Implementing the Core Chase for the Description Logic ALC

Maël Abily

June 21, 2021

The goal is to answer a query with a given database and a given set of rules by computing a universal model with an algorithm called the core chase. We are dealing with a restriction of FOL (Horn-$\mathcal{ALC}$ axioms).

Contents

1 Background

We only define what we need in first order logic but we do not redefine basics (interpretations, formulas,...).

1.1 Facts

1.1.1 Syntax

We considered a set of variables **Vars** (often noted $x, y, x_1, ...$). We define a *vocabulary* as a tuple (**Csts**,**Preds**) where **Csts** is a set of constants (often noted $a, b, c, c_1, ...$) and **Preds** is a set of predicates $(P, Q, R, P_1, ...)$. **Csts**, **Vars**, and **Preds** are pairwise disjoint. A *term* (often noted $t, t_1, ...$) 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 1.1. If $t_1, ..., t_n$ are terms and P is a predicate with $Ar(P) = n$, then $P(t_1, ..., t_n)$ is an *atom*. The atom $P(t_1, ..., t_n)$ is said to be *ground* if $t_1, ..., t_n$ are constants.

Definition 1.2. A *factbase* F is an existentially closed conjunction of atoms, that is F is a formula of the form $\exists x_1, ..., x_n.P_1(t_1^1, ..., t_{k_1}^1) \wedge ... \wedge P_m(t_1^m, ..., t_{k_m}^m)$ where t_i^j are terms and P_i are predicates. A factbase is *ground* if eauch atom is ground.

In some articles, the factbase are always considered as ground but it will not be the case here. 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 to use set notions such as the inclusion on sets of facts. 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 ***Vars***(A) (respectively ***Csts***(A), and ***Terms***(A)) be the set of variables (resp. constants, and terms) that occur in A. ***FB*** is the set of factbases.

1.1.2 Semantics

Definition 1.3. A factbase F *entails* a factbase F' (often noted $F \models F'$) if each interpretation satisfying F satisfies F'.

1.1.3 Homomorphism

Definition 1.4 (Substitution)**.** A *substitution* $\sigma : X \rightarrow$ **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 $c \in$ **Csts**, then $\sigma(c) = c$;
- if $x \in$ **Vars** $\setminus X$, $\sigma(x) = x$;
- if $f = P(t_1, ..., t_n)$ is an atom, then $\sigma(f) = P(\sigma(t_1), ..., \sigma(t_n))$;
- if $F = \{f_1, ..., f_n\}$ is a factbase, then $\sigma(F) = \{\sigma(f_1), ..., \sigma(f_n)\}$

Definition 1.5. For two factbases F and F', a *homomorphism* from F to F' is a substitution $\sigma : var(F) \rightarrow term(F')$ where $\sigma(F) \subseteq F'$.

Definition 1.6. For two factbases F and F', an *isomorphism* h from F to F' is a bijective homomorphism where its inverse h^{-1} is an homomorphism from F' to F.

We identify sets of facts that are unique up to isomorphism.

Theorem 1.1 (Homomorphism Theorem)**.** *A factbase F entails a factbase Q (often noted $F \models Q$) if and only if there exists a homomorphism from Q to F.*

Proof. [1] □

Example 1.1. The factbase $F = \{P(b, a), A(x)\}$ entails the factbase $Q = \{P(x, a), P(y, z)\}$ due to the homomorphism $\{x \mapsto b, y \mapsto b, z \mapsto a\}$.

Remark 1.1. Given two factbases F and Q, the problem to know if $F \models Q$ is NP-complete [3].

1.1.4 Core

For a factbase F, let $id_{|F}$ be the substitution identity defined by: for all variables $x \in \mathbf{Vars}(F)$, $id_{|F}(x) = x$.

Definition 1.7. A subset $F' \subseteq F$ is a *retract* of F if $F' \models F$.

