Implementing the Core Chase for the Description Logic ALC

Maël Abily

July 21, 2021

1 Introduction

The conjunctive query entailment is an important issue in database. This problem can be described in a first order logic background. We work on conjunctive formulas (that are formulas constructed only with conjunction and existential quantification) and queries that are conjunctive formulas without free variables; the answer of a query is either yes or no. For example $\exists y.Human(y) \wedge IsTheBrotherOf(Pierre, y)$ is a query asking if Pierre has a parent. We also work on existential rules that are formulas of the form $\forall \vec{x}.\forall \vec{y}.(A(\vec{x}, \vec{y}) \rightarrow \exists \vec{z}.B(\vec{x}, \vec{z}))$ where $\vec{u}$ represents a tuple of variables.

We can now define the conjunctive query entailment: Given a set T of existential rules and conjunctive formulas, and given a query Q determine if the set T entails the query Q. For example, the set $T = \{IsTheBrotherOf(Pierre,Marie), \forall x \forall y.IsTheBrotherOf(x, y) \rightarrow Human(x) \wedge Human(y)\}$ entails the query $Q = \exists y.Human(y) \wedge IsTheBrotherOf(Pierre, y)$.

We usually use reasoning algorithms to answer the conjunctive query entailment.Note that in practice, a lot of formulas can be express via conjunctive formulas. By definition, a set T of existential rules and conjunctive formulas entails a query Q if every model of T is a model of Q. It is not practical because a knowledge base often have an infinite number of models.

To deal with this problem, we can compute an universal model of the set T. That is a model of T that is entailed by all the models of T. If a such model U exists, we just need to show that U entails Q to conclude that T entails Q. Hence, to solve query entailment, we just have to compute a finite universal model U for a given input T and look if U entails Q.

To compute these models, we can use algorithms called the chase. We will present in this document the oblivious chase, the restricted chase and then the core chase. The last chase is the best in the sense of it terminates if and only if there exists a finite universal model.

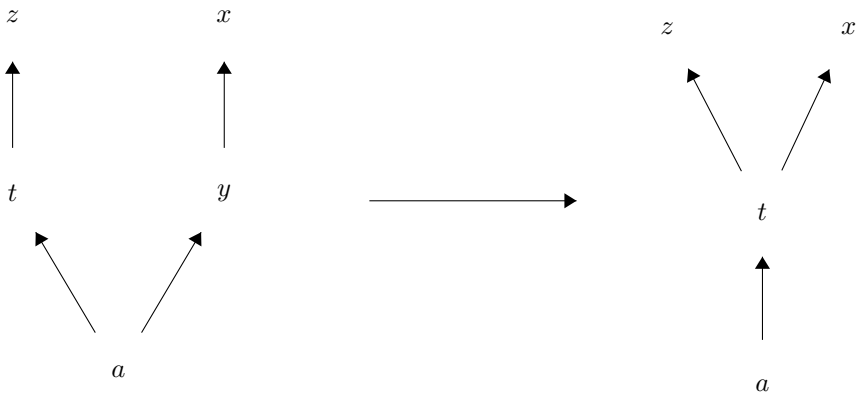

Figure 1: Merging Example

Nevertheless, it is the slowest so it it is never used for practical applications. We will then focus on a restricted type of set T where the set of conjunctive formulas is ground (that is there is no free-variable in the formula) and where the rules contained in T are **Horn-$\mathcal{ALCH}$** axioms. It is an interesting restriction because we can represent the results of the chase by a tree.

We will create a quicker chase for this restriction, that we called the merge chase, that would produce the same output than the core chase. It is based on the idea that in a tree, there exists a sibling relation between the nodes (that is two nodes are siblings if they have the same father) and we will see that, in some conditions, a sibling of a node can be merged on this node like in the Figure 1 where y is a sibling of t and is merged on it. We will then try to extend the merge chase on other type of knowledge bases.

Contents

First of all, we will define the usefull notions for this paper and give some well known properties. Then, we will move on our contributions.

2 Background

2.1 Facts

2.1.1 Syntax

We consider a set of variables **Vars**, a set of constants **Csts**, and a set of predicates **Preds**. **Csts**, **Vars**, and **Preds** are pairwise disjoint. A *term* is a variable or a constant. We note **Terms** the set of terms. If $t_1, \ldots, t_n$ are terms and P is a predicate of arity n, then $P(t_1, \ldots, t_n)$ is an *atom*. The atom $P(t_1, \ldots, t_n)$ is *ground* if $t_1, \ldots, t_n$ are constants. We can now define a factbase that is a notion omnipresent in all the paper.

Definition 2.1. A *factbase* F is an existentially closed conjunction of atoms, that is, a formula that does not contain occurrences of free variables and is of the form $\exists x_1, \ldots, x_n . P_1(t_1^1, \ldots, t_{k_1}^1) \wedge \ldots \wedge P_m(t_1^m, \ldots, t_{k_m}^m)$ where t_i^j are terms and P_i are predicates. A factbase is *ground* if each of its atoms is ground.

Note that a ground factbase does not contain any existential quantifiers. In some articles, the factbases are always considered as ground but in this document, we consider factbases that may not be ground. Consequently, a boolean conjonctive query will be a factbase, so we will only talk about factbases and not introduce the notion of query.

For convenience, we identify factbases as sets of atoms, which allows us to use set notions such as set inclusion. For example, we identify the factbase

$$\exists x, x_1, x_2, x_3 . P(x) \wedge Q(x, a) \wedge R(x_1, x_2, x_3, b)$$

with the set of facts

$$\{P(x), Q(x, a), R(x_1, x_2, x_3, b)\}$$

For a formula A, let $\boldsymbol{Vars}(A)$, $\boldsymbol{Csts}(A)$, and $\boldsymbol{Terms}(A)$ be the sets of variables, constants, and terms that occur in A, respectively.

Definition 2.2. A factbase F *entails* another factbase G (often noted $F \models G$) if each interpretation satisfying F satisfies G. F is equivalent to G if $F \models G$ and $G \models F$.

Intuitively, if $F \models G$, then F is logically stronger than G and from F, we can deduce G.

2.1.2 Homomorphism

A *substitution* $\sigma : X \to \mathbf{Terms}$ is a function where X is a set of variables. For example $\{x \mapsto z, y \mapsto a\}$ is a substitution from $\{x, y\}$ to **Terms**. By extension:

- if $c \in \mathbf{Csts}$, then $\sigma(c) = c$;
- if $x \in \mathbf{Vars} \setminus X$, $\sigma(x) = x$;
- if $f = P(t_1, \ldots, t_n)$ is an atom, then $\sigma(f) = P(\sigma(t_1), \ldots, \sigma(t_n))$; and
- if $F = \{f_1, \ldots, f_n\}$ is a factbase, then $\sigma(F) = \{\sigma(f_1), \ldots, \sigma(f_n)\}$.

This definition leads us to the central notion of Homomorphism.

Definition 2.3. For two factbases F and G, a *homomorphism* from F to G is a substitution $\sigma : \mathbf{Vars}(F) \to \mathbf{Terms}$ where $\sigma(F) \subseteq G$.

Sometimes, we will say that we *map* a variable x to a term t if $\sigma(x) = t$. An *isomorphism* h from F to G is a bijective homomorphism where its inverse is a homomorphism from G to F. For the remainder of this paper, we identify sets of facts that are unique up to isomorphism.

In practice, to know if $F \models G$, we use the following Theorem.

Theorem 2.1 (Homomorphism Theorem)**.** A factbase F *entails* another factbase Q if and only if there exists a homomorphism from Q to F.

The previous theorem has been proved in ([1],theorem 6.2.3). For example, the factbase $F = \{P(b, a), A(x)\}$ entails the factbase $Q = \{P(x, a), P(y, z)\}$ due to the homomorphism $\{x \mapsto b, y \mapsto b, z \mapsto a\}$.

