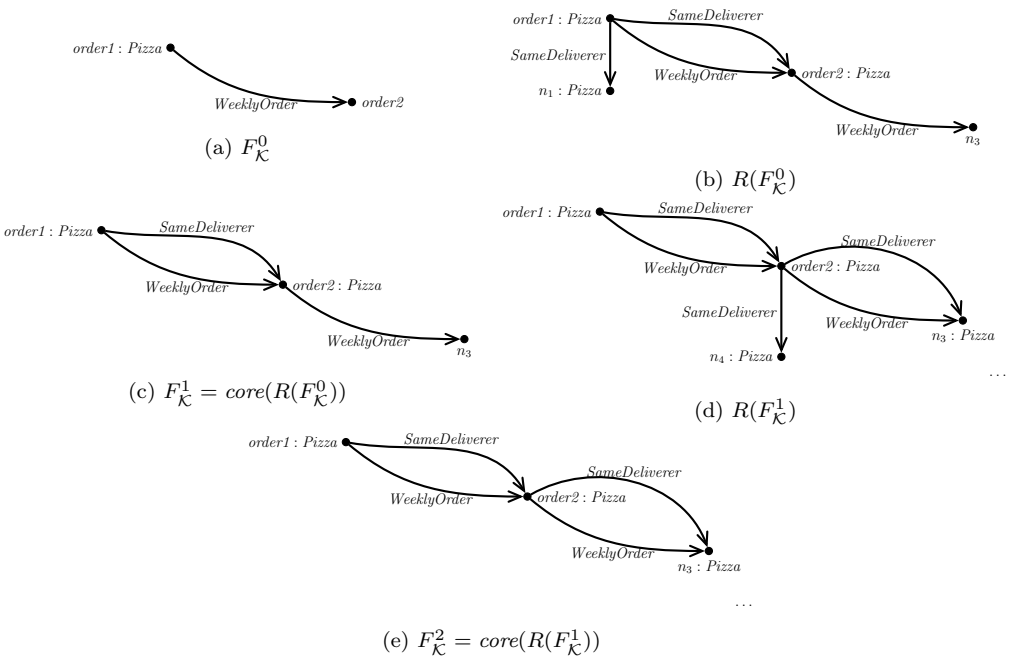
Definition 2.14. Consider a knowledge base $\mathcal{K} \coloneqq \langle R, I \rangle$. A *core-chase sequence* for $\mathcal{K}$ is a sequence of fact sets $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$ such that

1. $F^0_{\mathcal{K}} = I$ and
2. $F^{i+1}_{\mathcal{K}} = \mathit{core}(R(F^i_{\mathcal{K}}))$ for each $i \geq 0$.

Core chase sequences are always "fair" since there is no choice of triggers involved. Recall that a naive computation of *core*(...) that relies on finding homomorphisms over the whole fact set is computationally intensive in practice (Proposition 2.5). We discuss a possible practical improvement for this issue in Chapter 5.

For a *core*-chase sequence, we cannot just use the union over all fact sets as the result, since the core computation may remove some facts in later steps of the sequence. Thus, we define the *chase result* of a *core*-chase sequence $F^0_{\mathcal{K}}, F^1_{\mathcal{K}} \ldots$ as $F^\infty_{\mathcal{K}} := F^j_{\mathcal{K}}$ where $j \geq 0$ is the smallest number such that $F^j_{\mathcal{K}} = F^{j+1}_{\mathcal{K}}$. If such a j exists, we also say that the chase sequence *terminates*. This definition of termination also applies to *sk*/*res*-chase sequences. If such a j does not exist, $F^\infty_{\mathcal{K}}$ is not defined for the *core*-chase. We discuss chase termination in more detail in Section 2.4.

Example 2.15. *We present the unique core-chase sequence for the knowledge base $\mathcal{K}$ from Example 1.1 in Figure 2.3. The result of this sequence is not formally defined since it does not terminate, however, we approach the infinite universal core model from Example 1.4 step by step.*



(a) $F^0_{\mathcal{K}}$

(b) $R(F^0_{\mathcal{K}})$

(c) $F^1_{\mathcal{K}} = core(R(F^0_{\mathcal{K}}))$

(d) $R(F^1_{\mathcal{K}})$

(e) $F^2_{\mathcal{K}} = core(R(F^1_{\mathcal{K}}))$

Figure 2.3: Core Chase for Knowledge Base in Example 1.1

In general, we know that every chase sequence for a knowledge base yields a universal model. For the core chase, we only have the formal limitation that the sequence needs to terminate since otherwise there is no result defined.

Proposition 2.16. *Consider a knowledge base $\mathcal{K}$. The result $F_{\mathcal{K}}^{\infty}$ of a chase sequence (if defined in case of the core chase) is a universal model of $\mathcal{K}$.*

Proof Sketch. For a knowledge base $\mathcal{K} \coloneqq \langle R, I \rangle$, a chase sequence satisfies rules in R by introducing "missing" facts using fresh nulls if required. Hence, the chase result U is a model of $\mathcal{K}$. Similar facts necessarily exist in every model M of $\mathcal{K}$ to satisfy all rules in R. Thus, we can remap the nulls in U to terms in M to obtain a homomorphism from U into M. Since this is possible for an arbitrary model M, U is a universal model of $\mathcal{K}$. □

A result similar to Proposition 2.16 also exists in a work by Deutsch et al. [12, Theorem 5]. For the core chase, we obtain an interesting relation between termination and the existence of finite universal models of a knowledge base.

Proposition 2.17 ([12, Theorem 7])**.** *Consider a knowledge base $\mathcal{K}$. The unique core-chase sequence on $\mathcal{K}$ terminates iff $\mathcal{K}$ has a finite universal model.*

In this case, the unique *core*-chase sequence on $\mathcal{K}$ yields a finite universal core model.

It turns out that for certain rule sets, the restricted chase also yields cores. This has been shown in recent work [21] that is presented in Chapter 3. We strengthen these results further in Chapters 4 and 5 to develop a comprehensive procedure for computing universal core models that promises to be more efficient than naive implementations of the core chase.

Remark 2.18. *While we formally only define chase sequences for knowledge bases, i.e. pairs of rule sets and instances, we can in the same way use an arbitrary (possibly infinite) fact set instead of an instance for the skolem and restricted chase. For the core chase, this does not work since we only define core(F) for finite fact sets F and we want to obtain a unique chase sequence.*

We make use of infinite fact sets in the transfinite chase and hybrid chase in Chapters 4 and 5, respectively.

2.4 Chase Termination

We already noted that a chase sequence $F_{\mathcal{K}}^0, F_{\mathcal{K}}^1 \ldots$ for a knowledge base $\mathcal{K}$ *terminates* if there exists a $j \geq 0$ such that $F_{\mathcal{K}}^j = F_{\mathcal{K}}^{j+1}$. We introduce some further decision problems regarding termination that are studied throughout the literature. Beside considering a particular chase sequence, for a given knowledge base, it is interesting to know if any or all chase sequences on that knowledge base terminate. Furthermore, given a rule set, it is valuable to know if we find terminating chase sequences for all instances, i.e. for all knowledge bases featuring the rule set.

Formally, we consider the following decision problems CT_{is}^* with $i \in \{\forall\} \cup \{I \mid I \text{ is an instance}\}$ and $s \in \{\forall, \exists\}$ that are the sets of rule sets for which all/some chase sequences (s) on all instances / a given instance (i) terminate [18]. It has been shown that all of these problems are undecidable [6, 7, 15, 18]. However, we know that some chase variants terminate for more rule sets than others. In particular, the following relations exist between some of the decision problems [18].

$$\mathsf{CT}_{\forall\forall}^{sk} = \mathsf{CT}_{\forall\exists}^{sk} \subset \mathsf{CT}_{\forall\forall}^{res} \subset \mathsf{CT}_{\forall\exists}^{res} \subset \mathsf{CT}_{\forall\forall}^{core} = \mathsf{CT}_{\forall\exists}^{core}$$

We briefly argue why the equalities hold. For the skolem chase, the order of trigger application is irrelevant (as long as it is fair) since obsoleteness is not taken into account and triggers with the same rule and frontier mapping yield the same set of facts (up to bijective renaming of nulls). For the core chase, recall that there is a unique core chase sequence for every given knowledge base. Thus, for *sk* and *core*, we just write $\mathsf{CT}_{\forall}^{sk} \coloneqq \mathsf{CT}_{\forall\forall}^{sk} = \mathsf{CT}_{\forall\exists}^{sk}$ and $\mathsf{CT}_{\forall}^{core} \coloneqq \mathsf{CT}_{\forall\forall}^{core} = \mathsf{CT}_{\forall\exists}^{core}$. For the restricted chase, the order of applications indeed influences termination as shown in the following example.

Example 2.19. *We investigate an example similar to one by Gogacz et al. [16] which we will also examine later on in its original form and with some modifications. Consider the rule set R consisting of the following two rules*

$$\rho_1 \coloneqq S(x, y, y) \to \exists z.S(x, z, y) \wedge S(z, y, y)$$
$$\rho_2 \coloneqq S(x, y, z) \to S(x, x, z)$$

and the instance $I \coloneqq \{S(a, b, b)\}$. If we apply the only possible trigger for ρ_2 first, no trigger for ρ_1 is res-applicable anymore. However, we can alternate

between trigger applications for ρ_1 and ρ_2 starting with ρ_1 to obtain a fair res-chase sequence for $\langle R, I \rangle$ that does not terminate.

The original example [16] reveals another interesting insight about (un)fair *res*-chase sequences.

Example 2.20. *Consider the rule set R consisting of the following two rules*

$$\rho_1 \coloneqq S(x, y, y) \to \exists z.S(x, z, y) \wedge S(z, y, y)$$
$$\rho_2 \coloneqq S(x, y, z) \to S(z, z, z)$$

and the instance $I \coloneqq \{ S(a, b, b) \}$. There exists an unfair res-chase sequence for $\langle R, I \rangle$ that does not terminate by applying only triggers for ρ_1. However, every fair res-chase sequence for $\langle R, I \rangle$ terminates since any application of ρ_2 makes all triggers for ρ_1 obsolete.

This example is closely related to the so-called Fairness Theorem [16] that is examined in more detail in Chapter 4.

To tackle some of the decisions problems for chase termination, *acyclicity notions* and *cyclicity notions* for different chase variants have been introduced as sufficient conditions for termination or non-termination, respectively. For example for the skolem chase, if a rule set R is MFA [19], then $R \in \mathsf{CT}^{sk}_{\forall}$; and if R is MFC [10], then $R \notin \mathsf{CT}^{sk}_{\forall}$, respectively. Analogously, for the restricted chase, if R is RMFA [10], then $R \in \mathsf{CT}^{res}_{\forall\forall} \subset \mathsf{CT}^{res}_{\forall\exists}$; and if R is RMFC [10], then $R \notin \mathsf{CT}^{res}_{\forall\forall}$, respectively. In fact, we even find $R \notin \mathsf{CT}^{res}_{\forall\exists}$ if R is RMFC. This follows from the definition of RMFC [10] and we give a more detailed explanation why this holds in the proof of Corollary 4.15 without going too much into the formal details of RMFC. To the best of our knowledge, no acyclicity notions and especially no cyclicity notions tailored the core chase have been published. In terms of termination, MFA and RMFA are also sufficient conditions for the core chase but for non-termination, a rule set may still be in $\mathsf{CT}^{core}_{\forall}$ even though it is RMFC. Non-termination of the core chase is especially intriguing since non-membership of a rule set R in $\mathsf{CT}^{core}_{\forall}$ is equivalent to the existence of an instance I such that $\langle R, I \rangle$ does not have a finite universal model independent of the chosen chase variant. Later on, we show that RMFC is actually a sufficient condition for non-membership in $\mathsf{CT}^{core}_{\forall}$ for certain rule sets.

Beside acyclicity notions, it is also possible to consider only fragments of existential rules like (single-head) guarded or (single-head) linear rules for which the corresponding restrictions of some of the previously mentioned problems are known to be decidable. Namely, $\mathsf{CT}^{sk}_{\forall}$ is decidable for guarded (and linear) rules [9], $\mathsf{CT}^{res}_{\forall\forall}$ is decidable for single-head guarded (and single-head linear) rules [16], $\mathsf{CT}^{res}_{\forall\exists}$ is decidable for single-head linear rules [23], and $\mathsf{CT}^{core}_{\forall}$ is decidable for single-head linear rules [23]. In Chapter 4, we investigate in more detail if we can find a decidability result for $\mathsf{CT}^{core}_{\forall}$ for (single-head) guarded existential rules based on the decidability result for $\mathsf{CT}^{res}_{\forall\forall}$ [16].

In the next Chapter, notions by Krötzsch [21] are introduced that help us to relate restricted chase and core chase in particular in terms of termination in Chapter 4 and that also help us to introduce alternative computation procedures for the core chase in Chapter 5.

Chapter 3

Computing Cores with the Restricted Chase

From now on, we focus on the computation of universal core models. The core chase can be used to do this computation but it is rather expensive due to the necessity of computing intermediate cores of fact sets during the chase. Thankfully, recent work shows that the restricted chase also yields universal core models for certain rule sets [21].

Throughout this chapter, we recall important notions and results from a recent paper by Krötzsch [21] upon which we base our further considerations in the following chapters. One main result of the paper is a condition for when the restricted chase yields a finite universal core model for every instance on a given rule set. We mostly consider this chapter to be an extension to the preliminaries. For this sake, we largely refrain from the introduction of new results in this chapter and postpone most of them to Chapters 4 and 5 to clearly separate our contributions from existing work. Still, to benefit the overall reading flow, this chapter contains some smaller new contributions. Deviating from the original paper, we formulate decision problems for so-called "alternative matches" similar to those for chase termination. In addition, we show a particular result in more detail in the proof of Proposition 3.5 and we show new results in Propositions 3.6 and 3.9. We highlight these new contributions explicitly in this chapter.

3.1 Alternative Matches

The notion of *alternative matches* helps to identify cases when the restricted chase can be used to compute universal core models.

Definition 3.1. Consider a trigger $\lambda := \langle \rho, \theta \rangle$ and a fact set F. A homomorphism $h : F_\lambda \to F$ is an *alternative match* for λ w.r.t. F if $h(t) = t$ for every $t \in \theta(\mathit{frontier}(\rho))$ and there is a null in F_λ that does not occur in $h(F_\lambda)$.

A *res*-chase sequence $F^0_\mathcal{K}, F^1_\mathcal{K}, \ldots$ has an alternative match if there exists a $j \geq 0$ such that the trigger λ that is applied to $F^j_\mathcal{K}$ has an alternative match w.r.t. $F^\infty_\mathcal{K}$. Intuitively, an alternative match for a trigger λ that has already been applied indicates that the application of λ introduces nulls that may not be required once further facts are derived.

Example 3.2. *We find alternative matches in the restricted chase sequence from Example 2.13. In particular, all triggers that are applied featuring ρ_1 have alternative matches. These are marked by the dotted arrows in Figure 3.1. For example, the trigger that introduces n_1 has an alternative match that maps n_1 to order2.*

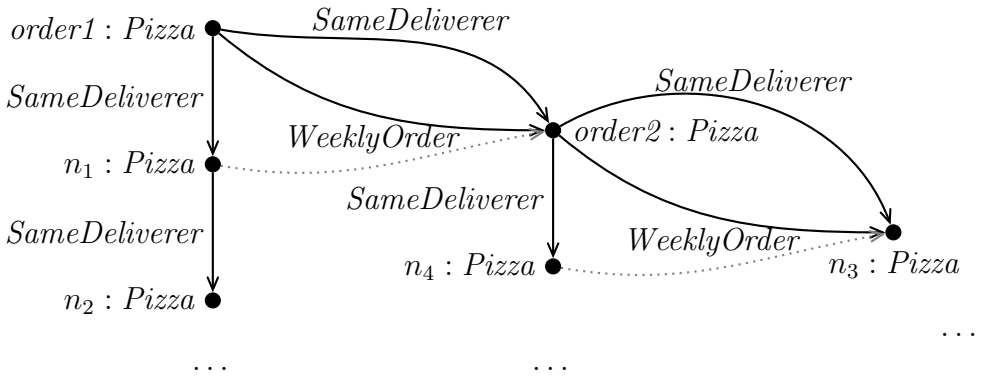


Figure 3.1: Alternative Matches for Chase in Example 2.13

Example 3.2 also shows that the infinite universal core model from Example 1.4 can be obtained by remapping the nulls according to the alternative matches. However, we may need to extend the alternative matches to endomorphisms over the whole fact set in the case of n_2. We go more into detail about this idea and its limitations in Chapter 5. Nevertheless, if a restricted

chase sequence does not have alternative matches, then it yields a universal core model without the need for any remappings.

Proposition 3.3 ([21, Theorem 2])**.** *If a res-chase sequence does not have an alternative match, then the res-chase result is a universal core model.*

The question of the existence of alternative matches in *res*-chase sequences gives rise to some decision problems similar to chase termination. We consider $\overline{\mathsf{AM}}_{is}$ with $i \in \{\forall, \exists\} \cup \{I \mid I \text{ is an instance}\}$ and $s \in \{\forall, \exists\}$. For example, a rule set R is in $\overline{\mathsf{AM}}_{I\exists}$ if some *res*-chase sequence for $\langle R, I\rangle$ does not have an alternative match and R is in $\overline{\mathsf{AM}}_{\forall\forall}$ if every *res*-chase sequence for every knowledge base featuring R does not have an alternative match. We observe that every rule set R is in $\overline{\mathsf{AM}}_{\exists\exists} = \overline{\mathsf{AM}}_{\exists\forall}$ since we can find an instance I with $I \not\models R$. Hence, I does not allow any restricted chase application so $F^{\infty}_{\langle R,I\rangle} = F^{0}_{\langle R,I\rangle} = I$ and thus no *res*-chase sequence has an alternative match. In practice, to compute finite universal core models using the restricted chase, we are mostly interested in rule sets in $\overline{\mathsf{AM}}_{\forall\exists}$. Although less general, we see in Chapter 4 that $\overline{\mathsf{AM}}_{\forall\forall}$ is an interesting class of rule sets as well.