Definition 1.8. A *retractation* from F to F' is a homomorphism σ from F to F' such that $\sigma_{|F'} = id_{|F'}$.

Proposition 1.1. *F' is a retract of F if and only if $F' \subseteq F$ and there exists a retractation from F to F'.*

Definition 1.9. 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 minimal retract of F that is a core.

Proposition 1.2. *The cores of a finite factbase F are unique up to isomorphism.*

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

Example 1.2. $F' = \{B(x, y), R(y, z)\}$ is the core of $F = \{B(x, y), R(y, z), B(x, w), R(w, z)\}$ because:

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

Proposition 1.3. *A factbase F is a core $\Leftrightarrow$ every homomorphism $\sigma : F \rightarrow F$ is a bijection.*

1.2 Existential rules

1.2.1 Syntax

Definition 1.10. Let $\vec{x}$, $\vec{y}$, and $\vec{z}$ be tuple of variables pairwise disjoint. An *(existential) rule* R 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 $\mathit{body}(R) = A$, $\mathit{head}(R) = B$, and the frontier of R $\mathit{fr}(R) = \vec{x}$ (that is the set of variables shared by the body and the head of R). We also note $\mathit{ev}(R)$ the set $\vec{z}$ of existential variables of the rule.

We omit the universal quantifiers when representing existential rules.

Definition 1.11. A *knowledge base* O is a pair (R, F) where R is a set of existential rules and F is a ground factbase.

1.2.2 Semantic

Definition 1.12 (Entailment)**.** A factbase F *entails* a rule α if each interpretation satisfying F statisfies α. We will note $F \models R$ if F entails each rule of the rule set R.

Theorem 1.2. 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 $\sigma : \mathbf{var(A)} \rightarrow \mathbf{term(F)}$ from A to F, there exists an extension of σ that is a homomorphism from B to F.

Definition 1.13 (Universal model)**.** A factbase M is a *model* for a knowledge base $O = (R, F)$ if $M \models F$ and $M \models R$. A model U for a knowledge base O is *universal* if for every model M of O, there exists a homomorphism $h : U \rightarrow M$.

Definition 1.14 (Entailment)**.** A knowledge base O *entails* a factbase B (often noted $O \models B$) if for each model M of O, $M \models B$.

Example 1.3. 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.

Proposition 1.4. *A knowledge base O entails a factbase B if there exists a universal model U 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 [3].

1.3 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 evenually 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, and since the problem of entailment is undecidable, this process may not halt.

Definition 1.15 (Trigger)**.** Let T be a rule set, α be a rule, σ be a subsitution, and F be a factbase. The tuple $t = (\alpha, \sigma)$ is an *oblivious trigger* for F (or α is *applicable* on F via σ) if:

- the domain of σ is the set of all variables occurring in $Body(\alpha)$.
- $\sigma(Body(\alpha)) \subseteq F$.

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

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

Definition 1.16 (application)**.** Let $t = (\alpha, \sigma)$ be an oblivious trigger of the factbase F. Let $\hat{\sigma}$ be a substitution that extends σ over $\mathbf{Vars}(Head(\alpha))$ such that for $y \in ev(R)$, $\hat{\sigma}(y) = y_t$ where y_t is a fresh variable unique with respect to the oblivious trigger t and the variable y. The factbase $\beta(F, t) = F \cup \hat{\sigma}(Head(\alpha))$ is called an *application* on the factbase F through the oblivious trigger $t = (\alpha, \sigma)$.

Example 1.4. . 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. $\beta(F, (\alpha, \sigma)) = \{A(b, c), B(b, z_{(\alpha,\sigma)})\}$ where $z_{(\alpha,\sigma)}$ is a fresh variable.

Definition 1.17 (Derivation)**.** An *oblivious derivation* (respectively a *restricted derivation*) from a knowledge base $O = (F, R)$ is a (possibly infinite) sequence $D = F_0, t_1, F_1, t_2, F_2, ...$ where F_i are factbases pairwise different, t_i are oblivious triggers (resp. restricted triggers), $F_0 = F$, and for $i > 0$, $F_i = \beta(F_{i-1}, t_i)$ is obtained by an application.