2.1.3 Core

For a factbase F, let $id_{|F}$ be the substitution mapping each variable in $\mathbf{Vars}(F)$ to itself. And for a subsitution σ defined on a factbase G containing F, let $\sigma_{|F}$ be the subsitution σ defined only on $\mathbf{Vars}(F)$. In practice, it is better to work on smaller factbases. It leads us to the notions of retracts and cores:

Definition 2.4. A factbase G is a *retract* of another factbase F if $G \subseteq F$ and $G \models F$. A *retractation* from F to G is a homomorphism σ from F to G such that $\sigma_{|G} = id_{|G}$. G is a *strict retract* of F if G is a retract of F and $G \neq F$.

If G is a rectract of F, then G is as strong as F and smaller. We impose that $\sigma_{|G} = id_{|G}$ in the definition of retractations because it helps us in the proof of Proposition 3.6 thanks to the well known property:

Proposition 2.1. *The factbase G is a retract of the factbase F if and only if $G \subseteq F$ and there exists a retractation from F to G.*

A core of a factbase F is then one of the smaller retract of F:

Definition 2.5. If a factbase F does not contain a strict retract, then we say that F is a *core*. A *core* of a factbase F (noted *core*(F)) is a subset of F that is a core.

The cores of a finite factbase F are unique up to isomorphism. Hence, we speak of "the" core of a factbase.

The factbase $G = \{B(x, y), R(y, z)\}$ is the core of $F = \{B(x, y), R(y, z), B(x, w), R(w, z)\}$ because:

- $G \subseteq F$;
- $\{x \mapsto x, y \mapsto y, z \mapsto z, w \mapsto y\}$ is a homomorphism from F to G, so G is a retract of F;
- all strict subsets of G are not retracts of G.

Note that $\{B(x, w), R(w, z)\}$ is also the core of F and is indeed isomorphic to G due to the homomorphism $\{x \mapsto x, y \mapsto w, z \mapsto z\}$.

2.2 Existential Rules

An existential rule is a formula of the form $A \to B$, more exactly:

Definition 2.6. Let $\vec{x}$, $\vec{y}$, and $\vec{z}$ be some tuples of variables that are pairwise disjoint. An *(existential) rule* α is a first-order formula of the form

$$\forall \vec{x}.\forall \vec{y}.(A(\vec{x}, \vec{y}) \to \exists \vec{z}.B(\vec{x}, \vec{z}))$$

where A and B are conjunctions of atoms. We define *body*(α) $= A$ and *head*(α) $= B$.

We omit the universal quantifiers when representing existential rules. A knowledge base represents then a set of rules and a set of facts:

Definition 2.7. A *knowledge base* K is a pair (R, F) where R is a set of existential rules and F is a ground factbase.

A factbase F *entails* a rule α if each interpretation satisfying F satisfies α. We will note $F \models R$ if F entails each rule of the rule set R. The following theorem is use in pratice to determine if $F \models \alpha$.

Theorem 2.2. A factbase F *entails* a rule $\alpha = A(\vec{x}, \vec{y}) \rightarrow \exists \vec{z}.B(\vec{x}, \vec{z})$ if and only if for every homomorphism σ from A to F, there exists an extension of σ that is a homomorphism from B to F.

Definition 2.8 (Model and universality)**.** A factbase M is a *model* for a knowledge base $K = (R, F)$ if $M \models F$ and $M \models R$. The factbase U is *universal* for K if for every model M of K, $M \models U$. The factbase M is an *universal model* for K if M is a model for K and M is universal for K.

For example, if we pose $K = (\{\alpha = A(x) \rightarrow \exists z.R(x, z) \wedge A(z)\}, \{A(b)\})$, then an universal model for K is

$$U = \{A(b), R(b, x_0)\} \cup \{A(x_i) \mid i \in \mathbb{N}\} \cup \{R(x_i, x_{i+1}) \mid i \in \mathbb{N}\}$$

Note that the knowledge base K does not admit finite universal models.

Definition 2.9 (Entailment)**.** A knowledge base K *entails* a factbase B (often noted $K \models B$) if for each model M of K, $M \models B$.

We can use universal models to solve the conjunctive query entailment:

Proposition 2.2. *A knowledge base K entails a factbase B if there exists a universal model U for K such that $U \models B$.*

An important problem that this document has to deal with is: Given a knowledge base $K = (R, F)$ and a factbase Q, does $K \models Q$? It is well-known that this problem is undecidable ([2], theorem 4).

2.3 The Chase

The process of applying rules on a factbase in order to infer more knowledge is called forward chaining. Forward chaining in existential rules is usually achieved via a family of algorithms called the chase. It can be seen as a two-steps process. It first repeatedly applies rules to the set of facts (and eventually computes sometimes the core to supress redundant facts). Then it looks for an answer to the query in this saturated set of facts. This saturated set of facts is an universal model of the knowledge base. The chase is sound and complete; so it must be non-terminating since the problem of entailment is undecidable. To determine how we apply a rule to a set of fact, we introduce the notion of trigger:

Definition 2.10 (Trigger)**.** Let T be a rule set, α be a rule, σ be a substitution, and F be a factbase. The tuple $t = (\alpha, \sigma)$ is an *oblivious trigger* for F if:

- the domain of σ is the set of all variables occurring in $Body(\alpha)$.
- σ is a homomorphism from $Body(\alpha))$ to F.

In this case, we say that t is *applicable* on F.

The tuple $t = (\alpha, \sigma)$ is a *restricted trigger* for F if t is an oblivious trigger for F and if for all $\hat{\sigma}$ that extend σ over $\mathbf{Vars}(Head(\alpha))$, $\hat{\sigma}(Head(\alpha)) \nsubseteq F$.

Notice that a restricted trigger is also an oblivious trigger. We will use the term *trigger* to talk about the oblivous and the restricted trigger.

The chase will considere triggers to infer new knowledge from an initial factbase. We explain now how it would apply a trigger, giving rise to the notion of application.

Definition 2.11 (application)**.** For a trigger $t = (\alpha, \sigma)$ for the factbase F where α is of the form $A(\vec{x}, \vec{y}) \rightarrow \exists \vec{z}.B(\vec{x}, \vec{z})$, we pose σ^s the substitution that extends σ over $\mathbf{Vars}(Head(\alpha))$ such that for $z \in \vec{z}$, $\sigma^s(z) = f^z_\alpha(\sigma(\vec{x}))$ where $f^z_\alpha(\sigma(\vec{x}))$ is a fresh variable unique with respect to the trigger t and the variable z. The factbase $\mathbf{appl}(F, t) = F \cup \sigma^s(Head(\alpha))$ is called an *application* on the factbase F through the trigger $t = (\alpha, \sigma)$. We say that the trigger t is *useless* on F if $\mathbf{appl}(F, t) = F$.

For example, if $\alpha = A(x, y) \rightarrow \exists z.B(x, z)$, $F = \{A(b, c)\}$, and $\sigma = \{x \mapsto b, y \mapsto c\}$, then (α, σ) is a restricted trigger for F. We have

$$\mathbf{appl}(F, (\alpha, \sigma)) = \{A(b, c), B(b, f^z_\alpha(b, c)\}$$

A sequence of rule applications is called a derivation. In this paper, we differentiate oblivious derivation and restricted derivation:

Definition 2.12 (Derivation)**.** An *oblivious derivation* (respectively a *restricted derivation*) for a knowledge base $K = (F, R)$ is a (possibly infinite) sequence $D = F_0, t_1, F_1, t_2, F_2, \ldots$ where

- $F_0, F_1, \ldots$ are factbases such that $F_i \subsetneq F_{i+1}$.
- $t_1, t_2, \ldots$ are oblivious triggers (resp. restricted triggers).
- $F_0 = F$.
- For all $i > 0$, $F_i = \mathbf{appl}(F_{i-1}, t_i)$.