We investigate decidability of these decision problems in the following mostly in this section. We only postpone $\overline{\mathsf{AM}}_{\forall\forall}$ to Section 3.2 since this becomes easier once we introduce further notions related to so-called core-stratification.

Rules can be decomposed in a way such that no alternative matches can occur [21]. However, the resulting rule set is not necessarily equivalent to the original one.

Definition 3.4. Consider a rule $\rho \coloneqq \mathit{Body}(\vec{x}, \vec{y}) \to \exists \vec{z}.\mathit{Head}(\vec{x}, \vec{z})$ as in Definition 2.1. The *existential decomposition* of ρ is the set of an existential and a Datalog rule $\{\, \mathit{Body}(\vec{x}, \vec{y}) \to \exists \vec{z}.P_\rho(\vec{x}, \vec{z}), P_\rho(\vec{x}, \vec{z}) \to \mathit{Head}(\vec{x}, \vec{z}) \,\}$ where P_ρ is a fresh predicate.

The *existential decomposition* of a rule set R is the union of the existential decompositions of all rules in R. For every instance, no *res*-chase sequence for such an existentially decomposed rule set can have an alternative match since the predicates occurring in existential rule heads never occur in any other rule head and the rule $\mathit{Body}(\vec{x}, \vec{y}) \to \exists \vec{z}.P_\rho(\vec{x}, \vec{z})$ cannot introduce alternative matches for triggers that feature this rule since at most one trigger can be applied for a given frontier mapping. In fact, the skolem chase and restricted

chase yield the same result for such rule sets when given an instance that does not feature any of the fresh predicates.

Utilizing existential decompositions, we can show the first decidability results. The existence of alternative matches for a given instance is undecidable [21, Theorem 4]. Even though this result has been established already, we give a more detailed proof, which was not presented before.

Proposition 3.5. *For all instances I, $\overline{\mathsf{AM}}_{I\forall}$ and $\overline{\mathsf{AM}}_{I\exists}$ are undecidable.*

Proof. Consider an instance I. We proof the undecidability of both $\overline{\mathsf{AM}}_{I\forall}$ and $\overline{\mathsf{AM}}_{I\exists}$ using a reduction from the complement of BCQ with the fixed instance I. We only show a single reduction that works for both decision problems.

Given a rule set R and a BCQ $\sigma \coloneqq \exists\vec{z}.\phi(\vec{z})$, let R' be the existential decomposition of R with a fresh rule $\rho \coloneqq \phi \to \exists z.A(z) \wedge A(c)$ where A is a fresh predicate, z is a fresh variable, and c is a fresh constant. If $\langle R, I, \sigma\rangle \in \mathsf{BCQ}$, then ρ enforces an alternative match for every *res*-chase sequence for $\langle R', I\rangle$. If $\langle R, I, \sigma\rangle \notin \mathsf{BCQ}$, then ρ is never applied in any *res*-chase sequence for $\langle R', I\rangle$ and since all other rules in R' are decomposed, no alternative matches occur in any *res*-chase sequence for $\langle R', I\rangle$. □

Using a similar proof, we show that $\overline{\mathsf{AM}}_{\forall\exists}$ is undecidable, which also is a new contribution.

Proposition 3.6. $\overline{\mathsf{AM}}_{\forall\exists}$ *is undecidable.*

Proof. We reduce BCQ to the complement of $\overline{\mathsf{AM}}_{\forall\exists}$. Given a rule set R, an instance I and a BCQ $\sigma \coloneqq \exists\vec{z}.\phi(\vec{z})$, we define R' to consist of the existential decomposition of R and additional rules as follows. We add the rule

$$\rho_I \coloneqq \ \to \exists z.I \wedge B(d) \wedge A(z)$$

that features a conjunction of all facts from I in its head as well as two atoms for fresh predicates A and B where d is a fresh constant and z is a fresh variable. We alter the body of every existing rule with empty body to be $B(d) \to \ldots$ to ensure that ρ_I is applied first on the empty instance. Also, we add the rule

$$\rho_{\mathsf{AM}} \coloneqq \phi \to A(c)$$

where c is a fresh constant. Now ρ_{AM} can enforce an alternative match if ρ_I has been applied. We also add auxiliary rules for each predicate P in R of the form $P(\vec{x}) \to I \wedge B(d)$ that ensure that I and $B(d)$ can be derived if any fact is already present without invoking ρ_I.

Now, we have that $R' \notin \overline{\mathsf{AM}}_{\forall\exists}$ iff $\langle R, I\rangle$ entails σ. We show both directions separately. Assume that $\langle R, I\rangle$ entails σ. For the empty instance, every *res*-chase sequence on $\langle R', \emptyset\rangle$ applies ρ_I first. Since $\langle R, I\rangle$ entails σ, ρ_{AM} becomes applicable at some point in every such *res*-chase sequence and introduces an alternative match for the first trigger in the sequence. Hence, $R' \notin \overline{\mathsf{AM}}_{\forall\exists}$.

For the other direction, we show the contrapositive. Assume that $\langle R, I\rangle$ does not entail σ. We show that $R' \in \overline{\mathsf{AM}}_{\forall\exists}$ i.e. for every instance I', we find a *res*-chase sequence for $\langle R', I'\rangle$ that does not have an alternative match. For every instance I', we can assume w.l.o.g. that instead of considering $\langle R', I'\rangle$, we can actually consider $\langle R', I' \cup I \cup \{\, B(d)\,\}\rangle$ since we can choose to apply one of the auxiliary rules first to derive the facts in I and $B(d)$. Now, $\langle R', I' \cup I \cup \{\, B(d)\,\}\rangle$ may entail σ or not. Note that for the special case of $I' = \emptyset$, we know that σ is not entailed. If σ is not entailed, then every *res*-chase sequence on $\langle R', I' \cup I \cup \{\, B(d)\,\}\rangle$ does not have an alternative match. If σ is entailed, then ρ_{AM} is applied in every *res*-chase sequence on $\langle R', I' \cup I \cup \{\, B(d)\,\}\rangle$ at some point. This necessarily happens after a finite amount of steps. However, the application of ρ_I can be delayed, since I and $B(d)$ have already been derived and the predicate A does not occur in any rule body, so it does not allow any new applications. Because ρ_{AM} is applied after a finite amount of steps, ρ_I becomes obsolete then. Hence, there exists a *res*-chase sequence on $\langle R', I' \cup I \cup \{\, B(d)\,\}\rangle$ that delays ρ_I long enough to become obsolete and thus, this chase sequence does not have an alternative match. So, whether σ is entailed or not, we find that $R' \in \overline{\mathsf{AM}}_{\forall\exists}$. □

Note that these results do not show undecidability for the corresponding decision problems of $\overline{\mathsf{AM}}_{I\exists}, \overline{\mathsf{AM}}_{I\forall}$, and $\overline{\mathsf{AM}}_{\forall\exists}$ for guarded rules since BCQ is known to be decidable in this case [3]. Thus, the existence of alternative matches may very well be decidable for guarded existential rule sets.

3.2 Core-Stratification

In this section, we recall a sufficient condition for the absence of alternative matches that leads to the main result of this chapter in Theorem 3.14 [21, Theorem 8]. Formally, for a rule set $R \in \mathsf{CT}^{res}_{\forall\forall}$, we present a sufficient condition for membership in $\overline{\mathsf{AM}}_{\forall\exists}$. In Chapter 4, we show that this condition is actually sufficient for membership in $\overline{\mathsf{AM}}_{\forall\exists}$ even for rule sets not in $\mathsf{CT}^{res}_{\forall\forall}$, which is also already hinted at in the work that introduces this condition [21].

Definition 3.7 (Restraining)**.** Consider two rules ρ_1 and ρ_2. The rule ρ_2 *restrains* ρ_1, written $\rho_2 \prec^{\square} \rho_1$, if there exist two fact sets F_1, F_2 and two triggers $\lambda_1 = \langle \rho_1, \theta_1 \rangle$ and $\lambda_2 = \langle \rho_2, \theta_2 \rangle$ that are *res*-applicable to F_1 and F_2, respectively, with $\lambda_1(F_1) \subseteq \lambda_2(F_2)$, such that (1) λ_1 has an alternative match w.r.t. $\lambda_2(F_2)$ and (2) λ_1 does not have an alternative match w.r.t. F_2.

Intuitively, an application of ρ_2 may introduce an alternative match for a trigger featuring ρ_1. Note that in particular, a rule can restrain itself.

Proposition 3.8 ([21, Lemma 6])**.** *Consider a res-chase sequence identified by its triggers $\lambda_1, \lambda_2, \ldots$. If this chase sequence has an alternative match, then there exist $0 \leq j \leq j'$ such that the rule in $\lambda_{j'}$ restrains the rule in λ_j.*

By the contrapositive, if we can ensure that rules are only applied in a way that respects restraining relations, then we prevent alternative matches, i.e. if $\rho_1 \prec^{\square} \rho_2$ then all applications of ρ_1 are done before any application of ρ_2. However, it may not always be possible to find such an application order. Additionally, we do not necessarily obtain a fair chase sequence by this strategy.

Before we go further into detail regarding these issues, we observe that the existence of restraining relations within a rule set decides $\overline{\mathsf{AM}}_{\forall\forall}$, which is a new contribution. This essentially follows directly from Definition 3.7.

Proposition 3.9. *Consider a rule set R. There exists rules $\rho_1, \rho_2 \in R$ with $\rho_2 \prec^{\square} \rho_1$ iff $R \notin \overline{\mathsf{AM}}_{\forall\forall}$.*

Proof. We show both directions of the claim separately. Assume that there exists an instance I such that some chase sequence for $\langle R, I \rangle$ has an alterna-

tive match. Then, there exists rules $\rho_1, \rho_2 \in R$ with $\rho_2 \prec^{\square} \rho_1$ by Proposition 3.8. The other direction essentially follows from the definition of $\prec^{\square}$ (Definition 3.7). Assume that there exist $\rho_1, \rho_2 \in R$ with $\rho_2 \prec^{\square} \rho_1$. Let $\lambda_1 = \langle \rho_1, \theta_1 \rangle$ and $\lambda_2 = \langle \rho_2, \theta_2 \rangle$ be triggers according to the definition of $\prec^{\square}$. Let F_1 and F_2 be the smallest fact sets that satisfy the condition of the definition of $\prec^{\square}$. We set $I \coloneqq F_2 \setminus F_{\lambda_1}$. We find a chase sequence for $\langle R, I \rangle$ that applies λ_1 followed by λ_2 (or only λ_2 in some special cases where $\rho_1 = \rho_2$), which immediately introduces an alternative match according to the definition of $\prec^{\square}$. □

To establish a sufficient condition for membership in $\overline{\mathsf{AM}}_{\forall\exists}$, we require some further notions. Our goal is to find a condition for when we are able to apply rules in an order that does not violate restraining relations. Then by Proposition 3.8, we know that no alternative matches are introduced in this sequence.

Definition 3.10 (Positive Reliance)**.** Consider two rules ρ_1 and ρ_2. The rule ρ_2 *positively relies* on ρ_1, written $\rho_1 \prec^{\triangle} \rho_2$, if there exists a fact set F and two triggers $\lambda_1 = \langle \rho_1, \theta_1 \rangle$ and $\lambda_2 = \langle \rho_2, \theta_2 \rangle$ such that λ_1 is *res*-applicable to F and λ_2 is *res*-applicable to $\lambda_1(F)$ but not *res*-applicable to F.

Intuitively, an application of ρ_1 may allow another application of ρ_2. We differ from the original notation [21] and use $\rho_1 \prec^{\triangle} \rho_2$ instead of $\rho_1 \prec^{+} \rho_2$ to prevent possible confusion with the transitive closure of relations.

Definition 3.11 (Downward Closure)**.** The *downward closure* of a rule ρ, written $\rho{\downarrow^{\square}}$, is the set of all rules ρ' with $\rho'((\prec^{\triangle})^{*} \circ \prec^{\square})^{+}\rho$ where $\circ$ denotes relation composition, $*$ denotes reflexive-transitive closure and $+$ denotes transitive closure.

Intuitively, we can construct the downward closure of ρ by initializing it with the rules that directly or indirectly restrain ρ and exhaustively adding all rules that directly or indirectly restrain a rule that is already contained in the downward closure.

Definition 3.12 (Core-Stratification)**.** A rule set R is *core-stratified* if $\rho \notin \rho{\downarrow^{\square}}$ for every rule $\rho \in R$.

Example 3.13. *The rule set from Example 1.1 is core-stratified. We have that* $\rho_3 \prec^{\square} \rho_1$*,* $\rho_2 \prec^{\triangle} \rho_3$*,* $\rho_3 \prec^{\triangle} \rho_1$ *and* $\rho_i \prec^{\triangle} \rho_i$ *for every* $1 \leq i \leq 3$*. By*

that, we find $\rho_1{\downarrow^{\square}} = \{\, \rho_2, \rho_3 \,\}$ *and* $\rho_2{\downarrow^{\square}} = \rho_3{\downarrow^{\square}} = \emptyset$.

Intuitively, if a rule set is core-stratified, we can find a rule application order that respects the restraining relations. Still, it is not straightforward to ensure that a resulting chase sequence is fair. We just sketch the proof in the following theorem since we show a more general version in Chapter 4.

Theorem 3.14 ([21, Theorem 8])**.** *Consider a knowledge base* $\mathcal{K} \coloneqq \langle R, I \rangle$*. If* R *is core-stratified and every res-chase sequence for* $\mathcal{K}$ *terminates, then there exists a res-chase sequence* $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$ *such that* $F^\infty_{\mathcal{K}}$ *is a finite universal core model.*

Proof Sketch. Since R is core-stratified, we can use the downward closures of the rules in R to obtain an order for the rules in R that respects the restraining relations. Since every chase sequence for $\mathcal{K}$ is finite, every rule in R is only applied a finite amount of times. Hence, there exists a (fair) *res*-chase sequence $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$ for $\mathcal{K}$ that applies rules in the given order exhaustively. The contrapositive of Proposition 3.8 yields that the *res*-chase sequence $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$ does not have an alternative match and with Proposition 3.3 the claim follows. □

The paper by Krötzsch [21] suggests that Theorem 3.14 can be generalized to also capture infinite *res*-chase sequences. We show a detailed proof of this in Chapter 4 as one of our main contributions.

For the practical relevance of core-stratification, the complexity of checking core-stratification for a rule set is also of interest.

Proposition 3.15 ([21, Theorem 7])**.** *Deciding if a rule set is core-stratified is in* Σ^{P}_2.

Proof Sketch. The complexity of checking core-stratification comes down to the complexity of checking restraining and positive reliance. We show that the complexity of the restraining check is Σ^{P}_2. The result for positive reliance is analogous.

All fact sets from Definition 3.7 can be guessed non-deterministically since the concrete names of the terms are not relevant and the size of each fact set is polynomial w.r.t. to the size of the rules if we restrict the fact sets to

predicates that occur in the rules. Similarly, substitutions for the triggers can be guessed non-deterministically. Together with the check whether they are *res*-applicable, this is possible in Σ_2^{P} (Proposition 2.11). The computation of the resulting fact sets from the trigger applications is then polynomial. Checking for the existence of alternative matches is possible in NP since we can guess an appropriate mapping and verify that it is a homomorphism and that it fulfills the remaining conditions of Definition 3.1 in polynomial time. In total, checking restraining for two given rules is in Σ_2^{P}.

□

In practice, the effort of the core-stratification check is reasonable compared to the checks for termination of the rule set; e.g. checking RMFA is 2ExpTime-complete [10].

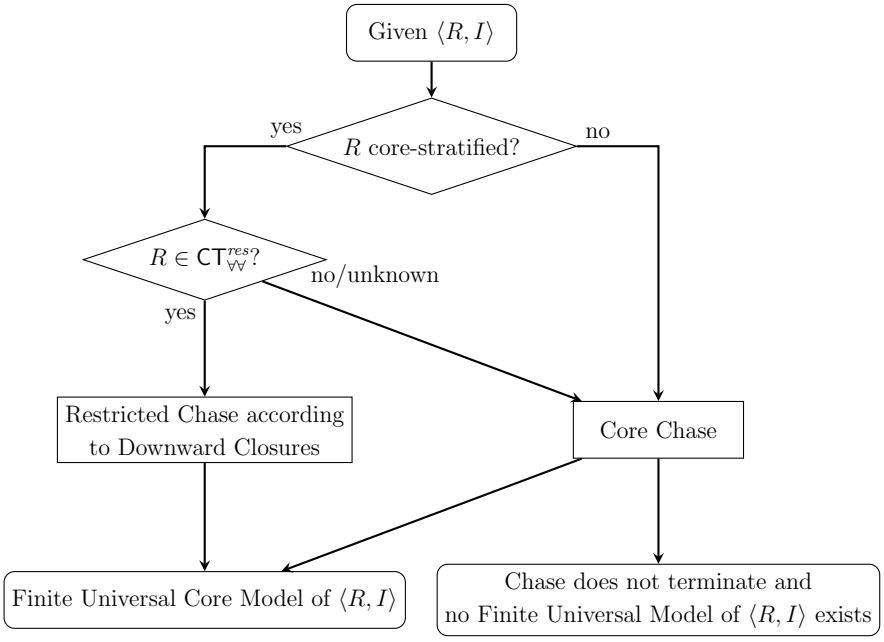


Figure 3.2: Basic Procedure for Core Computation

By Krötzsch [21], we obtain a procedure for computing universal core models of a given knowledge base in Figure 3.2. If a rule set is core-stratified and in $\mathsf{CT}^{res}_{\forall\forall}$, e.g. if it is RMFA, then we obtain a finite universal core model using the restricted chase in a way that applies rules w.r.t. the downward closures of the rules. Otherwise, we can still fall back to the core chase but in general

this is less efficient in practice and in this case we can also not be sure if the core chase terminates if the rule set is not in $\mathsf{CT}^{res}_{\forall\forall}$. To improve this, we continue to study the power of core-stratification to investigate in which cases we really have to fall back to the core chase in Chapter 4. Furthermore, in Chapter 5, we then also lay out possible improvements for the core chase for cases where we have to fall back on it.

