Unification in $\mathcal{FL}_0$ modulo a flat TBox*

Barbara Morawska

Ahmedabad University, India, barbara.morawska@ahduni.edu.in

Abstract. The small description logic $\mathcal{FL}_0$ deals with concepts constructed from concept names and roles with conjunction and value restriction for the roles. It is known that deciding the subsumption between concepts in this logic is polynomial, and unification is ExpTime-complete. If we consider the subsumption problem modulo a set of ground axioms the problem is ExpTime-complete. In this paper we study unification in $\mathcal{FL}_0$ modulo a set of ground axioms. We do not solve the problem in general, but focus on a very restricted case, namely the case where axioms are flat subsumptions between concept names. In this restricted case the subsumption is still polynomial, and we show that the unification is ExpTime-complete.

1 Introduction

The $\mathcal{FL}_0$ description logic is a logic which deals with the concepts constructed only from a set of concept names, $\mathbf{N}$, and a set of role names, $\mathbf{R}$, top symbol $\top$, and constructors: conjunction $\sqcap$, value restriction $\forall r.C$, where $r \in \mathbf{R}$ and C is an $\mathcal{FL}_0$ concept.

Unification of concepts in the description logic $\mathcal{FL}_0$ without a TBox (a set of ground axioms) was proved to be ExpTime complete in [5].

Unification in $\mathcal{FL}_0$ modulo a general TBox remains unsolved. In this paper, we will see how to solve this problem for a very restricted TBox.

The algorithm is going to use a decision procedure for subsumption between $\mathcal{FL}_0$ concepts modulo a TBox. Without a TBox (or with an empty TBox) subsumption in $\mathcal{FL}_0$ is polynomial. In [1] deciding subsumption between $\mathcal{FL}_0$ concepts modulo a TBox was proved to be ExpTime-complete. We will see that for the restricted TBox in this paper, the subsumption is still polynomial.

The content of this paper as related to the result in [4], where it was shown that the general unification in ACUI-theory modulo a set of ground axioms consisting of constants only, is NP-complete. $\mathcal{FL}_0$ can be treated as the equational theory ACUI (for conjunction) extended with homomorphism axioms for unary function symbols (for value restrictions). Our result differs form the above mentioned in this, that we do not allow free functions, but only homomorphisms. Hence this result is a step forward towards the goal of the authors of [4], to solve unification in $\mathcal{FL}_0$ modulo a general TBox.

On the other hand the presented result can be viewed as a generalization of our unification algorithm from [6]. In that paper we have encoded the $\mathcal{FL}_0$ unification problem as a set of restricted anti-Horn clauses, and we proved that a finite Herbrand model of these clauses defines a solution for the unification problem. In order to decide the existence of a finite Herbrand model, we use a concept of a *shortcut*, that defines an assignment of terms to predicates, which may be a part of a model.

In this paper we will analyze the unification problem directly, without an encoding into anti-Horn clauses. The idea though is the same as in [6], to use a kind of *shortcuts* to define an assignment for variables, which may constitute a part of a solution. Our algorithm is in ExpTime complexity class. The problem is also ExpTime-hard, because unification in $\mathcal{FL}_0$ with the empty TBox is already ExpTime-hard.[1] Hence $\mathcal{FL}_0$ unification modulo a *flat* TBox is ExpTime-complete.

We assume some knowledge of Description Logics and of $\mathcal{FL}_0$ in particular, and refer the reader to [2].

2 Normal form, TBox and subsumption

Normal form of an $\mathcal{FL}_0$ concept
An $\mathcal{FL}_0$ concept C is in *normal form*[2] if it is of the form: $\forall v_1.A_1 \sqcap \cdots \sqcap \forall v_n.A_n$, where v_i are words over a set of role names $\mathbf{R}$ and $A_i \in \mathbf{N}$ are concept names. If $v_i = \epsilon$, then $\forall v_i.A_i = \forall \epsilon.A_i = A_i$. Notice that the concept names can be repeated, as they can be *reachable* by different role words. A simple conjunct of a concept in this form, $\forall v_i.A_i$, with $v_i \in \mathbf{R}^*$, is called a **particle**.

Since an $\mathcal{FL}_0$ concept in the normal form is a conjunction of particles, we treat it as a **set of particles** and use the notation: $P \in C$ to indicate that P is a conjunct of a conjunction of particles C. Similarly, we use the subset relation to compare conjunctions. In accordance of the meaning of conjunction, the empty conjunction (or the empty set) of particles is understood as $\top$, the concept that subsumes every other concept.

Since in the usual $\mathcal{FL}_0$ concept, conjunction can appear also under universal value restriction, changing C into this normal form may increase the size, but only polynomially, since each particle in the normal form of C has to correspond to a unique leaf in the parse tree of the original concept.

Below we assume that all concepts are in this normal form. As sets of particles, we always understand them modulo associativity, commutatity and idempotence, even when we write them using conjunction constructor.

$\mathcal{FL}_0$ TBox and subsumption
We assume that an $\mathcal{FL}_0$ TBox is a set of general concept inclusions (GCIs) between concepts in normal form. These GCIs are also called axioms.

[1] Our algorithm solves also unification in $\mathcal{FL}_0$ with the empty TBox.
[2] There are different choices of normal forms for $\mathcal{FL}_0$ concepts possible. We choose the one that is most suitable for our purposes.

By properties of conjunction and equivalence, we can transform these axioms into the subsumptions with one particle on the right hand-side:
$\forall v_1.A_1 \sqcap \cdots \sqcap \forall v_n.A_n \sqsubseteq \forall w.B$, where v_i, w are words over the set of role names $\mathbf{R}$ and $A_i, B \in \mathbf{N}$ are concept names.

Equivalence between particles $\forall v.A \equiv \forall w.B$ is just a short form of two subsumptions in the opposite directions.

Rewrite step
We define a rewrite step as a notion describing an application of an axiom from a TBox $\mathcal{T}$.
$C \sqcap \forall v.C_1 \sqcap \cdots \sqcap \forall v.C_m \sqsubseteq_{\mathcal{T}} C \sqcap \forall v.D$ is called a *rewrite step at position p* iff $C_1 \sqcap \cdots \sqcap C_m \sqsubseteq D$ is a GCI in $\mathcal{T}$ and v is a word over $\mathbf{R}$ of the length p.
C in the above definition is any $\mathcal{FL}_0$ concept in normal form and $C_1, \ldots, C_m, D$ are particles.

Subsumption modulo a TBox
In the presence of a TBox, subsumption between two $\mathcal{FL}_0$ concepts $C_1 \sqsubseteq_{\mathcal{T}} C_2$ in normal forms occurs if and only if one of the following cases holds:

- $C_2 \subseteq C_1$ (inclusion step)
- There is a rewrite step between C_1 and C_2 at some position.
- There is a sequence of inclusion or rewrite steps:
  $C'_1 \sqsubseteq_{\mathcal{T}} C'_2, C'_2 \sqsubseteq_{\mathcal{T}} C'_3, \ldots, C'_{n-1} \sqsubseteq_{\mathcal{T}} C'_n$, such that $C_1 = C'_1$ and $C_2 = C'_n$. (transitivity of subsumption)

This characterization of subsumption in $\mathcal{FL}_0$ follows directly from the semantic properties of constructors of $\mathcal{FL}_0$.[3]

Now, we assume that a TBox has a **special form**. It contains only *flat* subsumptions of the form: $A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B$, where $A_1, \ldots, A_n, B$ are concept names. We call such TBox a **flat** TBox.