A fair derivation garantees that we consider every possible application:

Definition 2.13 (Fairness)**.** The oblivious (resp. restricted) derivation $D = F_0, t_1, F_1, t_2, F_2, \ldots$ is *fair* if for every i and every oblivious (resp. restricted) trigger t for F_i, there exists $k \geq i$ such that t is useless on F_k (resp. t is not anymore a restricted trigger for F_k).

We will now define the oblivious and restricted chase, It is defined in [3].

Definition 2.14. An *oblivious chase* (resp. a restricted chase) for a knowledge base $K = (F, R)$ is a fair oblivious (resp. restricted) derivation $D = F_0, t_1, F_1, t_2, F_2, \ldots$

For every oblivious chase $D = F_0, t_1, F_1, t_2, F_2, \ldots$ for K, we have $F_0 \subseteq F_1 \subseteq F_2 \subseteq \ldots$. T , and the result of an oblivious chase does not depend on the derivation D so we can pose $Obl(K) = \cup_{i\in\mathbb{N}} F_i$. We say that the oblivious chase *terminates* if $Obl(K)$ is finite.

It is well known that:

Theorem 2.3. For a knowledge base K, $Obl(K)$ is an universal model of K.

Consequently, the oblivious chase can be used to solve query entailment. It is more difficult to define the result of the restricted chase because the result depends on the order of the application of the rules. We define

$$Res(K) = \{\cup_{i\in\mathbb{N}} F_i \mid D = F_0, t_1, F_1, t_2, F_2, \ldots \text{is a retricted chase for } K\}$$

We say that the restricted chase *terminates* if there exists an element in $Res(K)$ that is finite. It is well known that:

Theorem 2.4. For a knowledge base K and for every $U \in Res(K)$, U is an universal model of K.

The oblivious chase can do a lot of applications that are useless:

Example 2.1. Suppose that we have the knowledge base $K = (\{\alpha = A(x, y) \rightarrow \exists z.A(y, z)\wedge A(z, y)\}, \{A(a, b)\})$. An oblivious chase derivation is $F_0, t_1, F_1, t_2, F_2, ...$ where

- $F_0 = F$,
- $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$,
- $F_1 = \{A(a, b), A(b, z_1), A(z_1, b)\}$ where $z_1 = f^z_\alpha(a, b)$,
- $t_2 = (\alpha, \{x \mapsto b, y \mapsto z_1\})$,
- $F_2 = F_1 \cup \{A(z_1, z_2), A(z_2, z_1)\}$ where $z_2 = f^z_\alpha(b, z_1)$,
- $\ldots$

It will never terminate because each new atom brings new rule applications. So the oblivious chase does not terminate on K whereas the restricted chase terminates. A restricted chase derivation is F_0, t_1, F_1 where $F_0 = F$, $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$, and $F_1 = \{A(a, b), A(b, z_{t_1}), A(z_{t_1}, b)\}$. This derivation is fair because there is not anymore any restricted trigger for F_1. We have $F_1 \in Res(K)$.

It is well known that:

Theorem 2.5. A knowledge base K entails a factbase B if $Obl(K) \models B$ or if there exists $U \in Res(K)$ such that $U \models B$. Conversely, if a knowledge base K entails a factbase B, then $Obl(K) \models B$ and for every $U \in Res(K)$, $U \models B$

The core chase has been firstly defined in [4].

Definition 2.15 (Core derivation and fairness)**.** A *core derivation* for a knowledge base $K = (R, F)$ is a (possibly infinite) sequence $D = F_0, F_1, F_2, \ldots$ where $F_0 = F$, and for $i > 0$, either $F_i = \mathbf{appl}(F_{i-1}, t_i)$ is obtained by an application with t_i an oblivious trigger, or F_i is the core of F_{i-1}.

the core derivation D is *fair* if:

- For every i, for every oblivious trigger t for F_i, there exists k such that t is useless on F_k.
- For every i, there exists $k \geq i$ such that F_k is a core.

The second condition in the definition of fairness is essential to garantee the terminaison of a fair core derivation when there exists a finite universal model.

Definition 2.16. A *core chase* for a knowledge base $K = (R, F)$ is a fair core derivation $D = F_0, F_1, F_2, \ldots$ The core chase *terminates* on K if it is a finite core derivation.

The article [4] has proven that the result of the core chase for a knowledge base is unique up to isomorphism. Therefore, we can define the result of the core chase:

Definition 2.17. If the core chase terminates on K due to the finite core derivation $D = F_0, F_1, F_2, \ldots, F_i$, then we pose $C(K) = F_i$. Otherwise, if the core chase does not terminate, $C(K)$ is undefined.

The following theorem has been proven in ([4], theorem 7)

Theorem 2.6. The knowledge base $K = (R, F)$ admits a finite universal model if and only if the core chase algorithm terminates on K.

There exists knowledge bases where the restricted chase does not terminate whereas the core chase terminates:

Example 2.2. Suppose that we have the knowledge base $K = (\{\alpha = A(x, y) \rightarrow \exists z.(A(x, x) \wedge A(y, z))\}, \{A(a, b)\})$. A restricted chase derivation is $F_0, t_1, F_1, t_2, F_2, ...$ where

- $F_0 = F$,

- $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$,
- $F_1 = F_0 \cup \{A(a,a), A(b,z_1)\}$ where $z_1 = f^z_\alpha(a,b)$,
- $t_2 = (\alpha, \{x \mapsto b, y \mapsto z_1\})$,
- $F_2 = F_1 \cup \{A(b,b), A(z_1,z_2)\}$ where $z_2 = f^z_\alpha(b,z_1)$,
- $t_3 = (\alpha, \{x \mapsto z_1, y \mapsto z_2\})$,
- $F_3 = F_2 \cup \{A(z_1,z_1), A(z_2,z_3)\}$ where $z_3 = f^z_\alpha(z_1,z_2)$,
- ...

It will never terminate because each new atom brings new restricted triggers. The core chase terminates on K: a core chase derivation is $F_0, F_1, F_2, F_3, F_4, F_5$ where

- $F_0 = F$,
- $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$,
- $F_1 = \mathbf{appl}(F_0, t_1) = F_0 \cup \{A(a,a), A(b,z_1)\}$ where $z_1 = f^z_\alpha(a,b)$,
- $t_2 = (\alpha, \{x \mapsto b, y \mapsto z_1\})$,
- $F_2 = \mathbf{appl}(F_1, t_2) = F_1 \cup \{A(b,b), A(z_1,z_2)\}$ where $z_2 = f^z_\alpha(b,z_1)$,
- $t_3 = \{x \mapsto a, y \mapsto a\}$, $F_3 = \mathbf{appl}(F_2, t_3)$, $t_4 = \{x \mapsto b, y \mapsto b\}$, and $F_4 = \mathbf{appl}(F_3, t_4) = F_2 \cup \{A(a, f^z_\alpha(a,a)), A(b, f^z_\alpha(b,b))\}$
- $F_5 = \mathit{Core}(F_4) = \{A(a,a), A(a,b), A(b,b)\}$.

There is not anymore any oblivious trigger for F_5 so the derivation is fair. Consequently the core chase terminates on K.

3 The Merge Chase

The core chase always terminates when there exists a finite universal model but this chase is very expensive in time because computing the core of a factbase is hard. Therefore we are presenting a more efficient way to solve this problem in the particular case of **Horn-$\mathcal{ALCH}$**.

Definition 3.1. For a factbase F and a term t, let $\boldsymbol{Preds}^1_F(t)$ be the set of unary predicates P such that $P(t) \in F$. For two terms t and u, we note $\boldsymbol{Preds}^2_F(t,u)$ the set of binary predicates P such that $P(t,u) \in F$.

For a factbase F and some $t, u \in \mathbf{Terms}(F)$, let $\mathbf{Preds}^1_F(t) = \{P \mid P(t) \in F\}$ and $\mathbf{Preds}^2_F(t,u) = \{P \mid P(t,u) \in F\}$.