Definition 1.18 (Fairness)**.** The oblivious or restricted derivation $D = F_0, t_1, F_1, t_2, F_2, ...$ is *fair* if for every i, for every restricted trigger t applicable on F_i, there exists $k > i$ such that 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.

We will now define the oblivious and restricted chase, It is defined in [2].

Definition 1.19. An *oblivious chase* (resp. a restricted chase) for a knowledge base $O = (F, R)$ is a fair oblivious (resp. restricted) derivation $D = F_0, t_1, F_1, t_2, F_2, ...$ $F_0 \subseteq F_1 \subseteq F_2 \subseteq ...$ so we can pose *Obl(O)* $= \cup_{i \in \mathbb{N}} F_i$ (resp. *Res(O)* $= \cup_{i \in \mathbb{N}} F_i$). We say that the oblivious (resp. restricted) chase *terminates* if *Obl(O)* (resp. *Res(O)*) is finite.

The oblivious chase is called this way because it forgets to check whether the rule is already satisfied... The restricted chase is less naive because a restricted trigger is applied only if it is not already satisfied.

1.3.1 The core chase

It has been firstly defined in [4].

Definition 1.20 (Core derivation)**.** A *core derivation* for a knowledge base $O = (R, F)$ is a (possibly infinite) sequence $D = F_0, F_1, F_2, ...$ where $F_0 = F$, and for $i > 0$, either $F_i = \beta(F_{i-1}, t_i)$ is obtained by an application with t_i a restricted trigger, or F_i is obtained by computing the core of F_{i-1}.

Definition 1.21 (Fairness)**.** The core derivation $D = F_0, F_1, F_2, ...$ is *fair* if:

- For every i, for every restricted trigger t applicable on F_i, there exists $k > i$ such that t is not anymore a restricted trigger on F_k.
- For every i, there exists $k \geq i$ such that F_k is a core.
- If the derivation is finite, then the last factbase is a core.

Definition 1.22. A *core chase* for a knowledge base $O = (R, F)$ is a fair core derivation $D = F_0, F_1, F_2, ...$ The core chase *terminates* on O if there exits $i \in \mathbb{N}$ such that there is not anymore any restricted trigger applicable on F_i. In this case, we pose $C(O) = F_i$. Otherwise, if the core chase does not terminate,$C(O)$ is undefined.

Theorem 1.3. The knowledge base $O = (R, F)$ admits a finite universal model if and only if the core chase algorithm terminates on O.

Proof. [4] □

1.3.2 Comparaison of the chase algorithms

The oblivious chase can do a lot of uninteresting applications and does not terminate on some trivial knowledge base.

Example 1.5. If we have the knowledge base $O = (\{\alpha\}, F)$ where $\alpha = A(x, y) \rightarrow \exists z.A(y, z) \wedge A(z, y)$ and $F = \{A(a, b)\}$, then the oblivious chase must use the triggers $(\alpha, \{x \mapsto a, y \mapsto b\})$, $(\alpha, \{x \mapsto b, y \mapsto x_0\})$,$(\alpha, \{x \mapsto x_0, y \mapsto x_1\})$,... in order to terminate, consequently, it will never stop. So the oblivious chase does not stop on O whereas the restricted chase stops and *res(O)* $= \{A(a, b), A(b, z_0), A(z_0, b)\}$ because at the initial step, there is only one restricted trigger: $(\alpha, \{x \mapsto a, y \mapsto b\})$, we apply it and there is not anymore any restricted trigger.

So the restricted chase is better than the oblivious chase. Nevertheless, there exists knowledge bases where the restricted chase does not terminate whereas there exists a finite universal model.

