Implementing the Core Chase for the Description Logic ALC

Maël Abily

July 5, 2021

1 Introduction

An important problem in database is the conjunctive query entailment. This problem can be described in a first order logic background: Given a knowledge base $O = (R, F)$ where F is a set of conjunctive formulas (that are formulas constructed only with conjunction and existential quantification) and R is a set of rules (that are, if $\vec{u}$ represents a tuple of variables, formulas of the form $\forall\vec{x}.\forall\vec{y}.(A(\vec{x},\vec{y}) \to \exists\vec{z}.B(\vec{x},\vec{z})))$, and given a query Q (that is a conjunctive formula), determine if the knowledge base O entails the query Q. We usually use reasoning algorithms to answer this problem. We can notice that in practice, a lot of formulas can be express via conjunctive formulas.

Example 1.1. For the knowledge base O composed of the set of conjunctive formulas $F = \{\mathit{Father}(\mathit{Michel})\}$ and the set of rules $R = \{\forall x.\mathit{Father}(x) \to \exists y.\mathit{IsTheSonOf}(y,x)\}$, and for the query $Q = \exists y.\mathit{IsTheSonOf}(y, Michel)$, the knowledge base O entails Q.

By definition, a knowledge base O entails a query Q if every model of O is a model of Q. It is not practical because a knowledge base can have an infinite number of model.

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

Nevertheless, it is the slowest so it it is never used for practical applications.

Therefore, we will focus on a restricted type of knowledge base $O = (R, F)$ where the set of conjunctive formulas F is ground (that is there is no variable in the formula) and where the rule contained in R are Horn-$\mathcal{ALC}$ axioms. It is an interesting restriction because we can represent the results of the chase by a tree.

We will create a quicker chase, 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 below where y is a sibling of t' and is merged on it.

$$\begin{tikzcd}
t & & x & & & t & & x \\
t' \arrow[u] & & y \arrow[u] & {} \arrow[r] & {} & & t' \arrow[ul] \arrow[ur] & \\
& t_0 \arrow[ul] \arrow[ur] & & & & & t_0 \arrow[u] &
\end{tikzcd}$$

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

This section deals with a lot of first order logic notions like interpretations and formulas.

2.1 Facts

2.1.1 Syntax

We considered a set of variables **Vars** (often noted $x, y, x_1, \ldots$), a set of constants **Csts** (often noted $a, b, c, c_1, \ldots$), and a set of predicates **Preds** ($P, Q, R, P_1, \ldots$). **Csts**, **Vars**, and **Preds** are pairwise disjoint. A *term* (often noted $t, t_1, \ldots$) is a variable or a constant. We note **Terms** the set of terms. We write $Ar(P)$ to denote the arity of the predicate P.

Definition 2.1. If $t_1, \ldots, t_n$ are terms and P is a predicate with $Ar(P) = n$, 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.

Definition 2.2. 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.

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 2.3. A factbase F *entails* another factbase F' (often noted $F \models F'$) if each interpretation satisfying F satisfies F'.

Definition 2.4. A factbase F is equivalent to another factbase F' if $F \models F'$ and $F' \models F$.

2.1.2 Homomorphism

Definition 2.5 (Substitution)**.** A *substitution* $\sigma : X \rightarrow \mathbf{Terms}$ 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 $\sigma(F) = \{\sigma(f_1), \ldots, \sigma(f_n)\}$.

Definition 2.6. For two factbases F and F', a *homomorphism* from F to F' is a substitution $\sigma : \mathbf{Vars}(F) \rightarrow \mathbf{Terms}$ where $\sigma(F) \subseteq F'$. Sometimes, we will say that we *map* a variable x to a term t if $\sigma(x) = t$.

Definition 2.7. For two factbases F and F', an *isomorphism* h from F to F' is a bijective homomorphism where its inverse is a homomorphism from F' to F.

For the remainder of this paper, we identify sets of facts that are unique up to isomorphism.

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 ([**?**],theorem 6.2.3).

Example 2.1. 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 F' 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 2.8. A factbase F' is a *retract* of another factbase F if $F' \subseteq F$ and $F' \models F$. A *retractation* from F to F' is a homomorphism σ from F to F' such that $\sigma_{|F'} = id_{|F'}$. F' is a *strict retract* of F is F' is a retract of F and $F' \neq F$.

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

Definition 2.9. 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.

Proposition 2.2. *The cores of a finite factbase F are unique up to isomorphism.*

Hence, we speak of "the" core of a factbase.

Example 2.2. $F_1 = \{B(x, y), R(y, z)\}$ is the core of $F = \{B(x, y), R(y, z), B(x, w), R(w, z)\}$ because:

- $F_1 \subseteq F$;
- $\{x \mapsto x, y \mapsto y, z \mapsto z, w \mapsto y\}$ is a homomorphism from F to F_1, so F_1 is a retract of F;
- all strict subsets of F_1 are not retracts of F_1.

Note that $F_2 = \{B(x, w), R(w, z)\}$ is also the core of F and is indeed isomorphic to F_1 due to the homomorphism $\{x \mapsto x, y \mapsto w, z \mapsto z\}$.

2.2 Existential Rules

2.2.1 Syntax

Definition 2.10. 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

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

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

We omit the universal quantifiers when representing existential rules.