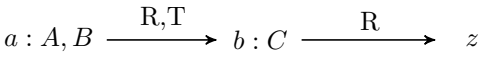

Figure 2: Factbase Representation

In this section, we only consider factbases with predicates of arity one or two. Hence, we will represent a factbase F by a labelled graph $G = (V, E)$ where $V = \{t \mid t \in \mathbf{Terms}\}$ and $E = \{(t, u) \mid t, u \in \mathbf{Terms} \wedge \mathbf{Preds}_F(t, u) \neq \emptyset\}$. We label each vertex $v \in V$ with the predicates in $\mathbf{Preds}^1_F(v)$ and each edge (t, u) with the predicates in $\mathbf{Preds}^2_F(t, u)$. For example, we represent $F = \{A(a), B(a), R(a, b), T(a, b), C(b), R(b, z)\}$ by the Figure 3.

3.1 Horn-$\mathcal{ALCH}$

Horn-$\mathcal{ALCH}$ has been introduced in [5].

Definition 3.2 (**Horn-$\mathcal{ALCH}$** axioms)**.** A ***Horn-$\mathcal{ALCH}$*** *axiom* is an existential rule of one of the following forms:

$$A_1(x) \wedge A_2(x) \rightarrow B(x) \tag{1}$$
$$A(x) \wedge R(x, y) \rightarrow B(y) \tag{2}$$
$$A(x) \rightarrow \exists y.R(x, y) \wedge B(y) \tag{3}$$
$$R(x, y) \wedge B(y) \rightarrow A(x) \tag{4}$$
$$R_1(x, y) \wedge R_2(x, y) \rightarrow S(x, y) \tag{5}$$

A ***Horn-$\mathcal{ALCH}$*** *knowledge base* $K = (R, F)$ is a knowledge base where R is a set of **Horn-$\mathcal{ALCH}$** axioms and F contains only predicates of arity one or two.

Display into 2 columns so it does not take so much vertical space. I tried but I do not succeed to do it with the package multicol

Note that the type 3 is the only type of rule that features existentially quantified variables. To introduce the merge chase, we first define the notion of mergeable variable. We then describe the atomic merging operation that merges a mergeable variable on a term. This atomic operation is used to define the merging that does atomic merging until it is not possible anymore. The merge chase is then a core chase where we replace the core operation by the merging. We want to show that the merge chase for a **Horn-$\mathcal{ALCH}$** knowledge base K computes a finite universal model when it terminates and that it terminates if and only if there exists a finite universal model for K.

Definition 3.3 (Mergeable variable)**.** For two terms u and v appearing in a factbase F, u is *mergeable* on v in F if:

- $u \neq v$ and u is a variable,

"merge on" sounds weird; see https://forum.wordreference.com/threads/merge-to-with-into.703507/ -¿ I do not like " x mergeable with t "

- $\mathbf{Preds}^1_F(u) \subseteq \mathbf{Preds}^1_F(v)$, and
- there exists a term t such that $\mathbf{Preds}^2_F(t,u) \neq \emptyset$ and $\mathbf{Preds}^2_F(t,u) \subseteq \mathbf{Preds}^2_F(t,v)$.

In this case, we say that u is a *mergeable variable*.

Definition 3.4. For a term t and a variable x, we say that $t \prec x$ if x is of the form $f^y_\alpha(t)$.

We write $\prec^+$ to denote the transitive closure of $\prec$. We note $\boldsymbol{Desc}(t)= \{y \in \mathbf{Vars} \mid t = y \vee t \prec^+ y\}$.

We can now define the notion of tree of a term t that is all the facts containing only variables in $\mathbf{Desc}(t)$:

Definition 3.5 (Tree)**.** For a term t occurring in a chase derivation of the **Horn-**$\mathcal{ALCH}$ knowledge base K, we pose:

$$\begin{aligned} \mathit{Tree}_F(t) = &\{A(u) \mid A(u) \in F \wedge u \in \mathbf{Desc}(t)\} \cup \\ &\{R(u,x) \mid R(u,x) \in F \wedge u, x \in \mathbf{Desc}(t)\} \end{aligned}$$

When x is mergeable on t in F, all the triggers applied on x and on the $\prec$-successor of x can be or have already been applied on t and on the $\prec$-successor of t. Therefore, all the facts containing x and the $\prec$-successor of x are or will be redundant. But we do not want to supress all these facts because we know that if they are not yet in the tree of t, they would be computed. Instead, we would like to "recycle" these facts. That is this intuition that leads us to define atomic merging:

Definition 3.6 (atomic merging)**.** If a variable x is mergeable on a term t in a factbase F, then we note $h_{x/t}$ the substitution defined on $\mathbf{Vars}(F)$ such that

- for every variable y in $\mathbf{Desc}(x)$, $h_{x/t}(y)$ is the variable y where the occurence of x is replaced by t, and
- for every variable y not in $\mathbf{Desc}(x)$, $h(y) = y$.

The *atomic merging* of x on t in F produces $h_{x/t}(F)$

For example, in the Figure 3.1, $f^y_\alpha(a)$ is mergeable on b in the left factbase. The atomic merging of $f^y_\alpha(a)$ on b in this factbase produces the right factbase.

The following definition describes the operation that will replace the computation of the core in the core chase. We will show that for a chase derivation $D = F_0, ..., F_k$ of K, if we apply our operation on F_k, then it computes the core of a factbase that could have been produced by continuing the derivation D.

Definition 3.7. For a factbase F, a *merge* sequence of F is a sequence $F_0, \ldots, F_n$ of factbases such that:

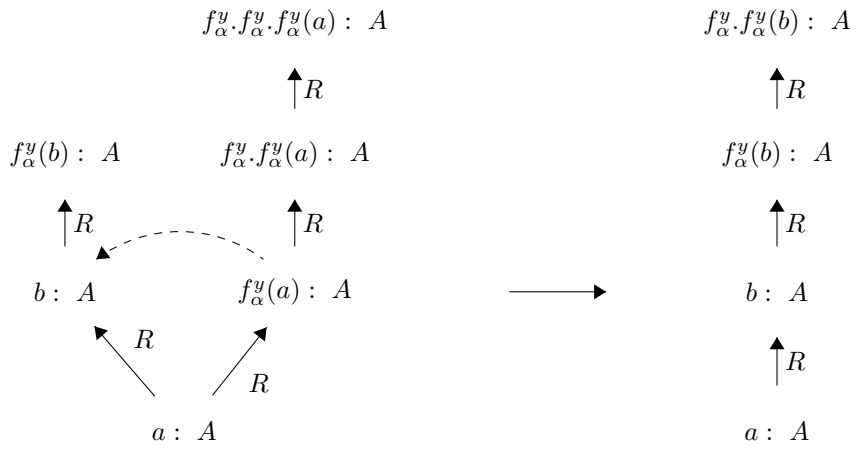

Figure 3: Atomic Merging Example

- $F_0 = F$.
- For $i \in \{1, \ldots, n\}$, the factbase F_i is the atomic merging of a variable x on a term t in F_{i-1}.
- The factbase F_n does not contain a mergeable variable.

Then, $\mathit{Merge}(F) = F_n$.

The result of the *Merge* operation is unique up to isomorphism.

Example 3.1. In the Figure 3.1, we merge the factbase of the left. The variable $f^z_\alpha(a)$ is mergeable on b, an atomic merging of $f^z_\alpha(a)$ on b yields the factbase of the middle. Now, $f^z_\beta(b)$ is mergeable on c, an atomic merging of $f^z_\beta(b)$ on c produces the factbase of the right. The last factbase is the result of the merge since it does not contain any mergeable variable.

We can now consider two new chases:

Definition 3.8 (derivation)**.** An *atomic merge derivation* (resp. a *merge derivation*) for a knowledge base $K = (R, F)$ is a (possibly infinite) sequence $D = F_0, F_1, F_2, \ldots$ where $F_0 = F$, and for $i > 0$, either $F_i = \mathbf{appl}(F_{i-1}, t_i)$ is obtained by an application with t_i an oblivious trigger, or F_i is obtained by an atomic merging of F_{i-1} (resp. the merging of F_{i-1}).