Chapter 4

The Power of Core-Stratification

If we consider only core-stratified rule sets, we know from Chapter 3 that we can use the restricted chase to compute a finite universal core model of a knowledge base if the underlying rule set is in $\mathsf{CT}^{res}_{\forall\forall}$.

In this chapter, we strengthen this result by showing that core-stratification is a sufficient condition for membership in $\overline{\mathsf{AM}}_{\forall\exists}$ even for rule sets that are not in $\mathsf{CT}^{res}_{\forall\forall}$ in Theorem 4.1. We provide a formal proof using the notion of a *transfinite chase*, which we use as a framework for chase sequences to enforce a certain application order of rules. Additionally, we obtain that $\mathsf{CT}^{res}_{\forall\exists}$ and $\mathsf{CT}^{core}_{\forall}$ coincide for core-stratified rule sets in Theorem 4.13. For core-stratified rule sets, this also implies that there exists a terminating restricted chase sequence that applies rules according to their downward closures iff there exists a finite universal model.

We also investigate if we can find stronger results for (single-head) guarded existential rule sets. We show in particular that membership in $\mathsf{CT}^{core}_{\forall}$ is decidable for single-head guarded existential rule sets without restraining relations in Corollary 4.17. This result makes use of the fact that $\mathsf{CT}^{res}_{\forall\forall}$ is decidable for single-head guarded existential rules [16]. Using a notion of *strong restraining*, we show that the requirement for single-heads can be relaxed at least for the so-called Fairness Theorem, which is an essential part of the

proof for decidability of $\mathsf{CT}^{res}_{\forall\forall}$ [16]. We conjecture that the rest of the proof for decidability of $\mathsf{CT}^{res}_{\forall\forall}$ can be adjusted to support arbitrary guarded existential rule sets, which implies that $\mathsf{CT}^{res}_{\forall\forall}$ is decidable for guarded existential rule sets without strong restraining relations (Conjecture 4.28), This also subsumes all single-head guarded existential rule sets and immediately yields that $\mathsf{CT}^{core}_{\forall}$ is decidable for arbitrary guarded existential rule sets without restraining relations.

Furthermore, we briefly investigate if we can even use the skolem chase to compute universal core models for core-stratified rule sets. One may think that this is possible similar to the restricted chase but it turns out that we can always find an instance for which a similar approach cannot work. We show this in Theorem 4.29.

The results of this chapter enable us to better decide when to use the restricted chase with respect to restraining relations to compute cores and when we may obtain a finite universal (core) model using the core chase as a fallback. In Chapter 5, we investigate if we can improve this fallback variant to be more efficient in practice.

4.1 Core-Stratification and Chase Termination

As suggested in Chapter 3, we aim to generalize Theorem 3.14. As main results of this section, we prove that core-stratification is a sufficient condition for $\overline{\mathsf{AM}}_{\forall\exists}$ and we show that $\mathsf{CT}^{res}_{\forall\exists}$ coincides with $\mathsf{CT}^{core}_{\forall}$ for rule sets in $\overline{\mathsf{AM}}_{\forall\exists}$. Even though $\mathsf{CT}^{res}_{\forall\exists}$ is undecidable and to the best of our knowledge not known to be decidable even for guarded existential rules, there still exist sufficient conditions for membership and non-membership of rule sets in $\mathsf{CT}^{res}_{\forall\exists}$. In particular, we show that RMFC is a sufficient condition for non-membership in $\mathsf{CT}^{core}_{\forall}$ and hence also a sufficient condition for the non-existence of finite universal models for rule sets in $\overline{\mathsf{AM}}_{\forall\exists}$. We also show that for core-stratified rule sets, there exists a terminating restricted chase sequence that prioritizes rules according to their downward closures iff there exists a finite universal model. We also introduce the *transfinite chase* to simplify the formal details of prioritization of certain rules.

4.1.1 Core-Stratification is sufficient for $\overline{\mathsf{AM}}_{\forall\exists}$

As the main theorem of this subsection, we show the following.

Theorem 4.1. *If a rule set R is core-stratified, then $R \in \overline{\mathsf{AM}}_{\forall\exists}$.*

For the proof of Theorem 4.1, we follow the intuition that we can find a chase sequence that respects the restraining relations for core-stratified rule sets. Then, we can immediately apply the contrapositive of Proposition 3.8 to obtain the result. The main problem that we have is that such a chase sequence is not necessarily fair. We introduce some notions in the following that help us to tackle this problem. In particular, we formalize what it means to apply rules according to restraining relations and we introduce the *transfinite chase* as a framework for chase sequences that prefer certain rules over others.

Definition 4.2 (Restrained Partitioning)**.** Consider a rule set R. A *restrained partitioning* of R is a list of sets $R_1, \ldots, R_n$ that form a partitioning of R (i.e. R is the disjoint union $R_1 \dot\cup \ldots \dot\cup R_n$) such that for every $1 \leq i \leq n$ and rule ρ we have that if $\rho \in R_i$ then $(\bigcup_{i \leq j \leq n} R_j) \cap \rho{\downarrow^{\square}} = \emptyset$.

The intuition for restrained partitionings is that we can apply rules in a way that respects restraining relations by applying only rules from R_1, then R_2 and so on. Note however that rules may positively rely on rules from "later" rule sets in the partitioning. We show in Lemma 4.6 that this does not lead to violations regarding the restraining relations. We find that every core-stratified rule set has a restrained partitioning.

Lemma 4.3. *Consider a rule set R. If R is core-stratified, then there exists a restrained partitioning of R.*

Proof. We construct a restrained partitioning for R as a list of sets $R_1, \ldots, R_n$ inductively:

- $R_1 \coloneqq \{\, \rho \in R \mid \rho{\downarrow^{\square}} = \emptyset \,\}$
- $R_{i+1} \coloneqq \{\, \rho \in R \setminus \bigcup_{1 \leq j \leq i} R_j \mid \rho{\downarrow^{\square}} \subseteq \bigcup_{1 \leq j \leq i} R_j \,\}$

We set n to be the smallest number such that $R_{n+1} = \emptyset$. Since there is only a finite number of rules in R, such an n necessarily exists. By construction, the sets $R_1, \ldots, R_n$ are pairwise disjoint.

We show for every $1 \leq i \leq n$ and every rule $\rho \in R_i$ that $(\bigcup_{i \leq j \leq n} R_j) \cap \rho\downarrow^{\square} = \emptyset$. If $\rho \in R_i$, then we have that $\rho\downarrow^{\square} \subseteq \bigcup_{1 \leq k \leq i-1} R_k$ by construction. Because $R_1, \ldots, R_n$ are pairwise disjoint, we have $(\bigcup_{i \leq j \leq n} R_j) \cap (\bigcup_{1 \leq k \leq i-1} R_k) = \emptyset$ and thus $(\bigcup_{i \leq j \leq n} R_j) \cap \rho\downarrow^{\square} = \emptyset$.

We still need to show that every rule in R occurs in $\bigcup_{1 \leq i \leq n} R_i$. Suppose for a contradiction that there exists a rule ρ_1 in R with $\rho_1 \notin \bigcup_{1 \leq i \leq n} R_i$. In particular, $\rho_1\downarrow^{\square} \not\subseteq \bigcup_{1 \leq i \leq n} R_i$. Then, there exists a rule $\rho_2 \in \rho_1\downarrow^{\square}$ with $\rho_2 \notin \bigcup_{1 \leq i \leq n} R_i$ which in turn implies that $\rho_2\downarrow^{\square} \not\subseteq \bigcup_{1 \leq i \leq n} R_i$. By this argument, we find an infinite list of rules $\rho_1, \rho_2, \ldots$ with $\rho_k \notin \bigcup_{1 \leq i \leq n} R_i$ and $\rho_j \in \rho_k\downarrow^{\square}$ for all $1 \leq k < j$. Since R is finite, at least one rule ρ occurs multiple times in this list. But then, $\rho \in \rho\downarrow^{\square}$, which contradicts the core-stratification of R.

□

Example 4.4. *Consider the rule set from Example 1.1. Recall from Example 3.13 that the downward closures of the rules are* $\rho_1\downarrow^{\square} = \{ \rho_2, \rho_3 \}$ *and* $\rho_2\downarrow^{\square} = \rho_3\downarrow^{\square} = \emptyset$*. A restrained partitioning of this rule set according to the construction in the proof of Lemma 4.3 is the list of rule sets* R_1, R_2 *with* $R_1 = \{ \rho_2, \rho_3 \}$ *and* $R_2 = \{ \rho_1 \}$*.*

We formalize the computation of the chase over a list of rule sets using the transfinite chase.

Definition 4.5 (Transfinite Chase Sequence)**.** Consider a list of rule sets $\mathfrak{R} = \langle R_1, \ldots, R_n \rangle$ and an instance I. We define a *transfinite chase sequence* inductively via a sequence of fact sets $\mathfrak{T}^0_{\langle \mathfrak{R}, I \rangle}, \ldots, \mathfrak{T}^n_{\langle \mathfrak{R}, I \rangle}$ such that

- $\mathfrak{T}^0_{\langle \mathfrak{R}, I \rangle} = I$ and
- $\mathfrak{T}^{i+1}_{\langle \mathfrak{R}, I \rangle} = T_i^\infty$ for all $0 \leq i < n$ where T_i^∞ is the result of a *res*-chase sequence $T_i^1, T_i^2, \ldots$ for the pair $\langle \mathfrak{R}^{\leq i+1}, \mathfrak{T}^i_{\langle \mathfrak{R}, I \rangle} \rangle$

where $\mathfrak{R}^{\leq m} \coloneqq \bigcup_{1 \leq j \leq m} R_j$ for all $m > 0$.

Outside of this definition, we also use the notation T_k^j in the context of a transfinite chase sequence to denote the fact set in step j of the $k+1$-th *res*-chase sequence within the transfinite chase sequence. Note that within the definition, the pair $\langle \mathfrak{R}^{\leq i+1}, \mathfrak{T}^i_{\langle \mathfrak{R}, I \rangle} \rangle$ may not formally be a knowledge base since $\mathfrak{T}^i_{\langle \mathfrak{R}, I \rangle}$ can be an infinite fact set with nulls, which is formally not an

instance. Still, as noted in Remark 2.18, the definitions of *res*-chase sequences also work for infinite starting fact sets. The *transfinite chase result* for $\mathfrak{R}$ and I is defined as $\mathfrak{T}^{\infty}_{\langle \mathfrak{R}, I\rangle} \coloneqq \mathfrak{T}^{n}_{\langle \mathfrak{R}, I\rangle}$. Essentially, a transfinite chase sequence is just a finite sequence of (possibly infinite) *res*-chase sequences. For a transfinite chase sequence, we can try to obtain a corresponding *res*-chase sequence for $\langle \mathfrak{R}^{\leq n}, I\rangle$ by concatenating the individual *res*-chase sequences. However, this is only possible up until the first infinite sequence within the transfinite chase sequence. Hence, the resulting *res*-chase sequence may be unfair since e.g. for one particular k there may be infinitely many chase steps necessary to obtain T^{∞}_{k} but then the triggers for all rules that occur in some rule set R_ℓ with $\ell > k$ are not applied after finitely many steps. Note that we can mix *res*-chase sequences with e.g. *core*-chase sequences to a certain extent in the transfinite chase sequence. We make use of this in Chapter 5. For this chapter however, we only consider transfinite chase sequences with *res*-chase sequences.

It may not be obvious that we can use the transfinite chase to apply rules in a way that respects the restraining relations even for restrained partitionings since the transfinite chase steps make use of unions over rule sets within the partitioning, e.g. $\mathfrak{R}^{\leq m}$, and rules may positively rely on rules from "later" rule sets in the partitioning. We show that this indeed holds.

Lemma 4.6. *Consider a restrained partitioning $\mathfrak{R} = \langle R_1, \ldots, R_n\rangle$ and an instance I. If, for two rules ρ and ρ', ρ is applied in $\mathfrak{T}^{k}_{\langle \mathfrak{R}, I\rangle}$ and ρ' is applied in $\mathfrak{T}^{k'}_{\langle \mathfrak{R}, I\rangle}$ with $k \leq k'$, then $\rho' \not\prec^{\square} \rho$.*

Proof. Assume for a contradiction that ρ is applied in $\mathfrak{T}^{k}_{\langle \mathfrak{R}, I\rangle}$ and ρ' is applied in $\mathfrak{T}^{k'}_{\langle \mathfrak{R}, I\rangle}$ with $k \leq k'$ but $\rho' \prec^{\square} \rho$.

Since $\rho' \prec^{\square} \rho$, there exist j, j' with $j' < j \leq k$ such that $\rho' \in R_{j'}$ and $\rho \in R_j$ because $\mathfrak{R}$ is a restrained partitioning. Since rules of each rule set in the partitioning are applied exhaustively, there is a rule $\rho'' \in \bigcup_{j \leq j'' \leq k'} R_{j''}$ that is applied in $\mathfrak{T}^{k'}_{\langle \mathfrak{R}, I\rangle}$ with $\rho''(\prec^{\triangle})^{+}\rho'$. But then $\rho'' \in \rho{\downarrow^{\square}}$ which contradicts the assumption that $\mathfrak{R}$ is a restrained partitioning. □

We prove that the transfinite chase on a restrained partitioning does not have alternative matches using this result. At first, we need to extend the

definition of alternative matches onto transfinite chase sequences.

Definition 4.7. Consider a list of rule sets $\mathfrak{R} = \langle R_1, \ldots, R_n \rangle$ and an instance I. For the transfinite chase sequence $\mathfrak{T}^0_{\langle \mathfrak{R}, I \rangle}, \ldots, \mathfrak{T}^n_{\langle \mathfrak{R}, I \rangle}$, an *alternative match* at step $\langle k, \ell \rangle$ is an alternative match for the trigger that is applied to T^ℓ_k w.r.t. $\mathfrak{T}^\infty_{\langle \mathfrak{R}, I \rangle}$.

The following lemma holds similarly to Proposition 3.8.

Lemma 4.8. *Consider a restrained partitioning $\mathfrak{R} = \langle R_1, \ldots, R_n \rangle$ and an instance I. The transfinite chase sequence $\mathfrak{T}^0_{\langle \mathfrak{R}, I \rangle}, \ldots, \mathfrak{T}^n_{\langle \mathfrak{R}, I \rangle}$ does not have an alternative match.*

Proof. Suppose for a contradiction that there exists a transfinite chase sequence $\mathfrak{T}^0_{\langle \mathfrak{R}, I \rangle}, \ldots, \mathfrak{T}^n_{\langle \mathfrak{R}, I \rangle}$ that has an alternative match at step $\langle k, \ell \rangle$ for the trigger $\lambda = \langle \rho, \theta \rangle$. Assume w.l.o.g. that this is the first such step. Let $k' \geq k, \ell' [> \ell \text{ if } k' = k]$ be the smallest numbers such that λ has an alternative match w.r.t. $T^{\ell'}_{k'}$. Let λ' be the trigger that yields $T^{\ell'}_{k'}$ and let ρ' be the rule in λ'.

We find that $\rho' \prec^\square \rho$ as follows. We set the rules and fact sets of Definition 3.7 such that $\rho_1 = \rho$, $F_1 = T^\ell_k$, $\rho_2 = \rho'$, and $\lambda'(F_2) = T^{\ell'}_{k'}$. We have that $\lambda(F_1) \subseteq \lambda'(F_2)$ and λ has an alternative match w.r.t. $\lambda'(F_2) = T^{\ell'}_{k'}$. Since k' and ℓ' are minimal, λ does not have an alternative match w.r.t. F_2.

Hence, we obtain the desired contradiction because $\rho' \prec^\square \rho$ contradicts Lemma 4.6. Therefore, the transfinite chase sequence does not have an alternative match. □

To conclude the proof of Theorem 4.1, we essentially need to show that we can transform a transfinite chase sequence on a restrained partitioning into a (fair) *res*-chase sequence. We show an essential auxiliary result for reordering triggers at first that we also reuse later on in this chapter.

Lemma 4.9. *Consider two fact sets F_1 and F_2 and two triggers λ_1 and λ_2 such that λ_1 is res-applicable to F_1, λ_2 is res-applicable to F_1 and F_2, and $\lambda_1(F_1) \subseteq F_2$. If λ_1 does not have an alternative match w.r.t. $\lambda_2(F_2)$, then λ_1 is not obsolete w.r.t. $\lambda_2(F_1)$.*

Proof. We show the contrapositive of the claim. Assume that $\lambda_1 = \langle \rho_1, \theta_1 \rangle$ is obsolete w.r.t. $\lambda_2(F_1)$. By definition, there exists a substitution θ_1' that agrees with θ_1 in all variables in $\mathit{frontier}(\rho_1)$ such that $\theta_1'(\mathit{head}(\rho_1)) \subseteq \lambda_2(F_1)$. The substitution θ_1' induces an alternative match for λ_1 w.r.t. $\lambda_2(F_2)$ that maps each null n_z, which λ_1 introduces for the variable z, to $\theta_1'(z)$. □

Note that λ_1 does not have alternative matches w.r.t. $\lambda_2(F_2)$ if the rule of λ_2 does not restrain the rule of λ_1.

The goal of Lemma 4.9 is to show that we can safely shift a trigger λ further to the beginning of a *res*-chase sequence as long as λ does not introduce alternative matches for any trigger that it is shifted before. Safely means here that all triggers in the sequence are still *res*-applicable, i.e. no trigger is made obsolete by λ after shifting. This idea is crucial for transforming a transfinite chase sequence into a fair *res*-chase sequence. We show this transformation in detail in the proof of the following lemma.

Lemma 4.10. *Consider a restrained partitioning $\mathfrak{R} = \langle R_1, \ldots, R_n \rangle$ of a core-stratified rule set R, an instance I and a transfinite chase sequence $\mathfrak{T}^0_{\langle \mathfrak{R}, I \rangle}, \ldots, \mathfrak{T}^n_{\langle \mathfrak{R}, I \rangle}$. There exists a (fair) res-chase sequence $F^0_{\langle R, I \rangle}, F^1_{\langle R, I \rangle}, \ldots$ such that $F^\infty_{\langle R, I \rangle} = \mathfrak{T}^\infty_{\langle \mathfrak{R}, I \rangle}$.*

For better readability in the proof, we use the following notation. For a *res*-chase sequence $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$ of a knowledge base $\mathcal{K}$, we define $\mathcal{K}^i \coloneqq F^i_{\mathcal{K}}$ and $\mathcal{K}^\infty \coloneqq F^\infty_{\mathcal{K}}$. We do not use this notation anywhere else beside this proof.