Obviously, in such a TBox, every rewrite step can only have the form
$C' \sqcap \forall v.A_1 \sqcap \cdots \sqcap \forall v.A_n \sqsubseteq_{\mathcal{T}} C' \sqcap \forall v.B$, where each particle has the same role prefix v.

Hence if this rewrite step is applied to C, $C \sqsubseteq_{\mathcal{T}} D$, then there must be particles of the same height: $\forall v.A_1, \ldots, \forall v.A_n$ in C, and they are replaced by a particle of the same height, $\forall v.B$ in D.

Now we show that the subsumption problem w.r.t. a flat TBox is in P.[4]

We define a saturation of an $\mathcal{FL}_0$ concept C w.r.t. GCIs in a TBox $\mathcal{T}$. In this definition $[\forall v.C]$ denotes a concept $\forall v.C$ brought to the normal form. Hence e.g. $[\forall v.(A \sqcap B)]$ denotes $\forall v.A \sqcap \forall v.B$ or equivalently a set of particles $\{\forall v.A, \forall v.B\}$.

[3] Without a TBox, subsumption in $\mathcal{FL}_0$ is characterized by the first and third condition. In the presence of a TBox, rewrite steps allow us to show additional subsumptions between $\mathcal{FL}_0$ concepts. Since we assume that concepts are in normal form, they are treated as sets of particles.

[4] We follow here the idea from [4].

1. We start with $C^* := C$.
2. As long as there is a GCI $C_1 \sqsubseteq C_2$ in $\mathcal{T}$, such that $[\forall v.C_1] \subseteq C^*$ and $[\forall v.C_2] \not\subseteq C^*$, redefine $C^* := C^* \cup [\forall v.C_2]$.

Saturation terminates (because $\mathcal{T}$ is finite and flat) and $C^* \equiv_{\mathcal{T}} C$.
Obviously, if $\mathcal{T}$ is not flat, this process may not terminate.

Lemma 1. *Let C, D be $\mathcal{FL}_0$ concepts and $\mathcal{T}$ a flat $\mathcal{FL}_0$ TBox. Then $C \sqsubseteq_{\mathcal{T}} D$ if and only if $D \subseteq C^*$.*

Example 1. Let $\mathcal{T} = \{A \sqcap B \sqsubseteq C,\ B \sqcap C \sqsubseteq D\}$.
Let $C = \{\forall r.A, \forall r.B\}$.
Then $C^* = \{\forall r.A, \forall r.B, \forall r.C, \forall r.D\}$. Obviously, $C \sqsubseteq_{\mathcal{T}} \forall r.D$

Corollary 1. *Subsumption problem in $\mathcal{FL}_0$ modulo a flat TBox is polynomial.*

Proof. In order to decide if $C \sqsubseteq_{\mathcal{T}} D$, we saturate C. This process terminates in polynomial time, because $\mathcal{T}$ is finite and the saturation may be executed only the number of times corresponding to the number of GCIs in $\mathcal{T}$ times the number of prefixes present in the particles of C. □

Most crucial observation for our unification procedure is that for every subsumption of the form $\forall v_1.C_1 \sqcap \cdots \sqcap \forall v_n.C_n \sqsubseteq_{\mathcal{T}} \forall v.D$, also the subsumption $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq_{\mathcal{T}} D$ holds, where $C_1, \ldots, C_n, D$ are constants. This is true for a flat TBox. We can prove the following, even stronger result.

Lemma 2. *Let $\mathcal{T}$ be a flat TBox. Then for every $v, v' \in \mathbf{R}^*$,*
$\forall v.A_1 \sqcap \cdots \sqcap \forall v.A_n \sqsubseteq_{\mathcal{T}} \forall v.B$ if and only if $\forall v'.A_1 \sqcap \cdots \sqcap \forall v'.A_n \sqsubseteq_{\mathcal{T}} \forall v'.B$.

Proof. If $\forall v.A_1 \sqcap \cdots \sqcap \forall v.A_n \sqsubseteq_{\mathcal{T}} \forall v.B$ then $A_1 \sqcap \cdots \sqcap A_n \sqsubseteq_{\mathcal{T}} B$ because $\mathcal{T}$ is flat and then $\forall v'.A_1 \sqcap \cdots \sqcap \forall v'.A_n \sqsubseteq_{\mathcal{T}} \forall v'.B$ by $\mathcal{FL}_0$ properties.[5] □

3 Unification problem

Unification problem is defined as a set of goal subsumptions between $\mathcal{FL}_0$ concepts in normal form: $\Gamma = \{C_1 \sqsubseteq^?\ D_1, \ldots, C_n \sqsubseteq^?\ D_n\}$.

Each of the goal subsumptions contains concepts in normal form, hence it is of the form: $\forall v_1.A_1 \sqcap \cdots \sqcap \forall v_m.A_m \sqsubseteq^?\ \forall v.B$, where v_i (or v) may be empty and A_i (or B) is a concept name.

Some of the concept names in the goal are **variables**. The concept names that are not variables are called **constants**. The set of goal variables is denoted by **Var**. $\mathcal{FL}_0$ concepts, subsumptions and sets of subsumptions that do not contain variables are called **ground**. We assume a flat, ground TBox $\mathcal{T}$.

A solution for the unification problem modulo $\mathcal{T}$ is a substitution of ground concepts for the variables, such that the goal subsumptions are true modulo the TBox $\mathcal{T}$. Notice that we are searching for ground substitutions, hence if a

[5] Value restrictions behave like homomorphism.

variable is assigned an empty conjunction of particles, we understand that it is substituted by the top constructor, $\top$. If a goal subsumption is of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, D is a variable, and a substitution γ assigns $\top$ to D, we say that γ satisfies this subsumptions *voidly*.

Flattening

The flattening of goal subsumptions is a kind of decomposition, which is a part of unification procedure. We flatten the goal subsumptions by a transformation using fresh variables and adding new goal subsumptions. A goal subsumption $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, where $C_1, \ldots, C_n, D$ are particles, is **not flat** if D is of the form $\forall r.D'$ or there is an i, $1 \leq i \leq n$, such that C_i is of the form $\forall r.C_i'$. In order to flatten such a subsumption, we will introduce *decomposition* variables, e.g. for a variable X, a decomposition variable would be denoted X^r. Such a variable will be defined by an increasing subsumption $X \sqsubseteq^? \forall r.X^r$ and a decreasing rule 1 introduced later. An increasing subsumption is added automatically to the unification problem, at the moment of the creation of a decomposition variable. The set of increasing subsumptions is maintained separately from other goal subsumptions, and is not subject to flattening. For a given role r and a variable X, X^r is unique, if it exists.

In a flattening process we will use the following notation. If P is a particle and r a role name ($r \in \mathbf{R}$), we define P^{-r} in the following way:

$$P^{-r} = \begin{cases} P^r & \text{if } P \text{ is a variable and } P^r \text{ is a decomposition variable} \\ P_i' & \text{if } P_i = \forall r.P_i' \\ \top & \text{in all other cases} \end{cases}$$

If s is a goal subsumption, $s = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, where $C_1, \ldots, C_n, D$ are particles, we define $s^{-r} = C_1^{-r} \sqcap \cdots \sqcap C_n^{-r} \sqsubseteq^? D^{-r}$.