D is *fair* if the first condition is respected (resp. the two conditions are respected):

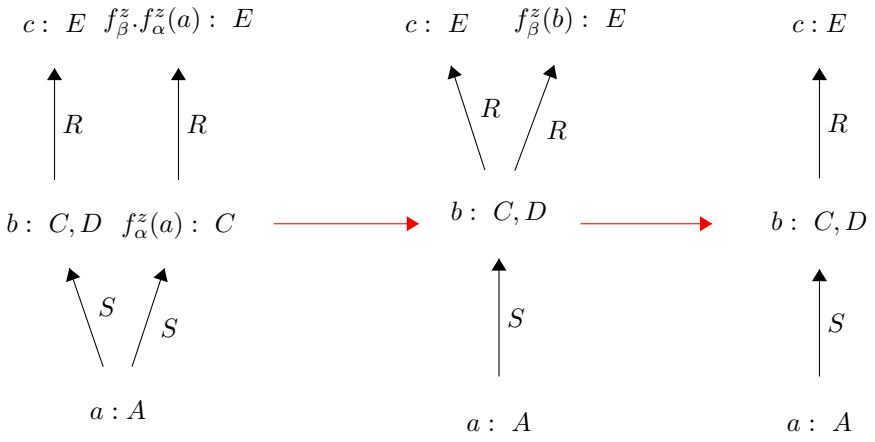

Figure 4: Merging Example

- For every i, for every oblivious trigger t applicable on F_i, there exists $k \leq i$ or $k > i$ such that $\mathbf{appl}(F_k, t) = F_k$; or there exists $k > i$ such that t is not anymore an oblivious trigger for F_k.
- For every i, there exists $k \geq i$ such that F_k is a core.

An *atomic merge chase* (resp. a *merge chase*) for a knowledge base $K = (R, F)$ is a fair atomic merge derivation (resp. a fair merge derivation) $D = F_0, F_1, F_2, \ldots$

Note that a merge chase is an atomic merge chase.

Definition 3.9. The atomic merge chase (resp. the merge chase) *terminates* on K if it is a finite derivation $D = F_0, F_1, F_2, \ldots, F_k$. We note, in this case, $\mathbf{MC}(K) = F_k$ the result of the merge chase.

The result of the merge chase is unique up to isomorphism.

Proposition 3.1. *Consider a variable t that occurs in a merge chase derivation D of some **Horn-$\mathcal{ALCH}$** knowledge base K. Then, t is of the form $f^y_\alpha(u)$ where u is a term, α is a rule of the form 3 in K, and y is the only existentially quantified variable occurring in α. Moreover, u is the only term in D such that $u \prec t$.*

Proposition 3.2. *For a factbase F that occurs in a merge chase derivation of some **Horn-$\mathcal{ALCH}$** knowledge base, $\prec^+$ is a strict partial order over the set of terms of F.*

Do we really need to define fairness for atomic merge derivations? Discuss. -¿ I think it is necessary to prove that the atomic merge chase computes a universal model

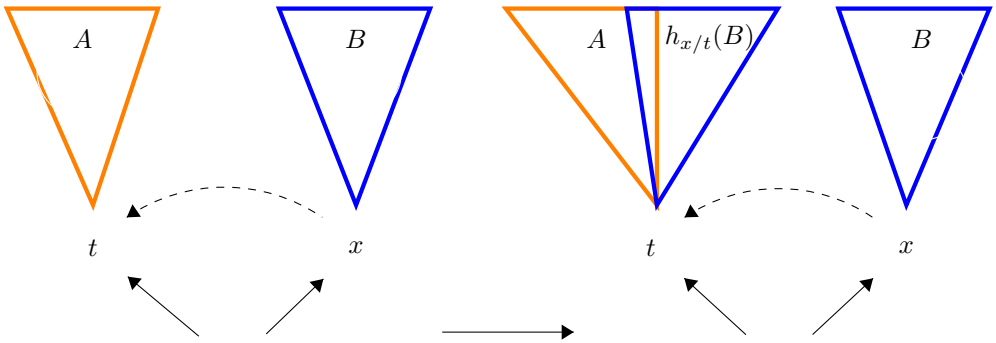

Figure 5: Illustration of the proof

Proof. Suppose for a contradiction that there exists a term t such that $t \prec^+ t$. There exists then $n \in \mathbb{N} \setminus \{0\}$ and terms $t_0, \ldots, t_n$ such that $t = t_0 \prec t_1 \prec \cdots \prec t_n = t$. By Proposition 3.1, there exists rules $\alpha_1, \ldots, \alpha_n$ and variables $v_1, \ldots, v_n$ such that $t_n = f_{\alpha_n}^{v_n}(f_{\alpha_{n-1}}^{v_{n-1}}(\cdots f_{\alpha_1}^{v_1}(t_0)\cdots))$. It is a contradiction since $t_0 = t$ and $t_n = t$. Therefore $\prec^+$ is irreflexive. By construction, $\prec^+$ is transitive; So $\prec^+$ is a strict partial order □

We have shown in the proof that the graph induced by $\prec$ does not contain any cycle. We obtain the following result from Propositions 3.1 and 3.2:

Proposition 3.3. *For a factbase F that occurs in a merge chase derivation of some **Horn-**$\mathcal{ALCH}$ knowledge base, the graph induced by $\prec$ is a forest of trees and the root of each tree is a constant.*

We show that the atomic merge is an operation that preserves the universality thanks to the two next propositions.

Proposition 3.4. *Let $D = F_0, F_1, \ldots, F_m$ be an atomic merge chase derivation of the knowledge base K. Let x be mergeable on t in F_m. There exists an atomic merge chase derivation $D' = D, F_{m+1} \ldots, F_k$ of K prolonging D such that*

- $F_{m+1} = \textbf{\textit{appl}}(F_m, t_m), \ldots, F_k = \textbf{\textit{appl}}(F_{k-1}, t_{k-1})$ *have been obtained by a rule application, where $t_m, \ldots, t_{k-1}$ are oblivious triggers,*
- $F_k = F_m \cup h_{x/t}(\mathit{Tree}_{F_m}(x))$.

We pose $\mathbf{Fut}(F_m, t, x) = F_k$.

Graphically, in the Figure 3.1, if we note $A = \mathit{Tree}_{F_m}(t)$ and $B = \mathit{Tree}_{F_m}(x)$, then we want do an atomic merge derivation from the left factbase to the right factbase.

To prove the Proposition 3.4, we will look all the triggers dealing with the variables in $\mathbf{Desc}(x)$ and apply them on the variables in $\mathbf{Desc}(t)$ if the triggers have not already been applied.

Proof. Let $tr_1 = (\alpha_1, \sigma_1), \ldots, tr_n = (\alpha_n, \sigma_n)$ be all the oblivious triggers applied on F_m such that

- the range of each σ_i contains only variables in $\mathbf{Desc}(x)$.
- if $F_k = \mathbf{appl}(F_{k-1}, tr_i)$ and $F_l = \mathbf{appl}(F_{l-1}, tr_j)$, where $k < l$, then $i < j$ (that is, $tr_1, \ldots, tr_n$ are oredered by application time).

We note $h_{x/t}(tr_i) = (\alpha_i, h_{x/t} \circ \sigma_i)$. Let $t_1, \ldots, t_r$ be the maximal subsequence of $h_{x/t}(tr_1), \ldots, h_{x/t}(tr_n)$ such that for all i, $\mathbf{appl}(F_m, t_i) \neq F_m$.

