Technische Universität Dresden

Master Thesis

Planning problems in Petri Nets and Fluent Calculus

Author:
Ferdian Jovan

Supervisor:
Prof. Dr. rer. nat. habil.
Steffen Hölldobler
Dr. rer. nat. habil.
Bertram Fronhöfer

Knowledge Representation and Reasoning
International Center for Computational Logic

April 2014

Declaration of Authorship

I, Ferdian Jovan, declare that this thesis titled, 'Planning problems in Petri Nets and Fluent Calculus' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at Technische Universität Dresden.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at Technische Universität Dresden or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Author	:	Ferdian Jovan
Matrikel-Nr	:	3828288
Date of Submission	:	4th of April 2014

Acknowledgements

I would like to give my gratitude and appreciation to my supervisor, Prof. Steffen Hölldobler, whose advices, encouraging suggestions and great stories have helped me in all the time of research for and writing of this thesis. I also thank him for his concern and motivation which contributed greatly to improving my English and helped me during my study in EMCL program. Until the end, I am indebted to him for his patience dealing with me.

I also would like to give my special thanks to Bertram Fronhöfer as my second supervisor who has helped a lot for this last one year of my studies. His ideas and corrections have helped me finishing and polishing this thesis in a better way. I thank him for his patience and his time reading, commenting and giving inputs to my report. From him, I got a lot of experience in writing a paper.

I wish to express my appreciation to Dr. Wisnu Jatmiko and Dr. Yohanes Stefanus from Faculty of Computer Science, Universitas Indonesia. They gave me the motivation to pursue master degree and encouraged me to apply in this master program. Prof. Yohanes Stefanus also the first lecturer who introduced computational logic to me. I am really grateful for his teaching which brought me into this field.

Also, I want to give my deep gratefulness to Mrs. Sylvia, Ms. Julia, and Mrs. Diecke for all their big help from the very beginning up to the very end of my study. I apologize for all the trouble I made with Ausländerbehörde which resulted many tasks to do. Their care had helped me solve the misunderstanding between I and Ausländerbehörde.

Also, many thanks should go to the European Master in Computational Logic programme and its director, Prof. Steffen Hölldobler, for their support and help during my study. I thank the European Comission for their Erasmus Mundus scholarship to support my study.

I also thank my fellow friends Andy Rivkin, Ismail Ilkan Ceylan, Adil Abdrahmanov, and Ario Santoso who have always motivated me to finish this program. My Indonesian friends in Bolzano and Dresden who have been helping me in my life both in Bolzano and Dresden.

Finally, I wish to express very special thanks to my family, who has always been patient and heartedly supported me, listened to me and given good advice, in both my life and my study. Without their encouragement, I would never be here.

TECHNISCHE UNIVERSITÄT DRESDEN

Abstract

Faculty of Computer Science
International Center for Computational Logic

Master Thesis

Planning problems in Petri Nets and Fluent Calculus

by Ferdian Jovan

In this thesis we discuss conjunctive planning problems in the context of the fluent calculus and Petri nets. We show that both formalisms are equivalent in solving these problems. Thereafter, we extend actions to contain preconditions as well as obstacles. This requires to extend the fluent calculus as well as Petri nets. Again, we show that both extended formalisms are equivalent. Inspired from Petri nets, we add real-valued information to conjunctive planning problems. We show that the fluent calculus is capable in solving conjunctive planning problems with real-valued information.

Keyword: Petri nets, fluent calculus, equational logic programming, conjunctive planning problems

Contents

List of Figures

Chapter 1

Introduction

Many human activities are planned and structured which will later be enveloped in an event. Roughly speaking, an event is a sequence of considered actions that people take to reach a desired situation. For example, an event of rescuing an ill man from his apartment is built of an initial situation where the ill man is in his apartment. The sequence of actions starts with helping him to an ambulance, driving him to the hospital, and ends with the situation where the ill man is hospitalized.

Now imagine that this event is given to an autonomous agent. The autonomous agent must come up with a plan which does not necessarily need to be an efficient one but has the ability to solve the problem. In this case, the plan has to work in such a way that the result could lead to the ill man being in a hospital.

The story and the method to solve the story with an autonomous agent show a mainstream problem of AI planning. The big question now lies on how to build an intelligent and efficient system with well defined semantics that can generate plans for complicated domains. This usually starts with simple questions that comprise on how domains/-worlds and such plans should look like?

In most studies that have been primarily accomplished so far, a world is internally represented by states, and a plan which is defined as simply a sequence of so-called primitive actions, that transforms one state into another [1–4]. For example, an ill man and an apartment, an ill man and an ambulance car, and an ill man and a hospital are the world situations, whereas driving the ambulance car to the hospital after helping him to the ambulance is the plan.

Hölldobler et al. showed in [5] that several approaches are capable of solving planning problems like the example above. They have shown that those approaches are equal in the way how states are represented, how actions are executed, and more importantly how the approaches solve planning problems. Furthermore, they defined planning problems with what so-called conjunctive planning problems.

In another approach, Leon Rubin Barret in his PhD thesis designed an architecture for structured, concurrent, and real-time actions in Petri networks and Bayesian networks [3]. This design can also be used to solve our example. He revealed that some concepts can only be specified procedurally with the help of Petri nets. He expanded ordinary Petri nets by adding inhibitor arcs, test arcs, continuous places, and transitions possessing arithmetic expressions. All of these extensions were there to help integrating Petri nets with Bayesian networks. These two networks working together are able to capture essential properties in actions such as compositionality, concurrency, quick reactions, and resilience in the face of unexpected events.

1.1 Motivation

A central notion in Petri nets are tokens which are consumed and produced when executing an action. Likewise, in the equational logic programming approach to actions and causality presented in [6] resources are used. The approach was later called *fluent calculus* in [7]. It is clear that there are similarities between Petri nets and the fluent calculus.

As Petri nets are able to show their capability to solve planning problems, we aim to show that the fluent calculus is also able to solve planning problems. We will use the planning problem framework introduced in [5]. As the planning problems defined in [5] are simple, we will rigorously define various classes of planning problems and map these problems into the fluent calculus. In the end, we will show the soundness and completeness of the fluent calculus in solving such problems.

We will also investigate the corresponding Petri nets introduced in [3] for some classes of planning problems. We will show how planning problems are mapped into Petri nets. We use them as a comparison to the fluent calculus in solving such problems. We will formally prove that there is one-to-one correspondence between Petri nets with their extensions and the fluent calculus.

1.2 Contribution

This thesis expands the theorem shown in [5] by adding Petri nets. It is shown in Chapter 3 that Petri nets and the fluent calculus are equivalent in solving simple planning problems. An example is given to illustrate the representation of planning problems in both, the fluent calculus and Petri nets. Furthermore, from the example, the planning problems transformation from the fluent calculus to Petri nets and vice versa can be demonstrated.

We extend our planning problems in such a way that they become more expressible. These extensions are originated from Petri nets designed by Barret in [3]. The planning problems now include new features such as preconditions and obstacles. We define formally some extensions which appear in Barret's Petri nets starting form those which have a good formalization to the one without any formalization [3]. A formal definition for test arcs in these Petri nets is necessary because Barret in his report did not provide one [3]. In the end, we extend the correspondence theorem stating that the fluent calculus and Petri nets are equivalent in solving so-called advanced planning problems. We have proven that the advanced Petri nets and advanced fluent calculus are equivalent in solving planning problems with these new features.

Barret altered the interaction between places with real-valued tokens and transitions. Inspired from his design, we introduce real-valued fluents in the fluent calculus and add theory actions to advanced planning problems. We prove that our advanced fluent calculus with real values are able to represent and solve advanced planning problems with real values inside. One should know that we do not show the equivalence between Petri nets combining advanced Petri nets and Petri nets calculating mathematical expressions and the fluent calculus due to informal definition of these Petri nets in [3].

1.3 Thesis Structure

We will start with preliminaries in Chapter 2 by introducing the notation and terminology in the fluent calculus and Petri nets. Chapter 3 introduces simple planning problems. It also addresses their representation both, the fluent calculus and Petri nets. At the end of this chapter, we will show the correspondence between the fluent calculus and Petri nets in solving simple planning problems.

The second part of this thesis shows how the development of simple planning problems towards more sophisticated planning problems. It also shows how these problems can still be solved with the extension of the fluent calculus and Petri nets. In Chapter 4, this thesis shows how advanced fluent calculus and advanced Petri nets are equivalently able to represent and elegantly solve planning problems equipped with side requirements for actions such as preconditions and obstacles. An example is given to help readers comprehend the problem. Chapter 5 combines planning problems with the appearance of real values. Some modifications to the fluent calculus are explained in details. As usual, an example is given to illustrate the approach. In the end, we show that the final modification of the fluent calculus is capable of solving planning problems with real-valued information.

In the final chapter, all findings are summarized. Some future works such as adding a time feature to the fluent calculus, the possibility to show the equivalence between timed Petri nets and timed fluent calculus are pointed out. Some considerations to combine the fluent calculus with Bayesian networks are also included.

Chapter 2

Preliminaries

The chapter starts with the basic notions of multisets. We continue with the introduction of Petri nets. Notions of first order formulae with equality follow after. Fluent calculus is presented at the end of this chapter.

2.1 Multisets

The notion of multiset comes from a generalization of the notion of sets in which each element is allowed to appear more than once. They are one of data structures that can represent resources flow. We define multiset as follows:

Definition 2.1.1. Let $\dot{\emptyset}$ denote the empty multiset and let the parentheses $\dot{\{}$ and $\dot{\}}$ be used to enclose the elements of a multiset. Let $\mathcal{M}, \mathcal{M}_1, \mathcal{M}_2$ be finite multisets. The following relations apply:

- *Membership*: $X \in_k \mathcal{M}$ iff X occurs precisely k-times in $\mathcal{M}$, for $k \geq 0$.
- *Equality*: $\mathcal{M}_1 \doteq \mathcal{M}_2$ iff for all X we find $X \in_k \mathcal{M}_1$ iff $X \in_k \mathcal{M}_2$.
- *Union*: $X \in_m \mathcal{M}_1 \dot{\cup} \mathcal{M}_2$ iff there exist $k, l \geq 0$ such that $X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2$, and $m = k + l$.
- *Difference*: $X \in_m \mathcal{M}_1 \dot{\setminus} \mathcal{M}_2$ iff there exist $k, l \geq 0$ such that either $X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2, k > l$, and $m = k - l$ or $X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2, k \leq l$, and $m = 0$.
- *Intersection*: $X \in_m \mathcal{M}_1 \dot{\cap} \mathcal{M}_2$ iff there exist $k, l \geq 0$ such that $X \in_k \mathcal{M}_1, X \in_l \mathcal{M}_2$, and $m = min\{k, l\}$, where min maps $\{k, l\}$ to its minimal element.
- *Submultiset*: $\mathcal{M}_1 \dot{\subseteq} \mathcal{M}_2$ iff $(\mathcal{M}_1 \dot{\cap} \mathcal{M}_2) \doteq \mathcal{M}_1$.

The number of times an element belongs to a multiset is the *multiplicity* of that element. Two multisets are equal if and only if all different elements in both multisets have the same multiplicity. In multisets, as in sets and in contrast to sequence or list, the order of elements is irrelevant.

2.2 Petri Nets

Petri nets were firstly introduced as a mathematical description of distributed systems, and analysis of their properties [8]. Lately, they are heavily used as a tool for modeling and validation of systems in which concurrency, communication and synchronization play a major role [9, 10].

A Petri net is a directed bipartite graph, in which nodes represent *transitions* and *places*. Transitions are events which are symbolized by squares. Places are conditions (preconditions and postconditions) of transitions symbolized by circles. The *directed arcs* describe which places are pre- and/or postconditions for transitions. Arcs are symbolized by arrows. Formally,

Definition 2.2.1. A Petri net $\mathcal{N}$ is a tuple $(\mathcal{P}, \mathcal{T}, \mathcal{F})$, where:

- $\mathcal{P}$ and $\mathcal{T}$ are finite sets;
- $\mathcal{P} \cap \mathcal{T} = \emptyset$;
- $\mathcal{F} \dot{\subseteq} (\mathcal{P} \times \mathcal{T}) \dot{\cup} (\mathcal{T} \times \mathcal{P})$.

$\mathcal{P}$ is a set of *places* and $\mathcal{T}$ is a set of *transitions*. $X_{\mathcal{N}} = \mathcal{P} \cup \mathcal{T}$ consists of *elements* of the net $\mathcal{N}$. $\mathcal{F}$ is a set of arcs connecting elements of $\mathcal{N}$.

More advanced Petri nets are usually equipped with a specified weight for each arc and a maximum capacity for each place. Nevertheless, in this work we consider *ordinary Petri Nets* where the weight for each arc is set to 1 and the capacity for each place is set to infinity. Murata stated that both ordinary and non-ordinary Petri nets have the same modeling power [11]. The only difference is modeling efficiency or convenience.

The next definition helps us to specify the concept of *pre-set* and *post-set* for a particular element of the net.

Definition 2.2.2. Let $\mathcal{N} = (\mathcal{P}, \mathcal{T}, \mathcal{F})$ be a Petri net and $x \in \mathcal{T}$.

- A *pre-set* of x, denoted $\bullet x$, is a finite multiset such that

$$y \in_k \bullet x \text{ iff } y \in \mathcal{T} \wedge (y, x) \in_k \mathcal{F}$$

holds.

- A *post-set* of x, denoted $x\bullet$, is a finite multiset such that

$$y \in_k x\bullet \text{ iff } y \in \mathcal{T} \wedge (x, y) \in_k \mathcal{F}$$

holds.

A Petri net comes with resources put in each place. These resources for each place are called *tokens*. The state describing how many tokens are in each place is called *marking*. Formally, a Petri net with a marking is described as follows:

Definition 2.2.3. Let $\mathcal{N} = (\mathcal{P}, \mathcal{T}, \mathcal{F})$ be a Petri net, and $\mathcal{M} \mathrel{\dot{\subseteq}} \dot{\{} x \mid x \in \mathcal{P} \dot{\}}$ be a multiset over places. A *marked Petri net* is a tuple $(\mathcal{N}, \mathcal{M})$.

A marking is a multiset of places representing how many tokens currently present in each place. A Petri net is usually equipped with an initial marking.

Definition 2.2.4. Let $\mathcal{N} = (\mathcal{P}, \mathcal{T}, \mathcal{F})$ be a Petri net, $t \in \mathcal{T}$ be a transition, and $\mathcal{M}, \mathcal{M}'$ be markings. The following holds:

- t is *enabled* at $\mathcal{M}$ in $\mathcal{N}$, denoted by $\mathcal{M} \xrightarrow{[t]}$, iff $\bullet t \mathrel{\dot{\subseteq}} \mathcal{M}$.
- An enabled transition t in Petri net $\mathcal{N}$ with marking $\mathcal{M}$ can *fire* leading to marking $\mathcal{M}'$, denoted by $\mathcal{M} \xrightarrow{[t]} \mathcal{M}'$, iff $\mathcal{M}' \doteq (\mathcal{M} \mathbin{\dot{\setminus}} \bullet t) \mathbin{\dot{\cup}} t\bullet$.

A transition is enabled if and only if incoming arcs of the transition from each place are fewer than the number of tokens currently present in the place. Given a marking, a *firing rule* determines whether a transition can be executed and what the resulting new marking is. We extend the notion of firing so that it gives us a *firing sequence* as follows [11, 12]:

Definition 2.2.5. Let $\mathcal{N}$ be a Petri net, and $\mathcal{M}, \mathcal{M}', \mathcal{M}''$ be markings. Let w be a list of transitions, and $t \in \mathcal{T}$. A firing sequence is defined as follows:

- $\mathcal{M} \xrightarrow{[]} \mathcal{M}$.
- If $\mathcal{M} \xrightarrow{[t]} \mathcal{M}'$, and $\mathcal{M}' \xrightarrow{w} \mathcal{M}''$ then $\mathcal{M} \xrightarrow{[t|w]} \mathcal{M}''$.