Definition 2.11. A *knowledge base* O is a pair (R, F) where R is a set of existential rules and F is a ground factbase.

2.2.2 Semantics

Definition 2.12 (Entailment)**.** 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.

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 2.13 (Model)**.** A factbase M is a *model* for a knowledge base $O = (R, F)$ if $M \models F$ and $M \models R$.

Example 2.3. We pose $O = (\{\alpha\}, F)$ where $\alpha = A(x) \rightarrow \exists z.R(x, z) \wedge A(z)$ and $F = \{A(b)\}$. $U = \{A(b), R(b, x_0)\} \cup \{A(x_i) \mid i \in \mathbb{N}\} \cup \{R(x_i, x_{i+1}) \mid i \in \mathbb{N}\}$ is a universal model of O. This knowledge base does not admit finite universal models.

Definition 2.14 (Entailment)**.** A knowledge base O *entails* a factbase B (often noted $O \models B$) if for each model M of O, $M \models B$.

Definition 2.15 (Universal model)**.** A factbase U is an *universal model* for a knowledge base $O = (R, F)$ if for every model M of O, $M \models U$.

We are interested by universal models because they can be used to solve fact entailment:

Proposition 2.3. *A knowledge base O entails a factbase B if there exists a universal model U for O such that $U \models B$.*

An important problem that this document has to deal with is: Given a knowledge base $O = (R, F)$ and a factbase Q, does $O \models Q$? It is well-known that this problem is undecidable ([**?**], 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 a 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 2.16 (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* if t is an oblivious trigger 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. So future definitions that are dealing with oblivious trigger, deal also with restricted trigger. We will therefore 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 2.17 (application)**.** Let $t = (\alpha, \sigma)$ be a trigger of the factbase F. α is of the form $A(\vec{x}, \vec{y}) \to \exists \vec{z}.B(\vec{x}, \vec{z})$. We pose σ^s the substitution that extends σ over $\mathbf{Vars}(Head(\alpha))$ such that for $y \in \vec{z}$, $\sigma^s(y) = y_t$ where y_t is a fresh variable unique with respect to the trigger t and the variable y. 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 also say that the trigger t has been applied on F.

Example 2.4. . If $\alpha = A(x, y) \to \exists z.B(x, z)$, $F = \{A(b, c)\}$, and $\sigma = \{x \mapsto b, y \mapsto c\}$ then (α, σ) is a restricted trigger for F. $\mathbf{appl}(F, (\alpha, \sigma)) = \{A(b, c), B(b, z_{(\alpha,\sigma)})\}$ where $z_{(\alpha,\sigma)}$ is a fresh variable.

Definition 2.18 (Derivation)**.** An *oblivious derivation* (respectively a *restricted derivation*) from a knowledge base $O = (F, R)$ is a (possibly infinite) sequence $D = F_0, t_1, F_1, t_2, F_2, \dots$ where $(F_i)_{i \in \mathbb{N}}$ are factbases such that $F_i \subsetneq F_{i+1}$, t_i are oblivious triggers pairwise different (resp. restricted triggers), $F_0 = F$, and $F_i = \mathbf{appl}(F_{i-1}, t_i)$ for all $i > 0$.

Definition 2.19 (Fairness)**.** The oblivious (resp. restricted) derivation $D = F_0, t_1, F_1, t_2, F_2, \dots$ is *fair* if for every i and every oblivious (resp. restricted) trigger t applicable on F_i, there exists $k \geq i$ such that $\mathbf{appl}(F_k, t_k) = F_k$ (resp. t is not anymore a restricted trigger on F_k).

A fair derivation garantees that we consider every possible application. An easy way to have a fair derivation is to do a breadth-first search (BFS) on the terms, that is, always apply the oldest triggers before applying other triggers.

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

Definition 2.20. An *oblivious chase* (resp. a restricted chase) for a knowledge base $O = (F, R)$ is a fair oblivious (resp. restricted) derivation $D = F_0, t_1, F_1, t_2, F_2, \dots$

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

It is well known that:

Theorem 2.3. For a knowledge base O, $\mathit{Obl}(O)$ is an universal model of O.

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

We say that the restricted chase *terminates* if there exists an element in *Res(O)* that is finite.

It is well known that:

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

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

Example 2.5. Suppose that we have the knowledge base $O = (\{\alpha\}, F)$ where $\alpha = A(x, y) \to \exists z.A(y, z) \wedge A(z, y)$ and $F = \{A(a, b)\}$. An oblivious chase derivation is $F_0, t_1, F_1, t_2, F_2, ...$ where $F_0 = F$, $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$, $F_1 = \{A(a, b), A(b, z_{t_1}), A(z_{t_1}, b)\}$, $t_2 = (\alpha, \{x \mapsto b, y \mapsto z_{t_1}\})$, $F_2 = \{A(a, b), A(b, z_{t_1}), A(z_{t_1}, b), A(z_{t_1}, z_{t_2}), A(z_{t_2}, z_{t_1})\}$, $\ldots$ It will never terminate because each new atom brings new rule applications. So the oblivious chase does not terminate on O 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 *Res(O)* $= F_1$.

The oblivious chase is called this way because it can make naive applications (that is $F_i \vDash F_{i+1}$): in the previous example, $F_1 \models F_2$. The restricted chase is less naive because a restricted trigger is applied only if it really adds information (that is $F_i \nvDash F_{i+1}$).