Proof. We show $\langle \mathfrak{R}^{\leq i}, I \rangle^\infty = \mathfrak{T}^i_{\langle \mathfrak{R}, I \rangle}$ via induction over $1 \leq i \leq n$. Note that $R = \mathfrak{R}^{\leq n}$ and $\mathfrak{T}^\infty_{\langle \mathfrak{R}, I \rangle} = \mathfrak{T}^n_{\langle \mathfrak{R}, I \rangle}$. The base of the induction for $i = 1$ holds since $\langle \mathfrak{R}^{\leq 1}, I \rangle^\infty = T^\infty_0 = \mathfrak{T}^1_{\langle \mathfrak{R}, I \rangle}$ follows from Definition 4.5. Thus, the fair *res*-chase sequence is the sequence $T^0_0, T^1_0, \ldots$. For the induction step for $i = k + 1$, we assume that the claim holds for $i = k$, i.e. we have a *res*-chase sequence $\langle \mathfrak{R}^{\leq k}, I \rangle^0, \langle \mathfrak{R}^{\leq k}, I \rangle^1, \ldots$ with $\langle \mathfrak{R}^{\leq k}, I \rangle^\infty = \mathfrak{T}^k_{\langle \mathfrak{R}, I \rangle}$. We show that we have a sequence $\langle \mathfrak{R}^{\leq k+1}, I \rangle^0, \langle \mathfrak{R}^{\leq k+1}, I \rangle^1, \ldots$ with $\langle \mathfrak{R}^{\leq k+1}, I \rangle^\infty = \mathfrak{T}^{k+1}_{\langle \mathfrak{R}, I \rangle} = T^\infty_k$. If $\mathfrak{T}^k_{\langle \mathfrak{R}, I \rangle}$ is finite, then there exists $m \geq 0$ with $\langle \mathfrak{R}^{\leq k}, I \rangle^m = \mathfrak{T}^k_{\langle \mathfrak{R}, I \rangle}$. Since $\mathfrak{R}^{\leq k} \subseteq \mathfrak{R}^{\leq k+1}$ we can set $\langle \mathfrak{R}^{\leq k+1}, I \rangle^j \coloneqq \langle \mathfrak{R}^{\leq k}, I \rangle^j$ for all $0 \leq j \leq m$ and $\langle \mathfrak{R}^{\leq k+1}, I \rangle^j \coloneqq T^{j-m}_k$ for $j > m$, i.e. we concatenate the sequences. Hence, the claim follows.

If $\mathfrak{T}^k_{\langle \mathfrak{R}, I\rangle}$ is infinite, the fact set $\mathfrak{T}^k_{\langle \mathfrak{R}, I\rangle}$ results from an infinite *res*-chase sequence $\langle \mathfrak{R}^{\leq k}, I\rangle^0, \langle \mathfrak{R}^{\leq k}, I\rangle^1, \ldots$. We also have a (possibly infinite) *res*-chase sequence $T^0_k, T^1_k, \ldots$ that yields $\mathfrak{T}^{k+1}_{\langle \mathfrak{R}, I\rangle}$. We construct a *res*-chase sequence $\langle \mathfrak{R}^{\leq k+1}, I\rangle^0, \langle \mathfrak{R}^{\leq k+1}, I\rangle^1, \ldots$ in the following by interleaving both chase sequences and shifting triggers from the second sequence into the first one. We identify the three chase sequences by the used triggers, i.e. $\Lambda_k = \langle \lambda^1_k, \lambda^2_k, \ldots\rangle$ for the sequence that yields $\mathfrak{T}^k_{\langle \mathfrak{R}, I\rangle}$, $\Lambda_{k+1} = \langle \lambda^1_{k+1}, \lambda^2_{k+1}, \ldots\rangle$ for the sequence that yields $\mathfrak{T}^{k+1}_{\langle \mathfrak{R}, I\rangle}$, and the newly constructed sequence $\Lambda_\dagger = \langle \lambda^1_\dagger, \lambda^2_\dagger, \ldots\rangle$, respectively.

We construct $\Lambda_\dagger$ as follows. For each $\ell > 0$:

$$\lambda^\ell_\dagger = \begin{cases} \lambda^j_{k+1}, & \text{if } \lambda^{\ell-1}_\dagger \in \Lambda_k \\ & \text{and } \lambda^j_{k+1} \text{ is } \textit{res}\text{-applicable to } \langle \mathfrak{R}^{\leq k+1}, I\rangle^{\ell-1} \text{ ;} \\ & \text{for the smallest } j > 0 \text{ with } \lambda^j_{k+1} \notin \Lambda^{\leq \ell-1}_\dagger \\ \lambda^j_k, & \text{otherwise; for the smallest } j > 0 \text{ with } \lambda^j_k \notin \Lambda^{\leq \ell-1}_\dagger \end{cases}$$

We write $\Lambda^{\leq \ell}_\dagger$ to denote the sequence $\Lambda_\dagger$ up until $\lambda^\ell_\dagger$. Intuitively, we pick the next unused application from Λ_{k+1} whenever possible. Otherwise we continue using Λ_k. Note that the first case also ensures that we do not only use triggers from Λ_{k+1} by requiring $\lambda^{\ell-1}_\dagger \in \Lambda_k$.

We show three properties for $\Lambda_\dagger$ that conclude the proof.

1. Each $\lambda^i_\dagger$ is *res*-applicable within $\Lambda_\dagger$.
2. The sequence $\Lambda_\dagger$ has the same result as Λ_{k+1} after Λ_k.
3. The sequence $\Lambda_\dagger$ is fair.

First, we show that each $\lambda^i_\dagger$ is indeed *res*-applicable within the new sequence. In particular we only need to show this for the $\lambda^i_\dagger \in \Lambda_k$ since if $\lambda^i_\dagger \in \Lambda_{k+1}$ this is ensured by the definition. For $\lambda^i_\dagger \in \Lambda_k$, we find a $j \leq i$ with $\lambda^i_\dagger = \lambda^j_k$. By definition, we find $\Lambda^{\leq j}_k \subseteq \Lambda^{\leq i}_\dagger$. Suppose now for a contradiction that $\lambda^i_\dagger$ cannot be applied in $\Lambda_\dagger$ and assume w.l.o.g. that $\lambda^i_\dagger$ is the first such trigger. We know that $\lambda^i_\dagger$ is active in $\Lambda_\dagger$ because λ^j_k is active in Λ_k. Therefore $\lambda^i_\dagger$ must be obsolete in $\Lambda_\dagger$ to not be *res*-applicable. Hence, there is at least one $\lambda^\ell_{k+1} \in \Lambda^{\leq i}_\dagger$ that leads to the obsoleteness of $\lambda^i_\dagger$. But then, by the contrapositive of Lemma

4.9, $\lambda^i_\dagger$ has an alternative match w.r.t. T^ℓ_k and by Definition 3.7 the rule in λ^ℓ_{k+1} restrains the rule in $\lambda^i_\dagger = \lambda^j_k$. This contradicts Lemma 4.6.

Second, we show that the result of $\Lambda_\dagger$ is indeed the same as applying Λ_{k+1} after Λ_k by showing that $\Lambda_k \cup \Lambda_{k+1} = \Lambda_\dagger$. The direction $\Lambda_k \cup \Lambda_{k+1} \supseteq \Lambda_\dagger$ follows from the construction. By the first case of the definition of $\Lambda_\dagger$, we obtain that $\Lambda_k \subseteq \Lambda_\dagger$. Hence, we only need to show $\Lambda_{k+1} \subseteq \Lambda_\dagger$.

(‡) We show via induction over $\ell \geq 1$ for each $\lambda^\ell_{k+1} \in \Lambda_{k+1}$ that there exists a $j \geq 0$ such that λ^ℓ_{k+1} is a *res*-applicable trigger after applying $\Lambda^{\leq j}_\dagger$. For the base of the induction with $\ell = 1$, there are only finitely many facts required for λ^1_{k+1} to be active. Hence, there exists a j such that λ^1_{k+1} is active after $\Lambda^{\leq j}_k = \Lambda^{\leq j}_\dagger$. Also, λ^1_{k+1} is not obsolete since otherwise it is also obsolete after Λ_k which is a contradiction since then $\lambda^1_{k+1} \notin \Lambda_{k+1}$. For the induction step with $\ell = n$, we assume that the claim holds for all $n' < n$. There are only finitely many facts required for λ^n_{k+1} to be active. We can choose a j such that all $\lambda^{n'}_{k+1} \in \Lambda^{\leq j}_\dagger$ by the induction hypothesis and all triggers from Λ_k that are required for λ^n_{k+1} to be active are also in $\Lambda^{\leq j}_\dagger$ analogously to the base case. Also, λ^n_{k+1} is not obsolete since otherwise it is also obsolete after Λ_k and $\Lambda^{\leq n-1}_{k+1}$.

Third, we show that $\Lambda_\dagger$ is fair. From the second part, we directly obtain that each trigger in $\Lambda_k \cup \Lambda_{k+1}$ is applied after finitely many steps in $\Lambda_\dagger$. This follows from the fact, that $\Lambda_k \cup \Lambda_{k+1} = \Lambda_\dagger$ and we only define elements in $\Lambda_\dagger$ for finite indices. Still, we need to show that every trigger that is not in $\Lambda_k \cup \Lambda_{k+1}$ but *res*-applicable to $\langle \mathfrak{R}^{\leq k+1}, I\rangle^m$ for some m is obsolete w.r.t. $\langle \mathfrak{R}^{\leq k+1}, I\rangle^{m'}$ for some $m' > m$. Assume that $\lambda^\star \notin \Lambda_k \cup \Lambda_{k+1}$ is a trigger that is *res*-applicable to $\langle \mathfrak{R}^{\leq k+1}, I\rangle^m$ for some m. Then, $\lambda^\star$ is active w.r.t. $\langle \mathfrak{R}^{\leq k}, I\rangle^j$ for some j or T^ℓ_k for some ℓ. In either case and since $\lambda^\star \notin \Lambda_k \cup \Lambda_{k+1}$, $\lambda^\star$ is obsolete for all chase steps above and including $\lambda^{j'}_k$ with some $j' \geq j$ or $\lambda^{\ell'}_{k+1}$ with some $\ell' \geq \ell$, respectively. Otherwise, Λ_k or Λ_{k+1} is not a fair chase sequence. In either case, there is only a finite amount of facts necessary to make $\lambda^\star$ obsolete. But then there is an $m' > m$ such that $\lambda^\star$ is obsolete for $\langle \mathfrak{R}^{\leq k+1}, I\rangle^{m'}$ (by a similar argument as for (‡)).

In summary, $\Lambda_\dagger$ describes a (fair) chase sequence $\langle \mathfrak{R}^{\leq k+1}, I\rangle^0, \langle \mathfrak{R}^{\leq k+1}, I\rangle^1, \ldots$ with $\langle \mathfrak{R}^{\leq k+1}, I\rangle^\infty = \mathfrak{T}^{k+1}_{\langle \mathfrak{R}, I\rangle} = T^\infty_k$. This concludes the induction step and the proof. □

Note that we require Lemma 4.6 and hence a restrained partitioning to show that each trigger $\lambda_\dagger^i$ is *res*-applicable within $\Lambda_\dagger$. Lemma 4.10 does indeed not hold if we do not enforce a restrained partitioning.

Example 4.11. *We use the following rule set R from an example by Gogacz et al. [16]:*

$$\rho_1 \coloneqq S(x, y, y) \to \exists z.S(x, z, y) \land S(z, y, y)$$
$$\rho_2 \coloneqq S(x, y, z) \to S(z, z, z)$$

The rule set R has the following restraining and positive reliance relations $\rho_2 \prec^\square \rho_1$, $\rho_1 \prec^\triangle \rho_2$, and $\rho_1 \prec^\triangle \rho_1$. Considering the instance $\{ S(a, b, b) \}$ as in the paper by Gogacz et al. [16], R admits only finite fair res-chase sequences since, intuitively speaking, a single application of ρ_2 blocks all further applications of ρ_1. Still, R allows for an infinite transfinite chase sequence if we pick $\{ \rho_1 \}, \{ \rho_2 \}$ as a partitioning. This partitioning is not a restrained partitioning since $\rho_2 \prec^\square \rho_1$.

An interesting observation for this example is that the relation $\rho_1 \prec^\triangle \rho_2$ breaks the core-stratification of R. Technically $\rho_1 \prec^\triangle \rho_2$ is correct since ρ_1 introduces a new trigger for ρ_2. Still, a closer look reveals that ρ_2 is already applicable once ρ_1 becomes applicable so the results of the triggers for ρ_2 are the same. Thus, the transfinite chase on $\{ \rho_2 \}, \{ \rho_1 \}$ always yields a finite universal core model even though $\{ \rho_2 \}, \{ \rho_1 \}$ is technically not a restrained partitioning.

Theorem 4.1 follows by combining the previous results as follows.

Proof of Theorem 4.1. Since R is core-stratified, there exists a restrained partitioning $\mathfrak{R}$ for R by Lemma 4.3. Every transfinite chase sequence for $\mathfrak{R}$ does not have an alternative match by Lemma 4.8. Additionally, for every transfinite chase sequence for $\mathfrak{R}$, there exists an equivalent (fair) *res*-chase sequence by Lemma 4.10. From Definition 3.1, we obtain that this (fair) *res*-chase sequence also does not have an alternative match, since it uses the same triggers as the transfinite chase sequence and has the same result. Thus, we have $R \in \overline{\mathsf{AM}}_{\forall\exists}$. □

In fact, for a core-stratified rule set R and every instance, we can use any transfinite chase sequence for a restrained partitioning of R directly to obtain an (unfair) *res*-chase sequence without alternative matches. In practice, we do not need this sequence to be fair since we know that a fair *res*-chase sequence with the same result exists.

Example 4.12. *Consider the restrained partitioning* $R_1 = \{ \rho_2, \rho_3 \}$ *and* $R_2 = \{ \rho_1 \}$ *from Example 4.4 for the rule set from Example 1.1. For the instance* $\{ \mathit{Pizza}(\mathit{order1}), \mathit{WeeklyOrder}(\mathit{order1}, \mathit{order2}) \}$*, the transfinite chase applies rules* ρ_2 *and* ρ_3 *exhaustively in the first transfinite chase step and thus builds up an infinite "chain" of weekly orders with same deliverer relations as it occurs in the infinite universal core model from Example 1.4. Then for the second transfinite chase step, which applies* ρ_1 *exhaustively, all active triggers are already obsolete. Hence, the result of the transfinite chase sequence is exactly the infinite universal core model from Example 1.4. In contrast to the core chase in Example 2.15, the result of this transfinite chase sequence is well defined since it only uses res-chase sequences.*

4.1.2 Restricted and Core Chase Termination coincide for $\overline{\mathsf{AM}}_{\forall\exists}$

We show for rule sets in $\overline{\mathsf{AM}}_{\forall\exists}$, e.g. core-stratified rule sets, that restricted and core chase termination coincide. In particular, for a rule set $R \in \overline{\mathsf{AM}}_{\forall\exists}$, we have that $R \in \mathsf{CT}^{res}_{\forall\exists}$ iff $R \in \mathsf{CT}^{core}_{\forall}$. However, we do not get a similar result for $\mathsf{CT}^{res}_{\forall\forall}$. If $R \in \mathsf{CT}^{res}_{\forall\forall}$, then $R \in \mathsf{CT}^{core}_{\forall}$ holds but if $R \in \overline{\mathsf{AM}}_{\forall\exists}$ and $R \notin \mathsf{CT}^{res}_{\forall\forall}$, $R \in \mathsf{CT}^{core}_{\forall}$ could still hold. We show the main theorem for this subsection in the following.

Theorem 4.13. *For a rule set* $R \in \overline{\mathsf{AM}}_{\forall\exists}$*, we have* $R \in \mathsf{CT}^{res}_{\forall\exists}$ *iff* $R \in \mathsf{CT}^{core}_{\forall}$*.*

Proof. Let $R \in \overline{\mathsf{AM}}_{\forall\exists}$. We show both directions of the claim separately. If $R \in \mathsf{CT}^{res}_{\forall\exists}$, then $R \in \mathsf{CT}^{core}_{\forall}$ holds immediately. If $R \notin \mathsf{CT}^{res}_{\forall\exists}$, then for some instance I no *res*-chase sequence terminates for $\langle R, I \rangle$. In particular some *res*-chase sequence without alternative matches, which exists since $R \in \overline{\mathsf{AM}}_{\forall\exists}$, does not terminate and thus yields an infinite universal core model by Proposition 3.3. By Lemma 2.6, no finite universal model of $\langle R, I \rangle$ exists which is

the case iff the core chase does not terminate for $\langle R, I\rangle$ by Proposition 2.17. Thus, $R \notin \mathsf{CT}^{core}_{\forall}$. □

Note that if $R \in \mathsf{CT}^{res}_{\forall\exists}$ and $R \in \overline{\mathsf{AM}}_{\forall\exists}$, then every chase sequence that does not have alternative matches is also finite for the restricted chase. Otherwise, this contradicts Lemma 2.6. In practice, we can always use the rule application order that we use in the transfinite chase on a restrained partitioning of a core-stratified rule set to obtain a finite universal core model if one exists.

Corollary 4.14. *Consider a core-stratified rule set R and an instance I. A transfinite chase sequence on a restrained partitioning of R and I terminates (yielding a finite universal core model) iff $\langle R, I\rangle$ has a finite universal (core) model.*

Proof. In the proof of Theorem 4.1, we show that every transfinite chase sequence for a restrained partitioning of R and I is equivalent to a fair *res*-chase sequence that does not have an alternative match. Hence, a transfinite chase sequence on a restrained partitioning of R and I yields a universal core model. This universal core model is finite iff the transfinite chase sequence terminates. In particular, if the transfinite chase sequence does not terminate, by Lemma 2.6 no finite universal (core) model of $\langle R, I\rangle$ exists. □

For the knowledge base $\mathcal{K}$ from Example 1.1, we stress that $\mathcal{K}$ does not have a finite universal (core) model since we find a transfinite chase sequence for one of its restrained partitionings that yields an infinite universal core model in Example 4.12.