If P is a particle, then we define P^{cons} in the following way:

$$P^{cons} = \begin{cases} P & \text{if } P \text{ is a constant or variable} \\ \top & \text{in all other cases} \end{cases}$$

If s is a goal subsumption, $s = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, where $C_1, \ldots, C_n, D$ are particles, we define $s^{cons} = C_1^{cons} \sqcap \cdots \sqcap C_n^{ccons} \sqsubseteq^? D^{cons}$.[6]

The subsumptions obtained in the process of flattening are called:

- **Start subsumptions:** of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ and D is a constant.
- **Flat subsumptions:** of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ and D is a variable.
- **Increasing subsumptions:** of the form $C \sqsubseteq^? \forall r.C^r$ for a role name r.

Now we define our flattening procedure in Figure 1.

After the flattening is done on Γ in an exhaustive way, we can remove the goal subsumptions with $\top$ on the right hand-side, as trivially satisfied by any substitution.

The flattening process is non-deterministic because of guessing in 3b. This is a polynomial guess, hence the process adds a non-deterministic polynomial step

[6] This means that the particles that are not constants or variables are deleted from s.

Given a **non flat** goal subsumption: $s = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$,

1. If D is of the form $\forall r.D'$,
   then replace s with s^{-r}.
2. If D is a constant (s is not flat, hence there is C_i of the form $\forall r.C_i'$), then replace s with s^{cons}.
3. If D is a variable (s is not flat, hence there is C_i of the form $\forall r.C_i'$), delete s from Γ and add the following goal subsumptions:
   (a) for each $r \in \mathbf{R}$, add s^{-r},
   (b) guess a set of constants A_D in the constants of $\mathcal{T}$ and Γ,
   (c) for each $C \in A_D$, add
   - $D \sqsubseteq^? C$ and
   - $C_1^{cons} \sqcap \cdots \sqcap C_n^{cons} \sqsubseteq^? C$

Fig. 1. Flattening of Γ

to the unification procedure. The complexity of the unification is dominated by the algorithm explained in the next section.

The flat subsumptions of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, where D is a variable, may have constants or variables on the left hand sides. We call such subsumptions with constants on the left hand sides, *mixed subsumptions*.

Now we define *pure* subsumptions by deleting all constants from mixed subsumptions in the goal. For example, if $X \sqcap A \sqsubseteq^? Y$ is a mixed subsumption in Γ, then $X \sqsubseteq^? Y$ is the corresponding *pure subsumption*.

Pure subsumptions are not part of the goal, but they have the following property: if γ unifies pure subsumptions, then it also unifies the corresponding mixed subsumptions. We will use the pure subsumptions later on in our unification procedure.

The meaning of a decomposition variable Z^r is that in a solution it should hold exactly those particles P, for which $\forall r.P$ is in the substitution for Z. The process of solving the unification problem will determine which particles should be in the solution of Z.

An increasing subsumption does not suffice to express this relation between the substitution for Z and that for Z^r. In order to properly characterize the meaning of a decomposition variable Z^r, we need to add another restriction on a substitution, which cannot be expressed as a goal subsumption, but rather as an implication, which we call a **decreasing rule:**

$$Z \sqsubseteq \forall r.P \implies Z^r \sqsubseteq P \tag{1}$$

where P is a ground particle. The meaning of this restriction is that whenever a ground particle of the form $\forall r.P$ is in the substitution for Z, then P has to be in the substitution for the decomposition variable Z^r. The reason is illustrated in the next example.

Example 2. Let our unification problem contain the goal subsumptions:
$Z \sqsubseteq^? \forall r.A, \forall r.A \sqsubseteq^? Z, Z \sqsubseteq^? X, X \sqsubseteq^? \forall r.B$. In this example, we assume that the

TBox is empty. The flattened goal is then:
start subsumptions: $Z^r \sqsubseteq^? A, \quad X^r \sqsubseteq^? B$; flat subsumptions: $A \sqsubseteq^? Z^r$, $Z \sqsubseteq^? X$; increasing subsumptions: $Z \sqsubseteq^? \forall r.Z^r$, $X \sqsubseteq^? \forall r.X^r$.

The first start subsumption forces A into a substitution for Z^r and thus by the first increasing subsumption $\forall r.A$ must be in the substitution for Z. Similarly, X^r gets B and X gets $\forall r.B$. By the second flat subsumption we know that $\forall r.B$ must be also in the substitution for Z. But there is nothing that can force B into Z^r, if we do not use the decreasing rule. If we do apply the decreasing rule 1, then B is forced into Z^r, but then we discover that the goal is not unifiable, because $A \not\sqsubseteq_\emptyset B$.

Without applying the decreasing rule 1, the following substitution would be wrongly accepted as a solution:
$Z \mapsto \{\forall r.A, \forall r.B\}, Z^r \mapsto \{A\}, X \mapsto \{\forall r.B\}, X^r \mapsto \{B\}$

The process of flattening of the unification problem obviously terminates in polynomial time with polynomial increase of the size of the goal. It is bounded by the size of the original problem. We call a flat unification problem, **normalized**.

Lemma 3. *Let Γ be a unification problem which contains a goal subumption which is not flat. Let $\mathcal{T}$ be a flat $\mathcal{FL}_0$ TBox.*

There is a right application of a rule from Figure 1, such that and Γ' a unification problem obtained from Γ by this application satisfies the following claim: γ is a ground solution of Γ w.r.t. $\mathcal{T}$ iff there is a substitution γ' that is a solution of Γ' modulo $\mathcal{T}$, where γ' is an extension of γ to some new variables.

Notice that the decreasing rule is not mentioned in the formulation of the above lemma. This is because if a substitution γ (or γ') is a ground unifier of Γ (or Γ'), then even if the decreasing rule is not satisfied by the assignments of γ (or γ'), this can be easily repaired by redefining the assignment for every decomposition variable: $\gamma(X^r) := \{P \mid \forall r.P \in \gamma(X)\}$. Because of this, knowing that a substitution γ is a solution for Γ, we can assume about it that the decreasing rule is satisfied by it.

Example 3. We illustrate in this example the flattening process.
Let $\Gamma = \{A \sqcap \forall s.X \sqsubseteq^? \forall r.Y,\ B \sqcap \forall r.Y \sqsubseteq^? A,\ \forall s.Y \sqcap \forall r.X \sqsubseteq^? X\}$

- Let $s = A \sqcap \forall s.X \sqsubseteq^? \forall r.Y$. The first case of flattening applies and s is replaced by s^{-r}. $s^{-r} = \top \sqsubseteq^? Y$.
- Let $s = B \sqcap \forall r.Y \sqsubseteq^? A$. The second case of flattening applies and s is replaced by s^A. $s^A = B \sqsubseteq^? A$
- Let $s = \forall s.Y \sqcap \forall r.X \sqsubseteq^? X$. The third case applies. We guess that $A_X = \emptyset$. Then s is replaced by s^{-s} and s^{-r}. $s^{-s} = Y \sqsubseteq^? X^s$. $s^{-r} = X \sqsubseteq^? X^r$. Since two decomposition variables are introduced, we have to add two increasing goal subsumptions. $X \sqsubseteq^? \forall s.X^s$ and $X \sqsubseteq^? \forall r.X^r$. Additionally we will require that the decreasing rule 1 restricts the assignment for the decomposition variables.