Theorem 2.5. A knowledge base O entails a factbase B if $\mathit{Obl}(O) \models B$ or $\mathit{Res}(O) \models B$.

2.3.1 The core chase

It has been firstly defined in [**?**].

Definition 2.21 (Core derivation)**.** A *core derivation* for a knowledge base $O = (R, F)$ 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}.

Definition 2.22 (Fairness)**.** A core derivation $D = F_0, F_1, F_2, \dots$ is *fair* if:

- For every i, for every oblivious trigger t applicable on F_i, there exists $k > i$ such that $\mathbf{appl}(F_k, t) = F_k$ or such that t has been applied on $\mathbf{appl}(F_k, t)$.
- For every i, there exists $k \geq i$ such that F_k is a core.

Definition 2.23. A *core chase* for a knowledge base $O = (R, F)$ is a fair core derivation $D = F_0, F_1, F_2, \dots$ The core chase *terminates* on O if it is a finite core derivation.

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

Definition 2.24. If the core chase terminates on O due to the finite core derivation $D = F_0, F_1, F_2, \dots, F_i$, then we pose $C(O) = F_i$. Otherwise, if the core chase does not terminate, $C(O)$ is undefined.

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

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

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

Example 2.6. Suppose that we have the knowledge base $O = (\{\alpha\}, F)$ where $\alpha = A(x, y) \rightarrow \exists z.(A(x, x) \wedge A(y, z))$ and $F = \{A(a, b)\}$. A restricted chase derivation is $F_0, t_1, F_1, t_2, F_2, \dots$ where

- $F_0 = F$,
- $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$,
- $F_1 = F_0 \cup \{A(a, a), A(b, z_{t_1})\}$,
- $t_2 = (\alpha, \{x \mapsto b, y \mapsto z_{t_1}\})$,
- $F_2 = F_1 \cup \{A(b, b), A(z_{t_1}, z_{t_2})\}$,
- $t_3 = (\alpha, \{x \mapsto z_{t_1}, y \mapsto z_{t_2}\})$,
- $F_3 = F_2 \cup \{A(z_{t_1}, z_{t_1}), A(z_{t_2}, z_{t_3})\}$,
- $\dots$

It will never terminate because each new atom brings new restricted triggers.

The core chase terminates on O: a core chase derivation is F_0, F_1, F_2, F_3 where

- $F_0 = F$,
- $t_1 = (\alpha, \{x \mapsto a, y \mapsto b\})$,
- $F_1 = \mathbf{appl}(F_0, t_1) = F_0 \cup \{A(a, a), A(b, z_{t_1})\}$,
- $t_2 = (\alpha, \{x \mapsto b, y \mapsto z_{t_1}\})$,
- $F_2 = \mathbf{appl}(F_1, t_2) = F_1 \cup \{A(b, b), A(z_{t_1}, z_{t_2})\}$,
- $F_3 = \mathit{Core}(F_2) = \{A(a, a), A(a, b), A(b, b)\}$.

There is not anymore any restricted trigger for F_3 so the derivation is fair. Consequently the core chase terminates on O.

3 The Merge Chase

The core chase always terminates when there exists a finite universal model but this core 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-$\mathcal{ALC}$.

Definition 3.1. For a factbase F and a term t, we note $\boldsymbol{Preds}^1_F(t)$ the set of unary predicates P such that $P(t) \in F$. For two terms t and t', we note $\boldsymbol{Preds}^2_F(t, t')$ the set of binary predicates P such that $P(t, t') \in F$.

In this section, we only consider logical theories 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 $E = \{(t_1, t_2) \mid t_1, t_2 \in \mathbf{Terms} \wedge \mathbf{Preds}_F(t_1, t_2) \neq \emptyset\}$. We label the vertex $v \in V$ by exactly the elements in $\mathbf{Preds}^1_F(v)$ and we label the edge between the terms t_1 and t_2 by exactly the elements in $\mathbf{Preds}^2_F(t_1, t_2)$. For example with $F = \{A(a), B(a), R(a, b), T(a, b), C(b), R(b, z)\}$:

$$\begin{tikzcd}
 & b : C \arrow[dr, "R"] & \\
a : A, B \arrow[ur, "{R,T}"] & & z
\end{tikzcd}$$

3.1 Horn-$\mathcal{ALC}$

Horn-$\mathcal{ALC}$ has been introduced in [**?**]

Definition 3.2 (Horn-$\mathcal{ALC}$ axioms)**.** A *Horn-$\mathcal{ALC}$ axiom* is an existential rule of the form:

$$A_1(x) \land A_2(x) \to B(x) \tag{1}$$
$$A(x) \land R(x,y) \to B(y) \tag{2}$$
$$A(x) \to \exists y.R(x,y) \land B(y) \tag{3}$$
$$R(x,y) \land B(y) \to A(x) \tag{4}$$

We fix $O = (R, F)$ a knowledge base for this section where R is a Horn-$\mathcal{ALC}$ rule set.

We can notice that all the variables are introduced by a Horn-$\mathcal{ALC}$ axiom of the form (3).

We will now introduce some notions in order to introduce the new chase and prove that it does what we want.