Theorems 4.1 and 4.13 also enable us to use sufficient conditions for non-membership in $\mathsf{CT}^{res}_{\forall\exists}$ for a rule set R, e.g. RMFC, to show the existence of an instance I such that $\langle R, I\rangle$ does not have a finite universal models.

Corollary 4.15. *Consider a core-stratified rule set R. If R is RMFC, then there exists an instance I such that $\langle R, I\rangle$ does not have a finite universal model.*

Proof Sketch. If R is RMFC, then $R \notin \mathsf{CT}^{res}_{\forall\forall}$ [10]. From the definition of RMFC, we even find that $R \notin \mathsf{CT}^{res}_{\forall\exists}$. Without going into the formal definitions, RMFC checks if there exists a rule ρ such that the instance I_ρ allows

an infinite *res*-chase sequence where I_ρ contains only facts that are required to make at least one trigger for ρ active. This check consist of a chase-like procedure on I_ρ that applies triggers that are "guaranteed to be applied" in an actual *res*-chase sequence. In other words, triggers that may become obsolete are ignored. If such a chase-like procedure is already infinite, then also every res-chase sequence on $\langle R, I_\rho \rangle$ is infinite. Hence, if R is RMFC, then $R \notin \mathsf{CT}^{res}_{\forall\exists}$ and thus, by Theorems 4.1 and 4.13, the claim follows. □

In the following, we investigate some special cases for fragments of existential rules. For single-head linear rules, we know that $\mathsf{CT}^{res}_{\forall\exists}$ is decidable [23] but the same work already shows decidability of core chase termination for single-head linear rules as well. To the best of our knowledge, $\mathsf{CT}^{res}_{\forall\exists}$ is not known to be decidable for guarded existential rules but it is also not known to be undecidable in this case. We investigate in the following if we can still find stronger results for guarded existential rules since at least $\mathsf{CT}^{res}_{\forall\forall}$ is known to be decidable for single-head guarded existential rules [16].

4.2 Restraining and Guarded Rules

In this section, we focus on $\overline{\mathsf{AM}}_{\forall\forall}$ instead of $\overline{\mathsf{AM}}_{\forall\exists}$ and relate it to $\mathsf{CT}^{res}_{\forall\forall}$ instead of $\mathsf{CT}^{res}_{\forall\exists}$ because $\mathsf{CT}^{res}_{\forall\forall}$ is decidable for single-head guarded existential rules [16]. Since $\overline{\mathsf{AM}}_{\forall\forall} \subseteq \overline{\mathsf{AM}}_{\forall\exists}$, $\overline{\mathsf{AM}}_{\forall\forall}$ is less general than $\overline{\mathsf{AM}}_{\forall\exists}$. Still, in contrast to $\overline{\mathsf{AM}}_{\forall\exists}$, we know that $\overline{\mathsf{AM}}_{\forall\forall}$ is decidable by Proposition 3.9, namely by checking for the existence of restraining relations within a rule set. For rule sets in $\overline{\mathsf{AM}}_{\forall\forall}$, we show that $\mathsf{CT}^{res}_{\forall\forall} = \mathsf{CT}^{res}_{\forall\exists} = \mathsf{CT}^{core}_{\forall}$. These results together with a result by Gogacz et al. [16] immediately imply that core chase termination is decidable for single-head guarded existential rules without restraining relations. We are even able to generalize the Fairness Theorem by Gogacz et al. [16] and hence we conjecture that core chase termination is decidable for guarded existential rules without restraining relations.

4.2.1 Restricted and Core Chase Termination coincide for Rule Sets without Restraining Relations

As an important result for rule sets in $\overline{\mathsf{AM}}_{\forall\forall}$, we learn that restricted and core chase termination coincide for such rule sets.

Theorem 4.16. *Consider a rule set $R \in \overline{\mathsf{AM}}_{\forall\forall}$. We have that $R \in \mathsf{CT}^{res}_{\forall\exists}$ iff $R \in \mathsf{CT}^{res}_{\forall\forall}$ iff $R \in \mathsf{CT}^{core}_{\forall}$.*

Proof. We assume that $R \in \overline{\mathsf{AM}}_{\forall\forall}$. We show the three following claims that immediately yield the result of the theorem:

1. If $R \in \mathsf{CT}^{res}_{\forall\exists}$, then $R \in \mathsf{CT}^{res}_{\forall\forall}$.
2. If $R \in \mathsf{CT}^{res}_{\forall\forall}$, then $R \in \mathsf{CT}^{core}_{\forall}$.
3. If $R \in \mathsf{CT}^{core}_{\forall}$, then $R \in \mathsf{CT}^{res}_{\forall\exists}$.

We show the first and third claim in a similar way. If $R \in \mathsf{CT}^{res}_{\forall\exists}$ or $R \in \mathsf{CT}^{core}_{\forall}$, respectively, then, for all knowledge bases featuring R, there exists a finite universal model. Suppose for a contradiction that $R \notin \mathsf{CT}^{res}_{\forall\forall}$ or $R \notin \mathsf{CT}^{res}_{\forall\exists}$, respectively. In either case, there exists an instance I such that $\langle R, I\rangle$ has a non-terminating *res*-chase sequence. This sequence yields an infinite universal core model by Proposition 3.3 since $R \in \overline{\mathsf{AM}}_{\forall\forall}$. By Lemma 2.6, we obtain the desired contradiction.

For the second claim, assume that $R \in \mathsf{CT}^{res}_{\forall\forall}$. Then, a finite universal (core) model of $\langle R, I\rangle$ exists for every instance I. This is the case iff $R \in \mathsf{CT}^{core}_{\forall}$ by Proposition 2.17, which yields the second claim.

Since the three claims hold, $R \in \mathsf{CT}^{res}_{\forall\exists}$ iff $R \in \mathsf{CT}^{res}_{\forall\forall}$ iff $R \in \mathsf{CT}^{core}_{\forall}$. □

Since $\mathsf{CT}^{res}_{\forall\forall}$ is known to be decidable for single-head guarded existential rules [16], we immediately obtain an analogous result for $\mathsf{CT}^{res}_{\forall\exists}$ and $\mathsf{CT}^{core}_{\forall}$.

Corollary 4.17. *Consider a single-head guarded existential rule set R without restraining relations. It is decidable if $R \in \mathsf{CT}^{res}_{\forall\exists}$ and if $R \in \mathsf{CT}^{core}_{\forall}$.*

Proof. The claim follows since $\mathsf{CT}^{res}_{\forall\forall}$ is decidable for single-head guarded existential rule set according to Gogacz et al. [16, Theorem 5.1] and $R \in \mathsf{CT}^{res}_{\forall\forall}$

iff $R \in \mathsf{CT}^{res}_{\forall\exists}$ iff $R \in \mathsf{CT}^{core}_{\forall}$ according to Theorem 4.16 and Proposition 3.9. □

Note that even though $\overline{\mathsf{AM}}_{\forall\forall}$ is decidable, we cannot give a decision procedure for $\mathsf{CT}^{core}_{\forall}$ for single-head guarded existential rule sets in general, since for the case $R \notin \mathsf{CT}^{res}_{\forall\forall} \cup \overline{\mathsf{AM}}_{\forall\forall}$, it is not clear whether $R \in \mathsf{CT}^{core}_{\forall}$ or not.

In the following, we investigate if the requirement for single-head rules is necessary for Corollary 4.17.

4.2.2 Generalizing the Fairness Theorem

In the previous considerations, we use a main result by Gogacz et al. [16] as is. One key part of the corresponding proof is the so-called Fairness Theorem.

Proposition 4.18 (Fairness Theorem [16, Theorem 4.1])**.** *Consider a knowledge base $\langle R, I\rangle$ such that R consists only of single-head rules. If there exists a possibly unfair non-terminating res-chase sequence for $\langle R, I\rangle$, then there also exists a fair non-terminating res-chase sequence for $\langle R, I\rangle$.*

In the decidability proof of $\mathsf{CT}^{res}_{\forall\forall}$ for single-head guarded existential rules, the Fairness Theorem is the part where the single-head requirement is most crucial. The rest of the proof also makes the assumption of single-heads but we think that this requirement may be relaxed there, although then the proof becomes more technical. Hence, we conjecture that $\mathsf{CT}^{res}_{\forall\forall}$ is decidable for a slightly larger fragment of guarded existential rules and that the result by Gogacz et al. [16] can be generalized.

As a simple result very close to our earlier considerations, we observe that rule sets without restraining relations allow for arbitrary reordering of rule applications in any chase sequence as long as the corresponding triggers are active. In particular, no trigger makes another trigger obsolete. We sketch the proof idea here and we give a detailed proof in a more general version of this result later on in Theorem 4.26.

Theorem 4.19 (Fairness based on non-restraining)**.** *Consider a rule set R without restraining relations and an instance I. If there exists a possibly unfair non-terminating res-chase sequence for $\langle R, I\rangle$, then there also exists a fair non-terminating res-chase sequence for $\langle R, I\rangle$.*

Proof Sketch. If a rule set has no restraining relations, then by Lemma 4.9, for every pair of triggers λ_1, λ_2 in a chase sequence with λ_2 being applied at some point after λ_1, the application of λ_2 before λ_1 does not make λ_1 obsolete. This means that triggers within a *res*-chase sequence can be shifted to the beginning of the sequence as soon as they become active, which is after a finite amount of steps. In this way, we can reorder trigger applications in any (unfair) non-terminating *res*-chase sequence to obtain a fair non-terminating *res*-chase sequence. □

This variation of the Fairness Theorem does not generalize the original version [16]. In particular, single-head guarded rule sets may contain restraining relations.

Example 4.20. *Consider the rule set R with the rules*

$$\rho_1 \coloneqq P(x, y) \to \exists z.P(y, z)$$
$$\rho_2 \coloneqq P(x, y) \to P(y, x)$$

We find that $\rho_2 \prec^{\square} \rho_1$ even though the rules are guarded (even linear) and only feature a single atom in their heads.

In practice, we can also assume that only very few rule sets have no restraining relations at all. Thus, we aim to find a more general condition using a similar idea.

Definition 4.21 (Strong Restraining)**.** Consider two rules ρ_1 and ρ_2. The rule ρ_2 *strongly restrains* ρ_1, written $\rho_2 \prec^{\times} \rho_1$, if there exists an infinite list of fact sets $(F_1^i)_{i\geq 1}$, a fact set F_2, an infinite list of triggers $(\lambda_1^i = \langle \rho_1, \theta_1^i \rangle)_{i\geq 1}$, and a trigger $\lambda_2 = \langle \rho_2, \theta_2 \rangle$ that are *res*-applicable to $(F_1^i)_{i\geq 1}$, and F_2, respectively, with $F_1^i \subseteq F_1^j$ for all $i < j$ and $F_1^i \subseteq F_2$ for all $i \geq 1$, such that θ_1^i and θ_1^j do not agree in all frontier variables of ρ for all $i < j$ and λ_1^i has an alternative match w.r.t. $\lambda_2(F_2)$ but does not have an alternative match w.r.t. F_2 for all $i \geq 1$.

Intuitively, if a rule set has no pair of rules that strongly restrain each other, then if $\rho_2 \prec^{\square} \rho_1$, there is only a finite amount of triggers featuring ρ_1 that differ in their mapping of the frontier of ρ_1 for which a particular trigger featuring ρ_2 introduces alternative matches. Hence, a particular trigger featuring ρ_2 can also only make a finite amount of triggers featuring ρ_1 obsolete.

Example 4.22. *We show an example by Gogacz et al. [16]. Consider the rule set R consisting of the following rules.*

$$\rho_1 \coloneqq S(x,y,y) \to \exists z.S(x,z,y) \wedge S(z,y,y)$$
$$\rho_2 \coloneqq S(x,y,z) \to S(z,z,z)$$

We have that $\rho_2 \prec^\times \rho_1$ by setting $F_1^1 \coloneqq \{ S(b_1,a,a) \}$, $F_1^i \coloneqq \lambda_1^{i-1}(F_1^{i-1}) \cup \{ S(b_i,a,a) \}$ with a fresh constant b_i for each $i > 1$ and $F_2 \coloneqq \bigcup_{i>0} \lambda_1^i(F_1^i)$ in Definition 4.21. Each trigger λ_1^i maps y to a and x to b_i. The trigger λ_2 can be chosen arbitrarily since all possible applications of ρ_2 in this context always result in $S(a,a,a)$.

In contrast, the rule set in Example 4.20 does not feature any strong restraining relations. In fact, every trigger λ_2 for ρ_2 makes only triggers for ρ_1 obsolete that map y to the same term as λ_2.

Instead of requiring infinite sequences in Definition 4.21, we can give similar conditions for finite lists of fact sets and triggers of arbitrary size n that we call n-restraining. For example, we may require only two fact sets and two triggers to obtain a practical checkable condition of 2-restraining. Note that 1-restraining is the same as restraining. We find that if two rules do not n-restrain each other for some $n > 0$, then they do also not n'-restrain each other for any $n' > n$. In particular they do also not strongly restrain each other.

One may think that this also holds vice versa, i.e. if two rules 2-restrain each other then they also strongly restrain each other because one can try to construct an infinite amount of triggers by mapping some frontier variables to fresh constants. However, this is not always possible as we see in the following counterexample.

Example 4.23. *Consider the following rules similar to Example 4.20.*

$$\rho_1 \coloneqq P(x,y) \to \exists z.P(y,z)$$
$$\rho_2 \coloneqq P(x,y) \to P(y,x) \wedge P(x,x)$$

We have that ρ_2 2-restrains ρ_1 by setting $F_1^1 \coloneqq \{ P(b,a) \}$, $F_1^2 \coloneqq F_1^1 \cup \{ P(a,n_1) \} = \lambda_1^1(F_1^1)$, $F_2 \coloneqq F_1^2 \cup \{ P(n_1,n_2) \} = \lambda_1^2(F_1^2)$, and $\lambda_2(F_2) = F_2 \cup \{ P(n_1,a), P(a,a) \}$ where λ_1^1, λ_1^2, and λ_2 are appropriate triggers.

We observe that every trigger for ρ_2 that maps x and y to two different terms, e.g. a and n_1, introduces an alternative match for exactly two triggers (up to renaming of non-frontier terms) for ρ_1 which map y to a or n_1, respectively. Furthermore, every trigger for ρ_2 that maps x and y to the same term only introduces an alternative match for a single trigger (up to renaming of non-frontier terms) for ρ_1. Thus, ρ_2 2-restrains ρ_1 but does not n-restrain ρ_1 for any $n > 2$. In particular, ρ_2 does not strongly restrain ρ_1.

Similar to restraining, 2-restraining is decidable in Σ_2^{P} using a similar proof idea as in Proposition 3.15.

Proposition 4.24. *Consider two rules ρ_1 and ρ_2. Deciding if ρ_2 2-restrains ρ_1 is in Σ_2^{P}.*

Proof Sketch. Similar to the proof of Proposition 3.15, the complexity of the 2-restraining check is subsumed by the complexity for checking RuleApplicability (Proposition 2.11). Here, we need to guess one more fact set and one more trigger. The rest of the proof is analogous. □

We show that non-2-restraining already generalizes single-head rules.

Proposition 4.25. *Single-head rule sets do not have 2-restrained rules.*

Proof. Suppose for a contradiction that a single-head rule set with a pair of rules ρ_1, ρ_2 exists such that ρ_2 2-restrains ρ_1. Let λ_1^1, λ_1^2, and λ_2 be triggers according to Definition 4.21. If ρ_1 does not feature frontier variables in its head, then only a single trigger is *res*-applicable to ρ_1. Therefore, we assume that the head of ρ_1 features at least one frontier variable. Since ρ_2 (2-)restrains ρ_1, ρ_1 and ρ_2 feature the same (single) head predicate P with arity n. We assume w.l.o.g. that the head of ρ_1 has the form $P(x_1, \ldots, x_n)$ and we define $\vec{i}$ to be the list of all i such that x_i is a frontier variable in ρ_1. Let $P_1^1(s_1, \ldots, s_n)$, $P_1^2(t_1, \ldots, t_n)$, and $P_2(u_1, \ldots, u_n)$ be the facts that result from the application of λ_1^1, λ_1^2, and λ_2, respectively. Since λ_1^1 has an alternative match w.r.t. $\lambda_2(F_2)$, we have that $s_i = u_i$ for all $i \in \vec{i}$. But since λ_1^2 has an alternative match w.r.t. $\lambda_2(F_2)$, we have that $t_i = u_i$ for all $i \in \vec{i}$. Hence, we have $s_i = t_i$ for all $i \in \vec{i}$ and thus, the substitutions of λ_1^1 and λ_1^2 agree in all their frontier variables which is a contradiction to the definition of 2-restraining (see Definition 4.21). □

Using strong restraining, we generalize our result from earlier and obtain a version of the Fairness Theorem, which also generalizes the original one because of Proposition 4.25.

Theorem 4.26 (Fairness based on Non-Strong-Restraining)**.** *Consider a rule set R without strong restraining relations and an instance I. If there exists a possibly unfair non-terminating res-chase sequence for $\langle R, I\rangle$, then there also exists a fair non-terminating res-chase sequence for $\langle R, I\rangle$.*

Proof. Let $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$ be an unfair *res*-chase sequence for $\langle R, I\rangle$ that we represent by the list of applied triggers Λ. There exists a trigger $\lambda_\dagger$ that is not applied after a finite amount of steps but that is *res*-applicable after a finite amount of steps. Since R does not feature rules that strongly restrain each other, for each rule in R, there are only finitely many triggers that do not agree in all of their frontier variables for which the application of $\lambda_\dagger$ may introduce alternative matches. Furthermore, there are only finitely many triggers $\lambda_1, \ldots \lambda_n$ in Λ for which the application of $\lambda_\dagger$ may introduce alternative matches since only one trigger for each rule with the same mapping of frontier variables can be applied in a *res*-chase sequence. Thus, $\lambda_\dagger$ can be inserted into Λ as soon as it becomes *res*-applicable and after all triggers $\lambda_1, \ldots \lambda_n$ are applied. This is the case after a finite amount of chase steps. We know that $\lambda_\dagger$ does not introduce alternative matches for any trigger that $\lambda_\dagger$ is inserted before. Hence, all consecutive triggers in Λ are still *res*-applicable (in particular not obsolete) by Lemma 4.9. This insertion can be done exhaustively for all triggers that are not applied after a finite amount of steps but *res*-applicable after a finite amount of steps. By this procedure, we describe a fair chase sequence that subsumes Λ and is thus non-terminating as well. □

Note that we can still find rule sets with strong restraining relations and instances such that there exists both an unfair and a fair non-terminating chase sequence.