A *firing sequence from* $\mathcal{M}$ *to* $\mathcal{M}'$ of $\mathcal{N}$ is a firing sequence w which starts from a marked Petri net $(\mathcal{N}, \mathcal{M})$ to a marked Petri net $(\mathcal{N}, \mathcal{M}')$.

At every step, starting with an initial marking, there are many enabled transitions. One of them is chosen non-deterministically like in a token game. By iterating for each possible enabled transition in each produced marking, we obtain all the reachable markings from the initial marking we start with. It gives a rise to the following problem:

Definition 2.2.6. The *reachability problem* consists of a Petri net $\mathcal{N} = (\mathcal{P}, \mathcal{T}, \mathcal{F})$, an initial marking $\mathcal{M}$, and a marking $\mathcal{M}'$. It is the question of whether $\mathcal{M}'$ is reachable from $\mathcal{M}$ in $\mathcal{N}$, i.e., there exists a firing sequence P such that $\mathcal{M} \xrightarrow{P} \mathcal{M}'$. $\mathcal{R}(\mathcal{M})$ denotes the smallest set of markings of $\mathcal{N}$ such that the markings are reachable from $\mathcal{M}$.

Reachability is a fundamental basis for studying the dynamic properties of any system. A solution for reachability problems in a Petri net determines an execution of a net and provides information of how the net behaves.

2.3 First Order Theories

Definition 2.3.1. An *alphabet* of first order logic is the union of six disjoint sets of symbols:

1. the countably infinite set of *variables* $\mathcal{V}$,
2. the finite or countably infinite set of *function symbols* $\mathcal{F}$,
3. the finite or countably infinite set of *predicate symbols* $\mathcal{R}$,
4. the set of connectives $\{\neg,\ \vee,\ \wedge,\ \leftarrow,\ \leftrightarrow\}$,
5. the set of quantifiers $\{\exists, \forall\}$, and
6. the set of punctuation symbols {"(", ",", ")"}.

The sets (4)-(6) are the same for each alphabet, whereas (1)-(3) may vary from alphabet to alphabet. Relation and function symbols are assigned a natural number that we will call *arity* of the respective symbol. We indicate the arity directly after the respective symbol separated by a slash.

We denote by $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ the language defined over $\mathcal{R}, \mathcal{F}, \mathcal{V}$. We turn to the definition of the constituents of our first order language over an alphabet.

Definition 2.3.2. The set of *terms* of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ is the smallest set satisfying the following conditions:

1. Every variable is a term,

2. If $f/n \in \mathcal{F}$ and $t_1, ..., t_n$ are terms then $f(t_1, ..., t_n)$ is a term.

A *constant* is a function symbol with arity 0. A term is *closed* or *ground(instantiated)*, if it does not contain any variables.

Definition 2.3.3. The set of *atomic formulae* (or, briefly, *atoms)* of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ is the set of strings of the form $p(t_1, ..., t_n)$, where $p/n \in \mathcal{R}$ and $t_1, ..., t_n$ are terms.

Definition 2.3.4. The set of (*well-formed*) *formulae* of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ is the smallest set defined by the following rules:

1. Every atom of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ is a formula of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$.
2. If F and G are formulae of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$, then so are $(\neg F)$, $(F \wedge G)$, $(F \vee G)$, $(F \leftarrow G)$, and $(F \leftrightarrow G)$.
3. If F is a formula of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$, then so are $(\exists F)$, and $(\forall F)$.

A *ground* formula is a formula with ground terms and no quantifiers occurred. A *literal* is an atom A or its negation $\neg A$. We assume that in a first order language there is at least one constant. Moreover, all formulae are assumed to be universally closed.

Definition 2.3.5. An *equation* is an expression of the form $s \approx t$, where s and t are terms. An *equational system* $\mathcal{E}$ is a set of universally closed equations.

We consider a first-order language over an alphabet which contains the binary relation symbol $\approx$ called *equality.* Equality has the properties of reflexivity, symmetry, transitivity and substitutivity. This can be expressed within a first-order logic by the equational system

$$\begin{aligned} \mathcal{E}_{\approx} \;=\{\;\; & X \approx X \\ & X \approx Y \rightarrow Y \approx X \\ & X \approx Y \wedge Y \approx Z \rightarrow X \approx Z \\ & \textstyle\bigwedge_{i=1}^{n} X_i \approx Y_i \rightarrow f(X_1, ..., X_n) \approx f(Y_1, ..., Y_n) \\ & \textstyle\bigwedge_{i=1}^{n} X_i \approx Y_i \wedge r(X_1, ..., X_n) \rightarrow r(Y_1, ..., Y_n) \quad \} \end{aligned}$$

which consists of the so-called *axioms of equality.*

Definition 2.3.6. A *substitution* is a mapping from a set of variables to a set of terms which coincides with the identity mapping except for finitely many points. It can be represented by a finite set of pairs

$$\sigma = \{X_1 \mapsto t_1, \ldots, X_n \mapsto t_n\}$$

The identity mapping, i.e. the *empty substitution*, is denoted by ε.

Definition 2.3.7. Let t_1, t_2 be two terms. t_1 is called a *variant* of t_2 if there is a substitution σ such that $t_1 = t_2\sigma$.

A substitution σ is a ground substitution if $t\sigma$ is a ground term for all t in terms. A substitution can be used to transform a term such that one term is a variant of another.

First Order Logic

The semantics of first order logic that we consider is based on Herbrand theory.

Definition 2.3.8. An *interpretation* I is a mapping from first order language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ to the set of truth values $\{\mathit{true}, \mathit{false}\}$. A *Herbrand interpretation* is an interpretation where the domain of the interpretation is a set of ground terms and for every ground term t in $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$ it holds that $t^I = t$.

We define an interpretation I as a subset of the Herbrand base, i.e., a set of all ground atoms. All atoms which are in the set are considered true, whereas others which are not are considered false.

Definition 2.3.9. Let I be an interpretation of the language $\mathcal{L}(\mathcal{R}, \mathcal{F}, \mathcal{V})$, $\mathcal{F}$ be a set of formulae, and F a formula. I is a *model* of a formula F, denoted by $I \models F$, if $I(F) = \mathit{true}$. $\models F$ if and only if $I \models F$ for all interpretations I.
F is called a logical consequence of $\mathcal{F}$, denoted by $\mathcal{F} \models F$, if every model I of $\mathcal{F}$ is also a model of F.

We define an equality between two terms under an equational system $\mathcal{E}$ using the notion of logical consequence as follows.

Definition 2.3.10. Let s, t be terms, and $\forall$ denote the universal closure.

$$s \approx_{\mathcal{E}} t \text{ iff } \mathcal{E} \cup \mathcal{E}_{\approx} \models \forall s \approx t.$$

$\approx_{\mathcal{E}}$ is the *least congruence relation on terms generated by* $\mathcal{E}$.

Unification under Equality

Unification problems are concerned with a way of finding a substitution, called a unifier, such that two terms become equal. We consider unification theory where equality axioms are included in the unification process.

Definition 2.3.11. An *$\mathcal{E}$-unification problem* consists of an equational system $\mathcal{E}$ and an equation $s \approx t$ and is the question of whether

$$\mathcal{E} \cup \mathcal{E}_{\approx} \models \exists s \approx t,$$

where the existential quantifier denotes the existential closure of $s \approx t$. An $\mathcal{E}$-unifier for this problem is a substitution θ such that

$$s\theta \approx_{\mathcal{E}} t\theta$$

and is a solution for the $\mathcal{E}$-unification problem.

Definition 2.3.12. Let $\mathcal{V}$ be a set of variables and θ and σ be two substitutions. σ is called an *$\mathcal{E}$-instance of θ on $\mathcal{V}$*, denoted by $\sigma \leq_{\mathcal{E}} \theta[\mathcal{V}]$, iff there exists a substitution τ such that $X\sigma \approx_{\mathcal{E}} X\theta\tau$ for all $X \in \mathcal{V}$.

We denote by $U_{\mathcal{E}}(s,t)$ the set of all $\mathcal{E}$-unifiers for the terms s and t.

Definition 2.3.13. Let $\mathcal{V}$ be the set of variables occurring in terms s and t. *A complete set of $\mathcal{E}$-unifiers for s and t*, denoted by $cU_{\mathcal{E}}(s,t))$, is a set $\mathcal{S}$ of substitutions where

1. $\mathcal{S} \subseteq U_{\mathcal{E}}(s,t)$; and
2. for all $\sigma \in U_{\mathcal{E}}(s,t)$ there exists $\theta \in \mathcal{S}$ such that $\sigma \leq_{\mathcal{E}} \theta[\mathcal{V}]$

hold.

We define *a unification complete with respect to equational system $\mathcal{E}$ ($\mathcal{E}^*$)* based on [13] as follows:

Definition 2.3.14. Let $\mathcal{E}$ be an equational system. A consistent set of formulas $\mathcal{E}^*$ is called unification complete wrt $\mathcal{E}$ if it consists of the axioms in $\mathcal{E}$, $\mathcal{E}_{\approx}$, and a number of equational formulae such that for any two terms s and t the following holds:

1. If s and t are not $\mathcal{E}$-unifiable then $\mathcal{E}^* \models \neg\exists\ s \approx t$.
2. If s and t are $\mathcal{E}$-unifiable then for each complete set of unifiers $cU_{\mathcal{E}}(s,t)$,

$$\mathcal{E}^* \models \forall\ (s \approx t \rightarrow \bigvee_{\theta \in cU_{\mathcal{E}}(s,t)} \exists \mathcal{V}_\theta\ \theta)$$

   where $\mathcal{V}_\theta$ denotes the variables which occur in θ but they are not part of the universal closure of s and t.

The completion of an equational system $\mathcal{E}$ is needed to deal with logic programs with negation in the body. In such a completion, we will be able to prove inequalities like $X \not\approx 1$.

2.4 Fluent Calculus

The fluent calculus is a first order logic where the axioms of equality and a particular equational system are included. The fluent calculus was originated from equational Horn logic used for deductive planning to solve the frame problems [4–6, 14]. It was later named as fluent calculus in action and causality [7]. We start this subsection by defining fluents.

Definition 2.4.1. Fluents are the non-variable elements of terms where the set of function symbols does not include $\circ$ and 1. *Simple fluents* are fluents consisting only of constants. The *ground* fluents are fluents not containing any variables.

In the fluent calculus, a binary function symbol $\circ$ is introduced. It is designed to represent terms which have the property of associativity, commutativity, and admitting a unit element (constant) 1. Formally,

Definition 2.4.2. The set of *fluent terms* is the smallest set meeting the following conditions

1. 1 is a fluent term,
2. each fluent is a fluent term, and
3. if s and t are fluent terms, then $s \circ t$ is a fluent term.

The *ground* fluent terms are fluent terms not containing any variables. The *simple* fluent terms are fluent terms constructed by simple fluents. As the sequence of fluents occurring in a fluent term is not important, we consider the following equational system:

$$\begin{aligned}\mathcal{E}_{AC1} = \{ \quad & X \circ (Y \circ Z) \approx (X \circ Y) \circ Z \\ & X \circ Y \approx Y \circ X \\ & X \circ 1 \approx X \qquad\qquad \}.\end{aligned}$$

Fluent terms can be represented in multisets by restricting the elements of multisets to fluents. A mapping $\cdot^I$ transforming a fluent term into a multiset of fluents is defined as follows.

Definition 2.4.3. Let t be a fluent term:

$$t^I = \left\{ \begin{array}{ll} \dot{\emptyset} & \text{if } t = 1 \\ \dot{\{}\ t\ \dot{\}} & \text{if } t \text{ is a fluent, and} \\ u^I \,\dot{\cup}\, v^I & \text{if } t = u \circ v. \end{array} \right\}$$

Although two fluent terms are said to be equivalent whenever they have the same fluents with the same amount, those two fluent terms are still two different fluent terms. Thus, to define an inverse mapping $\cdot^{-I}$ from multisets of fluents to fluent terms, we assume there is a linear order in fluents occurring in multiset of fluents.

Definition 2.4.4. Let $\mathcal{M}$ be a multiset of fluents, let s be the least element on some linear ordering in $\mathcal{M}$:

$$\mathcal{M}^{-I} = \left\{ \begin{array}{ll} 1 & \text{if } \mathcal{M} \doteq \dot{\emptyset} \\ s \circ \mathcal{N}^{-I} & \text{if } \mathcal{M} \doteq \dot{\{}\ s\ \dot{\}} \mathbin{\dot{\cup}} \mathcal{N}. \end{array} \right\}$$

If the order in fluents occuring in multisets is not insisted then picking fluents in different order when constructing the fluent terms will produce different fluent terms. Hence, $\cdot^{-I}$ can not be called a mapping. With definitions of $\cdot^{I}$ and $^{-I}$, there is a one-to-one correspondence between fluent terms and multisets of fluents as shown with the equations

$$t \approx_{AC1} (t^I)^{-I}$$

and

$$\mathcal{M} \doteq (\mathcal{M}^{-I})^I.$$

A unification problem in fluent terms can be divided into two separate problems: *fluent matching problem* and *fluent unification problem.* These notions are defined as follows:

Definition 2.4.5. The *fluent matching problem* consists of a fluent term s, a ground fluent term t, and a variable X not occurring in s. It is the question of whether there exists a substitution σ such that $(s \circ X)\sigma \approx_{AC1} t$.

Definition 2.4.6. The *fluent unification problem* consists of two fluent terms s and t and a variable X not occurring in s and t. It is the question of whether there exists a substitution σ such that $(s \circ X)\sigma \approx_{AC1} t\sigma$.

Fluent unification and matching problem are decidable, finitary, and there always exists a minimal complete set of matchers and unifiers [4]. As a consequence, there always is a unification complete theory with respect to fluent calculus [2]. Techniques to construct the minimal complete set of matchers and unifiers can be found in [15].

With the help of mapping ^{I}, fluent matching and fluent unification problems can be transformed into

Definition 2.4.7. The *submultiset matching problem* consists of a multiset $\mathcal{M}$ and a ground multiset $\mathcal{N}$. It is the question of whether there exists a substitution σ such that $\mathcal{M}\sigma \mathrel{\dot{\subseteq}} \mathcal{N}$.

Definition 2.4.8. The *submultiset unification problem* consists of two multiset $\mathcal{M}$ and $\mathcal{N}$. It is the question of whether there exists a substitution σ such that $\mathcal{M}\sigma \dot{\subseteq} \mathcal{N}\sigma$.

Logic Programs and SLDENF-resolution

The fluent calculus roots in the logic programming formalism of [6, 16]. The fluent calculus represents problems in the form of logic programs. We define a logic program as a set of clauses in particular form.

Definition 2.4.9. Let A be an atom, and each B_i, $1 \leq i \leq n$, a literal. A (program) clause is a formula of the form

$$A \leftarrow B_1 \ \wedge ... \wedge \ B_n \quad (n \geq 0)$$

where A is called *head* and $B_1 \ \wedge ... \wedge \ B_n$ is called *body* of the clause. A *definite clause* is a clause where every B_i, $1 \leq i \leq n$, is an atom. A fact is a clause without the body. A *(logic) program* $\mathcal{P}$ is a finite set of clauses.

Definition 2.4.10. A *goal*, or *query*, is a clause without the head i.e, of the form

$$\leftarrow B_1 \ \wedge ... \wedge \ B_n \quad (n \geq 1)$$

where each of $B_i, i \geq 1$ is a literal and is called *subgoal.* A definite goal is a goal clause where each of B_i is an atom.

To handle negative literals in a logic program, Clark introduced a program completion, in which, loosely speaking "if" is interpreted as "if and only if" [17].