Example 1.6. If we have the knowledge base $O = (\{\alpha\}, F)$ where $\alpha = A(x, y) \rightarrow \exists z.(A(x, x) \wedge A(y, z))$ and $F = \{A(a, b)\}$, then the restricted chase will use the restricted triggers : $(\alpha, \{x \mapsto a, y \mapsto b\})$, $(\alpha, \{x \mapsto b, y \mapsto z_0\})$,$(\alpha, \{x \mapsto z_0, y \mapsto z_1\})$, etc... and so *res(O)* $= \{A(a, a), A(a, b), A(b, b), A(b, z_0)\} \cup \{A(z_i, z_i), A(z_i, z_{i+1}) \mid i \in \mathbb{N}\}$. So the restricted chase does not terminate whereas there exists a universal model $U = \{A(a, a), A(a, b), A(b, b)\}$. The core chase terminates on O: at the first step $F_1 = \{A(a, a), A(a, b), A(b, z_0)\}$. At the second step, if we do an active application, $F_2 = \{(a, a), A(a, b), A(b, b), A(b, z_0), A(z_0, z_1)\}$ (if at this step, we will have computed the core, $F_2 = F_1$ and we will have continued the chase). If at the third step, we compute the core of F_1, then $F_2 = U$. There is not anymore any restricted trigger so the core chase terminates.

The core chase always terminates when there exists a finite universal model but this core chase is very expensive in time and it is dificult to define the result of the algorithm when there is no finite universal models because the computing factbases are not monotonic in comparaison to the factbases computing by the oblivious and restricted chase.

2 Horn-$\mathcal{ALC}$

2.1 Rules

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

Definition 2.1 (Horn-$\mathcal{ALC}$ axioms)**.** A *Horn-$\mathcal{ALC}$ axiom* is an existential rule of the form:

$$A_1(x) \wedge ... \wedge A_n(x) \rightarrow B(x) \tag{1}$$

$$A(x) \wedge R(x, y) \rightarrow B(y) \tag{2}$$

$$A(x) \rightarrow \exists y.R(x, y) \wedge B(y) \tag{3}$$

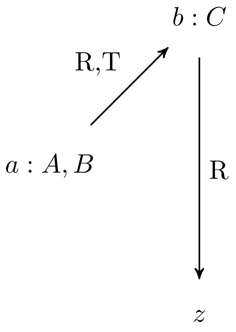
$$R(x, y) \wedge B(y) \rightarrow A(x) \tag{4}$$

Definition 2.2. For a factbase F and a term t, we note $C_F(t)$ the set of unary predicates P such that $P(t) \in F$.

We sometimes call this set : the *type* of t

In the Horn-$\mathcal{ALC}$ theory, the rules create only predicates of arity one or two. So, we will considere only factbases with predicates of arity one or two. Hence, we will represent a database F by a labelled graph $G = (V, E)$ where $V = \{t : A_1, ..., A_n/t \in$ **Terms** and $A_1, ..., A_n$ are exactly the elements in $C_F(t)\}$ and $E = \{(t_1, t_2)/t_1, t_2 \in$ **Terms** and there exists at least a binary predicate P such that $P(t_1, t_2) \in F$. In this case, we label the edge with exactly the binary predicates P such that $P(t_1, t_2) \in F\}$. For example with

$F = \{A(a), B(a), R(a,b), T(a,b), C(b), R(b,z)\}$:



2.2 Algorithm

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.

Definition 2.3. Let F' be a factbase that occured in a core chase derivation of the knowledge base O. For two terms t_1, t_2 appearing in F', we say that $t_1 \prec t_2$ if there exists a predicate R such that $R(t_1, t_2) \in F'$. We write $\prec^+$ to denote the transitive closure of $\prec$.