Definition 3.3. Let F' be a factbase that occurs in a chase derivation of the knowledge base O. For a term t_1 and a variable x_2 appearing in F', we say that $t_1 \prec x_2$ if there exists a restricted trigger $tr = (A(x) \to \exists y.R(x,y) \land B(y), \{x \mapsto t_1\})$ such that $x_2 = y_t$. We write $\prec^+$ to denote the transitive closure of $\prec$.

Concretely, $t_1 \prec x_2$ if x_2 has been introduced by t_1 due to a rule of the form (3). Therefore:

Proposition 3.1. *Let F' be a factbase that occurs in a chase derivation of the knowledge base O. For every variable x, there exists exactly one predecessor for $\prec$.*

Proposition 3.2. *Let $D = F_0, F_1, \dots, F_k$ be a chase derivation of the knowledge base O. $\prec^+$ is a strict partial order over the set of variables of F_k.*

Proof. We write $\prec_i$ to denote the relation $\prec$ over the factbase F_i. We show by induction on i that $\prec_i^+$ is a strict partial order over the set of variables of F_i.

F_0 is ground so $\mathbf{Vars}(F_0) = \emptyset$ so the initialisation is true.

Suppose that $\prec_i^+$ is a strict partial order over the set of variables of F_i. By construction, $\prec_{i+1}^+$ is transitive over $\mathbf{Vars}(F_{i+1})$. We just have to show that $\prec_{i+1}^+$ is irreflexive.

If F_{i+1} is the core of F_i, we just take off some facts so $\prec_{i+1}^+ \subseteq \prec_i^+$. Then, $\prec_{i+1}^+$ is irreflexive.

Otherwise, $F_{i+1} = \mathbf{appl}(F_i, t_i)$ with $t_i = (\alpha, \sigma)$ a restricted trigger.

- If α is not of the form of the rule 3, then $\prec_{i+1}^{+}=\prec_i^{+}$ so $\prec_{i+1}^{+}$ is also irreflexive by induction hypothesis.
- Otherwise, $\alpha = A(x) \to \exists y.R(x,y) \wedge B(y)$ so $F_{i+1} = F_i \cup \{R(\sigma(x), y_{t_i}), B(y_{t_i})\}$ and $\prec_{i+1}=\prec_i \cup (\sigma(x), y_{t_i})$. So $\prec_{i+1}^{+}=\prec_i^{+} \cup \{(z, y_{t_i}) \mid z = \sigma(x) \text{ or } z \prec_i^{+} \sigma(x)\}$. We can see that y_{t_i} has no successor for the relation $\prec_{i+1}$ so, as $\prec_i^{+}$ is irreflexive by induction hypothesis, $\prec_{i+1}^{+}$ is irreflexive.

Consequently, $\prec_{i+1}^{+}$ is a strict partial order over the set of variables of F_{i+1}. So the heredity is true.

□

We have shown in the proof that the graph induced by $\prec$ does not contain any cycle. Therefore, with the last proposition and Proposition 3.1:

Proposition 3.3. *Let F' be a factbase that occurs in a chase derivation of the knowledge base O. The graph induced by $\prec$ is a forest of trees.*

We can now define the notion of tree of a term t that is all the facts containing at least one $\prec$-sucessor of t:

Definition 3.4 (Tree)**.** Let F' be a factbase that occurs in a chase derivation of the knowledge base O. For a term t, we pose

$$\begin{aligned} \mathit{Tree}_{F'}(t) = &\{A(t') \mid A(t') \in F' \wedge t' \in \textbf{Terms} \wedge t \prec^{+} t'\} \\ &\cup \{R(t', x) \mid R(t', x) \in F' \wedge x \in \textbf{Vars} \wedge t' \in \textbf{Terms} \wedge (t = t' \vee t \prec^{+} t')\} \end{aligned}$$

We now define the notion of mergeable variable in order to consider an algorithm.

Definition 3.5 (Mergeable variable)**.** Let F' be a factbase that occurs in a chase derivation of the knowledge base O. For two terms t_1 and t_2 such that $t_1 \neq t_2$, t_1 is *mergeable* on t_2 in F' if:

- t_1 is a variable,
- $\textbf{Preds}^1_{F'}(t_1) \subseteq \textbf{Preds}^1_{F'}(t_2)$,
- there exists a term t such that
  - $\textbf{Preds}^2_{F'}(t, t_1) \neq \emptyset$, and
  - $\textbf{Preds}^2_{F'}(t, t_1) \subseteq \textbf{Preds}^2_{F'}(t, t_2)$.

In this case, we say that t_1 is a *mergeable variable.*

We imposed that t_1 is mergeable on t_2 only if t_1 is a variable because, we will latter map t_1 on t_2 and it is not possible to map a constant to another term.

When x is mergeable on t in F', we want to say that 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 so we would like to "recycle" these facts. That is this intuition that leads us to define atomic merging:

Definition 3.6 (atomic merging)**.** For a factbase F', a term $t \in \mathbf{Terms}(F')$, a variable $x \in \mathbf{Vars}(F')$, and a substitution h where $h(x) = t$ and for all $y \neq x, h(y) = y$, if x is mergeable on t in F', then we say that $h(F')$ is the *atomic merging* of x over t in F'.