Definition 2.4.11. Let $\mathcal{P}$ be a logic program, and p/n be a predicate symbol occurring in the head of a clause in $\mathcal{P}$, and q/n' be a predicate symbol not occurring in the head of any clause in $\mathcal{P}$. The *completed definition of* p is a formula of the form

$$(\forall X_1) \dots (\forall X_n)(p(X_1, \dots, X_n) \leftrightarrow G_1 \ \vee \dots \vee \ G_m)$$

where m is the number of clauses with p as the head, and $B_i, 1 \leq i \leq m$ is of the form

$$(\exists Y_1) \dots (\exists Y_k)(X_1 \approx t_1 \ \wedge \dots \wedge \ X_n \approx t_n \ \wedge \ B_1 \ \wedge \dots \wedge \ B_k).$$

$Y_1, \dots, Y_k$ are all variables occurring in a clause C where the head of the clause is p, $t_1, \dots, t_n$ are terms occurring in the head of clause C, and $B_1, \dots, B_k$ are literals occurring in the body of the clause C.

The completed definition of q is a formula of the form

$$(\forall X_1)\dots(\forall X_{n'})\neg q(X_1,\dots,X_{n'})$$

The *completion of* $\mathcal{P}$ is the generalized conjunction of the completed definitions of all relation symbols occurring in $\mathcal{P}$. We denote by $\mathcal{P}^*$ a completion of $\mathcal{P}$.

To know whether literals in the goal clause are a logical consequence of a logic program, we use SLD-resolution. This technique is the processing mechanism employed in Prolog system. The logical consequence can be traced by proving the unsatisfiability between a goal and a logic program with an SLD-resolution [16, 18].

We consider *SLDENF-resolution*, the variant of SLD-resolution with negation as failure added to handle negation in the body of clauses [17]. Negation as failure in SLDENF-resolution adopts the definition used in [13], whereas the standard unification procedure in SLDENF-resolution is replaced by an appropriate E-unification procedure [16, 19]. In contrast to SLDE-resolution, SLDENF-resolutions trace the logical consequence of a completion of both the logic program and the equational system [13, 20].

Definition 2.4.12. Let $\mathcal{P}$ be a logic program. Let C be a new variant $A \leftarrow B_1 \wedge ... \wedge B_m$ of a clause in $\mathcal{P}$, G a goal clause $\leftarrow G_1 \wedge ... \wedge G_n$. If A and an atom $G_i, 1 \leq i \leq n$, are $\mathcal{E}$-unifiable with $\mathcal{E}$-unifier θ, then

$$\leftarrow (G_1 \wedge ... \wedge G_{i-1} \wedge B_1 \wedge ... \wedge B_m \wedge G_{i+1} \wedge ... \wedge G_n)\theta$$

is called *SLDE-resolvent* of C and G.

Without loss of generality, we assume that literals are selected from left to right. There is a dependency definition between the definition of a finitely failed SLDENF-tree and the definition of SLDENF-resolution. We start with the definition of a finitely failed SLDENF-tree.

Definition 2.4.13. A finitely failed SLDENF-tree of rank r ($r \geq 0$) for program $\mathcal{P}$ and a goal clause G is a finite tree such that the following holds:

1. The root is labeled with G and each node is labeled with a non-empty goal;
2. For each leaf node $\leftarrow G_1 \ \wedge \dots \wedge \ G_n$ where G_i is selected:
    - If G_i is a positive literal then it does not $\mathcal{E}$-unify with the head of a new variant of any program clause; and
    - If G_i is a negative ground literal $\neg A$ then there exists an SLDENF-resolution of rank less than r for $\mathcal{P}$ and $\leftarrow A$.

3. If an inner node $\leftarrow G_1 \ \wedge \ldots \wedge \ G_n$ where G_i is selected then
   - If G_i is a positive literal then the child nodes are SLDE-resolvents between a new variant of a clause $C \in \mathcal{P}$ and the inner node.
   - If G_i is a negative ground literal $\neg A$ then there exists a finitely failed SLDENF-tree of rank less than r for $\mathcal{P}$ and $\leftarrow A$, and the only child of the inner node is labeled with $\leftarrow G_1 \ \wedge \ldots \wedge \ G_{i-1} \ \wedge \ G_{i+1} \ \wedge \ldots \wedge \ G_n$.

A finitely failed SLDENF-tree of rank 0 is where all successive child nodes are SLDE-resolvent of a clause in a logic program and the parent nodes.

Having defined a finitely failed SLDENF-tree, we turn to derivations where negative literals are selected.

Definition 2.4.14. An SLDENF-resolution of rank r $(r \geq 1)$ for a logic program $\mathcal{P}$ and a goal clause G consists of a sequence $G_0, \ldots, G_n$ of goals such that $G = G_0$ and G_n is an empty clause and for each $i = 1, \ldots, n$ the following holds:

- If the literal B of G_{i-1} is positive then there is a new variant of a program clause C such that G_i is an SLDE-resolvent of C and G_{i-1}.
- If the literal B of G_{i-1} is a negative ground literal $\neg A$ then there exists a finitely failed SLDENF-tree of rank less than r for $\mathcal{P}$ and $\leftarrow A$, and G_i is as G_{i-1} except that it does not contain B.

A resolution is said to *flounder* if the derivation yields a goal which contains only non-ground negative literals [21].

The fluent calculus is a logic program where axioms of equality and (AC1) axioms are included. Therefore, $\mathcal{E}$-unification problems which have to be solved within the fluent calculus are either fluent matching or fluent unification problems. Furthermore, θ is not called $\mathcal{E}$-unifier but rather AC1-unifier. The soundness of SLDENF-resolution for fluent calculus has been proven in [13]. However, the completeness of SLDENF-resolution for fluent calculus requires that no derivation flounders or is infinite and the completion of the logic program is satisfiable [2].

Definition 2.4.15. An *answer substitution* for a goal clause G is a substitution σ where the domain of the substitution σ is a subset of the set of variables occurring in G.

A goal clause may come with variables in it. When we apply SLDE(NF)-resolution to a logic program together with the goal clause, we will get a substitution to replace the variables in a goal clause. This substitution for the variables occurring in a goal clause is called an answer substitution.

2.5 Append Function

A function *append* is a common function used in many programming languages including Prolog. Generally, the function takes two finite list as inputs and produces a list which concatenates the first list with the second one. Formally,

Definition 2.5.1. $append : List \times List \mapsto List$ is a function which receives two finite lists and returns the first list with the second list attached at the end of the first list.

In Prolog, the append function is implemented as clauses with the head of clauses is the predicate append. The implementation goes as follows:

$$append([\,], L, L), \tag{appendbase}$$

$$append([H \mid T], L, [H \mid R]) \leftarrow append(T, L, R). \tag{appendinductive}$$

The first clause tells that an empty list appended with list L gives L as a result. The second clause tells that if list R is a result of appending list T with L then list $[H \mid R]$ is a result of appending $[H \mid T]$ with L.

Clauses (appendbase) and (appendinductive) are originally coming from Prolog system. Those clauses do not cause infinite SLDENF-derivations as long as either the length of the list of the first argument or the length of the list of the third argument is finite. The derivations may go forever if either the first or the third argument is not instantiated. This is shown in the following propositions.

Proposition 2.1. *Let l, nl be arbitrary fluent terms. No SLDENF-resolution proof of $\leftarrow append([a_n, \ldots, a_1], l, nl)$ is infinite.*

Proof. The proposition is proven by induction on the length of the first argument.
I.B To show: No SLDENF-resolution proof of $\leftarrow append([\,], l, nl)$ is infinite.
In the case where $n = 0$, i.e., the list is empty, $append([\,], l, nl)$ can only be AC1-unified with (appendbase) and only if $l \approx nl$. Hence, we obtain the empty clause or the derivation fails immediately.

I.H No SLDENF-resolution proof of $\leftarrow append([a_n, \ldots, a_1], l, nln)$ is infinite.
I.C To show: No SLDENF-resolution proof of $\leftarrow append([a_{n+1}, \ldots, a_1], l, nl)$ is infinite.
I.S In the case $n + 1 > 0$, $append([a_{n+1}, \ldots, a_1], l, nl)$ can only be AC1-unified with the head of clause (appendinductive) where the successful derivation yields

$$\leftarrow append([a_n, \ldots, a_1], l, nln) \tag{2.1}$$

where $nl = [a_{n+1} \mid nln]$. We can apply our induction hypothesis to (2.1) and learn that no SLDENF-resolution proof of $\leftarrow \mathit{append}([a_n, \ldots, a_1], l, nln)$ is infinite. By combining the result from the induction hypothesis and the SLDENF-derivations from $\leftarrow \mathit{append}([a_n, \ldots, a_1], l, nl)$ to (2.1) we can conclude that there is no SLDENF-resolution proof of $\leftarrow \mathit{append}([a_n, \ldots, a_1], l, nl)$ is infinite. $\square$

Proposition 2.2. *Let l, nl be arbitrary fluent terms. No SLDENF-resolution proof of $\leftarrow \mathit{append}(nl, l, [a_n, \ldots, a_1])$ is infinite.*

Proof. The proposition is proven by induction on the length of the third argument.
I.B To show: No SLDENF-resolution proof of $\leftarrow \mathit{append}(nl, l, [\,])$ is infinite.
In the case where $n = 0$, i.e., the list is empty, $\mathit{append}(nl, l, [\,])$ can only be AC1-unified with (appendbase) and only if $nl = r = [\,]$. Hence, we obtain the empty clause or the derivation fails immediately.

I.H No SLDENF-resolution proof of $\leftarrow \mathit{append}(nln, l, [a_n, \ldots, a_1])$ is infinite.
I.C To show: No SLDENF-resolution proof of $\leftarrow \mathit{append}(nl, l, [a_{n+1}, \ldots, a_1])$ is infinite.
I.S In the case $n + 1 > 0$, $\mathit{append}(nl, l, [a_{n+1}, \ldots, a_1])$ can only be AC1-unified with the head of clause (appendinductive) where the successful derivation yields

$$\leftarrow \mathit{append}(nln, l, [a_n, \ldots, a_1]) \tag{2.2}$$

where $nl = [a_{n+1} \mid nln]$. We can apply our induction hypothesis to (2.2) and learn that there is no SLDENF-resolution proof of $\leftarrow \mathit{append}(nln, l, [a_n, \ldots, a_1])$ is infinite. By combining the result from the induction hypothesis and the SLDENF-derivations from $\leftarrow \mathit{append}(nl, l, [a_{n+1}, \ldots, a_1])$ to (2.2) we can conclude that there is no SLDENF-resolution proof of $\leftarrow \mathit{append}(nl, l, [a_{n+1}, \ldots, a_1])$ is infinite. $\square$

One should note that if the third argument is finite then so are the first and the second argument of *append*. Following the definition of *append*, the first and the second argument of *append* are limited to finite lists.

Chapter 3

Planning Problems

Conjunctive planning problems were firstly introduced in [5] as a part of showing how to solve the technical frame problem. This problems describe an initial situation, a goal situation, a set of actions, and a question to find a series of appropriate actions changing the initial situation to the goal one. The initial and goal situations and the conditions as well as the effects of actions are modeled with multisets.

Definition 3.0.2. Let $\mathcal{S}$ be a set of simple fluents, and $\mathcal{S}_m$ be a finite multiset constructed from $\mathcal{S}$. *A conjunctive planning problem (cpp)* is a tuple $\langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$, where

- $\mathcal{I} \dot{\subseteq} \mathcal{S}_m$;
- $\mathcal{G} \dot{\subseteq} \mathcal{S}_m$; and
- $\mathcal{A}$ is a finite set of *actions* of the form $A : \mathcal{C} \Rightarrow \mathcal{E}$ where $\mathcal{C} \dot{\subseteq} \mathcal{S}_m, \mathcal{E} \dot{\subseteq} \mathcal{S}_m$.

$\mathcal{I}$ is called the *initial state* and $\mathcal{G}$ is called the *goal state*. An action $A : \mathcal{C} \Rightarrow \mathcal{E}$ has the *name* A, the *conditions* $\mathcal{C}$, and the *effects* $\mathcal{E}$. Multisets containing simple fluents occurring in cpp are called *states*. A state represents what simple fluents hold at a time.

Definition 3.0.3. Let $\mathcal{S}$ be a multiset of simple fluents. An action

$$A : \mathcal{C} \Rightarrow \mathcal{E}$$

is *applicable* to $\mathcal{S}$ iff

$$\mathcal{C} \dot{\subseteq} \mathcal{S}.$$

The *application of the action* leads to the multiset

$$(\mathcal{S} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}.$$

The consecutive application of actions to a multiset leading to another multiset is called a *plan.*

Definition 3.0.4. A plan is a sequence $[a_1, ..., a_j]$ of actions transforming state $\mathcal{S}$ into $\mathcal{S}'$ iff $\mathcal{S}'$ is the result of successively applying the actions in $[a_1, ..., a_j]$ to $\mathcal{S}$. A plan p is a *solution* for the planning problem iff the plan can transform the initial state $\mathcal{I}$ into a goal state $\mathcal{G}$.

A cpp is mainly about the question of whether there exists a sequence of actions such that its execution transforms the initial state into the goal state [5]. Hölldobler et al. showed that conjunctive planning problem can be solved with the linear connection method, equational logic and the linear logic approach [5]. Their main theorem states that the following four statements are equivalent:

1. Plan p is a solution for a conjunctive planning problem $\mathcal{Q}$.
2. p is generated by a linear connection proof of the representation of cpp in the linear connection method.
3. p is generated by an SLDE-resolution proof of the representation of cpp in equational logic.
4. p is generated by a linear logic proof of the representation of cpp in the linear logic.

These approaches also elegantly solve frame axiom problem as stated in [5, 22]. The linear connection method as well as the linear logic approach are not within the scope of this work. Interested readers are referred to [5] for formal definitions.

We will extend what Hölldobler et al., have done by adding sentence "p is a firing sequence of the representation of cpp in Petri nets" into their theorem. First, we present how conjunctive planning problems are represented and solved within both fluent calculus and Petri nets. Second, we provide an example of how we transform a conjunctive planning problem into both fluent calculus and Petri nets. We also give a solution achieved by these two approaches. In the end, we will show that there is a one-to-one correspondence between fluent calculus and Petri nets in finding the solution for conjunctive planning problems.

3.1 CPP in Petri Nets

Petri nets are a formalism for modelling discrete event systems. Such formalisms can be used to model planning problems. Places in Petri nets represent all types of resources needed to apply actions. An action is encoded by a transition. A name of a transition represents the name of the action, incoming arcs of a transition encode the conditions, and outgoing arcs of a transition encode the effects.

Definition 3.1.1. Let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ be a cpp. A tuple $\mathcal{N}_\mathcal{Q} = (\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{I}, \mathcal{G})$ is a representation of $\mathcal{Q}$ in a (marked) Petri net where

- $\mathcal{P}$ is a set of all simple fluents occurring in $\mathcal{Q}$;
- $\mathcal{T}$ is a set of all action names in $\mathcal{Q}$;
- $(p, t) \in_k \mathcal{F}$ iff $\exists (t : \mathcal{C} \Rightarrow \mathcal{E}) \in \mathcal{A}$ s.t $p \in_k \mathcal{C}$; and
- $(t, p) \in_k \mathcal{F}$ iff $\exists (t : \mathcal{C} \Rightarrow \mathcal{E}) \in \mathcal{A}$ s.t $p \in_k \mathcal{E}$.

One should observe that $(\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{I})$ and $(\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{G})$ are marked Petri nets. Markings represent states of conjunctive planning problems. Tokens in each place show how many resources are available at a state. Because the representation of the initial state $\mathcal{I}$ in cpp is in the form of multisets and so is the goal state $\mathcal{G}$, therefore, we can treat $\mathcal{I}$ and $\mathcal{G}$ as markings. Furthermore, we know that for every action $t : \mathcal{C} \Rightarrow \mathcal{E}$ in $\mathcal{A}$, we have a transition $t \in \mathcal{T}$ in the Petri net $\mathcal{N}_\mathcal{Q}$ with $\bullet t \doteq \mathcal{C}$ and $t\bullet \doteq \mathcal{E}$. Conversely, whenever a transition t is enabled at marking $\mathcal{M}$ in $\mathcal{N}_\mathcal{Q}$, there exists an action t in $\mathcal{Q}$ where t is applicable in $\mathcal{M}$.