Proposition 2.1. *In the Horn-$\mathcal{ALC}$ theory, let $D = F_0, F_1, ...$ be a core chase derivation of the knowledge base O. We write $\prec_i^+$ to denote the relation $\prec^+$ over the factbase F_i. $\prec_i^+$ is a strict partial order over the set of variables of F_i.*

Proof. We show it by induction on 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.
    - $\prec_{i+1}^+$ is transitive over $\mathbf{Vars}(F_{i+1})$ by construction.
    -
        - If F_{i+1} is the core of F_i, we just take off some facts so $\prec_{i+1}^+ \subseteq \prec_i^+$ so $\prec_{i+1}^+$ is irreflexive.
        - Otherwise, $F_{i+1} = \beta(F_i, t_i)$ 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 irreflexive.
            - Otherwise, $\alpha = A(x) \to \exists y.R(x,y) \land B(y)$ so $F_{i+1} = F_i \cup \{R(x, y_{t_i}), B(y_{t_i})\}$ so $\prec_{i+1}^+ = \prec_i^+ \cup \{(z, y_{t_i}) \mid z = x \text{ or } z \prec_i^+ x\}$. It is clear that $\prec_{i+1}^+$ 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 graphs ($\mathbf{Vars}(F_i)$,$\prec_i$) does not contain any cycle. Therefore this graph is a forest of trees.

Definition 2.4 (Siblings)**.** Let F' be a factbase that occured in a core chase derivation of the knowledge base O. Two terms t_1 and t_2 such that $t_1 \neq t_2$ are *siblings* if $C_{F'}(t_1) \subseteq C_{F'}(t_2)$ or $C_{F'}(t_2) \subseteq C_{F'}(t_1)$, and if there exists a term t and a predicate R such that $R(t, t_1) \in F'$ and $R(t, t_2) \in F'$. In this case, t_1 is a *powerful sibling* of t_2 if $C_{F'}(t_2) \subseteq C_{F'}(t_1)$. Otherwise, it is a *powerless sibling.*

Definition 2.5 (Merging)**.** Merging a factbase F' that occured in a core chase derivation of the knowledge base O.

Algorithm 1: Merge(F'):

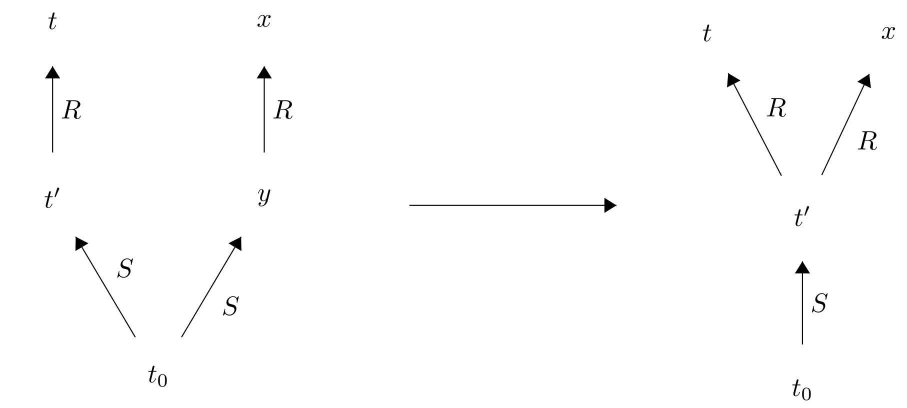
```
1  Let Terms(F') = $\{t_1, ...t_n\}$ be such that $(t_i \in \mathbf{cst} \wedge t_j \in \mathbf{var}) \Rightarrow i < j$
     and $(t_i \prec^+ t_j \wedge t_i, t_j \in \mathbf{var}) \Rightarrow i < j$ ;
2  for $i \in \{1, ..., n\}$ do
3      if $t_i$ is a variable then
4          for all powerfull siblings $t$ of $t_i$ do
5              $F' \leftarrow h(F')$ where $h$ is a substitution such that $h(t_i) = t$ and
                for all $x \neq t_i$, $h(x) = x$.
6          end
7      end
8      for all powerless siblings $x$ of $t_i$ such that $x$ is a variable do
9          $F' \leftarrow h(F')$ where $h$ is a substitution such that $h(x) = t_i$ and for
            all $y \neq x$, $h(y) = y$.
10     end
11 end
```

At line 1, we can sort terms like that because, by proposition 2.1, $\prec^+$ is a strict partial order over the set of variables of F'.