Finally the flattened Γ has the form:
start subsumption: $B \sqsubseteq^? A$; flat subsumptions: $\top \sqsubseteq^? Y,\ Y \sqsubseteq^? X^s,\ X \sqsubseteq^? X^r$; increasing subsumptions: $X \sqsubseteq^? \forall s.X^s,\ X \sqsubseteq^? \forall r.X^r$

We can notice that the goal has a solution for any TBox $\mathcal{T}$ for which $B \sqsubseteq_{\mathcal{T}} A$. For example a solution assigns $\top$ to all variables in Γ.

Shortcuts
From a syntactic point of view a shortcut is just a pair $(\mathcal{X}, \bar{t})$ where $\mathcal{X}$ is a set of variables and $\bar{t}$ is a vector of sets of particles of the same height. It represents an assignment of corresponding sets of particles from $\bar{t}$ to variables from $\mathcal{X}$. $\mathcal{X}(\bar{t}) = [X_1 \mapsto t_1, X_2 \mapsto t_2, \ldots, X_n \mapsto t_n]$. We require that $\mathcal{X}(\bar{t})$ satisfies the flat clauses of a given unification problem. We will also make sure that for each valid shortcut $(\mathcal{X}, \bar{t})$ there is a substitution γ of which $\mathcal{X}(\bar{t})$ is a part such that all minimal particles of in the range of γ are in $\bar{t}$, and such that γ satisfies all subsumptions of Γ, including decreasing rule for the particles of greater heights.

The idea behind our unification algorithm is that a solution γ may be divided into assignments of the particles of the same height to variables in the unification problem. These particles behave in an independent way as far as the satisfiability of flat clauses is concerned. The only connections between particles of different heights is by the increasing subsumptions and the decreasing rule.

If $(\mathcal{X}, \bar{t})$ is a valid shortcut and it satisfies the decreasing rule for the particles in $\bar{t}$ it is called **closed**. If additionally it satisfies all start subsumptions, it is called **complete**. Obviously, there is a solution for a unification problem Γ, if there is a shortcut that is complete.

In the following part, we will show that we do not need to look for shortcuts for all possible assignments of all possible particles to variables, which would be impossible, given infinitely many possible particles. We can restrict ourselves to the shortcuts with assignment of sets of constants only. These shortcuts of the form $(\mathcal{X}, \bar{c})$, where $\bar{c}$ is a vector of sets of constants, are called c-shortcuts.

They are defined as follows.

Definition 1. *Let $\mathcal{X} \subseteq$* **Var**. *The pair of set of variables $\mathcal{X}$ and a vector of sets of constants $\bar{c}$, $(\mathcal{X}, \bar{c})$ is called a* c-shortcut (for Γ) *if there is a substitution γ satisfying the following conditions:*

S1 *γ assigns the constants* ***only*** *from $\bar{c}$ and* ***only*** *to the variables from $\mathcal{X}$, according to the assignment $\mathcal{X}(\bar{c})$ defined by $(\mathcal{X}, \bar{c})$.*

S2 *All subsumptions in Γ and the decreasing rule are satisfied by γ, with possible exception of start subsumptions.*

The height *of γ is* $\max\{|w| \mid \forall w.A \in range(\gamma)$ *with A a constant* $\}$. *The* height *of the shortcut $(\mathcal{X}, \bar{c})$ is the smallest height of a substitution satisfying* ***S1*** *and* ***S2****.*

A special c-shortcut is of the form $(\emptyset, \emptyset)$, hence the empty assignment. This corresponds to the solution that substitutes all variables in the unification problem with $\top$. Notice that this substitution satisfies all flat subsumptions, increasing subsumptions and the decreasing rule. Hence it is a legitimate shortcut. We say that this c-shortcut has height 0.

Theorem 1. *A unification problem Γ has a solution modulo a flat TBox $\mathcal{T}$ iff there is a complete c-shortcut for Γ.*

An interesting property of some of the c-shortcuts is the following: if $(\mathcal{X}, \bar{c})$ is a c-shortcut for some vector $\bar{c}$, then it provides a substitution γ satisfying the conditions of Definition 1. Now for some of these c-shortcuts we can *lift* this substitution in such a way, that instead of constants it assigns the same constants prefixed with a common word v from $\mathbf{R}^*$ (the particle is $\forall v.C$). We construct a new substitution γ', which assigns to each variable in $\mathcal{X}$ particles of the form $\forall v.C$, whenever γ assigned a constant C. γ' satisfies the same subsumptions as γ with exception of start subsumptions and the decreasing rule activated by the particles replacing constants.[7]

Not all c-shortcuts have this property. Some subsumptions maybe satisfied by γ because of constants present on the left side of flat, mixed subsumptions. We cannot *substitute* for constants. Hence in such case, we cannot lift γ to γ'. The possibility of *lifting* of a substitution to particles of bigger height occurs only if γ satisfies *pure* counterparts of all flat subsumptions in the goal, where the pure subsumptions with $\top$ on the left hand-side are satisfied voidly by γ.

The idea for solving $\mathcal{FL}_0$ unification w.r.t. a flat TBox is the following: compute all possible c-shortcuts of height 0, check if there is a complete c-shortcut among them, and if not then identify the c-shortcuts that can be used to compute c-shortcuts for height 1, compute the c-shortcuts of height at most 1 with the help of the shortcuts of height 0, and so on, until we find a c-shortcut which is complete.

For constructing new c-shortcuts, we define a notion of an **f-output of a shortcut** $(\mathcal{X}, \bar{c})$ as a pair $(\mathcal{X}^{+f}, \bar{c})$, where $\mathcal{X}^{+f} = \{X^f \in \mathbf{Var} \mid X \in \mathcal{X}\}$ such that $\bar{c}[i] = \{C \mid [X_i \mapsto C] \in \mathcal{X}(\bar{c})\}$. This means that in $\mathcal{X}^{+f}(\bar{c})$ the same constant C which was assigned to X is now assigned to X^f. With the help of f-output, we make sure that the decreasing rule will be always satisfied by the solution we search for.

Example 4. In this example we see how c-shortcuts can be used to construct a solution. Let $\mathcal{T} = \{A \sqcap C \sqsubseteq B\}$.
$\Gamma = \{$start subsumptions: $X \sqsubseteq^? C$; flat subsumptions: $Y^f \sqcap A \sqsubseteq^? X, Z \sqcap X \sqsubseteq^? Y$; increasing subsumption: $Y \sqsubseteq^? \forall f.Y^f$ $\}$

The pure counterparts for the flat subsumptions are: $Y^f \sqsubseteq^? X,\ Z \sqcap X \sqsubseteq^? Y$.

First we compute all c-shortcuts of height 0. Among others we will discover the c-shortcut $S_1 = (\{Y, Z\}, < \{C\}, \{C\} >)$. Since this shortcut defines an assignment that satisfies the *pure* subsumptions, we label it as *usable*. Since this is not a complete c-shortcut (the start subsumption is not satisfied), then we look for other c-shortcuts.

We check if $S_2 = (\{X, Y^f, Y, Z\}, < \{C\}, \{C\}, \{B\}, \{A\} >)$ is a c-shortcut. For that we need a c-shortcut with Y included, and f-output in S_2. We can use S_1, because its f-output, $(\{Y^f\}, < \{C\} >)$ is included in S_2. S_2 is thus a legitimate c-shortcut. It happens also that S_2 is complete.