We pose $F_{m+1} = \mathbf{appl}(F_m, t_1), \ldots, F_{m+r} = \mathbf{appl}(F_{m+r-1}, t_r)$ and for $i \in \{1, \ldots, n\}$,

$$B'_i = h_{x/t} \circ \sigma_1^s(Head(\alpha_1)) \cup \ldots \cup h_{x/t} \circ \sigma_i^s(Head(\alpha_i))$$

We note $D_0 = D$, $r(0) = 0$, and for $i \in \{1, \ldots, n\}$, if there exists $r(i) \in \{1, \ldots, r\}$ such that $h_{x/t}(tr_i) = t_{r(i)}$, then we note $D_i = D_{i-1}, F_{m+r(i)}$; Otherwise, we note $D_i = D_{i-1}$ and we pose $r(i) = r(i-1)$. The integer $r(i)$, for $i > 0$, verifie the condition $\mathbf{appl}(F_{m+r(i)}, tr_i) = F_{m+r(i)}$.

I am getting lost around here; let's discuss it in the next meeting. ¡- I change a litle bit, is it better ?

We show by induction on $i \in \{0, \ldots, n\}, H(i)$: The sequence D_i is an atomic merge chase and $B'_i \subseteq F_{m+r(i)}$. For i = 0, $D_0 = D$ is an oblivious derivation and $B'_0 = \emptyset$, so $H(0)$ is true. Suppose that $H(i-1)$ is true for $i \in \{1, \ldots, n\}$.

If $\mathbf{appl}(F_m, h_{x/t}(tr_i)) = F_m$ then $h_{x/t} \circ \sigma_i^s(Head(\alpha_i)) \subseteq F_{m+r(i)}$ so $B'_i = B'_{i-1} \cup h_{x/t} \circ \sigma_i^s(Head(\alpha_i)) \subseteq F_{m+r(i)}$. Therefore $D_i = D_{i-1}$ is suitable, $H(i)$ is true.

Otherwise, the oblivious trigger $h_{x/t}(tr_i)$ has not been applied on F_m, there exists j such that $h_{x/t}(tr_i) = t_j$. Depending on the form of the rule α_i, there exists four different cases:

- If α_i is of the form $A(u) \to \exists v.R(u, v) \wedge B(v)$. We have $A(\sigma_i(u)) \in F_m$.
  - If $\sigma_i(u) = x$, then $A(x) \in F_m$. As t is a strong sibling of x in F_m, $A(t) \in F_m$. So $A(h_{x/t}(\sigma_i(u))) \in F_{m+r(i-1)}$ since $h_{x/t}(\sigma_i(u)) = t$.
  - If $x \prec^+ \sigma_i(u)$, then the fact $A(\sigma_i(u)))$ is in $\sigma_j^s(Head(\alpha_j))$ where $j < i$. We have then $A(h_{x/t}(\sigma_i(u))) \in B'_{i-1}$ so by induction hypothesis, $A(h_{x/t}(\sigma_i(u))) \in F_{m+r(i-1)}$.

  Therefore $h_{x/t}(tr_i)$ is an oblivious trigger for $F_{m+r(i-1)}$.
- If α_i is of the form $A_1(u) \wedge A_2(u) \to B(u)$. We have $A_1(\sigma_i(u)), A_2(\sigma_i(u)) \in F_m$.

– If $\sigma_i(u) = x$, then $A_1(x), A_2(x) \in F_m$. As t is a strong sibling of x in F_m, we have $A_1(t), A_2(t) \in F_m$. So $A_1(h_{x/t}(\sigma_i(u))), A_2(h_{x/t}(\sigma_i(u))) \in F_{m+r(i-1)}$ since $h_{x/t}(\sigma_i(u)) = t$.
– If $x \prec^+ \sigma_i(u)$, then the facts $A_1(\sigma_i(u)))$ and $A_2(\sigma_i(u)))$ are in $\sigma^s_j(Head(\alpha_j))$ where $j < i$. We have then $A_1(h_{x/t}(\sigma_i(u))), A_2(h_{x/t}(\sigma_i(u))) \in B'_{i-1}$ so by induction hypothesis, $A_1(h_{x/t}(\sigma_i(u))), A_2(h_{x/t}(\sigma_i(u))) \in F_{m+r(i-1)}$.

Therefore $h_{x/t}(tr_i)$ is an oblivious trigger for $F_{m+r(i-1)}$.

- If α_i is of the form $A(u) \wedge R(u, v) \rightarrow B(v)$.
  – If $\sigma_i(u) = x$, then $A(x) \in F_m$. As t is a strong sibling of x in F_m, $A(t) \in F_m$. Thus $A(h_{x/t}(\sigma_i(u))) \in F_{m+r(i-1)}$.
  – If $x \prec^+ \sigma_i(u)$, then the fact $A(\sigma_i(u)))$ is in $\sigma^s_j(Head(\alpha_j))$ where $j < i$. We have then $A(h_{x/t}(\sigma_i(u))) \in B'_{i-1}$ so by induction hypothesis, $A(h_{x/t}(\sigma_i(u))) \in F_{m+r(i-1)}$.

  The fact $R(\sigma_i(u), \sigma_i(v))$ is in $\sigma^s_j(Head(\alpha_j))$ where $j < i$. We have then $R(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v))) \in B'_{i-1}$ so by induction hypothesis, $R(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v))) \in F_{m+r(i-1)}$.

  Therefore, the facts $A(h_{x/t}(\sigma_i(x)))$ and $R(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v)))$ are in $F_{m+r(i-1)}$. Thus $h_{x/t}(tr_i)$ is an oblivious trigger for $F_{m+r(i-1)}$.

- If α_i is of the form $R(u, v) \wedge B(v) \rightarrow A(u)$. $\sigma_i(u) \neq x$ since $A(t) \in F_m$

  The facts $R(\sigma_i(u), \sigma_i(v)), B(\sigma_i(v))$ have been introduced by oblivious triggers in $\{tr_1, \ldots, tr_{i-1}\}$. So, by induction hypothesis, the facts $R(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v)))$ and $B(h_{x/t}(\sigma_i(v)))$ are in $F_{m+r(i-1)}$. Therefore $h_{x/t}(tr_i)$ is an oblivious trigger for $F_{m+r(i-1)}$.

- If α_i is of the form $R_1(u, v) \wedge R_2(u, v) \rightarrow S(u, v)$.

  The facts $R_1(\sigma_i(u), \sigma_i(v)), R_2(\sigma_i(u), \sigma_i(v))$ are in $\sigma^s_j(Head(\alpha_j))$ where $j < i$. We have then $R_1(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v))), R_2(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v))) \in B'_{i-1}$ so by induction hypothesis, $R_1(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v))), R_2(h_{x/t}(\sigma_i(u)), h_{x/t}(\sigma_i(v))) \in F_{m+r(i-1)}$.

As $h_{x/t}(tr_i)$ is an oblivious trigger for $F_{m+r(i-1)}$, D_i is an atomic merge derivation. By definition of an application, we have $h_{x/t} \circ \sigma^s_i(Head(\alpha_i)) \subseteq F_{m+r(i)}$ and so $B'_i = B_{i-1} \cup h_{x/t} \circ \sigma^s_i(Head(\alpha_i)) \subseteq F_{m+r(i)}$. Therefore, $H(i)$ is true.

We have proved the heredity. So, D_n is the oblivious derivation that we was looking for. We have $F_{m+k} = F_m \cup h_{x/t}(Tree_{F_m}(x))$.