The question of whether there exists a plan P solving a conjunctive planning problem $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ is represented by the question whether $\mathcal{G} \in \mathcal{R}(\mathcal{I})$ of $\mathcal{N}_\mathcal{Q}$, i.e., whether there exists a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of a Petri net $\mathcal{N}_\mathcal{Q}$.

With Petri nets, we can also find a sequence of firing for a modified planning problem, where the initial state $\mathcal{I}$ is the result of executing a transition to the previous state where the goal state $\mathcal{G}$ is unchanged. This follows immediately from the inductive definition of a firing sequence of a Petri net.

3.2 CPP in Fluent Calculus

A simple fluent calculus is a first order calculus, where conjunctive planning problems can be represented and solved [6].

Actions in cpp are represented using a ternary action symbol *action*/3 and are of the form

$$action(C, A, E)$$

where C encodes the conditions, A encodes the name, E encodes the effects of the action. These clauses are in the set $\mathcal{K}_A$.

A state in cpp is represented by a fluent term. An applicable action is represented with the help of ternary relation symbol *applicable*/3.

$$applicable(C \circ S, A, E \circ S) \;\leftarrow\; action(C, A, E) \tag{3.1}$$

The clause (3.1) is read that an action A is applicable in state $C \circ S$ leading to state $E \circ S$ if there is an action named A with conditions C and effects E. With the help of ternary relation symbols *causes*/3, we can express that a current state is transformed into future one by applying a sequence of actions.

$$causes(S, [\,], S) \tag{3.2}$$

$$causes(I, [A \mid P], G) \leftarrow applicable(I, A, S) \wedge causes(S, P, G) \tag{3.3}$$

Clause (3.2) states that there is nothing to do ([ ]), if the future state S is the current state S. The clause (3.3) is read declaratively as

> The execution of the plan $[A \mid P]$ transforms state I into state G if action A is applicable in state I and its application yields state S and there is a plan P which transforms state S into state G.

We pack these three clauses under the set $\mathcal{K}_C$.

The question of whether there exists a plan P solving a conjunctive planning problem $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ is represented by the question of whether

$$(\exists P) causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{3.4}$$

is the logical consequence of $\mathcal{K}_A \cup \mathcal{K}_C \cup \mathcal{E}_{AC1} \cup \mathcal{E}_{\approx}$. As clauses in $\mathcal{K}_C \cup \mathcal{K}_A$ are definite clauses, we may apply SLDE-resolution in order to solve the planning problem.

Coming back to answer substitution of (3.4) at this state, an answer substitution contains a binding for P. In fact, P is bound to a list of action names, which is called the plan generated by an SLDE-resolution proof. Generally, the plan which is generated by an SLDE-resolution proof is of the form $[a_1, ..., [a_n, []]...]$. We abbreviate such a plan by $[a_1, ..., a_n]$.

When we do an SLDE-resolution proof, we may encounter either fluent unification problems or fluent matching problems. However, the substitution processes are mostly between simple ground fluent terms and variables because of the restriction in conjunctive planning problems to simple fluents. Hence, from AC1-unifier construction shown in [15], the substitution forms being the AC1-unifiers are unique.

One should consider that the length of SLDE-resolutions for cpp is always greater than the generated plan. However, we assume that we only count the number of SLDE-refutation steps involving subgoals *causes*. Therefore, the length of an SLDE-resolution is the number of SLDE-refutation steps involving subgoals *causes*. Hence, the length of an SLDE-resolution proofs is as long as the generated plan.

3.3 An Instance of Conjunctive Planning Problems

A simple example about an ill man in an apartment who needed to be helped to hospital is considered to illustrate conjunctive planning problems. We will show how this problem can be represented and solved within the fluent calculus and Petri nets. Lets consider the following story:

Suppose there was a man who was severely ill living in an apartment. He could not go by himself to see a doctor. An ambulance car was asked to bring him to hospital. He was carried by the ambulance men to the ambulance car. The ambulance car is driven to the hospital.

This problem is considered as a conjunctive planning problem where the ill man (*ill*), the apartment (*apt*), the ambulance car (*amb*), and the hospital (*hos*) are the fluents. The possible actions are described by *carrying the patient to the ambulance car* (c), and *driving to the hospital* (d). The only solution for this ill man is to *carry him to the ambulance car* and *drive the car to the hospital.* Within the conjunctive planning problem we obtain

$$\mathcal{Q} = \langle \quad \dot{\{} ill, apt \dot{\}}, \dot{\{} ill, hos \dot{\}}, \{c : \dot{\{} ill, apt \dot{\}} \Rightarrow \dot{\{} ill, amb \dot{\}}, d : \dot{\{} ill, amb \dot{\}} \Rightarrow \dot{\{} ill, hos \dot{\}}\} \quad \rangle$$

If we apply c and d in this order to the initial state $\dot{\{} ill, apt \dot{\}}$, we end up with $\dot{\{} ill, hos \dot{\}}$. Therefore, c and then d are the right actions to help him to go to hospital.

In the fluent calculus, the actions of help-to-ambulance and drive-to-hospital are represented by the set of clauses

$$\mathcal{K}_A \quad = \{ \quad action(ill \circ apt, c, ill \circ amb), action(ill \circ amb, d, ill \circ hos) \quad \}$$

If we ask, whether there exists a plan P such that its execution transforms state $\dot{\{}ill, apt\dot{\}}$ into state $\dot{\{}ill, hos\dot{\}}$, i.e.

$$\exists P : causes(\dot{\{}ill, apt\dot{\}}^{-I}, P, \dot{\{}ill, hos\dot{\}}^{-I})$$

then we obtain the refutation shown in Figure 3.1 yielding the answer substitution

$$\{P \mapsto [c, [d, []]]\}$$

Hence, the ill man has to be carried to the ambulance car (c) first, the ambulance car is driven to hospital (d) afterwards, and they have reached the goal ($[\,]$).

In Petri nets, the ill man problem can be represented as

$$\begin{aligned} \mathcal{N}_{\mathcal{Q}} = (\ & \{ill, apt, amb, hos\}, \{c, d\}, \dot{\{}(ill, c), (c, ill), (ill, d), (d, ill), (apt, c), (c, amb), \\ & (amb, d), (d, hos)\dot{\}}, \dot{\{}ill, apt\dot{\}}, \dot{\{}ill, hos\dot{\}} \qquad) \end{aligned}$$

If we ask, whether there exists a firing sequence P such that its execution transforms marking $\dot{\{}ill, apt\dot{\}}$ into $\dot{\{}ill, hos\dot{\}}$, i.e.

$$\dot{\{}ill, apt\dot{\}} \xrightarrow{P} \dot{\{}ill, hos\dot{\}}$$

then we obtain the firing sequence shown in Figure 3.2 where the ambulance men have to carry the ill man to an ambulance car (c) first, then they drive the car to the hospital (d) afterwards before we can reach the marking $\dot{\{}ill, hos\dot{\}}$, i.e., $P = [c, d]$.

3.4 A Correspondence Between Fluent Calculus and Petri Nets

We have shown that a Petri net is able to represent a conjunctive planning problem. The question is whether a Petri net solves a conjunctive planning problem by the firing sequence from the initial marking to the goal marking. That question can be solved by showing a correspondence between the fluent calculus and Petri nets. The correspondence is shown by Theorem 3.1.

Throughout this section let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ denote a conjunctive planning problem and let $\mathcal{FC}_{\mathcal{Q}}$, and $\mathcal{N}_{\mathcal{Q}}$ denote its presentation in the fluent calculus and Petri nets, respectively.

Theorem 3.1. *The following statements are equivalent for a plan p of $\mathcal{Q}$* [1]

[1]For interested reader the proofs that a solution plan for conjunctive planning problem can be generated by a linear connection proof, a linear logic proof and an SLDE-resolution proof can be found in [5].

$$
\begin{array}{rcl}
\leftarrow causes(\dot{\{}ill, apt\dot{\}}^{-I}, P, \dot{\{}ill, hos\dot{\}}^{-I}) & \text{———} & causes(I, [A \mid P], G) \\
\sigma_1 = \{I \mapsto ill \circ apt, P \mapsto [A \mid P_1], & | & \\
G \mapsto ill \circ hos\} & | & \\
\leftarrow applicable(ill \circ apt, A, S) \wedge & \text{———} & applicable(C \circ S_1, A_1, E \circ S_1) \\
causes(S, P_1, ill \circ hos) & | & \\
\sigma_2 = \{C \mapsto ill \circ apt, S_1 \mapsto 1, & | & \\
A_1 \mapsto A, S \mapsto E\} & | & \\
\leftarrow action(ill \circ apt, A, E) & \text{———} & action(ill \circ apt, c, ill \circ amb) \\
\wedge causes(E, P_1, ill \circ hos) & | & \\
\sigma_3 = \{A \mapsto c, E \mapsto ill \circ amb\} & | & \\
\leftarrow causes(ill \circ amb, P_1, ill \circ hos) & \text{———} & causes(I_1, [A_2 \mid P_2], G_1) \\
\sigma_4 = \{I_1 \mapsto ill \circ amb, P_1 \mapsto [A_2 \mid P_2], & | & \\
G_1 \mapsto ill \circ hos\} & | & \\
\leftarrow applicable(ill \circ amb, A_2, S_2) \wedge & \text{———} & applicable(C_1 \circ S_3, A_3, E_1 \circ S_3) \\
causes(S_2, P_2, ill \circ hos) & | & \\
\sigma_5 = \{C_1 \mapsto ill \circ amb, S_3 \mapsto 1, & | & \\
A_3 \mapsto A_2, S_2 \mapsto E_1\} & | & \\
\leftarrow action(ill \circ amb, A_2, E_1) & \text{———} & action(ill \circ amb, d, ill \circ hos) \\
\wedge causes(E_1, P_2, ill \circ hos) & | & \\
\sigma_6 = \{A_2 \mapsto d, E_1 \mapsto ill \circ hos\} & | & \\
\leftarrow causes(ill \circ hos, P_2, ill \circ hos) & \text{———} & causes(S_4, [\,], S_4) \\
\sigma_7 = \{S_4 \mapsto ill \circ hos, P_2 \mapsto [\,] & | & \\
 & \square &
\end{array}
$$

FIGURE 3.1: The SLDE-resolution of $\neg causes(\dot{\{}ill, apt\dot{\}}^{-I}, P, \dot{\{}ill, hos\dot{\}}^{-I})$ yields the answer substitution $\{P \mapsto [c, [d, []]]\}$.

1. p is a solution for $\mathcal{Q}$.

2. p is generated by an SLDE-resolution proof of $\mathcal{FC}_{\mathcal{Q}}$.

3. p is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_{\mathcal{Q}}$.

Proof. We find that statements 1 and 2 are equivalent, i.e. p is a solution for $\mathcal{Q}$ iff p is generated by an SLDE-resolution proof of $\mathcal{FC}_{\mathcal{Q}}$ [5]. Hence, to proof the theorem it suffices to show that 2 implies 3 and 3 implies 2. These implications are proven in Lemma 3.2 and 3.3.

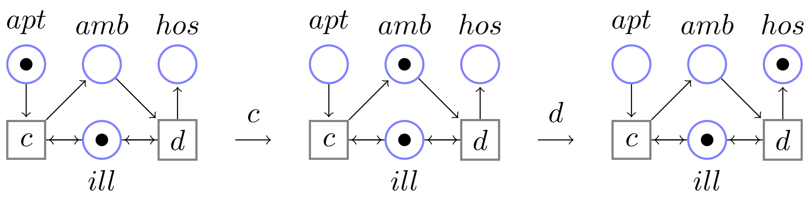

FIGURE 3.2: A firing sequence for the ill man problem with initial marking $\dot{\{} apt, ill \dot{\}}$.

Lemma 3.2. *If p is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$, then p can also be generated by an SLDE-resolution proof of $\mathcal{FC}_Q$.*

Proof. The lemma is proven by induction on the number j of transitions executed in the firing sequence of $\mathcal{N}_Q$.

I.B To show: If $[\,]$ is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$, then $[\,]$ is generated by an SLDE-resolution proof of $\mathcal{FC}_Q$.

If $[\,]$ is a firing sequence $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$ then there is no need to fire any transition to reach the goal marking because the goal marking coincides with the initial marking, i.e.,

$$\mathcal{I} \xrightarrow{[\,]} \mathcal{I}.$$

Let σ be the substitution

$$\{S \mapsto \mathcal{I}^{-I}, P \mapsto [\,]\}.$$

σ is the AC1-unifier for the atom occurring in

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{I}^{-I}) \tag{3.5}$$

and a new variant

$$causes(S, [\,], S)$$

of (3.2). Thus, there exists an SLDE-refutation of (3.5) using the variant of (3.2) and yielding the plan $[\,]$.

I.H If $[a_j, \ldots, a_1]$ is a firing sequence from $\mathcal{I}_j$ to $\mathcal{G}$ of $\mathcal{N}_Q$, then $[a_j, \ldots, a_1]$ can also be generated by an SLDE-resolution proof of $\mathcal{FC}_Q$.

I.C To show: If $[a_{j+1}, a_j, \ldots, a_1]$ is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$, then $[a_{j+1}, a_j, \ldots, a_1]$ can also be generated by an SLDE-resolution proof of $\mathcal{FC}_Q$.

I.S Suppose $[a_{j+1}, a_j, \ldots, a_1]$ is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$, i.e.,

$$\mathcal{I} \xrightarrow{[a_{j+1}, a_j, \ldots, a_1]} \mathcal{G}.$$

a_{j+1} is the first enabled transition and assume that $\bullet a_{j+1} \doteq \mathcal{C}$ and $a_{j+1}\bullet \doteq \mathcal{E}$ such that

$$\mathcal{I} \xrightarrow{a_{j+1}} (\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E}.$$

From the construction of $\mathcal{N}_\mathcal{Q}$, we may conclude that there is an action in $\mathcal{Q}$ in the form $a_{j+1} : \mathcal{C} \Rightarrow \mathcal{E}$. Therefore, we can find the corresponding fact

$$\mathit{action}(\mathcal{C}^{-I}, a_{j+1}, \mathcal{E}^{-I}). \tag{3.6}$$

in $\mathcal{F}C_\mathcal{Q}$ s.t. the conditions and effects are $\mathcal{C}$ and $\mathcal{E}$, respectively.

Let $\mathcal{I}' \doteq \mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}$, and σ be a substitution

$$\{I \mapsto \mathcal{I}^{-I}, G \mapsto \mathcal{G}^{-I}, P \mapsto [A \mid P']\}.$$

It is easy to see that σ is the AC1-unifier between the atom occurring in

$$\leftarrow \mathit{causes}(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{3.7}$$

and the head of a new variant

$$\mathit{causes}(I, [A \mid P'], G) \leftarrow \mathit{applicable}(I, A, S) \wedge \mathit{causes}(S, P', G)$$

of (3.3). Therefore, there is an SLDE-derivation from (3.7) to

$$\leftarrow \mathit{applicable}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, A, S) \wedge \mathit{causes}(S, P', \mathcal{G}^{-I}) \tag{3.8}$$

using the variant of (3.3) and AC1-unifier σ. There is an SLDE-derivation of (3.8) yielding

$$\leftarrow \mathit{action}(\mathcal{C}^{-I}, A, E) \wedge \mathit{causes}(E \circ \mathcal{I}'^{-I}, P', \mathcal{G}^{-I}) \tag{3.9}$$

using the head of a new variant

$$\mathit{applicable}(C \circ S', A', E \circ S') \leftarrow \mathit{action}(C, A', E)$$

of (3.1) and the AC1-unifier

$$\sigma' = \{C \mapsto \mathcal{C}^{-I}, S' \mapsto \mathcal{I}'^{-I}, A' \mapsto A, S \mapsto E \circ \mathcal{I}'^{-I}\}.$$

Once more, we can find an AC1-unifier

$$\sigma'' = \{E \mapsto \mathcal{E}^{-I}, A \mapsto a_{j+1}\}$$

between the atom of the first literal in (3.9) and (3.6). Thus, we find an SLDE-derivation from (3.9) to

$$\leftarrow causes(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P, \mathcal{G}^{-I}) \tag{3.10}$$

using (3.6) and the AC1-unifier σ''.