[7] Notice that constants never activate this rule.

Algorithm 1 MAIN$(\Gamma, \mathcal{T})$

Input: Γ a normalized unification problem, $\mathcal{T}$ a flat TBox
Output: True if there is a complete and closed c-shortcut for Γ, False otherwise

```
1: 𝔖_0 ← ∅
2: i = 0
3: for all i ≥ 0 do
4:     𝔖_{i+1} ← nextShortcuts(Γ, 𝔖_i, 𝒯),
5:     if there is a complete shortcut in 𝔖_{i+1} then
6:         return True
7:     else if 𝔖_{i+1} = 𝔖_i then
8:         return False
9:     end if
10: end for
```

From the two shortcuts S_2 and S_1 we can construct the following solution: $\gamma = [X \mapsto \{C\}, Y^f \mapsto \{C\}, Y \mapsto \{B, \forall f.C\}, Z \mapsto \{A, \forall f.C\}]$. Notice how the decreasing rule 1 is satisfied by the solution, because the f-output of S_1 is included in S_2.

4 Algorithm

The unification procedure consists of two stages.

1. Flattening of a given unification problem.
2. Running Algorithm MAIN 1 on the normalized problem.

Since the flattening process contains a non-deterministic component (it is in NP), in the case the main algorithm fails, one has to try different choice of constants for variables in the process of flattening, in the first step.

Now we look at the algorithms: Algorithm MAIN 1 and Algorithm NEXTSHORTCUTS 2. Γ_{flat} denotes the set of flat subsumptions from Γ and Γ_{pure} their pure counterparts.

Correctness of Algorithm Main 1
The **soundness** of the main algorithm follows from Theorem 1. If there is a complete c-shortcut, then there is a solution to the unification problem given by Definition 1. The **completeness** of this algorithm follows from the fact that given a ground solution γ for a problem, we can extract from it a c-shortcut $(\mathcal{X}, \bar{c})$, by grouping in $\mathcal{X}$ all variables assigned a constant by γ, and defining vector $\bar{c}$ as a vector of sets of constants assigned to variables by γ. This shortcut must be complete (start subsumptions are satisfied by γ and they are satisfied by constants mentioned in $(\mathcal{X}, \bar{c})$). This shortcut must have a height which is equal or smaller than the height of γ. Hence if Algorithm 2 is sound and complete, the main algorithm will detect the existence of this shortcut and terminate with success.

Algorithm 2 NEXTSHORTCUTS$(\Gamma, \mathfrak{S}_{in}, \mathcal{T})$

Input: Γ a normalized unification problem, $\mathfrak{S}_{in}$ a set of already computed c-shortcuts, $\mathcal{T}$ a flat TBox

Output: $\mathfrak{S}_{out}$ all c-shortcuts computed based on $\mathfrak{S}_{in}$ in the input

1: Let $\mathfrak{S}_{out} = \{\}$.
2: **for all** $(\mathcal{X}, \bar{c})$ where $\mathcal{X}$ is a set of variables and $\bar{c}$ an assignment of sets of constants to variables in $\mathcal{X}$ **do**
3: $\quad\gamma_{temp} \leftarrow \{P \mapsto C \mid P$ is assigned C in $(\mathcal{X}, \bar{c})\}$
4: $\quad$**if** $\mathcal{T} \models \gamma_{temp}(\Gamma_{flat})$ **then**
5: $\quad\quad$**for all** $f \in \mathbf{R}$ **do**
6: $\quad\quad\quad\mathcal{X}^{-f} \leftarrow \{P \in \mathbf{Var} \mid C \in \gamma_{temp}(P^f)\}$
7: $\quad\quad\quad$**if** $\mathcal{X}^{-f} \neq \emptyset$ **then** ▷ check if a suitable shortcut is present in $\mathfrak{S}_{in}$
8: $\quad\quad\quad\quad$**if** there is no $(\mathcal{Z}, \bar{c}) \in \mathfrak{S}_{in}$ labeled *usable*, such that $\mathcal{X}^{-f} \subseteq \mathcal{Z}$ and f-output of $(\mathcal{Z}, \bar{c})$ is included in $(\mathcal{X}, \bar{c})$. **then**
9: $\quad\quad\quad\quad\quad$fail for this choice of $(\mathcal{X}, \bar{c})$
10: $\quad\quad\quad\quad$**end if**
11: $\quad\quad\quad$**end if**
12: $\quad\quad$**end for**
13: $\quad\quad$**if** $\mathcal{T} \models \gamma_{temp}(\Gamma_{pure})$ and for every $\top \sqsubseteq^? X \in \Gamma_{pure}$, $\gamma_{temp}(X) = \top$ **then**
14: $\quad\quad\quad$label $(\mathcal{X}, \bar{c})$ as *usable*
15: $\quad\quad$**end if**
16: $\quad\quad\mathfrak{S}_{out} \leftarrow \mathfrak{S}_{out} \cup \{(\mathcal{X}, \bar{c})\}$
17: $\quad$**end if**
18: **end for**
19: **return** $\mathfrak{S}_{out}$

Somewhat technical proof of the correctness of Algorithm NEXTSHORTCUTS 2 is included in the appendix. Our main result is stated in the following theorem.

Theorem 2. *$\mathcal{FL}_0$-unification problem w.r.t. a flat TBox is* EXPTIME*-complete*

The argument for termination and complexity is based on the following observation. The number of all possible pairs $(\mathcal{X}, \bar{c})$ is exponential in the size of $\mathcal{T}$ and Γ. Hence the for-loop starting in line 2 in Algorithm 2 is executed exponentially many times and preforms polynomially many steps that take at most exponential time each. Since there are at most exponentially many c-shortcuts, the for-loop starting at line 3 may run only at most exponentially many times until the sets of subsequent shortcuts are the same.

The algorithm presented above is sound and complete for unification in $\mathcal{FL}_0$ modulo a flat TBox. It is also sound and complete for unification in $\mathcal{FL}_0$ modulo the empty TBox. This is because the subsumption checks in Algorithm 1 and Algorithm 2 are done modulo a given TBox. If a TBox is empty, then the subsumption checks will be just simpler.

In the case of the second algorithm, computing next shortcuts, the check is applied in line 4 ($\mathcal{T} \models \gamma_{temp}(\Gamma_{flat})$). We do this checks by applying substitution to the subsumptions and checking if they are true modulo the TBox. This check is polynomial, as shown in Section 2. The next check in line 13 ($\mathcal{T} \models \gamma_{temp}(\Gamma_{pure})$)

is done the same way. With the empty TBox, the checks are also polynomial of course.

In the main algorithm, we check if any shortcut in the set of shortcuts is complete (line 5). To do this, we apply the assignment of variables to the start subsumptions and check if they are true modulo the TBox. The start subsumptions can be satisfied only by constants, hence we need to check only the assignment defined be a c-shortcut, disregarding bigger particles. Hence we do this check by substituting variables given by the shortcut with constants and checking if the start subsumptions are true modulo the TBox. If the TBox is empty, this check is a bit simpler: we check inclusion, without computing the saturation of the left hand sides of the subsumptions.