The following algorithm is the operation that will replace the computation of the core in the core chase. We will show that for a core chase derivation $D = F_0, ..., F_k$ of O, if we apply our algorithm on F_k, then it computes the core of a factbase that could have been produced by continuing the derivation D. In this sense, it always computes the core or something better than the core.

Definition 3.7 (Merging)**.** Let F' be a factbase that occurs in a chase derivation of the knowledge base O.

Algorithm 1: Merge(F'):

```
1 Let Vars($F'$) = $\{x_1, \ldots, x_n\}$ be such that $(x_i \prec^+ x_j) \Rightarrow i < j$ ;
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             $F' \leftarrow$ the atomic merging of $x_i$ over $t$ 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'.

We will now consider two new chases:

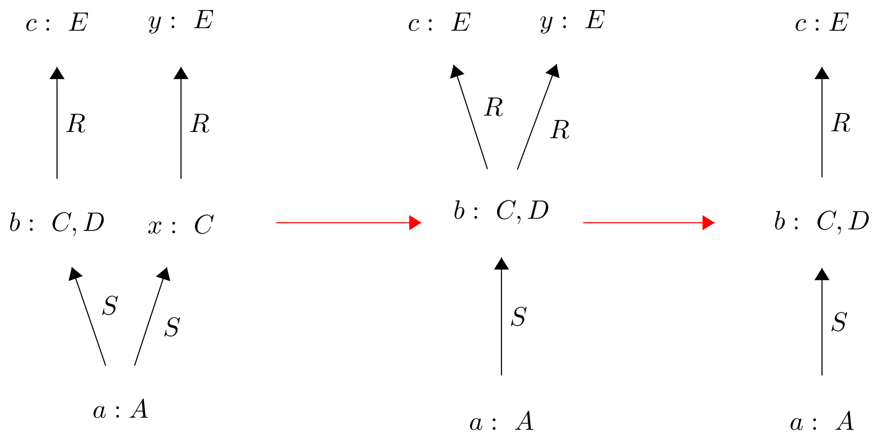
Definition 3.8 (derivation)**.** An *atomic merge derivation* (resp. a *merge derivation*) for a knowledge base $O = (R, F)$ is a (possibly infinite) sequence $D = F_0, F_1, F_2, \ldots$ where $F_0 = F$, and for $i > 0$, either $F_i = \mathbf{appl}(F_{i-1}, t_i)$ is obtained by an application with t_i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

The fairness of an atomic merge derivation will be the same that the fairness of an oblivious derivation and the fairness of a merge derivation will be the same that the fairness of a core derivation.

Definition 3.9. An *atomic merge chase* (resp. a *merge chase*)for a knowledge base $O = (R, F)$ is a fair atomic merge derivation (resp. a fair merge derivation) $D = F_0, F_1, F_2, \ldots$

Definition 3.10. The merge chase *terminates* on O if it is a finite core derivation.

Example 3.1. The figure below is an example of merging. We merge the factbase of the left, $x \prec y$ so we first look if x is mergeable on a term and then we look if y is mergeable on a term. The variable x is mergeable on b so we do an atomic merging of x over b that gives us the factbase of the middle. Now, y is mergeable on c so we do an atomic merging of y over c that gives us the factbase of the right.

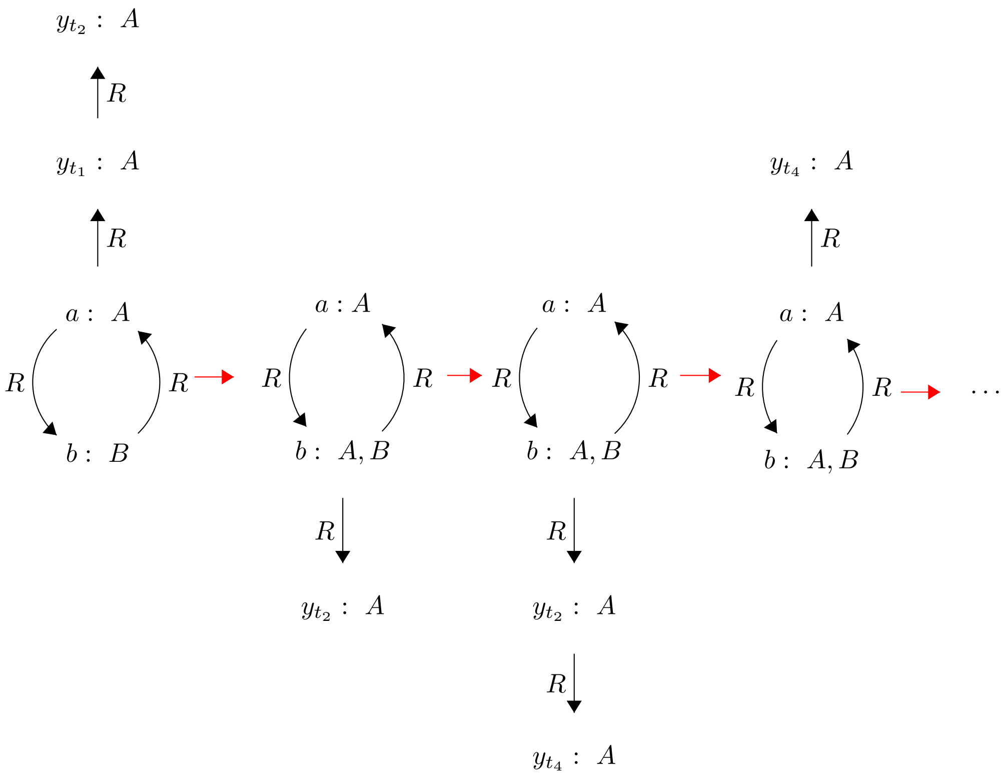


We can notice that the order of variables we choose is really important because an atomic merging can create mergeable variables. In this example, if we treat y before x, then at the the moment where we treat y, it does not have any term t yet such that y is mergeable on t so at the end, we get the factbase of the middle and we will not have merged every possible mergeable variable.

It is really important that the merging does all the possible atomic merging that it can do. Otherwise, the merge chase may not terminates on some knowledge base whereas there exists a finite universal model (like in the next example) and we want that the merge chase terminates on all knowledge base that admits a finite universal model.

Example 3.2. We consider in the example that $F = \{R(a, b), R(b, a), A(a), A(b)\}$ and $R = \{\alpha, \beta\}$ where $\alpha = A(x) \to \exists y.R(x, y) \wedge A(y)$ and $\beta = B(x) \to A(x)$.