□

Proposition 3.5. *Let $D = F_0, trig_1, F_1, \ldots, trig_m, F_m$ be an atomic merge chase derivation for the knowledge base K. Assume that x is mergeable on t in F_m, the atomic merging of x on t in F_m is a retract of **Fut**(F_m, t, x) .*

Proof. The atomic merging of x on t in F_m is $h_{x/t}(F_m)$. By the Proposition 3.4, $\mathbf{Fut}(F_m, t, x) = F_m \cup h_{x/t}(Tree_{F_m}(x))$. So, $h_{x/t}(F_m) \subseteq \mathbf{Fut}(F_m, t, x)$ since $\mathbf{Fut}(F_m, t, x) = Tree_{F_m}(x) \cup h_{x/t}(F_m)$. Finally, as $h_{x/t}(\mathbf{Fut}(F_m, t, x)) = h_{x/t}(F_m)$, we have that $h_{x/t}(F_m) \models \mathbf{Fut}(F_m, t, x)$. □

The following theorem is a corollary of Propositions 3.4 and 3.5 :

Theorem 3.1. When applied to a factbase in an atomic merge chase sequence, the atomic merge operation preserves universality.

I would simply say that each fact set in an atomic merge sequence is universal. -¿ I define what mean that an operation preserve the universality and what are the properties at the end of thd part 2

Proof. Let K be a **Horn-$\mathcal{ALCH}$** knowledge base, $D = F_0, \ldots, F_m$ be an atomic merge chase derivation, and M be a model of K such that $M \models F_m$. Suppose that there exists a variable x mergeable on a term t in F_m. According to Proposition 3.4, The factbase $\mathbf{Fut}(F_m, t, x)$ has been obtained from F_m only by the use of the application operation. The Proposition **??** says that this operation preserves the universality. We have then $M \models \mathbf{Fut}(F_m, t, x)$. So M is a model of the atomic merging of x on t in F_m since, by Proposition 3.5, it is a subset of $\mathbf{Fut}(F_m, t, x)$. □

The merge operation uses only atomic merge, so by Theorem 3.1:

Theorem 3.2. When applied to a factbase in an atomic merge chase sequence, the merge operation preserves universality.

The main reason to prefer the merge chase instead of the atomic merge chase is that unless the atomic merging operations are applied exhaustively, the merge chase may not terminate on inputs that admit finite universal models:

Theorem 3.3. There exists a **Horn-$\mathcal{ALCH}$** knowledge base and an infinite atomic merge chase derivation for K such that K admits a finite universal model.

I think that it would be clearer to say that all of the elements in a merge sequence are universal.

Proof. Let $F = \{R(a, b), R(b, a), A(a), A(b)\}$ and $R = \{\alpha = A(x) \rightarrow \exists y.R(x, y) \wedge A(y), \beta = B(x) \rightarrow A(x)\}$. We consider the atomic merge chase derivation for $K = (R, F)$ $F_0 = F, F_1, F_2, ...$

- We applied to F_0 the oblivious trigger $t_1 = (\alpha, \{x \mapsto a\})$;
- we then applied to F_1, the oblivious trigger $t_2 = (\alpha, \{x \mapsto f^y_\alpha(a)\})$ giving rise to the first factbase of the Figure 3.1;
- we then apply the oblivious trigger $t_3 = (\beta, \{x \mapsto b\})$ to obtain the factbase F_3;
- At this moment, $f^y_\alpha(a)$ is mergeable on b so we do an atomic merging of $f^y_\alpha(a)$ on b to get F_4 that is the second factbase of the figure;

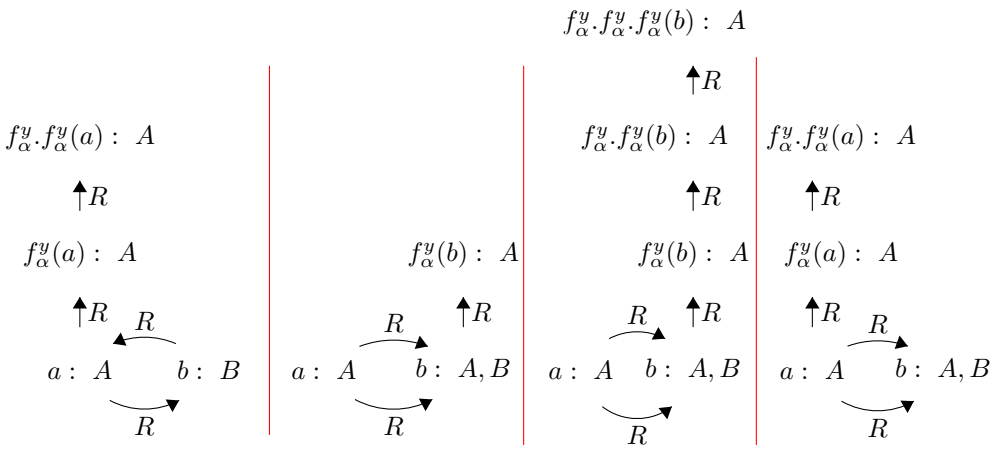

Figure 6: Example where the atomic merge chase is not efficient

- F_5 is obtained by the application of the oblivious trigger $t_4 = (\alpha, \{x \mapsto f^y_\alpha(b)\})$ and F_6 by the oblivious trigger $t_5 = (\alpha, \{x \mapsto f^y_\alpha(f^y_\alpha(b))\})$, F_6 is the third factbase of the figure;
- At this moment, $f^y_\alpha(b)$ is mergeable on a so we do an atomic merging of $f^y_\alpha(b)$ on a to get F_7 that is the last factbase of the figure;
- We can repeat this infinitely.

But K admits a finite universal model: $U = F \cup \{B(b)\}$.

□

Improve write-up; discuss. is it better ?

We want now prove that a merging computes a core:

Proposition 3.6. *For a factbase G that occurs in a merge chase derivation of some **Horn-$\mathcal{ALCH}$** knowledge base, Merge(G) is a core.*

Proof. Suppose for a contradiction that *Merge*(G) is not a core. There exists a factbase $G' \subsetneq$ *Merge*(G) such that G' is a retract of *Merge*(G). By Proposition 2.1, there exists then a retractation h from *Merge*(G) to G'. We have $var(\textit{Merge}(G)) \setminus var(G') \neq \emptyset$. Let x be a $\prec$-minimal variable of this set. The term x is a variable, so has been introduced by the chase due to a rule of the form 3. So there exists a term t such that $t \prec x$.

- We have $\mathbf{Preds}^2_{Merge(G)}(t,x) \neq \emptyset$. By $\prec$-minimality of x, $t \in \mathbf{Vars}(G')$. So, as h is a retractation: $h(t) = t$, so for $R \in \mathbf{Preds}^2_{Merge(G)}(t,x)$, $h(R(t,x)) = R(t,h(x)) \in Merge(G)$ and so $R \in \mathbf{Preds}^2_{Merge(G)}(t,h(x))$. Thus $\mathbf{Preds}^2_{Merge(G)}(t,x) \subseteq \mathbf{Preds}^2_{Merge(G)}(t,h(x))$ and $t \prec h(x)$.
- $x \notin G'$ and $h(x) \in G'$ so $h(x) \neq x$.
- Let $A \in \mathbf{Preds}^1_{Merge(G)}(x)$. $h(A(x)) \in Merge(G)$ so $A(h(x)) \in Merge(G)$ so $\mathbf{Preds}^1_{Merge(G)}(x) \subseteq \mathbf{Preds}^1_{Merge(G)}(h(x))$.

Consequently, x is mergeable on $h(x)$ in $Merge(G)$ which results on a contradiction. Hence, $Merge(G)$ is a core. □

$Merge(G)$ is a core but not necessarily the core of G.

Proposition 3.7. *The merge chase computes a finite universal model of K when it terminates.*

Proof. Let $D = F_0, \ldots, F_n$ be a merge chase for $K = (R, G)$.

- The merge chase never take off ground facts and $G = F_0$ so $G \subseteq F_n$. We have then $F_n \models G$. Assume for a contradiction that $F_n \not\models R$. There exists then a rule α in R not satisfied by F_n: $F_n \vDash Body(\alpha)$ and $F_n \not\models Head(\alpha)$. It means that there exists a substitution σ from $Body(\alpha)$ to F_n. Therefore, $t = (\alpha, \sigma)$ is an oblivious trigger for F_n. As the derivation D is fair, there exists k such that $\mathbf{appl}(F_k, t) = F_k$. Thus $\sigma^s(Head(\alpha)) \subseteq F_k$, so $F_k \models \alpha$. As $F_n \models F_k$, $F_n \models \alpha$ which lead us to a contradiction. We have then $F_n \models R$ so F_n is a model of K.
- According to Proposition **??** and 3.2, each operation of the merge chase conserves the universality so we can show by induction that F_n is a universal model of K.
- F_n is finite.