Example 4.27. *We show an adjusted version of an example by Gogacz et*

al. [16]. Consider the rule set R consisting of the following rules.

$$\rho_1 \coloneqq S(x,y,y) \to \exists z.S(x,z,y) \wedge S(z,y,y)$$
$$\rho_2 \coloneqq S(x,y,z) \to S(z,z,z)$$
$$\rho_3 \coloneqq P(x,y) \to \exists z.P(y,z)$$

From Example 4.22, we know that $\rho_2 \prec^{\times} \rho_1$.

On the instance $I \coloneqq \{ P(a,b), S(b,a,a) \}$, we find an unfair non-terminating res-chase sequence by only applying ρ_3 infinitely often. Similarly, we find a fair non-terminating res-chase sequence, which consists of finitely many applications for ρ_1 and ρ_2 followed by infinitely many applications of ρ_3.

We think that by using the adjusted Fairness Theorem in Theorem 4.26, the single-head requirement in the rest of the proof by Gogacz et al. [16] can be relaxed to support arbitrary guarded existential rules although the proof still needs to be carried out in detail.

Conjecture 4.28. *Consider a guarded rule set R without strong restraining relations. It is decidable if $R \in \mathsf{CT}^{res}_{\forall\forall}$.*

Note that we conjecture that a similar result also holds for so-called sticky existential rules, which we do not introduce here, but for which a decidability proof for $\mathsf{CT}^{res}_{\forall\forall}$ based on the Fairness Theorem is also shown by Gogacz et al. [16]. The detailed investigation of the proofs is a subject for future work.

Conjecture 4.28 allows us to decide if a rule set is in $\mathsf{CT}^{res}_{\forall\forall}$ for rule sets without restraining relations and rule sets without 2-restraining relations. In particular, for a guarded existential rule set R without restraining relations, this allows us to decide if $R \in \mathsf{CT}^{core}_{\forall}$ by Proposition 3.9, Theorem 4.16, and the fact that a rule set without restraining relations also does not feature any strong restraining relations. This is more general than Corollary 4.17.

4.3 Cores with the Skolem Chase

After only considering cases for when the restricted chase yields universal core models, it is natural to ask if other chase variants (beside restricted- and core-chase) yield universal core models as well for certain rule sets. In

this section, we investigate if a condition similar to core-stratification exists for the skolem chase. Recall that the order in which triggers are applied does not influence the result of the skolem chase up to bijective renaming of nulls.

We may aim for a result similar to Corollary 4.14 but we find instead that a similar result cannot hold for the skolem chase.

Theorem 4.29. *Consider a rule set R that features at least one rule that contains existentially quantified variables. There exists an instance I such that all sk-chase sequences on $\langle R, I\rangle$ do not yield a universal core model.*

Proof. Since the skolem chase always yields a universal model, we show in particular that the result is not a core.

Let $\rho \in R$ be a rule that features at least one existentially quantified variable. Let θ be a substitution that maps all (universally and existentially quantified) variables in ρ to fresh constants. Let $I \coloneqq \theta(\mathit{body}(\rho) \cup \mathit{head}(\rho))$. Let θ' be the same substitution as θ except that θ' maps the existentially quantified variables in ρ to themselves.

In any *sk*-chase sequence $F^0_{\langle R,I\rangle}, F^1_{\langle R,I\rangle}, \dots$ for $\langle R, I\rangle$, the trigger $\lambda = \langle \rho, \theta'\rangle$ is active in all steps. Since the *sk*-chase sequence is fair, λ or a trigger that agrees with λ in the variables in $\mathit{frontier}(\rho)$ is applied after a finite amount of steps producing the facts $\theta''(\mathit{head}(\rho))$ where θ'' is a substitution that agrees with θ' in $\mathit{frontier}(\rho)$ and θ'' maps the existentially quantified variables in ρ to fresh nulls.

There exists an endomorphism $h : F^\infty_{\langle R,I\rangle} \to F^\infty_{\langle R,I\rangle}$ that maps $\theta''(z)$ to $\theta(z)$ for every existentially quantified variable z in ρ, i.e. h maps the nulls introduced by $\langle \rho, \theta''\rangle$ to corresponding constants in I. In the case that $F^\infty_{\langle R,I\rangle}$ features a null n that depends on a null that is introduced by $\langle \rho, \theta''\rangle$, then a corresponding null n' was also introduced by the skolem chase that depends on the corresponding constant in I. Then, h also remaps n to n' and is still an endomorphism on $F^\infty_{\langle R,I\rangle}$. Since h is not injective, $F^\infty_{\langle R,I\rangle}$ is not a core. □

Chapter 5

Computing Cores for Non-Core-Stratified Rule Sets

After Chapter 4, we can already give a refined version of a practical procedure for core computation in Figure 5.1.

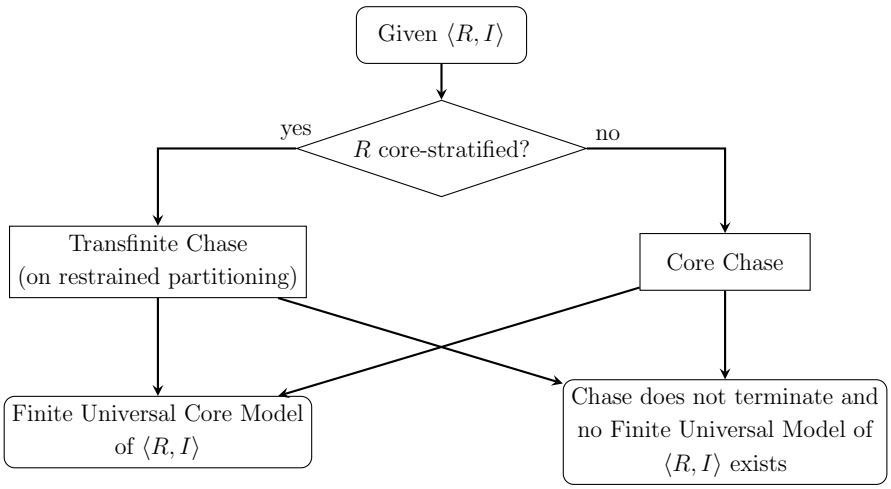


Figure 5.1: Procedure for Core Computation with Transfinite Chase

After studying cases in which the restricted chase is able to yield universal core models we aim to investigate the following questions in this chapter:

1. Can we remove self-restraining relations in non-core-stratified rule sets by rewriting them into equivalent rule sets to obtain a rule set in $\overline{\mathsf{AM}}_{\forall\exists}$?

2. Can we find an efficient way to compute the core chase when we cannot use the restricted chase to compute universal core models, i.e. when a rule set is not in $\overline{\mathsf{AM}}_{\forall\exists}$? In particular, can we make use of restraining relations even if a rule set if not core-stratified?

5.1 Remove Self-Retraining by Piece Decomposition

From our considerations regarding the skolem chase in Theorem 4.29, we notice that a similar issue may occur for the restricted chase as well.

Example 5.1. *Consider the singleton rule set R with the rule $\rho \coloneqq A(x) \to \exists z, z'.P(x,z) \wedge S(x,z')$. We have $R \notin \overline{\mathsf{AM}}_{\forall\exists}$ since for the instance $I \coloneqq \{ A(c), P(c,c) \}$ the first application of ρ on I leads to an alternative match.*

This issue is captured by restraining and thus by core-stratification. In fact, ρ restrains itself. Still, this issue can be resolved by decomposing the rule according to its existentially quantified variables using so-called *pieces* [2].

Definition 5.2 (Pieces)**.** Consider a rule ρ. The *pieces* of ρ, written $\mathit{pieces}(\rho)$, are the largest partitioning of $\mathit{head}(\rho)$ such that for each two atoms $A, B \in \mathit{head}(\rho)$, A and B occur in the same set of the partitioning, i.e. the same *piece*, if A and B feature a common existentially quantified variable.

The *piece decomposition* of a rule ρ with $\mathit{pieces}(\rho) = \{ \varphi_1, \ldots, \varphi_n \}$ is the set of rules $\{\rho_1, \ldots, \rho_n\}$ where $\rho_i \coloneqq \mathit{body}(\rho) \to \exists \vec{z_i}.\varphi_i$ for every $1 \leq i \leq n$ and $\vec{z_i}$ is the list of existentially quantified variables that occur in φ_i. The *piece decomposition* of a rule set is the union of the piece decompositions of all of its rules.

Example 5.3. *For the rule $\rho \coloneqq A(x) \to \exists z, z'.P(x,z) \wedge S(x,z')$ from Example 5.1, we obtain $\mathit{pieces}(\rho) = \{ \{ P(x,z) \}, \{ S(x,z') \} \}$. The piece decomposition of ρ consists of two rules:*

$$\rho_1 = A(x) \to \exists z.P(x,z)$$
$$\rho_2 = A(x) \to \exists z'.S(x,z')$$

Proposition 5.4 ([2, Property 4])**.** *Consider a rule set R. The piece decomposition R' of R is equivalent to R, i.e. for each fact set M, we have that $M \models R$ iff $M \models R'$.*

Proof. Let M be a fact set and let R' be the piece decomposition of R.

If $M \models R$, then $M \models \rho$ for each $\rho \in R$. By definition, for every substitution θ that is a homomorphism from $\mathit{body}(\rho)$ to F, there exists a substitution θ' that is a homomorphism from $\mathit{head}(\rho)$ to F such that $\theta(x) = \theta'(x)$ for every variable $x \in \mathit{frontier}(\rho)$. Then, θ' is also a homomorphism from $\mathit{head}(\rho')$ to M for every rule ρ' in R'. Thus, $M \models \rho'$ and in turn $M \models R'$.

If $M \not\models R$, then $M \not\models \rho$ for some $\rho \in R$. By definition, there exists a substitution θ that is a homomorphism from $\mathit{body}(\rho)$ to M but every substitution θ' with $\theta(x) = \theta'(x)$ for every variable $x \in \mathit{frontier}(\rho)$ is not a homomorphism from $\mathit{head}(\rho)$ to M. Hence, there exists an atom $a \in \mathit{head}(\rho)$ such that $\theta'(a) \notin M$. Since R' is the piece decomposition of R, there exists a rule $\rho' \in R'$ with $a \in \mathit{head}(\rho')$. But then, θ' is not a homomorphism from $\mathit{head}(\rho')$ to M. Thus, $M \not\models \rho'$ and in turn $M \not\models R'$ □

We observe that piece decomposition can eliminate self restraining in some cases as in Example 5.3. However, there are also rules with only one piece that restrain themselves.

Example 5.5 ([21, Example 5])**.** *Consider the rule $B(x) \to \exists v, w.P(x, v, w) \wedge P(x, x, w) \wedge A(v)$. We have $\mathit{pieces}(\rho) = \{\{ P(x, v, w), P(x, x, w), A(v) \}\}$ but the rule still restrains itself.*

In practice, since piece decompositions of rule sets are equivalent to the original rule sets by Proposition 5.4, it is preferable to always consider the piece decomposition of a given rule set when computing cores to be able to benefit from core-stratification more often.

5.2 A more efficient Core Chase

In the main part of this chapter, we point out how ideas like restraining can improve the computation of the core chase in practice. To recall, a *core*-chase

sequence is basically a *res*-chase sequence that applies triggers in parallel and computes cores of the intermediate results.

If a rule set is not core-stratified, we cannot use the restricted chase directly to obtain a core in general. However, we can use the idea of alternative matches as a heuristic for the actual core computation to a certain extent by checking for "local homomorphisms" instead of "global" ones. Additionally, we can use the idea of the transfinite chase to benefit from rule sets which are "partially core-stratified". In particular, we can combine both ideas.

5.2.1 Intermediate Cores with Extended Alternative Matches

For the actual core computation, we only consider a single chase step first. Since the core chase also relies on *res*-applicability, the complexity of checking RuleApplicability for the core chase is Σ_2^{P} as for the restricted chase according to Proposition 2.11.

The part where restricted chase and core chase differ is the definition of a single chase step. For the restricted chase, we obtain the next fact set for a trigger $\lambda = \langle \rho, \theta \rangle$ and a fact set F as $\lambda(F)$, i.e. the set of facts that is the union of F and F_λ. Hence, for the restricted chase, the computationally hard part of a rule application is to find an applicable trigger. On the other hand, for the core chase, we obtain the next fact set as $\mathit{core}(R(F))$. Essentially, the restricted chase application is performed as before but we apply all applicable triggers in parallel and we additionally compute a core of the resulting fact set. Since applying multiple triggers in parallel just requires to check applicability multiple times, we focus on the core computation since this is the major difference between the two chase variants.

According to Proposition 2.5, the underlying decision problem of CoreIdentification is DP-complete. While DP seems feasible on first sight in comparison to Σ_2^{P} for the RuleApplicability check, recall that Proposition 2.11 also states that the RuleApplicability check is in P w.r.t. the size of the given fact set if a rule is fixed. On the other hand, the complexity of CoreIdentification is already only w.r.t. to the size of the fact set. When computing the chase, we can infer that core computation becomes much harder than the applicability

check for growing fact sets.

An intuitive procedure for computing cores based on alternative matches could look as follows. During the chase, we check if a rule ρ_2 is applied that restrains another rule ρ_1 that has been applied before or in parallel. Instead of computing a core right away, we check for alternative matches of the trigger that features the restrained rule ρ_1, which promises to be more feasible than finding an endomorphism over all facts. Then, we remap nulls according to the alternative matches to obtain a core. However, even though this approach seems intuitive, we show that it does not work exactly as described by pointing out three key problems in the following. Afterwards, we present an adjusted procedure that solves these problems by sacrificing the use of restraining relations and by utilizing a more liberal notion of alternative matches.

The first problem is that an alternative match for a trigger featuring ρ_1 may only exist after further facts have been introduced by other rule applications.

Example 5.6. *Consider the instance* $I = \{ A(c) \}$ *and the rule set* R *with the rules*

$$\rho_1 \coloneqq A(x) \to \exists z.S(x,z) \wedge P(x,z)$$
$$\rho_2 \coloneqq S(x,y) \to S(x,x)$$
$$\rho_3 \coloneqq S(x,x) \to P(x,x)$$

We have the following sequence of fact sets if we just apply all rules in parallel without computing cores in between.

$$F_1 \coloneqq R(I) = \{ A(c), S(c,n_1), P(c,n_1) \}$$
$$F_2 \coloneqq R(F_1) = F_1 \cup \{ S(c,c) \}$$
$$F_3 \coloneqq R(F_2) = F_2 \cup \{ P(c,c) \}$$

We have that $\rho_2 \prec^{\square} \rho_1$ *and* $\rho_3 \prec^{\square} \rho_1$*. Still, we find that* ρ_2 *has been applied to obtain* F_2 *but there does not exist an alternative match. Only w.r.t.* F_3*, where* ρ_3 *has been applied, we find an alternative match for the trigger that has been applied for* ρ_1*.*

Second, even if we find an alternative match for an intermediate fact set, this fact set may still be a core already. Therefore, we need to check if the

alternative match can be extended to a valid endomorphism over the whole fact set. Again, there could be further rule applications necessary to introduce enough facts such that the alternative match can be extended to a valid endomorphism over the whole fact set or such an endomorphism may no exist at all.

Example 5.7 ([21, Example 4])**.** *Consider the empty instance $I = \emptyset$ and the rule set R with the following rules.*

$$\begin{aligned}
\rho_1 &\coloneqq \ \rightarrow \exists z.P(z) \\
\rho_2 &\coloneqq P(x) \rightarrow \exists z.S(x,z) \\
\rho_3 &\coloneqq S(x,y) \rightarrow P(y) \wedge S(y,x) \\
\rho_4 &\coloneqq S(x,y) \wedge S(y,x) \rightarrow S(x,x)
\end{aligned}$$

We have the following sequence of fact sets if we just apply all rules in parallel without computing cores in between.

$$\begin{aligned}
F_1 &\coloneqq R(I) = \{\, P(n_1) \,\} \\
F_2 &\coloneqq R(F_1) = F_1 \cup \{\, S(n_1,n_2) \,\} \\
F_3 &\coloneqq R(F_2) = F_2 \cup \{\, P(n_2), S(n_2,n_1) \,\} \\
F_4 &\coloneqq R(F_3) = F_3 \cup \{\, S(n_1,n_1), S(n_2,n_2) \,\}
\end{aligned}$$

We observe that F_3 already introduces an alternative match for the trigger that has been applied for ρ_1 to obtain $P(n_1)$ in F_1. However, F_3 is still a core. Only once we have F_4, we can find an endomorphism that does not have n_1 in its image. If we remove ρ_4 from R, such an endomorphism does not exist at all in the corresponding chase sequence but an alternative match still exists.

Third, the elimination of nulls by endomorphisms can lead to a situation where another null that was introduced by some rule ρ needs to be removed later on even though no rule that restrains ρ is applied after or in parallel to ρ. When remapping nulls, the notion of alternative matches is also not clear anymore. More severely than for the first issue, this means that we have to look for "homomorphisms in the spirit of alternative matches" after every chase step and we cannot rely on restraining relations to detect when such a homomorphism may occur.