Proposition 2.2. *After a merging, there do not exists a term t and a variable x such that t is a powerfull sibling of x.*

There can still have siblings between two constants.

Proof. Let G be a factbase that occured in a core chase derivation of the knowledge base O. Suppose by contradiction that there exists a term t and a variable x such that t is a powerfull sibling of x in *Merge*(G): there exists a term t' and a binary predicate R such that $R(t', x), R(t', t) \in$ *Merge*(G). There is still a such sibling in *Merge*(G) only if there exists a factbase G^1 computing during the merging such that x has already been studied by the algorithm, and there exists a variable $y \in \mathbf{Vars}(G^1)$ that will be merged on t and such that $R(y, x) \in G^1$. There exists then a term t_0 and a binary predicate S such that $S(t_0, t'), S(t_0, y) \in G^1$. We note G^2 the factbase obtained after the merging of t' and y. G^1 is the left figure and G^2 is the right figure (we do not represent all the graphs):



We have $t_0 \prec t' \prec t$ and $t_0 \prec y \prec x$ so x is treated by the algorithm after the merging of t' and y so the algorithm will merge t and x: contradiction.

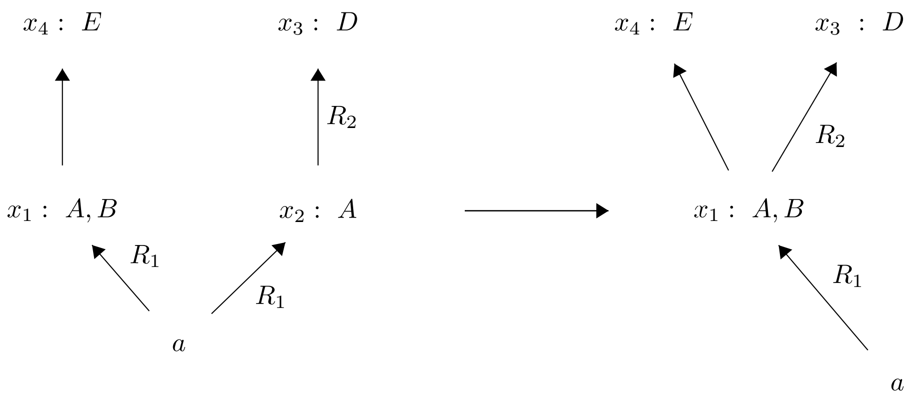
□

The merging is actually computing something better than the core of the factbase F': it is computing the core of a factbase that could have been obtained by applying some restricted triggers to F'.

We will now consider a new chase:

Definition 2.6. The *merge chase* is the core chase using the merging instead of computing the core, and can be applied only in the Horn-$\mathcal{ALC}$ theory.

Example 2.1. In the figure below, the merging of the factbase of the left gives the factbase of the right due to the homorphism $h = \{x_1 \mapsto x_1, x_2 \mapsto x_1, x_3 \mapsto x_3\}$.



Proposition 2.3. *Let G be a factbase that occured in a core chase derivation of the knowledge base O. $\mathit{Merge}(G)$ is a core.*