Of course, with the empty TBox, we could much simplify our procedures. First the flattening of the unification problems may be more radical. Constants can be treated independently from each other. Hence we could split subsumptions further on, or delete the constants that are irrelevant for a unifier. Moreover the notion of shortcuts may be simplified too, because now vectors of constants may be required to contain only one constant. (Constants behave independently from each other). Finally, usable shortcuts will be usable for particles with any constant, hence we do not even need to mention a constant in their definition. These ideas were used in our paper [6]. The number of such shortcuts is then bounded by the number of all possible subsets of variables, hence it is at most exponential. Thus we have the same complexity in this simpler case of unification in $\mathcal{FL}_0$ with the empty TBox, as in the case of the unification in $\mathcal{FL}_0$ modulo a flat TBox.

5 Conclusions

We have proved the $\mathcal{FL}_0$-unification problem modulo a flat TBox is solvable in ExpTime. The algorithm shown in this paper is based on the notion of shortcuts, i.e. substitutions which satisfy some part of the problem.

This method works for a flat TBox. It seems though that it can be applied to other forms of TBoxes, provided that they satisfy some form of the property expressed in Lemma 2. Can we define properties of a TBox which has a kind of *flat* normal form, but is not completely flat, such that the same algorithm would work for it too?

The result in this paper is of course a small step in the direction of hopefully solving unification in $\mathcal{FL}_0$ modulo general TBox. Hence one would like to explore how to extend the algorithm for less restricted TBoxes.

Another interesting line of research would also be to study matching in $\mathcal{FL}_0$ modulo flat TBoxes. In [3] matching between $\mathcal{FL}_0$ concepts w.r.t. a general TBox, was shown to be ExpTime-complete, but in a restricted case it is of a polynomial complexity. One can ask if this case applies for the flat TBoxes.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Kaelbling, L.P., Saffiotti, A. (eds.) IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005. pp. 364–369. Professional Book Center, http://ijcai.org/Proceedings/05/Papers/0372.pdf
2. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
3. Baader, F., Fernández Gil, O., Marantidis, P.: Matching in the description logic FL0 with respect to general tboxes (extended abstract). In: Simkus, M., Weddell, G. (eds.) Proceedings of the 32nd International Workshop on Description Logics (DL'19). CEUR Workshop Proceedings, vol. 2373. CEUR-WS (2019)
4. Baader, F., Marantidis, P., Mottet, A.: ACUI unification modulo ground theories. In: Ayala-Rincón, M., Balbiani, P. (eds.) Proceedings of the 32th International Workshop on Unification (UNIF 2018). Oxford, UK (2018)
5. Baader, F., Narendran, P.: Unification of concept terms in description logics. Journal of Symbolic Computation **31**(3), 277–305 (2001). https://doi.org/10.1006/jsco.2000.0426
6. Borgwardt, S., Morawska, B.: Finding finite Herbrand models. In: Bjrner, N., Voronkov, A. (eds.) Proc. of the 18th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR-18). Lecture Notes in Computer Science, vol. 7180, pp. 138–152. Springer (2012), https://doi.org/10.1007/978-3-642-28717-6_13

A Proofs omitted in the paper

Lemma 1 (Lemma 1 in the paper). *Let C, D be $\mathcal{FL}_0$ concepts and $\mathcal{T}$ a flat $\mathcal{FL}_0$ TBox. Then $C \sqsubseteq_\mathcal{T} D$ if and only if $D \subseteq C^*$.*

Proof. For the *only if* direction, let us assume that $C \sqsubseteq_\mathcal{T} D$. Let C^* be the saturation of C. Then since $C^* \equiv_\mathcal{T} C$, $C^* \sqsubseteq_\mathcal{T} D$. If $D \not\subseteq C^*$, then there is a rewrite step $C^* \sqsubseteq_\mathcal{T} D'$ with $D' \not\subseteq C^*$. But this is impossible since C^* is a saturation.

For the *if* direction, just notice that if $D \subseteq C^*$, then by inclusion step, $C^* \sqsubseteq_\mathcal{T} D$ and since $C \equiv_\mathcal{T} C^*$, we have $C \sqsubseteq_\mathcal{T} D$. □

Lemma 2 (Lemma 3 in the paper). *Let Γ be a unification problem which contains a goal subumption which is not flat. Let $\mathcal{T}$ be a flat $\mathcal{FL}_0$ TBox.*

There is a right application of a rule from Figure 1, such that and Γ' a unification problem obtained from Γ by this application satisfies the following claim: γ is a ground solution of Γ w.r.t. $\mathcal{T}$ iff there is a substitution γ' that is a solution of Γ' w.r.t. $\mathcal{T}$, where γ' is an extension of γ to some new variables.

Proof. In the *only if* direction we assume that γ is a unifier before a flattening step. We define extension of γ for some decomposition variables that are possibly introduced in this step as follows:

$\gamma'(X^r) := \{P \mid \forall r.P \in \gamma(X)\}$.

This yields immediately that all increasing subsumptions created in the course of flattening are satisfied by γ'. Also the applications of the decreasing rule 1 are correct. (If $\gamma'(X) \sqsubseteq_\mathcal{T} \forall r.P$, then $\gamma'(X^r) \sqsubseteq_\mathcal{T} P$.)

We assume also that in the process of flattening (Figure 1) if case 3 is to be applied, the set of constants A_D guessed in 3(b) is as follows:

$A_D = \{C \mid C$ is a constant of $\mathcal{T}$ or Γ and $C \in \gamma(D)\}$.

Now we consider the flattening steps.

1. Let the flattening step be taken as defined in point 1 of Figure 1. Then $s = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D \in \Gamma$, where $D = \forall r.D'$. The flattening step modifies the subsumption in Γ to $(C_1)^{-r} \sqcap \cdots \sqcap (C_n)^{-r} \sqsubseteq^? D'$ in Γ'.
   We have to show that if γ unifies the first subsumption, γ' unifies the second. If γ unifies $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? \forall r.D'$ it means that $\gamma(\forall r.D') \subseteq \gamma(C_1, \ldots, C_n)^*$. This means that also $\gamma'(D) \subseteq \gamma'((C_1)^{-r}, \ldots, (C_n)^{-r})^*$ and thus $(C_1)^{-r} \sqcap \cdots \sqcap (C_n)^{-r} \sqsubseteq^? D'$ is unified by γ'.
2. Let the flattening step be taken as defined in point 2 of Figure 1.
   Then $s = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D \in \Gamma$, where $C_i = \forall r.C_i'$ and D is a constant. Since γ unifies the subsumption, $\gamma(D) = D \subseteq \gamma(C_1, \ldots, C_n)^*$. By the properties of $\mathcal{FL}_0$ subsumption w.r.t. a flat TBox, we can remove $\gamma(C_i)$ from $\gamma(C_1, \ldots, C_n)^*$ and still $D \subseteq \gamma(\{C_1, \ldots, C_n\} \setminus \{C_i\})^*$. Hence γ' unifies the same subsumption with C_i deleted.
3. Let the flattening step be taken as defined in point 3 of Figure 1.
   Then $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D \in \Gamma$, where $C_i = \forall r.C_i'$ and D is a variable.
   We have to show that if γ unifies the subsumption w.r.t. $\mathcal{T}$, then γ' unifies the subsumptions constructed according to the flattening procedure for this case.