We consider the atomic merge chase derivation $F_0 = F, F_1, F_2, ...$ We applied to F_0 the restricted trigger $t_1 = (\alpha, \{x \mapsto a\})$, we then applied to F_1, the restricted trigger $t_2 = (\alpha, \{x \mapsto y_{t_1}\})$ giving rise to the first factbase of the figure below. We then apply the restricted trigger $t_3 = (\beta, \{x \mapsto b\})$ to obtain the factbase F_3. At this moment, y_{t_1} is mergeable on b so we do an atomic merging of y_{t_1} over b to get F_4 that is the second factbase of the figure. F_5 is obtained by the application of the restricted trigger $t_4 = (\alpha, \{x \mapsto y_{t_2}\})$ and is the third factbase of the figure. At this moment, y_{t_2} is mergeable on a so we do an atomic merging of y_{t_2} over a to get F_6 that is the last factbase of the figure. We can repeat this infinitely. But O admits a finite universal model: $U = F \cup \{B(b)\}$. So the version of the chase where we consider partial merging is not what we want.



The last example show the importance of doing a total merging. We have to prove that our merging algorithm does a total merging:

Proposition 3.4. *Let G be a factbase that occurs in a chase derivation of the knowledge base O. 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' and a binary predicate R such that $R(t', x), R(t', t) \in \mathit{Merge}(G)$. 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 over 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 over t' and we note G^2 the factbase just after the atomic merging. There exists a term t_0 and a binary predicate S such that $S(t_0, t'), S(t_0, y) \in G^1$. G^1 is the left figure and G^2 is the right figure (we do not represent all the graphs):

$$\begin{tikzcd}
t & & x & & & t & & x \\
t' \arrow[u, "R"] & & y \arrow[u, "R"] & {} \arrow[r] & {} & & t' \arrow[ul, "R"] \arrow[ur, "R"'] & \\
& t_0 \arrow[ul, "S"] \arrow[ur, "S"'] & & & & & t_0 \arrow[u, "S"] &
\end{tikzcd}$$

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: contradiction. □

We want now prove that a merging computes a core:

Proposition 3.5. *For a factbase G obtained by applying some Horn-$\mathcal{ALC}$ axioms on O, Merge(G) is a core.*

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

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

$h(R(t,x)) = R(t,h(x)) \in \mathit{Merge}(G)$ and so $R \in \mathbf{Preds}^2_{\mathit{Merge}(G)}(t,h(x))$. Thus $\mathbf{Preds}^2_{\mathit{Merge}(G)}(t,x) \subseteq \mathbf{Preds}^2_{\mathit{Merge}(G)}(t,h(x))$ and $t \prec h(x)$.

- $x \notin G'$ and $h(x) \in G'$ so $h(x) \neq x$.
- Let $A \in \mathbf{Preds}^1_{\mathit{Merge}(G)}(x)$. $h(A(x)) \in \mathit{Merge}(G)$ so $A(h(x)) \in \mathit{Merge}(G)$ so $\mathbf{Preds}^1_{\mathit{Merge}(G)}(x) \subseteq \mathbf{Preds}^1_{\mathit{Merge}(G)}(h(x))$.

Consequently, x is mergeable on $h(x)$ in $\mathit{Merge}(G)$. So, by proposition 3.4, the merging should have suppress x or $h(x)$, so $x \notin \mathit{Merge}(G)$ or $h(x) \notin \mathit{Merge}(G)$: contradiction. So $\mathit{Merge}(G)$ is a core. □

$\mathit{Merge}(G)$ is a core but not necessarily the core of G.

3.2 Corectness

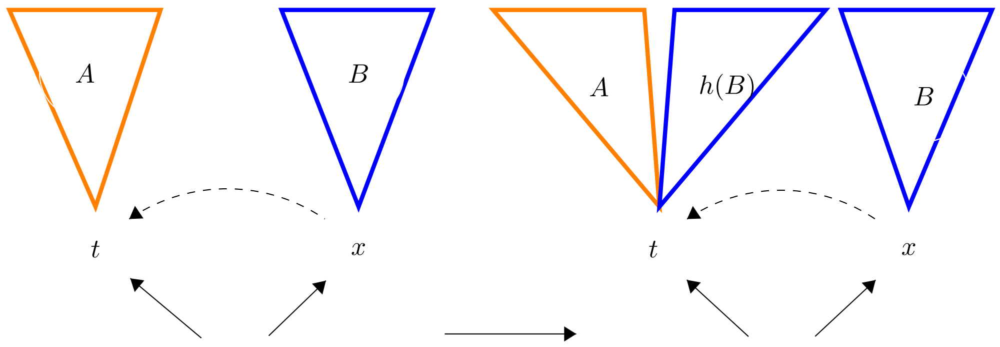
To prove that the merge chase is a correct algorithm, we have to show that:

- It preserves the universality.
- A merge step computes a core.

To simplify the proofs, we will allow us to use the same triggers. It will not really change the result because we can just add redundances and it preserves the terminaison. We cannot use anymore our definition of application so we redefine it. A naive trigger is a triple $t = (\alpha, \sigma, \hat{\sigma})$ where (α, σ) is an oblivious trigger and $\hat{\sigma}$ is a subsitution that extends σ over $\mathbf{Vars}(\mathit{Head}(\alpha))$. The factbase $\mathbf{appl}(F,t) = F \cup \hat{\sigma}(\mathit{Head}(\alpha))$ is an application on the factbase F through the trigger t.