Proof. Suppose by contradiction that $\mathit{Merge}(G)$ is not a core. There exists $G' \subsetneq \mathit{Merge}(G)$ such that G' is a retract of $\mathit{Merge}(G)$. By proposition 1.1, there exists then a retractation h from $\mathit{Merge}(G)$ 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 core chase due to the axiom 3. So there exists a term t and a relation R such that $R(t,x) \in \mathit{Merge}(G)$. By minimality of x, $t \in \mathbf{Vars}(G')$. So, as h is a retractation: $h(t) = t$, so $h(R(t,x)) = R(t, h(x))$. $x \notin G'$ and $h(x) \in G'$ so $h(x) \neq x$. Let $A \in C_{\mathit{Merge}(G)}(x)$. $h(A(x)) \in \mathit{Merge}(G)$ so $A(h(x)) \in \mathit{Merge}(G)$ so $C_{\mathit{Merge}(G)}(x) \subseteq C_{\mathit{Merge}(G)}(h(x))$. Consequently, $h(x)$ is a sibling of x in $\mathit{Merge}(G)$. So, by proposition 2.2, 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.

Proposition 2.4. *Let G be a factbase that occured in a core chase derivation of the knowledge base O. There exists a core chase derivation $D = F_0, F_1, ..., F_n$ of O such that $F_n = \mathit{Merge}(G)$.*

Sketch of the proof. We prove it only in the case where the merging algorithm merges only one couple of sibling. It will then be direct to prove by induction the general case where they are several merging of siblings during the algorithm. So there exists a unique term t and a unique variable x such that t is a powerfull sibling of x in G. By definition of G, there exists a core chase derivation F_0,...,F_k of O such that $G = F_k$. The intuitive idea is to extend this derivation by computing on t all the retricted triggers that have been applied to x and the succesors of x (by $\prec^+$) and then to terminate the derivation, we compute the core. It will work because all the pieces of information that x has, t has them too, and because the computation of the core will then supress all the branch starting at x but will not supress any variable of the branch of t (otherwise there would be more siblings). □

maybe I can do an induction on the number of trigger applied to x.

2.3 Generalisation

Definition 2.7 (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:

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

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

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

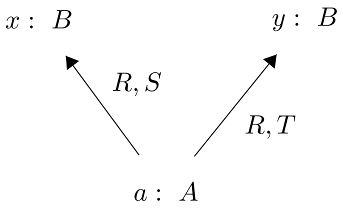
2.3.1 Horn-$\mathcal{ALCH}$

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:

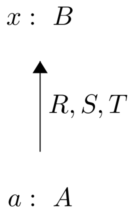
Example 2.2. If we have the knowledge base $O = (R, F)$ where $F = \{A(a)\}$ and the rules are :

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

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



Then, when we apply a merging:



The merging is bad because a core chase will never create this factbase.

Definition 2.8. For a factbase F' and two terms t and t', we note $R_{F'}(t, t')$ the set of binary predicates P such that $P(t, t') \in F'$.

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

Definition 2.9 ($\mathcal{ALCH}$-Siblings)**.** Let F' be a factbase that occured in a core chase derivation of the knowledge base O. Two terms t_1 and t_2 such that $t_1 \neq t_2$ are *$\mathcal{ALCH}$-siblings* if there exists a term t such that $R_{F'}(t, t_1) \neq \emptyset$, $R_{F'}(t, t_2) \neq \emptyset$, and $(C_{F'}(t_1) \subseteq C_{F'}(t_2) \wedge R_{F'}(t_1) \subseteq R_{F'}(t_2))$ or $(C_{F'}(t_2) \subseteq C_{F'}(t_1) \wedge R_{F'}(t_2) \subseteq R_{F'}(t_1))$. In this case, t_1 is a *powerful $\mathcal{ALCH}$-sibling* of t_2 if $C_{F'}(t_2) \subseteq C_{F'}(t_1) \wedge R_{F'}(t_1) \subseteq R_{F'}(t_2)$. Otherwise, it is a *powerless $\mathcal{ALCH}$-sibling.*

The merge chase will now use the $\mathcal{ALCH}$-siblings notion instead of the siblings notion.

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] 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.

[3] Ashok K. Chandra and Philip M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing*, STOC '77, page 77–90, New York, NY, USA, 1977. Association for Computing Machinery.

[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.