□

Proposition 3.8. *If there exists a finite universal model for a **Horn-$\mathcal{ALCH}$** knowledge base K, then the merge chase terminates.*

Proof.
- Suppose for a contradiction that K admits a finite universal model U for K and does not admit a finite merge chase sequence. There exists so an infinite merge chase sequence $F_1, F_2, \ldots$
- A term t is redundant with respect to this sequence if $t \in F_i$ and $t \notin F_j$ for some $i < j$.

- For each i, let G_i be the maximal subset of F_i that does not contain redundant terms.
- We pose $M = \cup_i G_i$, it is an universal model of K so $M \models U$. There exists then a homomorphism from U to M. As U is finite, the factbase $h(U) \subseteq M$ is also finite. The sequence $G_0, G_1, \ldots$ is monotonic (that is $G_0 \subseteq G_1 \subseteq \cdots$) so there exist i such that $h(U) \subseteq G_i$. We have then $F_i \models h(U)$ since $G_i \subseteq F_i$. We have $h(U) \models U$, so $F_i \models U$. As all the used operations during a merge chase preserves the universality and as $U \models F_0$, $U \models F_i$. Therefore F_i is a finite universal model for K.
- There exists a finite number of triggers for F_i and D is fair. So, there exists $j \geq i$ such that all the triggers for F_i has been applied in the derivation $F_0, F_1, \ldots, F_j$.
- As D is fair, there exists $k \geq j$ such that F_k is a core. We have $F_k \subseteq F_i$ so all the triggers for F_k has been applied in the derivation $F_0, F_1, \ldots, F_k$.
- The derivation $F_0, \ldots, F_k$ is therefore fair so it is a finite merge chase sequence which leads us to a contradiction.

□

This proof needs a bit more work; do you have any questions about it?

The following theorem is then a direct consequence of Propositions 3.7 and 3.8.

Theorem 3.4. The merge chase computes an universal model if and only if there exists an universal model.

We will now describe a deterministic algorithm to merge a factbase:

Definition 3.10 (Merging)**.** Let F be a factbase that occurs in a merge chase derivation of the knowledge base K.

Algorithm 1: Merge(F):

```
1 Let Vars(F) = {x_1, ..., x_n} be such that (x_i ≺⁺ x_j) ⇒ i < j ;
2 for i = 1 to n do
3 |  if x_i is still a variable in F then
4 |  |  for all term t such that x_i is mergeable on t do
5 |  |  |  F ← the atomic merging of x_i on t in F.
6 |  |  end
7 |  end
8 end
9 return F
```

At line 1, we can sort terms like that because, by proposition 3.2, $\prec^+$ is a strict partial order over the set of variables of F.

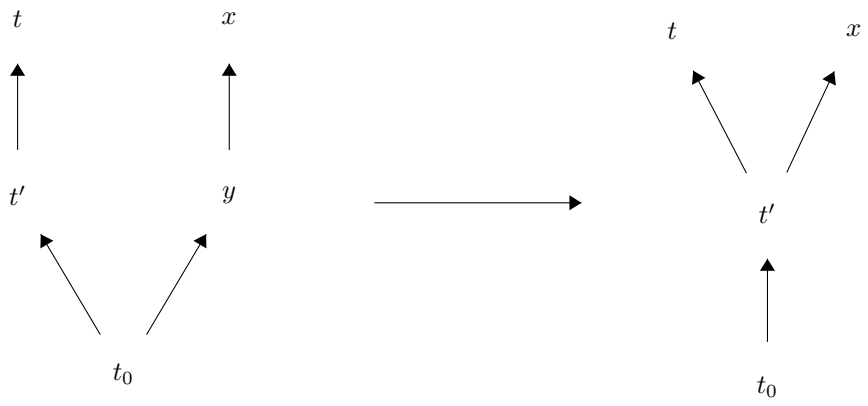

Figure 7: Illustration of the proof

Note that the application of an atomic merge may result in new mergeable variables. Therefore, we have to be carefull about the order of the variables in the Algorithm 1. In the Example 3.1, if we treat $f^z_\beta(f^z_\alpha(a))$ before $f^z_\alpha(a)$, then at the moment where we treat $f^z_\beta(f^z_\alpha(a))$, it does not have any term t yet such that $f^z_\beta(f^z_\alpha(a))$ is mergeable on t. At the end, we get the factbase of the middle and we will not have merged every possible mergeable variable.

The Example 3.3 shows the importance of doing a total merging. We have to prove that our merging algorithm does a total merging:

Proposition 3.9. *Let G be a factbase that occurs in a chase derivation of the knowledge base K. There does not exists a term t and a variable x such that x is mergeable on t in Merge(G).*

Proof. Suppose for a contradiction that there exists a term t and a variable x such that x is mergeable on t in *Merge*(G), that is, there exists a term t' such that $\mathbf{Preds}^2_{Merge(G)}(t', x) \neq \emptyset$ and $\mathbf{Preds}^2_{Merge(G)}(t', x) \subseteq \mathbf{Preds}^2_{Merge(G)}(t', t)$. This case can happen only if x became mergeable after that x has been traited by the merging algorithm. Thus, during the merging, there has been an atomic merging on t'. Let y be the variable merged on t' such that $y \prec x$. We note G^1 the factbase just before the atomic merging of y on t' and we note G^2 the factbase just after the atomic merging. There exists a term t_0 such that $\mathbf{Preds}^2_{G^1}(t_0, y) \neq \emptyset$ and $\mathbf{Preds}^2_{G^1}(t_0, y) \subseteq \mathbf{Preds}^2_{G^1}(t_0, t')$. The factbase G^1 is in the left of the Figure 9 and the factbase G^2 is in the right (we do not represent all the graphs):

We have $t_0 \prec t' \prec t$ and $t_0 \prec y \prec x$ so x should have been treated by the algorithm after the merging of t' and y so the algorithm will merge x on t. It is

a contradiction. □

3.2 Horn-$\mathcal{ALCHI}$

Definition 3.11 (**Horn-$\mathcal{ALCHI}$** axioms)**.** A ***Horn-$\mathcal{ALCHI}$*** *axiom* is either a **Horn-$\mathcal{ALCH}$** axiom or an existential rule of the form:

$$R_1(x, y) \wedge R_2(x, y) \rightarrow S(y, x) \tag{6}$$

A ***Horn-$\mathcal{ALCHI}$*** *knowledge base* $K = (R, F)$ is a knowledge base where R is a set of **Horn-$\mathcal{ALCHI}$** axioms and F contains only predicates of arity one or two.

We have to modify the merge chase because it does not work anymore:

We keep the same relation $\prec$. It is still a strict partial order over the set of variables.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases: The Logical Level*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1995.

[2] Catriel Beeri and Moshe Y. Vardi. The implication problem for data dependencies. In *Proceedings of the 8th Colloquium on Automata, Languages and Programming*, page 73–85, Berlin, Heidelberg, 1981. Springer-Verlag.

[3] A. Calì, G. Gottlob, and M. Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Artificial Intelligence Research*, 48:115–174, Oct 2013.

[4] Alin Deutsch, Alan Nash, and Jeffrey B. Remmel. The chase revisited. In Maurizio Lenzerini and Domenico Lembo, editors, *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008, June 9-11, 2008, Vancouver, BC, Canada*, pages 149–158. ACM, 2008.

[5] Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler. Complexities of horn description logics. *ACM Trans. Comput. Log.*, 14(1):2:1–2:36, 2013.