By executing a_{j+1} at the initial marking $\mathcal{I}$ in $\mathcal{N}_Q$, we have a firing sequence $[a_j, \ldots, a_1]$ of $\mathcal{N}_Q$ such that

$$\mathcal{I}' \dot{\cup} \mathcal{E} \xrightarrow{[a_j,\ldots,a_1]} \mathcal{G}. \tag{3.11}$$

Because this proof contains $[a_j, \ldots, a_1]$ and the goal marking in (I.H) is the same as in (3.11), we may conclude that $\mathcal{I}_j \doteq \mathcal{I}' \dot{\cup} \mathcal{E}$. We can apply the induction hypothesis and learn that there is an SLDE-refutation of (3.10) yielding the computed answer substitution $\{W \mapsto [a_j, ..., a_1]\}$.

Combining this refutation with derivation from (3.7) to (3.10) yields an SLDE-derivation of (3.7) with the computed answer substitution $\{P \mapsto [a_{j+1}, ..., a_1]\}$, i.e., the sequence of actions $[a_{j+1}, ..., a_1]$ is generated. □

The proof of this lemma directly defines a procedure to transform a firing sequence of Petri nets into SLDE-refutations. In particular, the firing sequence in Figure 3.2 is transformed into the SLDE-refutation in Figure 3.1.

Lemma 3.3. *If plan p is generated by an SLDE-refutation of $\mathcal{FC}_Q$, then p is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$.*

Proof. The lemma is an immediate consequence of the following claim.

If there is an SLDE-refutation of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ generating plan p, then there is a firing sequence p from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$.

The claim is proven by induction on the length j of the SLDE-refutation.
I.B To show: If there is an SLDE-refutation of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ with length 1 generating plan p, then there is a firing sequence p of $\mathcal{N}_Q$ from $\mathcal{I}$ to $\mathcal{G}$.

If the length of the SLDE-refutation is 1, then the atom occurring in

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{3.12}$$

and a variant

$$causes(S, [\,], S)$$

of (3.2) must be AC1-unifiable and SLDE-refutation of (3.12) generates the empty plan $[\,]$. Furthermore, the first and the third argument of the variant of (3.2) refer to the same fluent term. Therefore, we get that $\mathcal{I}^{-I} \approx_{AC1} \mathcal{G}^{-I}$. We may conclude that AC1-unifier for (3.12) and the variant of (3.2) is of the form

$$\{P \mapsto [\,], S \mapsto \mathcal{I}^{-I}\}.$$

Since $\mathcal{I}^{-I} \approx_{AC1} \mathcal{G}^{-I}$, we obtain $\mathcal{I} \doteq \mathcal{G}$. We may conclude that in Petri net $\mathcal{N}_Q$, the initial marking and the goal marking are the same. Therefore, there is no need to fire any transition, i.e,

$$\mathcal{I} \xrightarrow{[\,]} \mathcal{I}.$$

Hence, there is a firing sequence $[\,]$ from $\mathcal{I}$ to $\mathcal{I}$ of $\mathcal{N}_Q$.

I.H If there is an SLDE-refutation of $\leftarrow \mathit{causes}(\mathcal{I}_j^{-I}, P_j, \mathcal{G}^{-I})$ with length j generating plan p_j, then there is a firing sequence p_j from $\mathcal{I}_j$ to $\mathcal{G}$ of $\mathcal{N}_Q$.
I.C To show: If there is an SLDE-refutation of $\leftarrow \mathit{causes}(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ with length $j+1$ generating plan p, then there is a firing sequence p from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_Q$.
I.S Suppose there is an SLDE-refutation of

$$\leftarrow \mathit{causes}(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{3.13}$$

with length $j+1$ and generating plan $[a_1, ..., a_j]$.

The first atom occurring in (3.13) can only be AC1-unified with the head of a new variant

$$\mathit{causes}(I, [A \mid P'], G) \leftarrow \mathit{applicable}(I, A, S) \ \wedge \ \mathit{causes}(S, P', G)$$

of (3.3) with an AC1-unifier of the form

$$\sigma = \{I \mapsto \mathcal{I}^{-I}, G \mapsto \mathcal{G}^{-I}, P \mapsto [A \mid P']\}.$$

It gives an SLDE-derivation from (3.13) to

$$\leftarrow \mathit{applicable}(\mathcal{I}^{-I}, A, S) \wedge \mathit{causes}(S, P', \mathcal{G}^{-I}) \tag{3.14}$$

The first atom occurring in (3.14) can only be AC1-unified with the head of a new variant

$$\mathit{applicable}(C \circ S', A', E \circ S') \leftarrow \mathit{action}(C, A', E)$$

of (3.1) with an AC1-unifier of the form

$$\{C \mapsto c_1 \circ ... \circ c_l, S' \mapsto i_1 \circ \ldots \circ i_n, A' \mapsto A, S \mapsto E \circ i_1 \circ ... \circ i_n\}.$$

where $\mathcal{I}^{-I} \approx_{AC1} c_1 \circ \ldots \circ c_l \circ i_1 \circ \ldots \circ i_n$. The SLDE-derivation of (3.14) yields

$$\leftarrow action(c_1 \circ ... \circ c_l, A, E) \wedge causes(E \circ i_1 \circ ... \circ i_n, P', \mathcal{G}^{-I}) \tag{3.15}$$

Since we assume that there is an SLDE-refutation of (3.13), there must be a fact

$$action(c_1 \circ ... \circ c_l, a_1, e_1 \circ ... \circ e_k) \tag{3.16}$$

in $\mathcal{F}C_\mathcal{Q}$ which can be AC1-unified with the first atom occurring in (3.15). Let

$$\sigma' = \{A \mapsto a_1, E \mapsto e_1 \circ ... \circ e_k\}$$

be a substitution. σ' is the AC1-unifier between the atom of the first literal of clause (3.15) and (3.16) [15]. Hence, there is an SLDE-refutation from (3.15) to

$$\leftarrow causes(e_1 \circ ... \circ e_k \circ i_1 \circ ... \circ i_n, P', \mathcal{G}^{-I}) \tag{3.17}$$

From (3.16), we may conclude that there must be an action $a_1 : (c_1 \circ ... \circ c_l)^I \Rightarrow (e_1 \circ ... \circ e_k)^I$ in $\mathcal{Q}$. Furthermore, we know that a_1 is applicable in the initial state $\mathcal{I}$ from the fact that $(c_1 \circ ... \circ c_l)^I \dot{\subseteq} \mathcal{I}$. Hence, we can find the corresponding enabled transition a_1 in $\mathcal{N}_\mathcal{Q}$ with $\bullet a_1 \doteq \dot{\{}c_1, ..., c_l\dot{\}}$ and $a_1 \bullet \doteq \dot{\{}e_1, ..., e_k\dot{\}}$ such that

$$\mathcal{I} \xrightarrow{a_1} (i_1 \circ ... \circ i_n \circ e_1 \circ ... \circ e_k)^I \tag{3.18}$$

We can apply (I.H) to (3.17) since the SLDE-refutation only has length j. We learn a firing sequence $[a_2, \ldots, a_j]$ from $(i_1 \circ ... \circ i_n \circ e_1 \circ ... \circ e_k)^I$ to $\mathcal{G}$ of $\mathcal{N}_\mathcal{Q}$, i.e.,

$$(i_1 \circ ... \circ i_n \circ e_1 \circ ... \circ e_k)^I \xrightarrow{[a_2,...,a_j]} \mathcal{G} \tag{3.19}$$

Combining (3.18) and (3.19), we obtain a firing sequence $[a_1, \ldots, a_m]$ such that

$$\mathcal{I} \xrightarrow{[a_1,...,a_j]} \mathcal{G}$$

Because there is a firing sequence p from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{N}_\mathcal{Q}$, the claim is true. □

The proof defines the counter part procedure of Lemma 3.2 which transforms an SLDE-refutation into a firing sequence of Petri nets.

Chapter 4

Advanced Planning Problems

In the previous chapter we have shown that Petri nets and fluent calculus are capable to find a sequence of actions solving conjunctive planning problems. These kind of problems come with simple actions where actions are only equipped with conditions and effects. They do not distinguish between conditions that are consumed and conditions that must be present without being consumed. Furthermore, there is no such thing like obstacles which may hinder an action from being taken.

Lets modify the ill man problem presented in the previous chapter as follows:

Suppose now the ill man was really fat such that he could not fit through his apartment's door.

In the example above, being fat is an obstacle for the ill man to get rescued. It inhibits the action *carry-to-the-ambulance* from being taken. It is clear that a normal conjunctive planning problem is not able to model the problem above because there is a restriction to the action *carry-to-the-ambulance* named that the ill man can not be fat.

To model the modified example about the ill man in conjunctive planning problem, we introduce what so-called *obstacles* to an action. Informally, obstacles are resources at which if the number of them is enough then they can hinder an action from being taken. An action can be taken whenever the number of the resources is not enough to cause any trouble to the action. Unlike conditions and effects of an action, obstacles are neither consumed nor produced. Moreover, we also introduced so-called *preconditions.* Preconditions are conditions that must be present whenever an action is being taken. For instance, the man being ill in our example is a precondition for all considered actions.

This modification is focused on the set of actions. There is no alternation regarding the initial state and the goal state.

Definition 4.0.1. *An advanced conjunctive planning problem (acpp)* $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ is a conjunctive planning problem where $\mathcal{A}$ is a finite set of extended actions of the form

$$A : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E},$$

where $\mathcal{C}, \mathcal{R}, \mathcal{O}, \mathcal{E}$ are simple fluents and $\mathcal{C} \mathbin{\dot{\cap}} \mathcal{E} \doteq \dot{\emptyset}$. $A, \mathcal{C}, \mathcal{R}, \mathcal{O}, \mathcal{E}$ are the name, the conditions, the preconditions, the obstacles, and the effects of the action respectively.

We extend the definition of applicable action for acpp as follows:

Definition 4.0.2. Let $\mathcal{S}$ be a multiset of simple fluents, and $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ be an acpp. An extended action

$$A : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$$

is *applicable* to $\mathcal{S}$ iff

$$\mathcal{C} \mathrel{\dot{\subseteq}} \mathcal{S}, \mathcal{R} \mathrel{\dot{\subseteq}} \mathcal{S} \text{ and } \forall e \in_k \mathcal{O} \; (e \in_j \mathcal{S} \rightarrow j < k)$$

The *application of an extended action* leads to the multiset

$$(\mathcal{S} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E}.$$

We broaden the definition of applicable in acpp such that obstacles and preconditions are included as the prerequisite for an action. An action is applicable in a state whenever the state contains all conditions and preconditions which are needed to produce effects and it does not have enough elements of obstacles for each different obstacles which may hinder the action. One important point for an action is that once there are enough elements of one type of obstacles at particular states, they are enough to make an action not to be applicable. Definitions of a plan and a solution still follow definitions described in Chapter 3.

Some may argue why our definition of applicable actions separates $\mathcal{R} \mathrel{\dot{\subseteq}} \mathcal{S}$ from $\mathcal{C} \mathrel{\dot{\subseteq}} \mathcal{S}$ because it is possible to combine them together as $\mathcal{R} \mathbin{\dot{\cup}} \mathcal{C} \mathrel{\dot{\subseteq}} \mathcal{S}$. We separate $\mathcal{R} \mathrel{\dot{\subseteq}} \mathcal{S}$ from $\mathcal{C} \mathrel{\dot{\subseteq}} \mathcal{S}$ to obtain the possibility to have $\mathcal{S} \mathrel{\dot{\subseteq}} (\mathcal{C} \mathbin{\dot{\cup}} \mathcal{R})$. Our definition has more flexibility to model some real-life problems than the restriction $\mathcal{R} \mathbin{\dot{\cup}} \mathcal{C} \mathrel{\dot{\subseteq}} \mathcal{S}$ could have.

4.1 Advanced Petri Nets

An *extended Petri net* is a Petri net with *inhibitor arcs* [12]. Originally, the introduction of inhibitor arcs adds the ability to test "zero" (i.e., absence of tokens in a place) and increases the modelling power of Petri nets to the level of Turing machine [11]. David and Alla in [23] modified the original definition by adding weight to inhibitor arcs. Their definition states that a transition can only be fired at a particular marking, if the marking has fewer tokens than the weight of inhibitor arcs connected to the transition [23].

Apart from the existence of inhibitor arcs, Barret introduced new arcs called *test arcs*. These arcs were intended to aid discrete Petri nets to work with places holding real values. The appearance of test arcs in Petri nets adds the ability to check "threshold" of a place (i.e., whether a place has more tokens than the threshold). A test arc is a modified arc going from a place to a transition that does not consume any tokens when the transition fires but still disables the transition whenever that input place does not have enough tokens [3].

Combining these two special arcs in one Petri net gives us *advanced Petri nets*. Formally,

Definition 4.1.1. An *advanced Petri net* $\mathcal{N}$ is a tuple $(\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{H}, \mathcal{L})$, where:

1. $(\mathcal{P}, \mathcal{T}, \mathcal{F})$ is a marked Petri net;
2. $\mathcal{H} \dot{\subseteq} \mathcal{P} \times \mathcal{T}$; and
3. $\mathcal{L} \dot{\subseteq} \mathcal{P} \times \mathcal{T}$.

$\mathcal{H}$ are called inhibitor arcs whereas $\mathcal{L}$ are called test arcs. Inhibitor arcs and test arcs of an advanced Petri net $\mathcal{N}$ constitute submultisets of $\mathcal{P}_{\mathcal{N}} \times \mathcal{T}_{\mathcal{N}}$. We denote by $(\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{H}, \mathcal{L})$ an advanced Petri net (APN). From the definition of an advanced Petri net, we get that ordinary Petri nets are Petri nets with $\mathcal{H} \doteq \mathcal{L} \doteq \emptyset$. Graphically, an inhibitor arc is depicted by arrows with a diamond head, whereas a test arc is depicted by a dash arrow. An instance of Petri nets with inhibitor arcs and test arcs are shown in Figure 4.1.

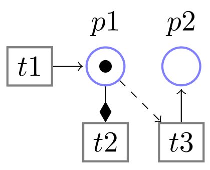

FIGURE 4.1: A Petri net with an inhibitor arc, connected to $t2$, and a test arcs, connected to $t3$.

Definition 4.1.2. Let $\mathcal{N} = (\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{H}, \mathcal{L})$ be an APN, we define the multiset of inhibitor places of transition t as follows

$$p \in_k \blacktriangledown t \text{ iff } p \in \mathcal{P} \ \wedge \ (p, t) \in_k \mathcal{H}$$

and the multiset of test places of transition t as follows

$$p \in_k \blacktriangle t \text{ iff } p \in \mathcal{P} \ \wedge \ (p,t) \in_k \mathcal{L}$$

It is well known that inhibitor arcs can properly increase the expressive power of a net. The argument is that two-counter machines can be modeled by deterministic finite nets with (at least two) unbounded places and adjacent inhibitor arcs [12]. Moreover, inhibitor arcs provide a formal way to represent coordination in parallel systems.

Definition 4.1.3. Let $\mathcal{N} = (\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{H}, \mathcal{L})$ be an advanced Petri net and $\mathcal{M}$ a marking. $t \in \mathcal{T}$ is *enabled* at $\mathcal{M}$ in $\mathcal{N}$ iff $\bullet t \dot{\subseteq} \mathcal{M}$, $\forall p \in \mathcal{P}((p,t) \in_k \mathcal{L} \ \wedge \ p \in_j \mathcal{M} \rightarrow k \leq j)$, and $\forall p \in \mathcal{P}((p,t) \in_m \mathcal{H} \ \wedge \ p \in_n \mathcal{M} \rightarrow m > n)$.

The enabling condition of the firing rule is extended by two constraints. $\forall p \in \mathcal{P}((p,t) \in_k \mathcal{L} \ \wedge \ p \in_j \mathcal{M} \rightarrow k \leq j)$ is to ensure no transition will violate test arcs, whereas $\forall p \in \mathcal{P}((p,t) \in_k \mathcal{H} \ \wedge \ p \in_j \mathcal{M} \rightarrow k > j)$ is to ensure no transition will violate inhibitor arcs. The notion *fire* and *firing sequence* are defined as before.