Example 5.8. *Consider the empty instance $I = \emptyset$ and the rule set R with the following rules.*

$$\rho_1 \coloneqq \ \rightarrow \exists z.P(z)$$
$$\rho_2 \coloneqq P(x) \rightarrow \exists z.Q(x,z)$$
$$\rho_3 \coloneqq Q(x,y) \rightarrow \exists z.Q(z,y) \wedge Q(z,c) \wedge P(z) \wedge S(z,y)$$
$$\rho_4 \coloneqq Q(x,y) \wedge S(x,z) \rightarrow S(x,y)$$

We have the following sequence of fact sets if we just apply all rules in parallel without computing cores in between.

$$F_1 \coloneqq R(I) = \{\, P(n_1) \,\}$$
$$F_2 \coloneqq R(F_1) = F_1 \cup \{\, Q(n_1,n_2) \,\}$$
$$F_3 \coloneqq R(F_2) = F_2 \cup \{\, Q(n_3,n_2), Q(n_3,c), P(n_3), S(n_3,n_2) \,\}$$
$$F_4 \coloneqq R(F_3) = F_3 \cup \{\, S(n_3,c), Q(n_4,c), P(n_4), S(n_4,c) \,\}$$

Once we derive F_3, we can remap n_1 to n_3 and once we derive F_4, we can additionally remap n_2 to c (and n_4 to n_3 to obtain a core). Still, the rule ρ_2, which introduces $Q(n_1,n_2)$, is not restrained by any rule. Thus, the necessity for the remapping of n_2 is not detected if we wait until a rule that restrains ρ_2 is applied. Note that without the remapping of n_1, the trigger for ρ_2, which introduces $Q(n_1,n_2)$, does not even have an alternative match w.r.t. to F_4. For a possible fix of this issue, it seems intuitive that n_2 depends on n_1 and that we could remap n_2 as soon as we remap n_1. However, this is not true since we can already remap n_1 to n_3 in F_3 where we cannot map n_2 to c yet.

Because of these limitations, the following procedure of checking for "homomorphisms in the spirit of alternative matches" and extending them to endomorphisms over the whole fact set is rather close to a more naive core computation. We view this approach as a basic heuristic for finding global endomorphisms that remove unnecessary nulls. A similar idea has been presented for data exchange settings [17, Procedure FindCore].

To clarify this approach, we define a chase sequence in the spirit of the core chase. By that, we aim to shrink the search space for the NP and coNP part of CoreIdentification in some cases. Formally, we use a relaxed version of alternative matches because our new chase sequence is allowed to remap

nulls and then the notion of alternative matches is not clear anymore as in Example 5.8. Throughout the chase sequence, we maintain a list of triggers Λ that contains all triggers that introduce fresh nulls and have been applied already or are applied in the current chase step. We also relax the notion of F_λ within the chase sequence and view F_λ to be a mutable set that is maintained during the chase. Once λ is applied, we initialize F_λ according to the formal definition. During the chase, F_λ is updated according to remappings of nulls.

Definition 5.9. Consider a trigger $\lambda \in \Lambda$, a fact set F, and a homomorphism $h : F_\lambda \to F$ such that there is a non-empty set of nulls N^- in F_λ that do not occur in $h(F_\lambda)$. We define an *extended alternative match* of h to be an endomorphism $h' : F \to F$ with $h'(n) = h(n)$ for every null $n \in N^-$ such that the nulls in N^- do not occur in the image of h'.

Intuitively, h can be considered an alternative match for λ w.r.t. F. However, this intuition may technically not be true anymore once remappings of nulls for the sake of obtaining a core have been done as in Example 5.8. Therefore, we call h *relaxed alternative match*.

Within the chase sequence, we can algorithmically look for extended alternative matches in a given fact set and remove them by exhaustively applying the corresponding endomorphisms. Note that the removal of a null may yield to further relaxed alternative matches. We sketch a basic procedure for the removal of extended alternative matches in Algorithm 1.

The key part of Algorithm 1 lies in the function *findSomeExtendedAlternativeMatch* that calls *relaxedAlternativeMatches* and *tryToBuildExtendedAlternativeMatch*.

The function *relaxedAlternativeMatches* looks for relaxed alternative matches according to Definition 5.9, i.e. for homomorphisms h from F_λ to F such that some null in F_λ does not occur in $h(F_\lambda)$. This can be done iteratively by trying to remap some null that occurs in the position of an existentially quantified variable of the rule in λ first and then extending this mapping to also remap other nulls in F_λ if necessary. It is beneficial for *relaxedAlternativeMatches* if the predicates in F_λ do not occur often in F. If *relaxedAlternativeMatches* does not find any such homomorphisms for any trigger in Λ, the algorithm already terminates.

Algorithm 1: *removeExtendedAlternativeMatches*

Input: fact set F that occurs in chase sequence

Output: modified fact set F' without extended alternative matches

```
1  def findSomeExtendedAlternativeMatch(F)
2      foreach λ ∈ Λ do
3          foreach h ∈ relaxedAlternativeMatches(F_λ, F) do
4              h' := tryToBuildExtendedAlternativeMatch(h, F)
5              if h' then return h'
6          end
7      end
8      return nil
9  end
10
11 F' := F
12 while h' := findSomeExtendedAlternativeMatch(F') do
13     F' := h'(F')
14     foreach λ ∈ Λ do F_λ := h'(F_λ)
15 end
16 return F'
```

If *relaxedAlternativeMatches* finds homomorphisms, i.e. relaxed alternative matches, the function *tryToBuildExtendedAlternativeMatch* then extends the relaxed alternative match h onto the whole fact set F if possible. We can assume w.l.o.g. that h maps all terms that do not occur in F_λ to themselves. Then, *tryToBuildExtendedAlternativeMatch* tries to iteratively extend h to an endomorphism h' on F by remapping only the nulls that occur in a common atom with another null that is already not mapped to itself by h'. All other nulls do not need to be considered and can be mapped to themselves. Hence, it is beneficial for *tryToBuildExtendedAlternativeMatch* if nulls in F_λ are not connected to many other nulls via common atoms. If no extended alternative match can be found, the function returns *nil*.

In the worst case, the functions *relaxedAlternativeMatches* and *tryToBuildExtendedAlternativeMatch* do not improve upon a naive core computation. In terms of complexity, the underlying decision problem of Algorithm 1 is essen-

tially CoreIdentification. While we have to keep in mind that the improvement of Algorithm 1 over a more naive core computation is rather limited because of the issues that we discussed in the examples above, e.g. Example 5.8, we still think that further investigation of these issues may lead to improvements for Algorithm 1.

For a fact set F, we denote the fact set that is obtained by the algorithm *removeExtendedAlternativeMatches* with *eam*(F) similar to *core*(F). Since every extended alternative match removes at least one null from F', the algorithm terminates. We define our new chase sequence formally similar to the core chase.

Definition 5.10. Consider a knowledge base $\mathcal{K} \coloneqq \langle R, I \rangle$. An *eam-chase sequence* for $\mathcal{K}$ is a sequence of fact sets $F^0_\mathcal{K}, F^1_\mathcal{K}, \ldots$ such that

1. $F^0_\mathcal{K} = I$ and
2. $F^{i+1}_\mathcal{K} = \mathit{eam}(R(F^i_\mathcal{K}))$ for each $i \geq 0$.

Example 5.11. *For the knowledge base $\mathcal{K}$ in Example 1.1, we sketch the computation of eam$(R(F^0_\mathcal{K}))$ in the following. The fact set $R(F^0_\mathcal{K})$ is presented in Figure 2.3b. Following Algorithm 1, we find that the trigger λ that introduces n_1 is the only one for which we find a homomorphism h from $F_\lambda = \{\, \mathit{Pizza}(n_1), \mathit{SameDeliverer}(\mathit{order1}, n_1) \,\}$ to $R(F^0_\mathcal{K})$ such that there is a non-empty set of nulls in F_λ that do occur in $h(F_\lambda)$. This set of nulls consists of n_1 here. In particular, h maps n_1 to order2. We can extend h to h' by mapping every term except n_1 to itself to obtain an extended alternative match for h. We obtain eam$(R(F^0_\mathcal{K})) = h'(R(F^0_\mathcal{K}))$, which is the same as core$(R(F^0_\mathcal{K}))$ in Figure 2.3c. In fact, the eam-chase sequence for $\mathcal{K}$ coincides with the core-chase sequence for $\mathcal{K}$ in Figure 2.3.*

We show in the following that the (unique) *eam*-chase sequence for a rule set is equivalent to the (unique) *core*-chase sequence up to isomorphism. It is clear that the *eam*-chase sequence yields a universal model (if it terminates) since we apply rules exhaustively and we only reduce the number of facts in the *eam* function using endomorphisms. We still need to show that the *eam*-chase sequence yields a core. Then, we immediately obtain that the result of the *eam*-chase sequence is a universal core model if it terminates. As for the core chase, the result of an *eam*-chase sequence is undefined if the sequence

does not terminate.

Theorem 5.12. *Every fact set in an eam-chase sequence $(F^i_{\mathcal{K}})_{i\geq 0}$ is a core.*

Proof. We show that $F^i_{\mathcal{K}}$ is a core for every $i \geq 0$. For $i = 0$, the claim holds, since I does not feature nulls.

For $i = k > 0$, suppose for a contradiction that $F^k_{\mathcal{K}}$ is not a core. We show that there exists an extended alternative match. If $F^k_{\mathcal{K}}$ is not a core then there exists an endomorphism $h : F^k_{\mathcal{K}} \to F^k_{\mathcal{K}}$ that is not surjective since $F^k_{\mathcal{K}}$ is finite. Then some null n in $F^k_{\mathcal{K}}$ does not occur in $h(F^k_{\mathcal{K}})$. Hence, there exists a trigger $\lambda \in \Lambda$ for that n occurs in F_λ and h is a homomorphism from F_λ to $F^k_{\mathcal{K}}$ such that n does not occur in $h(F_\lambda)$. But then, h is already an extended alternative match of h itself. This is a contradiction to the assumption that $F^k_{\mathcal{K}}$ is a fact set in an *eam*-chase sequence. □

Note that similar to Remark 2.18 for the core chase, we only define the eam chase for instances and not for arbitrary possibly infinite fact sets. This is why we assume that every fact set in $(F^i_{\mathcal{K}})_{i\geq 0}$ is finite in the proof. By Theorem 5.12, we also know that Algorithm 1 produces a core and that the (unique) *eam*-chase sequence of a knowledge base $\mathcal{K}$ is equivalent to the (unique) *core*-chase sequence of $\mathcal{K}$, i.e. for each step in the sequences the produced fact sets are isomorphic.

Corollary 5.13. *Consider a knowledge base $\mathcal{K}$. The eam-chase on $\mathcal{K}$ terminates (yielding a universal core model) iff the core-chase on $\mathcal{K}$ terminates (yielding a universal core model).*

In essence, the only difference to the core chase is that the *eam*-chase gives a more precise algorithm for intermediate core computation using a heuristic that specifies where to start searching for homomorphisms that remove nulls (Algorithm 1). Once no such homomorphism exists anymore, the algorithm terminates and we are guaranteed to have a core. It still remains to evaluate how the *eam*-chase compares to other possible implementations of the *core*-chase in practice. Also, it is probably worthwhile to investigate the problems that we discussed in the beginning of this subsection, e.g. Example 5.8, in further detail since they may lead to improvements for Algorithm 1.

5.2.2 Partial Core-Stratification and the Hybrid Chase

Beside considering the core computation within a single chase step, we can also influence the application order of rules. Even for rule sets that are not core-stratified, there may be many rules that do not occur in their own downward closure. We can think of such rule sets as "partially core-stratified".

We introduce a relaxed variant of restrained partitionings from Chapter 4. We aim to be able to use the restricted chase on all rule sets of the partitioning except for the last one where we can fall back to the eam/core chase.

Definition 5.14 (Relaxed Restrained Partitioning)**.** Consider a rule set R. A *relaxed restrained partitioning* of R is a list of sets $R_1, \ldots, R_n$ that form a partitioning of R (i.e. R is the disjoint union $R_1 \dot\cup \ldots \dot\cup R_n$) such that for every $1 \leq i \leq n-1$ and rule ρ we have that if $\rho \in R_i$ then $(\bigcup_{i \leq j \leq n} R_j) \cap \rho{\downarrow^{\square}} = \emptyset$.

The only difference to restrained partitionings is that in a relaxed restrained partitioning, rules may occur together in the last rule set even if they restrain each other. In particular, every rule set R can be seen as a relaxed restrained partitioning (consisting of the single rule set R).

Remark 5.15. *To obtain a more useful relaxed restrained partitioning for a rule set R, we order the rules in R according to their downward closures. We inductively construct such a relaxed restrained partitioning for R as follows:*

- $R_1 \coloneqq \{\, \rho \in R \mid \rho{\downarrow^{\square}} = \emptyset \,\}$
- *If* $\{\, \rho \in R \setminus \bigcup_{1 \leq j \leq i} R_j \mid \rho{\downarrow^{\square}} \subseteq \bigcup_{1 \leq j \leq i} R_j \,\} \neq \emptyset$*, then* $R_{i+1} \coloneqq \{\, \rho \in R \setminus \bigcup_{1 \leq j \leq i} R_j \mid \rho{\downarrow^{\square}} \subseteq \bigcup_{1 \leq j \leq i} R_j \,\}$*. Otherwise,* $R_{i+1} \coloneqq R \setminus \bigcup_{1 \leq j \leq i} R_j$*.*

For a core-stratified rule set, this construction yields a restrained partitioning as in the proof of Lemma 4.3.

The transfinite chase on a relaxed restrained partitioning that uses only restricted chase sequences does not necessarily produce a core. This is expected since we already noted that every rule set can be seen as a relaxed restrained partitioning and then this essentially means that the restricted chase and the eam/core chase coincide for every rule set. Still, we can use the eam/core chase in the last step of the transfinite chase sequence to obtain universal core models. We refer to this modified transfinite chase as the *hybrid chase*.

Consider a relaxed restrained partitioning $\mathfrak{R} = \langle R_1, \ldots, R_n \rangle$. We denote a *hybrid chase sequence* analogously to transfinite chase sequences (Definition 4.5) by $\mathfrak{H}^0_{\langle \mathfrak{R}, I \rangle}, \ldots, \mathfrak{H}^n_{\langle \mathfrak{R}, I \rangle}$ and its result by $\mathfrak{H}^\infty_{\langle \mathfrak{R}, I \rangle} \coloneqq \mathfrak{H}^n_{\langle \mathfrak{R}, I \rangle}$. For the i-th sequence in the hybrid chase, we use the restricted chase if $i \leq n-1$. For the last sequence in the hybrid chase, we use the eam/core chase but we treat nulls that have been introduced before as constants, i.e. the nulls that have been introduced before the last sequence are always mapped to themselves during the core computation. In case of an *eam*-chase sequence, this means that the list of triggers Λ is initially empty. The aim of this is to shrink the search space for the core computation in the last sequence of the hybrid chase. We show later on that this adjustment does not break core computation.

Note that the result of a hybrid chase sequence is only well defined if the last *eam/core*-chase sequence terminates since the result of non-terminating *eam/core*-chase sequences is not defined. Furthermore, the last *eam/core*-sequence of the hybrid chase sequence itself is only well defined if $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ is finite because the eam/core chase only supports finite starting fact sets (Remark 2.18).

Example 5.16. *We show a hybrid chase sequence for a relaxed restrained partitioning of the rule set R from Example 1.1. Note that this rule set is actually core-stratified and we can thus even find a restrained partitioning, e.g. according to Remark 5.15, but to show the idea of the hybrid chase with our main example, we pick another relaxed restrained partitioning. We partition R into $R_1 \coloneqq \{ \rho_3 \}$ and $R_2 \coloneqq \{ \rho_1, \rho_2 \}$. Since $\rho_2 \in \rho_1{\downarrow^\square}$, the partitioning $\mathfrak{R} \coloneqq \langle R_1, R_2 \rangle$ is not a restrained partitioning but still a relaxed restrained partitioning. For the instance I from Example 1.1, we present parts of a hybrid chase sequence in Figure 5.2. Since the last step in the hybrid chase sequence does not terminate, its result is formally not defined. We show the first step $\mathfrak{H}^1_{\langle \mathfrak{R}, I \rangle}$ in Figure 5.2b, which is the fact set that results from I by applying ρ_3 a single time. In the second step of the hybrid chase sequence, the eam/core-chase sequence applies all rules in parallel. Essentially, the second step of the hybrid chase sequence behaves similarly to the core chase in Figure 2.15 but every fact set in this eam/core-chase sequence has a "dangling" null that is introduced by ρ_1. We sketch the first two steps of the eam/core-chase sequence in Figures 5.2c, 5.2d, and 5.2e.*

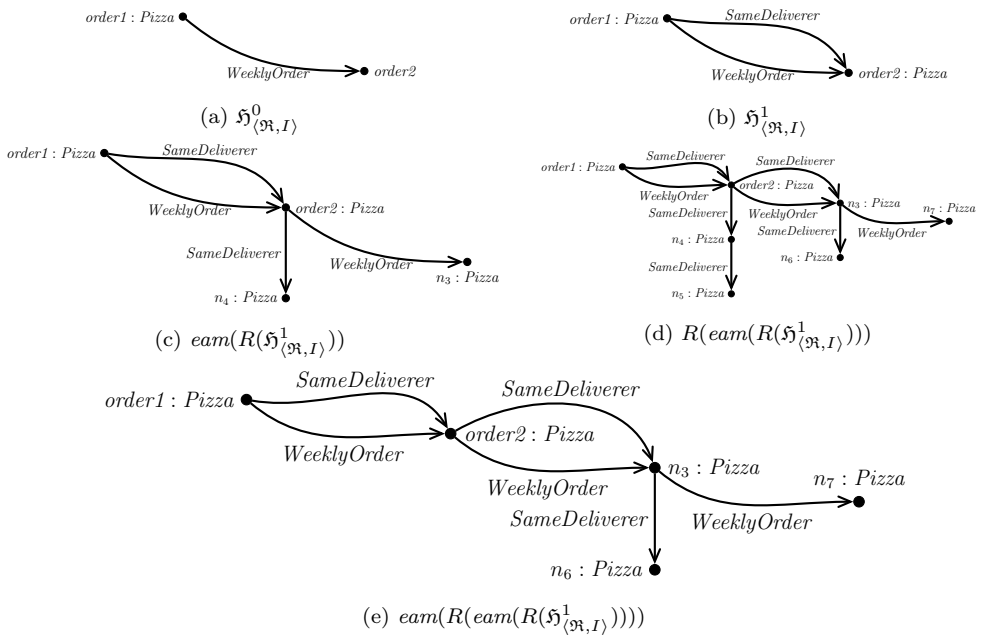


Figure 5.2: Possible Hybrid Chase for Knowledge Base in Example 1.1

We show in the following that the hybrid chase yields a finite universal core model iff one exists. Before the last *eam/core*-chase sequence, the hybrid chase is essentially a transfinite chase on a restrained partitioning so by Lemma 4.8, the restricted chase sequences in the hybrid chase do not feature alternative matches. By Lemma 4.10, there even exists a fair *res*-chase sequence that is equivalent to the hybrid chase part before the last sequence.