– For all particles of the form $\forall r.P \in \gamma(D)$, we can see that s^{-r} is satisfied by γ', because the substitution for D^r on the right side of s^r is such that $\gamma'(D^r) = \{P \mid \forall r.P \in \gamma(D)\}$.
If there is no particle of the form $\forall r.P$, for a particular role r, then s^r is satisfied, because then $\gamma'(D^r) = \top$.
– For all the constants $C \in \gamma(D)$, we have guessed $C \in A_D$, and thus Since $C \in \gamma'(D)$, γ' unifies $D \sqsubseteq^? C$ and also $C_1^{cons} \sqcap \cdots \sqcap C_n^{cons} \sqsubseteq^? C$.

For the *if* direction, we trace the flattening step backwards. If γ' is a solution for Γ', then γ' is also a unifier of Γ. We assume that all the increasing subsumptions are satisfied by γ' and also the decreasing rule 1, read as implication is true under γ'.

We look at what kind of flattening step was made.

1. In the first case we get s^r from s according to the first point of Figure 1. Since γ' unifies s^r and the increasing subsumptions are satisfied by γ' then γ' unifies also s.
2. In the second case, we get s^{cons} from s, where s^{cons} differs from s only in this that some non-flat particles are deleted from its left side. Since these particles cannot play any role in the subsumption of a constant, they are redundant. Hence if γ' solves s^{cons}, it also solves s.
3. In the third case, we have a set of subsumptions that replaced s in Γ'. The subcases of this case illustrate how different particles in $\gamma'(D)$ are solved. We consider them separately.
   – If a particle of the form $\forall r.P$ is in $\gamma'(D)$, then we know that $P \in \gamma'(D^r)$ and since γ' solves s^{-r}, then s is also solved by the same substitution if D on the right side of s is replaced by $\forall r.P$.
   – If a constant C of $\mathcal{T} \cup \Gamma'$ is in $\gamma'(D)$ (case (b)), γ' satisfies $D \sqsubseteq^? C$ and $C_1^{cons} \sqcap \cdots \sqcap C_n^{cons} \sqsubseteq^? C$, then γ' unifies also $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? C$.

   These are all possible particles in $\gamma'(D)$, hence γ' unifies s.

Now since all variables added in the flattening are not in the original goal and γ' is ground, we can restrict γ' to γ which is equal to γ' restricted to the goal variables. Then $\gamma = \gamma'|_{\mathbf{Var}}$ and γ is a unifier of the original unification problem before flattening.

□

Theorem 1 (Theorem 1 in the paper). *A unification problem Γ has a solution modulo a flat TBox $\mathcal{T}$ iff there is a complete c-shortcut for Γ.*

Proof. This theorem follows directly from the definition of c-shortcuts (Definition 1).

First *only if* direction. Let γ be a solution for Γ, then it satisfies the start clauses. If there are some constants assigned by γ to variables, we can extract a c-shortcut $(\mathcal{Z}, \bar{c})$ such that $\mathcal{Z}$ contains all variables such that γ assigns constants to them and $\bar{c}$ contains sets of constants assigned to these variables. γ satisfies all subsumptions and the decreasing rule, hence this shortcut is complete as required by the theorem.

If there are no constants assigned by γ to variables, this means that all start clauses are ground. Hence they are satisfied by $\mathcal{T}$. Then the c-shortcut we are looking for is defined as $(\emptyset, \emptyset)$. Notice that the empty substitution[8] satisfies all flat and increasing subsumptions and also the decreasing rule. Hence the shortcut $(\emptyset, \emptyset)$ is complete.

For the opposite direction, assume there is a c-shortcut satisfying the assumptions in theorem. Obviously, this shortcut provides a substitution γ that satisfies the conditions of Definition 1, hence Γ has a solution. □

Theorem 2 (Theorem 2 in the paper). *$\mathcal{FL}_0$-unification problem w.r.t. a flat TBox is* ExpTime*-complete*

Proof. Lemma 5, together with Lemmas 3 and 4 below show that the problem is in ExpTime. Unification in $\mathcal{FL}_0$ with the empty TBox is already ExpTime-hard. Hence $\mathcal{FL}_0$-unification modulo a flat TBox is ExpTime-complete. □

Lemma 3 (Soundness of Alg. Main). *If Algorithm 1 terminates with $True$, then there is a solution γ for Γ modulo $\mathcal{T}$.*

Proof. The algorithm terminates with $True$ only if there is a complete c-shortcut in $\mathfrak{S}_i$ for $i \geq 0$. The lemma follows by Theorem 1.

□

Lemma 4 (Completeness of Alg.Main). *If there is a solution γ for Γ modulo $\mathcal{T}$, then Algorithm 1 terminates with $True$.*

Proof. Assume there is a solution γ of Γ modulo $\mathcal{T}$. Then there is a c-shortcut defined as follows: $(\mathcal{X}, \bar{c})$, where $\mathcal{X} = \{P \mid [P \mapsto C] \in \gamma\}$ and $\bar{c}[i] = \{C \mid P_i \mapsto C \in \gamma\}$ for a $P_i \in \mathcal{X}$.

$(\mathcal{X}, \bar{c})$ is a complete c-shortcut, because γ satisfies the start subsumptions in Γ. By Lemma 7 (completeness of nextShortcuts), this shortcut will be computed for some height n (equal or smaller than the height of γ). Algorithm Main will detect that it is a complete shortcut in line 5. In order to detect this, it will substitute the start subsumptions with the assignment defined by the shortcut $(\mathcal{X}, \bar{c})$ and check if they are true modulo $\mathcal{T}$. Hence the algorithm will return $True$. □

Lemma 5 (Termination of Alg. Main). *Algorithm 1 terminates within time exponential in the size of Γ and $\mathcal{T}$.*

Proof. Since there are at most exponentially many c-shortcuts (compare the proof for Lemma 8), the for-loop starting at line 3 may run only at most exponentially many times until the sets of subsequent shortcuts provided are the same. And each round of the loop terminates in time exponential (Lemma 8). Hence in the worst case, the algorithm will return false after computing all possible c-shortcuts, therefore it will terminate in at most exponential time. □

[8] The substitution that assigns $\top$ to all variables.

Lemma 6 (Soundness of Alg. nextShortcuts). *Let S be in $\mathfrak{S}_{out}$ output by Algorithm 2. Then there is a substitution γ that satisfies conditions of Definition 1, and thus S is a c-shortcut.*

Proof. The proof is by induction of the height of the shortcuts in the input $\mathfrak{S}_{in}$ given to the algorithm.