ACPP Representation in APN

It is clear that ordinary Petri nets are not able to represent advanced conjunctive planning problems. The reason is that ordinary Petri nets do not have a mechanism to prevent a transition from being fired. To cope with this problem, we use an advanced Petri net. Barret showed that advanced Petri nets are able to model concurrent events where perceived information from an environment can affect what actions are and are not allowed to be taken [3].

ACPP representation in APN is divided into two parts where the ordinary cpp is represented using an ordinary Petri net, preconditions are represented by test arcs and obstacles are represented by inhibitor arcs. Formally,

Definition 4.1.4. Let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ be an acpp. A tuple $\mathcal{AN}_\mathcal{Q} = (\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{H}, \mathcal{L}, \mathcal{I}, \mathcal{G})$ is a representation of $\mathcal{Q}$ in a (marked) advanced Petri net where

- $(\mathcal{P}, \mathcal{T}, \mathcal{F}, \mathcal{I}, \mathcal{G})$ follows the definition (3.1.1);
- $(p,t) \in_k \mathcal{H}$ iff $\exists (t : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}) \in \mathcal{A}$ s.t $p \in_k \mathcal{O}$;
- $(p,t) \in_k \mathcal{L}$ iff $\exists (t : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}) \in \mathcal{A}$ s.t $p \in_k \mathcal{R}$.

From the definition above, we know that for every action $t : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$ in $\mathcal{Q}$, we have a transition $t \in \mathcal{T}$ in APN $\mathcal{AN}_\mathcal{Q}$ with $\blacktriangledown t \doteq \mathcal{O}$, $\blacktriangle t \doteq \mathcal{R}$, $\bullet t \doteq \mathcal{C}$, and $t \bullet \doteq \mathcal{E}$. One may notice that the requirements for the firing rule of a transition are the requirements for an applicable action in acpp. Hence, a transition t is enabled at marking $\mathcal{M}$ in $\mathcal{AN}_{\mathcal{Q}}$ iff there exists an action t in $\mathcal{Q}$ where t is applicable in $\mathcal{M}$

The question of whether there exists a plan P solving an acpp $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ is represented by the question whether there exists a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of APN $\mathcal{AN}_{\mathcal{Q}}$.

4.2 ACPP in Advanced Fluent Calculus

The representation of an acpp in fluent calculus is based on our specification given in Chapter 3. Actions are still represented using a ternary predicate symbol *action*/3 where the first argument encodes the condition, the second argument encodes the name, and the third argument encodes the effect of an action.

To cope with the new feature in acpp, we introduce three new binary predicates: *precon*/2, *hinder*/2, *inhib*/2. *inhib* is a predicate to serve obstacles of an action. For all actions of the form $(a : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E})$ in $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ and for each fluent $o \in_k \mathcal{O}$, we have a fact $inhib(O, A)$ where O encodes o appearing k times in $\mathcal{O}$ and A encodes the name a of the action. Hence, if there are n different fluents in the obstacles of an action then there are n different *inhib*-clause for that action.

The preconditions of an action are encoded with the help of binary predicate *precon* and are of the form

$$precon(R, A)$$

where R encodes the preconditions of an action and A encodes the name of the action. Both *inhib* and *precon* clauses are added in the set $\mathcal{K}_A$.

hinder is a predicate to give information whether an action is hindered by obstacles at a particular state. A clause involving *hinder* is of the form

$$hinder(O \circ S, A) \leftarrow inhib(O, A). \tag{4.1}$$

The clause above specifies that if the obstacles O of an action A are part of the state $O \circ S$ then action A is hindered in this state.

We modify $\mathcal{K}_C$ especially the clause (3.1) to cover wider requirements of an applicable action in acpp. The modification is done as follows

$$\begin{aligned} \mathit{applicable}(C \circ S, A, E \circ S) \leftarrow\ & \mathit{action}(C, A, E) \\ & \wedge\ \mathit{precon}(R, A) \\ & \wedge\ R \circ R' \approx C \circ S \\ & \wedge\ \neg \mathit{hinder}(C \circ S, A) \end{aligned} \tag{4.2}$$

The applicable clause is read declaratively as follows:

> An action A is applicable at state $C \circ S$ and its application yields a state $E \circ S$ if condition C of A is a part of $C \circ S$ and so is precondition R of A and there is no obstacle with action A at state $C \circ S$.

With this modification, $\mathcal{K}_C$ now contains a fact (3.2), clause (3.3), clause (4.1), clause (4.2), and $S \approx S$.

With the introduction of negation in the body of clause (4.2), we use SLDENF-resolution proofs instead of SLDE-resolution proofs. For that, we need to guarantee that our advanced fluent calculus is satisfiable. Furthermore, the SLDENF-resolutions must be free from floundering and infinite derivation [2]. This is all needed to ensure the completeness of SLDENF-resolution. We will show that the completion of $\mathcal{K}_A \cup \mathcal{K}_C \cup \mathcal{E}_{AC1} \cup \mathcal{E}_{\approx}$ is satisfiable.

Let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ be an acpp, and $\mathcal{AFC}_{\mathcal{Q}}$ the fluent calculus representation of $\mathcal{Q}$. We denote by $\mathcal{AFC}^*_{\mathcal{Q}}$ a completion of $\mathcal{K}_A \cup \mathcal{K}_C \cup \mathcal{E}_{AC1} \cup \mathcal{E}_{\approx}$.

Lemma 4.1. *$\mathcal{AFC}^*_{\mathcal{Q}}$ is satisfiable.*

Proof. A model of $\mathcal{AFC}^*_{\mathcal{Q}}$ can be constructed as follows. From [2], $AC1^*$ turns out to be satisfiable. Let

$$\mathcal{M}_{AC1^*} = \{ s \approx t \mid AC1^* \models s \approx t\ \wedge\ s, t \text{ are ground terms } \}$$

then $\mathcal{M}_{AC1^*}$ is a model of $AC1^*$. Let

$$\mathcal{M}_{\mathit{action}} = \mathcal{M}_{AC1^*} \cup \{ \mathit{action}(c, t, e) \mid \mathit{action}(c, t, e) \in \mathcal{K}_A \},$$

$$\mathcal{M}_{\mathit{inhib}} = \mathcal{M}_{\mathit{action}} \cup \{ \mathit{inhib}(o, t) \mid \mathit{inhib}(o, t) \in \mathcal{K}_A \},$$

and

$$\mathcal{M}_{\mathit{precon}} = \mathcal{M}_{\mathit{inhib}} \cup \{ \mathit{precon}(a, t) \mid \mathit{precon}(a, t) \in \mathcal{K}_A \},$$

then $\mathcal{M}_{precon}$ is a model of $\mathcal{K}_A \cup AC1^*$. Let

$$\mathcal{M}_{hinder} = \mathcal{M}_{precon} \cup \{hinder(s,t) \mid \exists o, s' \ \{inhib(o,t), o \circ s' \approx s\} \dot{\subseteq} \mathcal{M}_{precon}\},$$

$$\begin{aligned}\mathcal{M}_{app} = \mathcal{M}_{hinder} \cup \{applicable(s,t,n) \mid \ & \exists s', c, e, a, a' \ s.t \ \{action(c,t,e), c \circ s' \approx s, \\ & e \circ s' \approx n, precon(a,t), a \circ a' \approx s\} \subseteq \mathcal{M}_{hinder} \\ & \wedge \ hinder(s,t) \notin \mathcal{M}_{hinder}\},\end{aligned}$$

$$\mathcal{M}_0 = \mathcal{M}_{app} \cup \{causes(i,[\,],g) \mid i \approx g \in \mathcal{M}_{app}\},$$

and for all $k \geq 1$ let

$$\begin{aligned}\mathcal{M}_k = \mathcal{M}_{k-1} \cup \{causes(i,p,g) \mid \ & \exists p', t, e \ s.t \ \{applicable(i,t,e), causes(e,p',g), \\ & p \approx [t \mid p']\} \subseteq \mathcal{M}_{k-1}\}.\end{aligned}$$

Then, by a straightforward induction on the length of the second argument of *causes* we learn that $\mathcal{M} = \bigcup_{k \in \mathbb{N}} \mathcal{M}_k$ is a model of $\mathcal{K}_C \cup \mathcal{K}_A \cup AC1^*$ and, consequently, $\mathcal{M}$ is a model for $\mathcal{AFC}^*_{\mathcal{Q}}$. □

We would like to have a syntactical condition which guarantees that no derivation flounders or infinite. Unfortunately, clauses in $\mathcal{K}_C$ are recursive. The number of recursive calls to *causes* depends on the second argument of *causes*, which is a list. If the length of this list is known in advance, then each derivation of a goal clause of the form $\leftarrow causes(i,p,g)$ is finite, where i and g are (possibly uninstantiated) fluent terms denoting the initial and the goal state of acpp, respectively, and p is a (possibly uninstantiated) list of a given length. Furthermore, to avoid floundering in SLDENF-resolutions, the initial state i in $\leftarrow causes(i,p,g)$ must be a ground fluent term. These two observations regarding finiteness and non-floundering are combined in the following propositions.

Proposition 4.2. *Let s, a be two simple fluent terms. Then, no SLDENF-resolution proof of $\leftarrow hinder(s,a)$ is infinite.*

Proof. The goal clause $\leftarrow hinder(s,a)$ can only AC1-unify with the head of a new variant

$$hinder(O \circ S, A) \leftarrow inhib(O, A)$$

of (4.1) where it yields to

$$\leftarrow inhib(o,a)$$

Hence, the AC1-unifiers for $hinder(o,a)$ and the head of the clause (4.1) must be of the form

$$\sigma = \{O \mapsto o, S \mapsto s', A \mapsto a\}$$

where $s \approx_{AC1} o \circ s'$. At this point, the derivation either fails or the literal $inhib(o, a)$ can be solved with a fact in $\mathcal{K}_A$ to continue to an empty clause. $\square$

Proposition 4.3. *No SLDENF-resolution proof of* $\leftarrow causes(\mathcal{I}^{-I}, [a_1, \ldots, a_n], \mathcal{G}^{-I})$ *flounders or is infinite.*

Proof. The proposition is proven by induction on the length of the list of the second argument in *causes*.
I.B To show: No SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}^{-I}, [\,], \mathcal{G}^{-I})$ flounders or is infinite.
In case where $n = 0$, i.e., the list is empty, $causes(\mathcal{I}^{-I}, [\,], \mathcal{G}^{-I})$ can only be AC1-unified with a new variant

$$causes(S, [\,], S)$$

of (3.2) and only if $\mathcal{I}^{-I} \approx_{AC1} \mathcal{G}^{-I}$. Hence, we obtain the empty clause or the derivation fails immediately.

I.H No SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}_n^{-I}, [a_n, \ldots, a_1], \mathcal{G}^{-I})$ flounders or is infinite.

I.C No SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}^{-I}, [a_{n+1}, \ldots, a_1], \mathcal{G}^{-I})$ flounders or is infinite.

I.S In the case $n+1 > 0$, $causes(\mathcal{I}^{-I}, [a_{n+1}, \ldots, a_1], \mathcal{G}^{-I})$ can only be AC1-unified with the head of a new variant

$$causes(I, [A \mid P], G) \leftarrow applicable(I, A, S) \wedge causes(S, P, G)$$

of (3.3) where the successful derivation yields

$$\leftarrow applicable(\mathcal{I}^{-I}, a_{n+1}, S) \ \wedge \ causes(S, [a_n, \ldots, a_1], \mathcal{G}^{-I}) \tag{4.3}$$

with an AC1-unifier of the form

$$\{I \mapsto \mathcal{I}^{-I}, A \mapsto a_{n+1}, P \mapsto [a_n, \ldots, a_1], G \mapsto \mathcal{G}^{-I}\}$$

The first atom in (4.3) and the head of a new variant

$$\begin{aligned} applicable(C \circ S', A', E \circ S') \leftarrow \quad & action(C, A', E) \wedge precon(R, A') \\ & \wedge\ R \circ R' \approx C \circ S' \wedge \neg hinder(C \circ S', A') \end{aligned}$$

of (4.2) are AC1-unifiable with an AC1-unifier of the form

$$\{C \mapsto \mathcal{C}^{-I}, S' \mapsto \mathcal{I}'^{-I}, A' \mapsto a_{n+1}, S \mapsto E \circ \mathcal{I}'^{-I}\}$$

leading to

$$\begin{aligned}\leftarrow\quad & \mathit{action}(\mathcal{C}^{-I}, a_{n+1}, E) \;\wedge\; \mathit{precon}(R, a_{n+1}) \;\wedge\; R \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \\ & \wedge \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{n+1}) \;\wedge\; \mathit{causes}(E \circ \mathcal{I}'^{-I}, [a_n, \ldots, a_1], \mathcal{G}^{-I})\end{aligned} \tag{4.4}$$

where $\mathcal{I} \doteq \mathcal{C} \,\dot{\cup}\, \mathcal{I}'$. At this point, there are two possibilities. Either the derivation finitely fails, or the atom $\mathit{action}(\mathcal{C}^{-I}, a_{n+1}, E)$ and a fact

$$\mathit{action}(\mathcal{C}^{-I}, a_{n+1}, \mathcal{E}^{-I})$$

from $\mathcal{K}_A$ are AC1-unifiable with an AC1-unifier of the form

$$\{E \mapsto \mathcal{E}^{-I}\}.$$

If the latter is the case, then there is an SLDENF-derivation of (4.4) to

$$\begin{aligned}\leftarrow\quad & \mathit{precon}(R, a_{n+1}) \;\wedge\; R \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \;\wedge\; \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{n+1}) \\ & \wedge\, \mathit{causes}(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, [a_n, \ldots, a_1], \mathcal{G}^{-I})\end{aligned} \tag{4.5}$$

Once more, there are two possibilities. Either the derivation finitely fails, or the atom $\mathit{precon}(R, a_{n+1})$ and a fact

$$\mathit{precon}(\mathcal{R}^{-I}, a_{n+1})$$

from $\mathcal{K}_A$ are AC1-unifiable with an AC1-unifier of the form

$$\{R \mapsto \mathcal{R}^{-I}\}.$$

If the latter is the case, then there is an SLDENF-derivation of (4.5) to

$$\begin{aligned}\leftarrow\quad & \mathcal{R}^{-I} \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \;\wedge\; \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{n+1}) \\ & \wedge\, \mathit{causes}(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, [a_n, \ldots, a_1], \mathcal{G}^{-I})\end{aligned}$$

Again, there are two possibilities. Either the derivation finitely fails, or the atom
$\mathcal{R}^{-I} \circ A' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I}$ and a new variant

$$X \approx X$$

of $S \approx S$ in $\mathcal{K}_C$ are AC1-unifiable. If $\mathcal{I}$ is a multiset of simple fluents then $\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}$ is a ground fluent term. Hence, if the latter is the case we obtain an AC1-unifier of the form

$$\{R' \mapsto (\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{R})^{-I}, X \mapsto \mathcal{C}^{-I} \circ \mathcal{I}'^{-I}\}$$

and it yields

$$\leftarrow \neg hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{n+1}) \wedge causes(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, [a_n, \ldots, a_1], \mathcal{G}^{-I}).$$

According to the preceding definition in chapter 2, the negative literal $\neg hinder(\mathcal{I}^{-I}, a_{n+1})$ can be selected since it is ground. Proposition (4.2) proves that the corresponding SLDENF-tree is finite. Thus, there are two cases, either the derivation finitely fails or the derivation continues with

$$\leftarrow causes(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, [a_n, \ldots, a_1], \mathcal{G}^{-I}). \tag{4.6}$$

If the derivation continues with (4.6) then we can use the induction hypothesis where $\mathcal{I}_n^{-I} \approx_{AC1} \mathcal{I}'^{-I} \circ \mathcal{E}^{-I}$ to show that the derivation neither flounders nor is infinite since the plan $[a_n, \ldots, a_1]$ has the length n. □

Proposition 4.3 tells that the length of the second argument of *causes* must be fixed in advanced to prevent from having a possible infinite derivation. Based on this result, we refine the question of finding a plan solving an acpp to finding a plan with length n solving an acpp with an initial state $\mathcal{I}$ and a goal state $\mathcal{G}$. In advanced fluent calculus, this is a question of whether

$$(\exists A_1, \ldots, A_n) causes(\mathcal{I}^{-I}, [A_1, \ldots, A_n], \mathcal{G}^{-I})$$

is a logical consequence of the completion of the logic program $\mathcal{K}_A \cup \mathcal{K}_C$ and the (AC1) axioms $\mathcal{E}_{\approx} \cup \mathcal{E}_{AC1}$.