Proposition 3.6. *Let $D = F_0, trig_1, F_1, \ldots, trig_m, F_m$ be a chase derivation of the knowledge base O. Let x be mergeable on t in F_m. Let h be a substitution defined on $\boldsymbol{Vars}(\mathit{Tree}_{F_m}(x))$ such that $h(x) = t$ and for $y \neq x$, $h(y)$ is a fresh variable. There exists an oblivious chase derivation $D' = F_0, trig_1, F_1, \ldots, trig_k, F_k$ of O prolonging D such that $F_k = F_m \cup h(\mathit{Tree}_{F_m}(x))$. We pose $F_{\mathbf{Fut}(F_m,t,x)} = F_k$ and $h_{\mathbf{Fut}(F_m,t,x)} = h$.*

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



To prove the result, we will look all the triggers dealing with the successor of x and apply them on the successor of t even if the triggers has already been applied.

Proof. Let $tr_1 = (\alpha_1, \sigma_1, \hat{\sigma}_1), \ldots, tr_n = (\alpha_n, \sigma_n, \hat{\sigma}_n)$ be all the oblivious triggers in $\{trig_1, \ldots, trig_m\}$ such that

- the range of each $\hat{\sigma}_i$ contains at least one $\prec$-sucessor of x
- if $tr_i = trig_r$ and $tr_j = trig_l$, then $i < j$ implies $r < l$ (that is, $tr_1, \ldots, tr_n$ are oredered by application time).

We note $tr'_1 = (\alpha_1, \sigma'_1, \hat{\sigma}'_1), \ldots, tr'_n = (\alpha_n, \sigma'_n, \hat{\sigma}'_n)$ where $\sigma'_i = h \circ \sigma_i$ and $\hat{\sigma}'_i = h \circ \hat{\sigma}_i$. We pose $F_{m+1} = \mathbf{appl}(F_m, tr'_1), \ldots, F_{m+n} = \mathbf{appl}(F_{m+n-1}, tr'_n)$

We show by induction on $i \in \{0, \ldots, n\}, H(i)$: the sequence $D_i = D, tr'_1, F_{m+1}, \ldots, tr'_i, F_{m+i}$ is an oblivious derivation.

For i $= 0$, $D_0 = D$ is an oblivious derivation, so $H(0)$ is true.

Suppose that $H(i-1)$ is true for $i \in \{1, \ldots, n\}$. Depending on the form of the rule α_i, there exists four different cases:

- First case α_i is of the form $A(u) \rightarrow \exists v.R(u, v) \wedge B(v)$. 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 $A(\sigma'_i(u)) \in F_{m+i-1}$ since $\sigma'_i(u) = t$.
  - If $x \prec^+ \sigma_i(u)$, then the fact $A(\sigma_i(u)))$ has been introduced by an oblivious trigger in $\{tr_1, \ldots, tr_{i-1}\}$. We have then by induction hypothesis, $A(\sigma'_i(u)) \in F_{m+i-1)}$.

  Therefore tr'_i is an oblivious trigger on F_{m+i-1}. So $H(i)$ is true.

- Second case α_i is of the form $A_1(u) \wedge A_2(u) \rightarrow B(u)$. We have $x \prec^+ \sigma_i(u)$. The facts $A_1(\sigma_i(u))$ and $A_2(\sigma_i(u))$ have been introduced by two oblivious trigger in $\{tr_1, \ldots, tr_{i-1}\}$. So by induction hypothesis,$A_1(h(\sigma_i(u))), A_2(h(\sigma_i(u))) \in F_{m+i-1}$. So tr'_i is an oblivious trigger on F_{m+i-1}. Consequently, $H(i)$ is true.

- Third case 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, $A(t) \in F_m$. $A(\sigma'_i(u)) \in F_{m+i-1}$.
  - If $x \prec^+ \sigma_i(u)$, then the fact $A(\sigma_i(u))$ has been introduced by an oblivious trigger in $\{tr_1, \ldots, tr_{i-1}\}$. So, by induction hypothesis, $A(\sigma'_i(u)) \in F_{m+i-1}$.

  the fact $R(\sigma_i(u), \sigma_i(v))$ has been introduced by an oblivious trigger in $\{tr_1, \ldots, tr_{i-1}\}$.So by induction hypothesis, $R(\sigma'_i(u), \sigma_i(v)) \in F_{m+i-1}$.

  Therefore, the facts $A(\sigma_i(x))$ and $R(\sigma_i(u), \sigma_i(v))$ are in F_{m+i-1}. So tr'_i is an oblivious trigger on F_{m+i-1}. So H(i) is true.