Also, the last *eam/core*-chase sequence cannot introduce alternative matches for any trigger that is applied before the last *eam/core*-chase sequence by a similar argument as for Lemma 4.6. We show in the following lemma that this also implies that homomorphisms that potentially yield cores do not need to remap nulls that are introduced before the last *eam/core*-chase sequence.

Lemma 5.17. *Consider a knowledge base $\mathcal{K} \coloneqq \langle R, I \rangle$, a res-chase sequence $F^0_{\mathcal{K}}, F^1_{\mathcal{K}}, \ldots$, and a fact set $F \supseteq F^\infty_{\mathcal{K}}$. If no trigger in the chase sequence has an alternative match w.r.t. F, then there exists a core C of F with $F^\infty_{\mathcal{K}} \subseteq C$.*

Proof. Suppose for a contradiction that every core C of F does not subsume $F^\infty_{\mathcal{K}}$. For every homomorphism h from F to C there is a null in $F^\infty_{\mathcal{K}}$ that is not mapped to itself by h.

We assume w.l.o.g. that for any finite set of nulls N, the function $h_N : N \to N, n \mapsto h(n)$ is not a bijective (non-identity) renaming of nulls. In such a case, we can pick another homomorphisms that uses the identify mapping on N. This is because $h(F) \subseteq C \subseteq F$ and thus $h(\dots h(F) \dots) \subseteq C$.

There exists a largest $j \geq 0$ such that all nulls in $F^j_{\mathcal{K}}$ are mapped to themselves by h, i.e. $F^{j+1}_{\mathcal{K}}$ is the first fact set where some null is not mapped to itself by h. Hence, since we rule out bijective (non-identity) renaming, some null that is freshly introduced in $F^{j+1}_{\mathcal{K}}$ does not occur in $h(F^{j+1}_{\mathcal{K}})$. Thus, h is an alternative match for the trigger that yields $F^{j+1}_{\mathcal{K}}$ w.r.t. C. This implies that h is also an alternative match w.r.t. $F \supset C$ and by that we obtain the desired contradiction.

□

Note that this result rules out that an issue like in Example 5.8 occurs for some null that is introduced before the last *eam/core*-chase sequence during the hybrid chase. We show that the hybrid chase yields a finite universal core model iff one exists.

Theorem 5.18. *Consider a relaxed restrained partitioning $\mathfrak{R} = \langle R_1, \dots, R_n \rangle$ of a rule set R and an instance I. There exists a terminating hybrid chase sequence $\mathfrak{H}^0_{\langle \mathfrak{R}, I \rangle}, \dots, \mathfrak{H}^n_{\langle \mathfrak{R}, I \rangle}$ (yielding a finite universal core model) iff $\langle R, I \rangle$ has a finite universal (core) model.*

Proof. Consider the hybrid chase sequence $\mathfrak{H}^0_{\langle \mathfrak{R}, I \rangle}, \dots, \mathfrak{H}^n_{\langle \mathfrak{R}, I \rangle}$. Recall that by Lemma 4.8, $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ does not feature alternative matches since up until $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ the hybrid chase is essentially a transfinite chase on a restrained partitioning.

Assume at first that $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ is finite. Recall that we assume that the *eam/core*-chase sequence treats nulls from $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ as constants, i.e. the nulls from $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ are always mapped to themselves during the core computation.

($\star$) We show that every fact set F in the last *eam/core*-chase sequence is a core that subsumes $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ such that no trigger that is applied up until $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$ has an alternative match w.r.t. F. We show the claim via induction over the last *eam/core*-chase sequence $F_0, F_1, \dots$. For the base of the induction, we have $F_0 = \mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I \rangle}$, which is a core (and does not feature alternative matches) since the hybrid chase is a transfinite chase on a restrained partitioning up

until $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$. For the induction step, from k to $k+1$, we assume that the claim holds for F_k. We have that $R(F_k) \supseteq F_k$ does not introduce alternative matches for triggers up until $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$ since $\mathfrak{R}$ is a relaxed restrained partitioning. By Lemma 5.17, some core C of $R(F_k)$ subsumes $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$. But then there exists an endomorphism on $R(F_k)$ that yields this core and maps nulls in $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$ to themselves. Hence such a core is found by the functions *eam* or *core*, respectively. This concludes the proof of $(\star)$.

We show both directions of the theorem for the case that $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$ is finite. If $\mathfrak{H}^{0}_{\langle \mathfrak{R}, I\rangle}, \ldots, \mathfrak{H}^{n}_{\langle \mathfrak{R}, I\rangle}$ is terminating, then the last *eam/core*-chase sequence is terminating and thus yields a finite universal model C of $\langle R, I\rangle$. By $(\star)$, C is a core and thus a finite universal core model of $\langle R, I\rangle$. If a finite universal core model C of $\langle R, I\rangle$ exists, then the last *eam/core*-chase sequence only requires a finite amount of steps to obtain a fact set F that contains all facts in C (possibly with renamed nulls). By $(\star)$, F is a universal core model.

It remains to show that no finite universal (core) model of $\langle R, I\rangle$ exists if $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$ is infinite. In this case, we can obtain an infinite universal model U for $\langle R, I\rangle$ with a *res*-chase sequence on $\langle R, \mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}\rangle$. Since $\mathfrak{R}$ is a relaxed restrained partitioning, all triggers that are applied up until $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle}$ do not have alternative matches w.r.t. U. By Lemma 5.17, there exists a core C of U with $\mathfrak{H}^{n-1}_{\langle \mathfrak{R}, I\rangle} \subseteq C$. Thus, C is infinite. Since C is an infinite universal core model of $\langle R, I\rangle$, no finite universal model for $\langle R, I\rangle$ exists by Lemma 2.6. □

In practice, for a given knowledge base, we can compute the chase according to a hybrid chase sequence using a relaxed restrained partitioning that orders rules according to their downward closures as in Remark 5.15. For the last step of the hybrid chase, we can make use of eam-chase as a heuristic for the core chase, i.e. we use extended alternative matches to find endomorphisms that yield a core. The procedure that we create by combining both of these ideas promises to be feasible in practice for many rule sets.

Chapter 6

Conclusion

After elaborate considerations about chase termination for core-stratified rule sets in Chapter 4 and investigations of more practical aspects of core computation for rule sets that are not core-stratified in Chapter 5, we lay out an encompassing procedure for computing universal core models of knowledge bases in Figure 6.1. In this chapter, we summarize our main results and stress interesting open problems that can be addressed in future work.

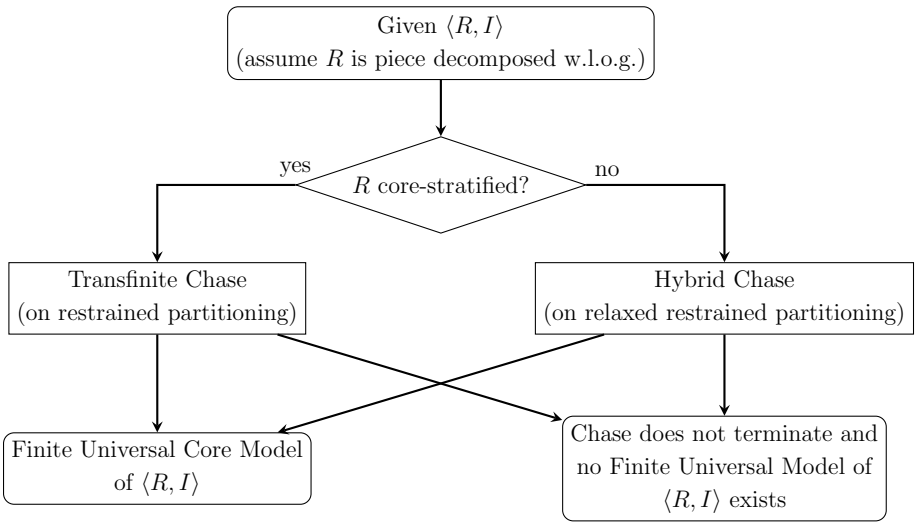


Figure 6.1: Comprehensive Procedure for Core Computation

6.1 Summary of Results

We give a recap about the theoretical main results of our thesis as well as the practical implications.

The most vital theoretical result of our work is that $\mathsf{CT}^{res}_{\forall\exists}$ and $\mathsf{CT}^{core}_{\forall}$ coincide for core-stratified rule sets according to Theorems 4.1 and 4.13. One main part of the proof is based on a framework that we call the transfinite chase that simplifies considerations for possibly unfair and possibly infinite chase sequences. In practice, this means that we can compute a universal core model with the restricted chase using a rule application order according to restraining relations given that a rule set is core-stratified. This computation terminates iff a finite universal (core) model exists (Corollary 4.14). Especially, we do not have to take fairness into account when computing the transfinite chase on a restrained partitioning because we know that a fair equivalent *res*-chase sequence exists according to Lemma 4.10. Another implication of this result is that RMFC is a sufficient condition for core chase non-termination for core-stratified rules or in other words, for a core-stratified rule set, RMFC is a sufficient condition for when there exists an instance such that the knowledge base that consists of the rule set and the instance does not have a finite universal model (Corollary 4.15).

Furthermore, we establish that for rule sets without restraining relations $\mathsf{CT}^{res}_{\forall\exists}$, $\mathsf{CT}^{res}_{\forall\forall}$, and $\mathsf{CT}^{core}_{\forall}$ coincide in Theorem 4.16. This result is especially interesting if we have single-head guarded existential rules since this special case of $\mathsf{CT}^{res}_{\forall\forall}$ is known to be decidable. This yields that core chase termination is decidable for single-head guarded existential rules without restraining relations (Corollary 4.17).

We also show a slightly more general version of the Fairness Theorem from Gogacz et al. [16] in Theorem 4.26. It is interesting to see that the idea of restraining is rather close to the idea of fairness and we conjecture that this yields a slightly bigger fragment of guarded rules for which restricted chase termination is decidable, namely guarded rules that do not feature 2-restraining (or strong-restraining) relations.

In cases where rule sets are not core-stratified, we can make use of (extended) alternative matches as a heuristic for the computation of cores for concrete

fact sets in practice, which we formalize with the eam chase while pointing out limitations of this approach. Furthermore, we introduce the hybrid chase as a mixed version of restricted and eam/core chase in spirit of the transfinite chase that makes use of restraining relations in relaxed restrained partitionings. This approach is especially promising when a rule set is "partially core-stratified" i.e. when it contains only few rules that occur in their own downward closure.

6.2 Open Questions and Future Work

Our research formulates new questions and stresses some existing open problems that may immediately yield useful corollaries depending on how they are resolved.

One interesting question is the decidability of the existence of alternative matches for guarded existential rule sets. If this is decidable, we do not need to consider core-stratification for guarded existential rules. Also, in cases where we know that the existence of alternative matches is undecidable, tighter conditions than restraining and core-stratification can probably be found since these notions are still an over-approximation.

A more intriguing questions that is also mentioned in particular by Gogacz et al. [16] is the decidability of $\mathsf{CT}^{res}_{\forall\exists}$ for (single-head) guarded existential rule sets. If this is found to be decidable, then this immediately also decides $\mathsf{CT}^{core}_{\forall}$ for core-stratified (single-head) guarded existential rule sets. For now, we only know that $\mathsf{CT}^{core}_{\forall}$ is decidable for single-head guarded existential rule sets without restraining relations but probably most "real-world" rule sets have restraining relations while potentially still being core-stratified. It also remains open to verify Conjecture 4.28, which states that the result by Gogacz et al. [16] can be extended to arbitrary guarded existential rules without strong restraining relations.

Another general issue is the case for when a rule set may have alternative matches, even if we could decide both the existence of alternative matches and restricted chase termination. If alternative matches occur, a universal core model may still exist but we do not necessarily find it using the restricted chase. Here we can fall back to the eam/core chase or the hybrid chase. Still,

it is interesting to investigate if this gap can be closed in another way.

Last but not least, our ideas for potentially more efficient computation of universal core models like the hybrid chase need to be evaluated in practice. If rule sets are "partially core-stratified", i.e. if few rules occur in their own downward closure, then the hybrid chase promises significant practical improvements. This involves an efficient implementation of the computation of cores during the last hybrid chase step. We layed out a heuristic for the core chase in form of the eam chase and it remains to evaluate in how far (extended) alternative matches can improve the performance of this computation in comparison to other possible implementations of the core chase. The key issues that we pointed out for the intuitive approach of the eam chase also motivate further theoretical investigation that may lead to improvements for the procedure.

In summary, our work encourages further research regarding restricted and core chase termination, especially for guarded existential rules and computation of universal core models in general. Our findings also enable new practical evaluations and implementations of procedures that compute universal core models for arbitrary rule sets. We think that for many rule sets our ideas and in particular our comprehensive procedure in Figure 6.1 can achieve good results in practice.

Bibliography

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

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

[3] Bárány, V., Gottlob, G., and Otto, M. Querying the guarded fragment. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom* (2010), IEEE Computer Society, pp. 1–10.

[4] Barceló, P., and Calautti, M., Eds. *22nd International Conference on Database Theory, ICDT 2019, March 26-28, 2019, Lisbon, Portugal* (2019), vol. 127 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[5] Bauslaugh, B. L. Core-like properties of infinite graphs and structures. *Discret. Math. 138*, 1-3 (1995), 101–111.

[6] Beeri, C., and Vardi, M. Y. The implication problem for data dependencies. In *Automata, Languages and Programming, 8th Colloquium, Acre (Akko), Israel, July 13-17, 1981, Proceedings* (1981), S. Even and O. Kariv, Eds., vol. 115 of *Lecture Notes in Computer Science*, Springer, pp. 73–85.

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

[8] Benedikt, M., Konstantinidis, G., Mecca, G., Motik, B., Papotti, P., Santoro, D., and Tsamoura, E. Benchmarking the chase. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017* (2017), E. Sallinger, J. V. den Bussche, and F. Geerts, Eds., ACM, pp. 37–52.

[9] Calautti, M., Gottlob, G., and Pieris, A. Chase termination for guarded existential rules. In *Proceedings of the 34th ACM Symposium on Principles of Database Systems, PODS 2015, Melbourne, Victoria, Australia, May 31 - June 4, 2015* (2015), T. Milo and D. Calvanese, Eds., ACM, pp. 91–103.

[10] Carral, D., Dragoste, I., and Krötzsch, M. Restricted chase (non)termination for existential rules with disjunctions. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017* (2017), C. Sierra, Ed., ijcai.org, pp. 922–928.

[11] Carral, D., Krötzsch, M., Marx, M., Ozaki, A., and Rudolph, S. Preserving constraints with the stable chase. In *21st International Conference on Database Theory, ICDT 2018, March 26-29, 2018, Vienna, Austria* (2018), B. Kimelfeld and Y. Amsterdamer, Eds., vol. 98 of *LIPIcs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 12:1–12:19.

[12] Deutsch, A., Nash, A., and Remmel, J. B. The chase revisited. In *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008, June 9-11, 2008, Vancouver, BC, Canada* (2008), M. Lenzerini and D. Lembo, Eds., ACM, pp. 149–158.

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

[14] Fagin, R., Kolaitis, P. G., and Popa, L. Data exchange: getting to the core. In *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 9-12,*

2003, San Diego, CA, USA (2003), F. Neven, C. Beeri, and T. Milo, Eds., ACM, pp. 90–101.

[15] Gogacz, T., and Marcinkowski, J. All-instances termination of chase is undecidable. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II* (2014), J. Esparza, P. Fraigniaud, T. Husfeldt, and E. Koutsoupias, Eds., vol. 8573 of *Lecture Notes in Computer Science*, Springer, pp. 293–304.

[16] Gogacz, T., Marcinkowski, J., and Pieris, A. All-instances restricted chase termination. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020* (2020), D. Suciu, Y. Tao, and Z. Wei, Eds., ACM, pp. 245–258.

[17] Gottlob, G., and Nash, A. Efficient core computation in data exchange. *J. ACM 55*, 2 (2008), 9:1–9:49.

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

[19] Grau, B. C., Horrocks, I., Krötzsch, M., Kupke, C., Magka, D., Motik, B., and Wang, Z. Acyclicity notions for existential rules and their application to query answering in ontologies. *J. Artif. Intell. Res. 47* (2013), 741–808.

[20] Hell, P., and Nesetril, J. The core of a graph. *Discret. Math. 109*, 1-3 (1992), 117–126.

[21] Krötzsch, M. Computing cores for existential rules with the standard chase and ASP. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020* (2020), D. Calvanese, E. Erdem, and M. Thielscher, Eds., pp. 603–613.

[22] Krötzsch, M., Marx, M., and Rudolph, S. The power of the terminating chase (invited talk). In Barceló and Calautti [4], pp. 3:1–3:17.

[23] Leclère, M., Mugnier, M., Thomazo, M., and Ulliana, F. A single approach to decide chase termination on linear existential rules. In Barceló and Calautti [4], pp. 18:1–18:19.

[24] Maier, D., Mendelzon, A. O., and Sagiv, Y. Testing implications of data dependencies. *ACM Trans. Database Syst. 4*, 4 (1979), 455–469.

[25] Pichler, R., and Savenkov, V. Demo: Data exchange modeling tool. *Proc. VLDB Endow. 2*, 2 (2009), 1606–1609.