One should remember that we only count the number of SLDENF-refutation steps involving subgoals *causes*. Therefore, the length of an SLDENF-resolution is the number of SLDENF-refutation steps involving subgoals *causes*. Furthermore, we are still dealing with simple fluents in advanced cpp. Hence, when we do SLDENF-resolution proofs, the substitutions being AC1-unifiers for two atoms are still unique [15].

4.3 An Instance of ACPP

Consider the simple modification of the ill man problem mentioned in the introduction of this chapter:

> *Suppose there was a man who was severely ill living in an apartment. He could not go by himself to see a doctor. An ambulance car was asked to bring him to hospital. He was supposed to be taken from his apartment to the ambulance car. However, he was really fat such that he could not fit through his apartment's door.*

This problem is no longer a cpp, but rather an acpp. The fluents are ill, apt, amb, hos, fat as the ill man, the apartment, the ambulance, the hospital, and being fat, respectively. The actions are *to carry him to the ambulance* (c), and *to drive the ambulance car to hospital* (d). The solution by carrying him to the ambulance and driving the ambulance car to hospital is no longer valid since now the ill man is really fat such that he can not pass through his door in his apartment. Within the advanced conjunctive planning problem we obtain

$$\begin{aligned}\mathcal{Q} = \langle \quad & \dot{\{} ill, fat, apt \dot{\}}, \dot{\{} ill, fat, hos \dot{\}}, \{ c : \dot{\{} apt \dot{\}} \xRightarrow{\dot{\{} ill \dot{\}}, \dot{\{} fat \dot{\}}} \dot{\{} amb \dot{\}}, \\ & d : \dot{\{} amb \dot{\}} \xRightarrow{\dot{\{} ill \dot{\}}, \emptyset} \dot{\{} hos \dot{\}} \} \qquad \rangle\end{aligned}$$

As it is shown here, action (c) has an obstacle $\dot{\{} fat \dot{\}}$. Since the initial state $\dot{\{} ill, fat, apt \dot{\}}$ has fat in it, we can not perform action (c). Furthermore, action (d) can not be performed because the initial state does not have amb in it at which it is needed as the condition to perform action (d). Therefore, there is no solution to this problem.

In the fluent calculus, the actions are represented by the set of clauses

$$\{action(apt, c, amb), action(amb, d, hos)\}.$$

Inhibitors of action c are represented by the set of clauses

$$\{inhib(fat, c)\}$$

and the preconditions of action c and d are represented by the set of clauses

$$\{precon(ill, c), precon(ill, d)\}$$

If we ask, whether there exists a plan P such that its execution transforms state $\dot{\{} ill, fat, apt \dot{\}}$ into state $\dot{\{} ill, fat, hos \dot{\}}$, i.e.

$$\exists P : causes(\dot{\{} ill, fat, apt \dot{\}}^{-I}, P, \dot{\{} ill, fat, hos \dot{\}}^{-I})$$

then we will not obtain any successful SLDENF-refutation. One potential SLDENF-refutation is failed because there is an SLDENF-refutation of $hinder(\dot{\{} ill, fat, apt \dot{\}}^{-I}, c)$.

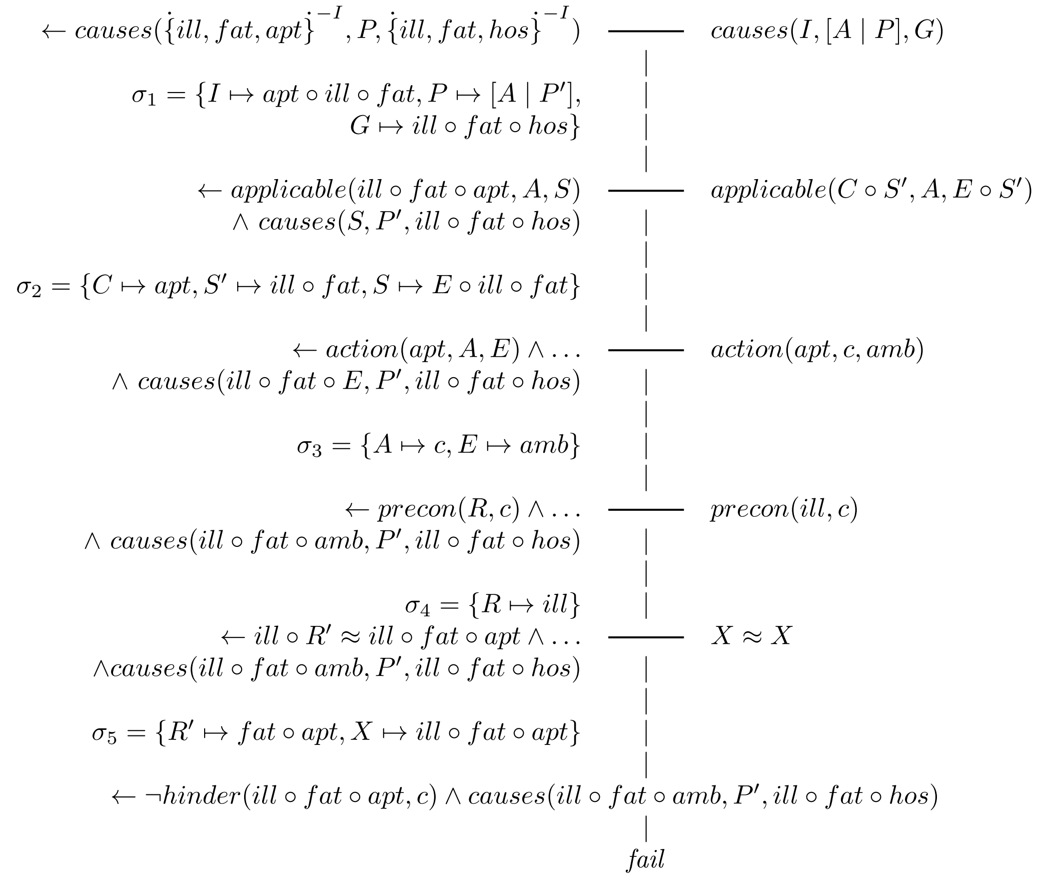
$\leftarrow causes(\dot{\{}ill, fat, apt\dot{\}}^{-I}, P, \dot{\{}ill, fat, hos\dot{\}}^{-I})$
$causes(I, [A \mid P], G)$
$\sigma_1 = \{I \mapsto apt \circ ill \circ fat, P \mapsto [A \mid P'], G \mapsto ill \circ fat \circ hos\}$
$\leftarrow applicable(ill \circ fat \circ apt, A, S) \wedge causes(S, P', ill \circ fat \circ hos)$
$applicable(C \circ S', A, E \circ S')$
$\sigma_2 = \{C \mapsto apt, S' \mapsto ill \circ fat, S \mapsto E \circ ill \circ fat\}$
$\leftarrow action(apt, A, E) \wedge \ldots \wedge causes(ill \circ fat \circ E, P', ill \circ fat \circ hos)$
$action(apt, c, amb)$
$\sigma_3 = \{A \mapsto c, E \mapsto amb\}$
$\leftarrow precon(R, c) \wedge \ldots \wedge causes(ill \circ fat \circ amb, P', ill \circ fat \circ hos)$
$precon(ill, c)$
$\sigma_4 = \{R \mapsto ill\}$
$\leftarrow ill \circ R' \approx ill \circ fat \circ apt \wedge \ldots \wedge causes(ill \circ fat \circ amb, P', ill \circ fat \circ hos)$
$X \approx X$
$\sigma_5 = \{R' \mapsto fat \circ apt, X \mapsto ill \circ fat \circ apt\}$
$\leftarrow \neg hinder(ill \circ fat \circ apt, c) \wedge causes(ill \circ fat \circ amb, P', ill \circ fat \circ hos)$
fail

FIGURE 4.2: A fail SLDENF-resolution proof of $causes(\dot{\{}ill, fat, apt\dot{\}}^{-I}, P, \dot{\{}ill, fat, hos\dot{\}}^{-I})$.

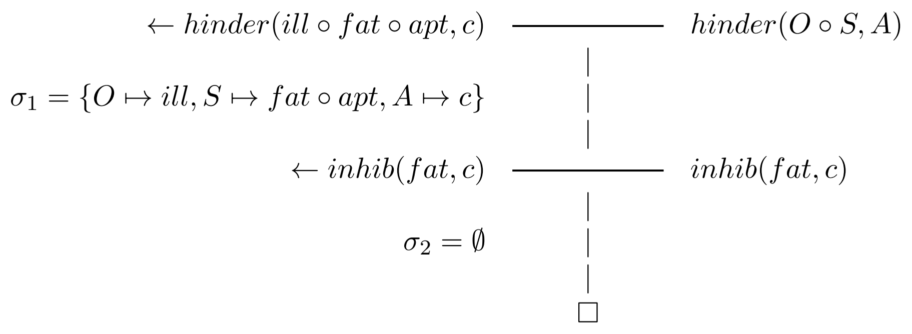
$\leftarrow hinder(ill \circ fat \circ apt, c)$
$hinder(O \circ S, A)$
$\sigma_1 = \{O \mapsto ill, S \mapsto fat \circ apt, A \mapsto c\}$
$\leftarrow inhib(fat, c)$
$inhib(fat, c)$
$\sigma_2 = \emptyset$
$\square$

FIGURE 4.3: An SLDENF-resolution proof of $hinder(\dot{\{}ill, fat, apt\dot{\}}^{-I}, c)$.

Figure 4.2 shows how the potential SLDENF-refutation of $causes(\dot{\{}ill, fat, apt\dot{\}}^{-I}, P, \dot{\{}ill, fat, hos\dot{\}}^{-I})$ fails. Figure 4.3 shows the SLDENF-refutation of $hinder(\dot{\{}ill, fat, apt\dot{\}}^{-I}, c)$.

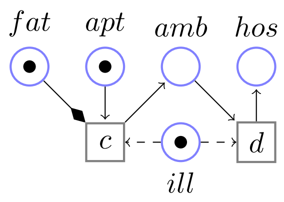

FIGURE 4.4: An advanced Petri net for the modified ill man problem.

In APN, the modified ill man problem can be represented as

$$\begin{aligned}\mathcal{AN}_{\mathcal{Q}} \;=(\;& \{ill, apt, amb, hos, fat\}, \{c, d\}, \dot{\{}(apt, c), (c, amb), (amb, d), (d, hos)\dot{\}},\\ & \dot{\{}(fat, c)\dot{\}}, \dot{\{}(ill, c), (ill, d)\dot{\}}, \dot{\{}ill, fat, apt\dot{\}}, \dot{\{}ill, fat, hos\dot{\}} \qquad)\end{aligned}$$

If we ask, whether there exists a firing sequence P such that its execution transforms marking $\dot{\{}ill, fat, apt\dot{\}}$ into $\dot{\{}ill, fat, hos\dot{\}}$, i.e.

$$\dot{\{}ill, fat, apt\dot{\}} \xrightarrow{P} \dot{\{}ill, fat, hos\dot{\}}$$

then we obtain no firing sequence because transition c can not be fired due to an inhibitory arc connecting fat and c and the initial marking which has token fat. Figure 4.4 shows the initial configuration of APN $\mathcal{AN}_{\mathcal{Q}}$.

Let's continue the story by adding the following information:
A helicopter was sent to help him. A crane was used to carry him out of his windows to the helicopter. The helicopter brought him to the hospital.

We have a helicopter (hel) as the additional simple fluent. We also have two additional actions: *carrying him out of his windows by a crane* (cr) and *piloting the helicopter to the hospital* (p). The acpp is

$$\begin{aligned}\mathcal{Q}' = \langle \;& \dot{\{}ill, fat, apt\dot{\}}, \dot{\{}ill, fat, hos\dot{\}}, \{c : \dot{\{}apt\dot{\}} \overset{\dot{\{}ill\dot{\}},\dot{\{}fat\dot{\}}}{\Longrightarrow} \dot{\{}amb\dot{\}},\\ & d : \dot{\{}amb\dot{\}} \overset{\dot{\{}ill\dot{\}},\emptyset}{\Longrightarrow} \dot{\{}hos\dot{\}}, cr : \dot{\{}apt\dot{\}} \overset{\dot{\{}ill\dot{\}},\emptyset}{\Longrightarrow} \dot{\{}hel\dot{\}}, p : \dot{\{}hel\dot{\}} \overset{\dot{\{}ill\dot{\}},\emptyset}{\Longrightarrow} \dot{\{}hos\dot{\}}\} \quad \rangle\end{aligned}$$

It is easy to see that the only solution for this problem is by *carrying him out of his windows by the crane* and *piloting the helicopter to the hospital.* In the fluent calculus, the additional actions are represented by the set of clauses

$$\{action(apt, cr, hel), action(hel, p, hos)\}.$$

and the preconditions of action cr and p are represented by the set of clauses

$$\{precon(ill, cr), precon(ill, p)\}$$

$$\begin{array}{rcl}
\leftarrow causes(\dot{\{}ill, fat, apt\dot{\}}^{-I}, P, \dot{\{}ill, fat, hos\dot{\}}^{-I}) & \text{———} & causes(I, [A \mid P'], G) \\
\sigma_1 = \{I \mapsto apt \circ ill \circ fat, P \mapsto [A \mid P'], & | & \\
G \mapsto ill \circ fat \circ hos\} & | & \\
\leftarrow applicable(ill \circ fat \circ apt, A, S) & \text{———} & applicable(C \circ S', A, E \circ S') \\
\wedge\ causes(S, P', ill \circ fat \circ hos) & | & \\
\sigma_2 = \{C \mapsto apt, S' \mapsto ill \circ fat & | & \\
S \mapsto E \circ ill \circ fat\} & | & \\
\leftarrow action(apt, A, E) \wedge \ldots & \text{———} & action(apt, cr, hel) \\
\wedge\ causes(ill \circ fat \circ E, P', ill \circ fat \circ hos) & | & \\
\sigma_3 = \{A \mapsto cr, E \mapsto hel\} & | & \\
 & \vdots & \\
\leftarrow causes(ill \circ fat \circ hel, P', ill \circ fat \circ hos) & \text{———} & causes(I', [A' \mid P''], G') \\
\sigma_4 = \{I' \mapsto ill \circ fat \circ hel, P' \mapsto [A' \mid P''], & | & \\
G' \mapsto ill \circ fat \circ hos\} & | & \\
\leftarrow applicable(ill \circ fat \circ hel, A', S') \wedge & \text{———} & applicable(C' \circ S'', A', E' \circ S'') \\
\wedge\ causes(S', P'', ill \circ fat \circ hos) & | & \\
\sigma_5 = \{C' \mapsto hel, S'' \mapsto ill \circ fat, & | & \\
S' \mapsto E' \circ fat \circ ill\} & | & \\
\leftarrow action(hel, A', E') \wedge \ldots & \text{———} & action(hel, p, hos) \\
\wedge\ causes(ill \circ fat \circ E', P'', ill \circ fat \circ hos) & | & \\
\sigma_6 = \{A' \mapsto p, E' \mapsto hos\} & | & \\
 & \vdots & \\
\leftarrow causes(ill \circ fat \circ hos, P'', ill \circ fat \circ hos) & \text{———} & causes(S''', [\,], S''') \\
\sigma_7 = \{S''' \mapsto ill \circ fat \circ hos, P'' \mapsto [\,]\} & | & \\
 & \square &
\end{array}$$

FIGURE 4.5: A succesful SLDENF-resolution proof of $causes(\dot{\{}ill, fat, apt\dot{\}}^{-I}, P, \dot{\{}ill, fat, hos\dot{\}}^{-I})$.