1. The **base case** is when $\mathfrak{S}_{in}$ is empty. If $S = (\emptyset, \emptyset)$, we know that it corresponds to the substitution assigning $\top$ for all variables, which satisfies the condition of Definition 1.
   Let then $S = (\mathcal{X}, \bar{c})$. The algorithm defines γ_{temp} in line 3. We show that satisfies all conditions of Definition 1.
   Since Algorithm did not fail for S, **S1** of is satisfied by this definition of γ.
   S2 also is satisfied, because γ satisfies all flat subsumptions, and the decomposition variables are not assigned any constants, hence increasing subsumptions are voidly satisfied. The decreasing rule 1 is also voidly satisfied. Some start subsumptions may not be satisfied.
   The height of γ is 0, hence the height of such c-shortcut is also 0.
2. Induction step. Let the heights of c-shortcuts in the input $\mathfrak{S}_{in}$ be maximally n. We assume for the induction argument that all of them are real c-shortcuts, hence for each of them there is a substitution that satisfies the conditions of Definition 1.
   Now we justify that $S = (\mathcal{X}, \bar{c})$ is also a c-shortcut. If S is already in the input $\mathfrak{S}_{in}$, then this is true by induction. Hence we can assume that $S \notin \mathfrak{S}_{in}$, but it is in the output.
   We show how to construct a substitution γ for S, which satisfies the conditions of Definition 1.
   We start with γ_{temp} defined in the algorithm and consider the unsatisfied increasing subsumptions.
   These subsumptions are not satisfied only because $P^f \in \mathcal{X}$ are assigned some constants. For such f, $\mathcal{X}^{-f} \neq \emptyset$. Hence the algorithm has to check if an appropriate c-shortcut is present in $\mathfrak{S}_{in}$.
   Since the algorithm did not fail for this S, we know that there was a c-shortcut $(\mathcal{Z}, \bar{c})$ labeled *usable* in the input $\mathfrak{S}_{in}$, such that the f-ouput of $(\mathcal{Z}, \bar{c})$ is included in S, satisfying the conditions in line 8.
   Let γ_f be a substitution provided by $(\mathcal{Z}, \bar{c})$ (the existence of which we can assume by induction assumption).
   Now we use γ_f to account for bigger particles created in the variables in $\mathcal{X}^{-f}$.
   We construct a substitution $\gamma^f = \{[X \mapsto \forall v f.c] \mid [X \mapsto \forall v.c] \in \gamma_f\}$.
   Finally we define $\gamma := \gamma_{temp} \cup \bigcup_{f \in \mathbf{R}} \gamma^f$
   We observe here that γ satisfies all the conditions of Definition 1.
   - Condition **S1** is satisfied because constants are only assigned by γ_{temp}.
   - Condition **S2** is satisfied as follows. For the constants:
     - all flat subsumptions are satisfied by γ_{temp} (which was checked by algorithm),
     - increasing subsumptions are satisfied by adding the assignments in γ^f, for each f, as needed,

- decreasing rule is satisfied because the f-output for a used shortcut is in $(\mathcal{X}, \bar{c})$,

For the higher particles of the form $\forall v f.c$, all flat subsumptions are satisfied by γ^f for every f, because γ_f satisfies all the flat subsumptions, and then by Lemma 2, γ^f satisfies them too. The increasing subsumptions and the decreasing rule are also satisfied by γ^f, because the adding of a function symbol in the increasing subsumptions or deleting one in the decomposition variables are always done at the top of a particle involved. Hence all subsumptions in Γ are satisfied by γ except possibly some start subsumptions.

□

Lemma 7 (Completeness of Alg. nextShortcuts 2). *Let γ be a substitution that satisfies the conditions of Definition 1. Then there is a c-shortcut S of some height n that is returned by Algorithm 2, when t $\mathfrak{S}_{in}$ contains all the c-shortcuts of heights smaller than n.*

Proof. The proof is by induction on the height n of γ in the lemma.

1. The base case is when $n = 0$. Hence γ assigns only constants. We define $(\mathcal{X}, \bar{c})$, the c-shortcut as: $\mathcal{X} = \{P \mid [P \mapsto C] \in \gamma\}$ and the vector $\bar{c}$ is defined as an array of sets of constants: $\bar{c}[i] = \{C \mid [P_i \mapsto C] \in \gamma\}$ for $P_i \in \mathcal{X}$.
   It is obvious that Algorithm 2 will not fail on $(\mathcal{X}, \bar{c})$, because flat subsumptions are satisfied and the decomposition variables are not in $\mathcal{X}$. Hence it will return the pair in the set of computed shortcuts.
2. Now we assume that for each substitution satisfying conditions in Definition 1, with the height smaller than n, where $n \geq 0$, there is a corresponding shortcut in the input $\mathfrak{S}_{in}$. Let γ be such a substitution of height $n + 1$ that satisfies the conditions of Definition 1.
   We define $\mathcal{X} = \{P \mid [P \mapsto C] \in \gamma, \text{ and } C \text{ is a constant }\}$. And the vector $\bar{c}$ is defined as an array, assuming an order on variables: $\bar{c}[i] = \{C \mid P_i \mapsto C \in \gamma\}$, where $P_i \in \mathcal{X}$.
   Note that if $\mathcal{X} = \emptyset$, the c-shortcut has the form $(\emptyset, \emptyset)$. This c-shortcut is of height 0 and it is returned by the algorithm, as required. Hence let us assume that $\mathcal{X} \neq \emptyset$.
   We have to show that $(\mathcal{X}, \bar{c})$ is computed in the set $\mathfrak{S}_{out}$ output by Algorithm 2. By the properties of the $\mathcal{FL}_0$ flat TBox, we know that γ_{temp} defined by the algorithm satisfies all flat subsumptions in Γ.
   Now for each $f \in \mathbf{R}$ we define X^{-f} as:
   $\mathcal{X}^{-f} = \{P \in \mathbf{Var} \mid P^f \in \mathcal{X}\}$.
   We would like to show that $(\mathcal{X}^{-f}, \bar{c})$ is a shortcut in $\mathfrak{S}$, but actually $\mathcal{X}^{-f}$ maybe a subset of a shortcut that we need here, because some flat subsumptions need more particles to be present in the substitution provided by the shortcut to satisfy them. These additional particles are of the form $\forall f.C$ and

they are assigned to variables Q, for which Q^f is not defined, hence even if $Q \mapsto \forall f.C$ is in γ, there is no decomposition variable Q^f.[9]
Hence we look for a usable c-shortcut $(\mathcal{Z}, \bar{c})$, such that $\mathcal{X}^{-f} \subseteq \mathcal{Z}$ and the f-output of $(\mathcal{Z}, \bar{c})$ is in $(\mathcal{X}, \bar{c})$. If f-output is as required, the decreasing rule will be satisfied.
Since γ is a model for all subsumptions in Γ, by Lemma 2, we know that such $(\mathcal{Z}, \bar{c})$ exists.
We can extract from γ the substitution $\gamma^f = \{[X \mapsto \forall v f.C] \in \gamma\}$ that corresponds to such a shortcut. The particles of minimal height in γ^f are of the form $\forall f.C$, where S is a constant. Hence this substitution satisfies pure subsumptions in Γ. Thus the shortcut is *usable*.
It has smaller height than $n+1$, hence by induction it is provided in the input $\mathfrak{S}_{in}$.
This is true for each function symbol $f \in \mathbf{R}$, hence the algorithm will not fail for $(\mathcal{X}, \bar{c})$ and it will be included in the output $\mathfrak{S}_{out}$.

□

Lemma 8 (Termination of Alg. nextShortcuts). *Algorithm 2 terminates in time at most exponential in the size of Γ and $\mathcal{T}$.*

Proof. The number of all possible pairs $(\mathcal{X}, \bar{c})$ is exponential in the size of $\mathcal{T}$ and Γ.

There are polynomially many variables in **Var** and there are polynomially many constants in Γ and $\mathcal{T}$. Hence we have exponentially many choices for each variable. Hence there are exponentially many such assignments.

Hence the output $\mathfrak{S}_{out}$ cannot contain more than exponentially many shortcuts. Hence the for-loop starting in line 2 is executed exponentially many times and preforms polynomially many steps that take at most exponential time each.

□

[9] Looking at Example 4, we can notice that S_2 needs a shortcut to account for the decomposition variable Y^f, but there is not shortcut with $\{Y\}$ alone as a variable component, rather $\{Y\}$ is a subset of variables in shortcut S_1.