- Fourth case α_i is of the form $R(u,v) \wedge B(v) \rightarrow A(u)$.

  The facts $R(\sigma_i(u), \sigma_i(v)), B(\sigma_i(v))$ has been introduced by an oblivious trigger in $\{tr_1, \ldots, tr_{i-1}\}$. So, by induction hypothesis, the facts $A(\sigma_i(x))$ and $R(\sigma_i(u), \sigma_i(v))$ are in F_{m+i-1}. So tr'_i is an oblivious trigger on F_{m+i}. We conclude that $H(i)$ is true.

We have proved the heredity. So, D_n is the oblivious derivation that we was looking for. We have $F_{m+n} = F_m \cup h_n(\mathit{Tree}_{F_m}(x))$.

□

Proposition 3.7. *Let $D = F_0, trig_1, F_1, \ldots, trig_m, F_m$ be an oblivious chase derivation of the knowledge base O. Assume that x is mergeable on t in F_m. $F_{\mathbf{Fut}(F_m,t,x)}$ is equivalent to the atomic merging of x over t in F_m.*

Proof. We note F' the atomic merging of x over t in F_m and we pose $h = h_{\mathbf{Fut}(F_m,t,x)}$. By the Proposition 3.6, $F_{\mathbf{Fut}(F_m,t,x)} = F_m \cup h(\mathit{Tree}_{F_m}(x))$. We have that h is a bijection on $\mathbf{Vars}(\mathit{Tree}_{F_m}(x)) - \{x\}$ so h^{-1} is well defined on this set. We pose:

$$\begin{aligned}
h_1 : \mathbf{Vars}(F_{\mathbf{Fut}(F_m,t,x)}) &\rightarrow \mathbf{Terms}(F') \\
y &\mapsto t \text{ if } y = x \\
y &\mapsto h^{-1}(y) \text{ if } y \neq x \wedge\ y \in \mathbf{Vars}(h(\mathit{Tree}_{F_m}(x))) \\
y &\mapsto y \text{ otherwise}
\end{aligned}$$

and

$$\begin{aligned} h_2 : \mathbf{Vars}(F') &\to \mathbf{Terms}(F_{\mathbf{Fut}(F_m,t,x)}) \\ y &\mapsto h(y) \text{ if } y \in \mathbf{Vars}(\mathit{Tree}_{F_m}(x)) \\ y &\mapsto y \text{ otherwise} \end{aligned}$$

h_1 is a homomorphism from $F_{\mathbf{Fut}(F_m,t,x)}$ to F' and h_2 is a homomorphism from F' to $F_{\mathbf{Fut}(F_m,t,x)}$.

inserer graphique

□

We have proved that the atomic merge chase preserves the universality, therefore:

Proposition 3.8. *The atomic merge chase and the merge chase computes an universal model when they terminate.*

By Proposition [**?**], the merge chase computes a core, therefore, we can use the the results of the paper [**?**] to have the following theorem:

Theorem 3.1. The merge chase computes an universal model if and only if there exists an universal model.

3.3 Generalisation

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

$$R_1(x,y) \wedge R_2(x,y) \wedge \ldots \wedge R_n(x,y) \to S(x,y) \tag{5}$$

A *Horn-$\mathcal{ALCHI}$ axiom* is either a Horn-$\mathcal{ALCH}$ axiom or an existential rule of the form:

$$R_1(x,y) \wedge R_2(x,y) \wedge \ldots \wedge R_n(x,y) \to S(y,x) \tag{6}$$

3.3.1 Horn-$\mathcal{ALCH}$

We fix $O = (R, F)$ a knowledge base for this section where R is a Horn-$\mathcal{ALC}$ rule set and F is a ground factbase with predicates of arity one or two. We have to modify the merge chase because it doesn't work anymore:

Example 3.3. If we have the knowledge base $O = (R, F)$ where $F = \{A(a)\}$ and the rules are :

$$\begin{aligned} A(x) &\to \exists y.S(x,y) \wedge B(y) \\ A(x) &\to \exists y.T(x,y) \wedge B(y) \\ S(x,y) &\to R(x,y) \\ T(x,y) &\to R(x,y) \end{aligned}$$

By applying the rules one, two, three and then four, we have the factbase G:

$$\begin{tikzcd}
x : B & & y : B \\
& a : A \arrow[ul, "{R,S}"'] \arrow[ur, "{R,T}"'] &
\end{tikzcd}$$

Then, when we apply a merging:

$$\begin{tikzcd}
x : B \\
a : A \arrow[u, "{R,S,T}"']
\end{tikzcd}$$

The merging is bad because this factbase is not a universal model of O.

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