No inhibitors are needed for these new actions. The solution for this problem is depicted in the SLDENF-resolution proof shown in Figure 4.5.

In APN, the modified problem with additional actions can be represented as

$$\begin{aligned}
\mathcal{AN}'_Q = (\ & \{ill, apt, amb, hos, fat, hel\}, \{c, d, cr, p\}, \dot{\{}(apt, c), (c, amb), (amb, d), \\
& (d, hos), (apt, cr), (cr, hel), (hel, p), (p, hos)\dot{\}}, \dot{\{}(fat, c)\dot{\}}, \dot{\{}(ill, c), (ill, d), \\
& (ill, cr), (ill, p)\dot{\}}, \dot{\{}ill, fat, apt\dot{\}}, \dot{\{}ill, fat, hos\dot{\}} \qquad)
\end{aligned}$$

Figure 4.6 shows the firing sequence of APN $\mathcal{AN}'_Q$ as the solution for $\mathcal{Q}'$.

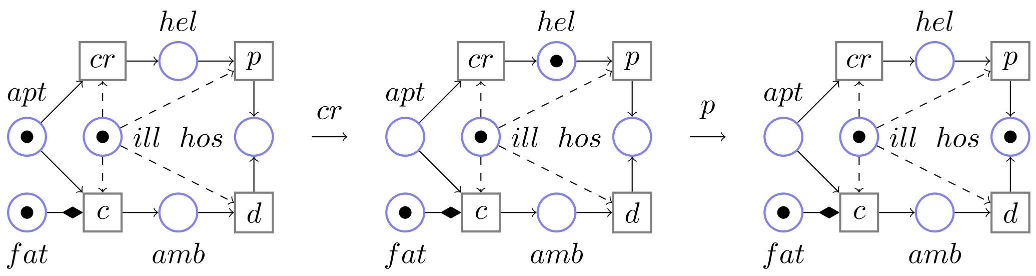

FIGURE 4.6: A solution for the modified ill man problem in APN $\mathcal{AN}'_{\mathcal{Q}}$.

4.4 A Correspondence Between AFC and APN

We have modeled a situation where some resources may hinder an action under advanced conjunctive planning problems and have shown that these problems can also be modeled in the fluent calculus. Here, we will show that a slight modification of Theorem 3.1 holds when we change cpp to acpp, ordinary Petri nets to APN, and fluent calculus to advanced fluent calculus. Throughout this section let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$ denote an advanced conjunctive planning problem and let $\mathcal{AFC}_{\mathcal{Q}}$, and $\mathcal{AN}_{\mathcal{Q}}$ denote its presentation in the advanced fluent calculus, and APN, respectively.

Proposition 4.4. *There are enough obstacles in a state $\mathcal{S}$ to hinder an action a in $\mathcal{Q}$ iff there is an SLDENF-resolution proof of $\leftarrow hinder(\mathcal{S}^{-I}, a)$ in $\mathcal{AFC}_{\mathcal{Q}}$.*

Proof. $\Rightarrow$ To show: there is an SLDENF-resolution proof of $\leftarrow hinder(\mathcal{S}^{-I}, a)$. Assume there are enough obstacles in a state $\mathcal{S}$ to hinder an action a. Furthermore, assume that the action a is of the form

$$a : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$$

At least one type of obstacles in $\mathcal{O}$ appears in state $\mathcal{S}$. Let $\mathcal{O}' \dot{\subseteq} \mathcal{S}$ contain only fluents o with multiplicity n where o is the obstacle. Hence, we have $o \in_m \mathcal{S}$, $o \in_n \mathcal{O}$ and $m \geq n$. Hence, there must be a corresponding fact

$$inhib(\mathcal{O}'^{-I}, a) \tag{4.7}$$

in $\mathcal{AFC}_{\mathcal{Q}}$. Let σ

$$\{O \mapsto \mathcal{O}'^{-I}, S \mapsto (\mathcal{S} \dot{\setminus} \mathcal{O}')^{-I}, A \mapsto a\}$$

be a substitution. σ is the AC1-unifier for the atom occurring in

$$\leftarrow hinder(\mathcal{S}^{-I}, a) \tag{4.8}$$

and the head of a new variant

$$hinder(O \circ S, A) \leftarrow inhib(O, A)$$

of (4.1). Therefore, there is an SLDENF-derivation from (4.8) to

$$\leftarrow inhib(\mathcal{O}'^{-I}, a) \tag{4.9}$$

We can do an SLDENF-derivation using (4.7) and an empty substitution to (4.9) to reach an empty clause. Hence, there is an SLDENF-resolution proof of (4.8).

$\Leftarrow$ To show: There are enough obstacles in a state $\mathcal{S}$ to hinder an action a.
Assume there is an SLDENF-resolution proof of a goal clause

$$\leftarrow hinder(\mathcal{S}^{-I}, a) \tag{4.10}$$

It means there is an SLDENF-derivation from (4.10) to

$$\leftarrow inhib(\mathcal{O}'^{-I}, a) \tag{4.11}$$

using a new variant

$$hinder(O \circ S, A) \leftarrow inhib(O, A)$$

of (4.1) and an AC1-unifier σ which must have the form

$$\{O \mapsto \mathcal{O}'^{-I}, S \mapsto \mathcal{S}'^{-I}, A \mapsto a\},$$

where $\mathcal{S} \doteq \mathcal{O}' \dot\cup \mathcal{S}'$. We can find a fact

$$inhib(\mathcal{O}'^{-I}, a) \tag{4.12}$$

in $\mathcal{AFC}_\mathcal{Q}$ to obtain an SLDENF-derivation from (4.11) to the empty clause using the empty substitution and (4.12).

From (4.12), we know that there is an action a in $\mathcal{Q}$. Assume that the action a is of the form

$$a : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E} \tag{4.13}$$

From (4.12) and (4.13), we conclude that $\mathcal{O}' \dot\subseteq \mathcal{O}$. Furthermore, (4.12) shows that $\mathcal{O}'^{-I}$ is a fluent term of the form $o \circ \ldots \circ o$. We obtain $o \in_n \mathcal{O}'$ and $o \in_n \mathcal{O}$. As $\mathcal{O}' \dot\subseteq \mathcal{S}$, we may conclude that $o \in_m \mathcal{S}$ where $m \geq n$. Hence, there are enough obstacles o at $\mathcal{S}$ to hinder the action a. $\square$

In the following theorem, we show that a solution for an acpp can either be generated by an SLDENF-resolution proof of $\mathcal{AFC}_\mathcal{Q}$ or be a firing sequence from the initial state to the goal state in $\mathcal{AN}_\mathcal{Q}$.

Theorem 4.5. *The following statements are equivalent for a plan p of acpp $\mathcal{Q}$.*

1. *p is a solution for $\mathcal{Q}$.*
2. *p is generated by an SLDENF-resolution proof of $\mathcal{AFC}_\mathcal{Q}$.*
3. *p is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_\mathcal{Q}$.*

Proof. To proof the theorem it suffices to show that 1 implies 2, 2 implies 3, and 3 implies 1. These implications are proven in Lemma 4.6, 4.7, and 4.8.

Lemma 4.6. *If p is a solution for $\mathcal{Q}$ then p is generated by an SLDENF-resolution proof of $\mathcal{AFC}_\mathcal{Q}$.*

Proof. We prove it by induction on length n of solution p.
I.B To show: If $[\,]$ is a solution for $\mathcal{Q}$ then $[\,]$ is generated by an SLDENF-resolution proof of $\mathcal{AFC}_\mathcal{Q}$.
If $[\,]$ is both a plan and a solution for $\mathcal{Q}$. It means that $\mathcal{I} \doteq \mathcal{G}$ so that there is no need to do any action. Let

$$\sigma = \{S \mapsto \mathcal{I}^{-I}, P \mapsto [\,]\}$$

be a substitution. Then σ is the AC1-unifier for the atom occurring in

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{I}^{-I}) \tag{4.14}$$

and a new variant

$$causes(S, [\,], S)$$

of (3.2). Thus, there exists an SLDENF-refutation of (4.14) generating the plan $[\,]$.

I.H If $[a_j, \ldots, a_1]$ is a solution for $\mathcal{Q}_j = \langle \mathcal{I}_j, \mathcal{G}, \mathcal{A} \rangle$ then $[a_j, \ldots, a_1]$ is generated by an SLDENF-resolution proof of $\mathcal{AFC}_\mathcal{Q}$.
I.C To show: If $[a_{j+1}, \ldots, a_1]$ is a solution for $\mathcal{Q}$ then $[a_{j+1}, \ldots, a_1]$ is generated by an SLDENF-resolution proof of $\mathcal{AFC}_\mathcal{Q}$.
I.S Suppose $[a_{j+1}, \ldots, a_1]$ is a solution for $\mathcal{Q}$. a_{j+1} is the first action taken. Lets assume that action a_{j+1} is of the form

$$a_{j+1} : \mathcal{C} \stackrel{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$$

such that by applying action a_{j+1} from the initial state $\mathcal{I}$, we obtain a state $(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$. For the action a_{j+1}, we find the corresponding facts

$$\mathit{action}(\mathcal{C}^{-I}, a_{j+1}, \mathcal{E}^{-I}) \tag{4.15}$$

$$\mathit{precon}(\mathcal{R}^{-I}, a_{j+1}) \tag{4.16}$$

in $\mathcal{AFC}_\mathcal{Q}$. Let

$$\sigma = \{I \mapsto \mathcal{I}^{-I}, G \mapsto \mathcal{G}^{-I}, P \mapsto [A \mid P']\}$$

σ is the AC1-unifier between the atom occurring in

$$\leftarrow \mathit{causes}(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{4.17}$$

and the head of a new variant

$$\mathit{causes}(I, [A \mid P'], G) \leftarrow \mathit{applicable}(I, A, S) \wedge \mathit{causes}(S, P', G)$$

of (3.3). Therefore, there is an SLDENF-derivation from (4.17) to

$$\leftarrow \mathit{applicable}(\mathcal{I}^{-I}, A, S) \ \wedge \ \mathit{causes}(S, P', \mathcal{G}^{-I}) \tag{4.18}$$

The first atom occurring in (4.18) and the head of a new variant

$$\begin{aligned} \mathit{applicable}(C \circ S', A', E \circ S') \leftarrow \ & \mathit{action}(C, A', E) \wedge \mathit{precon}(R, A') \\ & \wedge R \circ R' \approx C \circ S' \wedge \neg \mathit{hinder}(C \circ S', A') \end{aligned}$$

of (4.2) are AC1-unifiable with

$$\sigma_1 = \{A' \mapsto A, C \mapsto \mathcal{C}^{-I}, S' \mapsto \mathcal{I}'^{-I}, S \mapsto E \circ \mathcal{I}'^{-I}\}$$

as the AC1-unifier where $\mathcal{I} \doteq \mathcal{C} \dot{\cup} \mathcal{I}'$. The SLDENF-derivation of (4.18) is

$$\begin{aligned} \leftarrow \quad & \mathit{action}(\mathcal{C}^{-I}, A, E) \ \wedge \ \mathit{precon}(R, A) \ \wedge \ R \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \\ & \wedge \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, A) \ \wedge \ \mathit{causes}(\mathcal{I}'^{-I} \circ E, P', \mathcal{G}^{-I}) \end{aligned} \tag{4.19}$$

The first atom occurring in (4.19) and the fact (4.15) are AC1-unifiable with

$$\{A \mapsto a_{j+1}, E \mapsto \mathcal{E}^{-I}\}$$

yielding an SLDENF-derivation from (4.19) to

$$\begin{aligned} \leftarrow \quad & \mathit{precon}(R, a_{j+1}) \ \wedge \ R \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \\ & \wedge \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1}) \ \wedge \ \mathit{causes}(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}) \end{aligned} \tag{4.20}$$

There is an SLDENF-derivation of (4.20) yielding

$$\leftarrow \mathcal{R}^{-I} \circ R' \approx \mathcal{C}^{-I} \circ\ \mathcal{I}'^{-I} \wedge\ \neg hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1})\ \wedge\ causes(\mathcal{I}'^{-I} \circ\ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}) \tag{4.21}$$

between the first atom occurring in (4.20) and (4.16), using a substitution

$$\{R \mapsto \mathcal{R}^{-I}\}$$

as AC1-unifier.

Since a_{j+1} is applicable to the initial state $\mathcal{I}$ in acpp $\mathcal{Q}$, it means that the preconditions are parts of the initial state $\mathcal{I}$, i.e., $\mathcal{R} \dot{\subseteq} \mathcal{I}$. Hence, there is an SDLE-derivation of (4.21) yielding

$$\leftarrow \neg hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1})\ \wedge\ causes(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}) \tag{4.22}$$

between the first atom occurring in (4.21) and a new variant

$$X \approx X$$

of $S \approx S$ in $\mathcal{K}_C$, and with substitution

$$\{R' \mapsto (\mathcal{I} \dot{\setminus} \mathcal{R})^{-I}, X \mapsto \mathcal{C}^{-I} \circ \mathcal{I}'^{-I}\}.$$

We can safely say that there are not enough obstacles at $\mathcal{I}$ for the action a_{j+1} which can hinder a_{j+1} due to the fact that a_{j+1} is applicable in $\mathcal{I}$. Hence, we can use Proposition 4.4 to say that there is no SLDENF-resolution proof of $hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1})$. In other words, all SLDENF-resolutions of $hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1})$ finitely fail. Hence, we can construct the corresponding finitely failed SLDENF-tree of $hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1})$. Thus, there is an SLDENF-derivation from (4.22) to

$$\leftarrow causes(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}). \tag{4.23}$$

By applying a_{j+1} to the initial state $\mathcal{I}$, we end up to a new acpp with a new initial state $\mathcal{I}_j \doteq (\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$ with a plan $[a_j, \ldots, a_1]$ as the solution to go to the goal state $\mathcal{G}$. Hence, we can apply the induction hypothesis and learn that there is an SLDENF-refutation of (4.23) yielding the computed answer substitution $\{P' \mapsto [a_j, \ldots, a_1]\}$.

Combining this refutation with derivation from (4.17) to (4.23) yields an SLDENF-refutation of (4.17) with computed answer substitution $\{P \mapsto [a_{j+1}, \ldots, a_1]\}$, i.e, the plan $[a_{j+1}, \ldots, a_1]$ is generated. □

Lemma 4.7. *If p is generated by an SLDENF-resolution proof of $\mathcal{AFC}_Q$ then p is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_Q$.*

Proof. The lemma is an immediate consequence of the following claim.

If there is an SLDENF-refutation of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ generating plan p, then there is a firing sequence p from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_Q$.

The claim is proven by induction on the length j of the SLDENF-refutation.
I.B To show: If there is an SLDENF-refutation of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ with length 1 generating plan p, then there is a firing sequence p of $\mathcal{AN}_Q$ from $\mathcal{I}$ to $\mathcal{G}$.

If the length of the SLDENF-refutation is 1, then the atom occurring in

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{4.24}$$

and a new variant

$$causes(S, [\,], S)$$

of (3.2) must be AC1-unifiable and SLDENF-refutation of (4.24) generates the empty plan $[\,]$. Furthermore, the first and the third argument of *causes* in (4.24) refer to the same fluent term. Therefore, we get that $\mathcal{I}^{-I} \approx_{AC1} \mathcal{G}^{-I}$. We may conclude that the AC1-unifier for (4.24) and the variant of (3.2) is of the form

$$\{P \mapsto [\,], S \mapsto \mathcal{I}^{-I}\}.$$

Since $\mathcal{I}^{-I} \approx_{AC1} \mathcal{G}^{-I}$, we obtain $\mathcal{I} \doteq \mathcal{G}$. We may conclude that in Petri net $\mathcal{AN}_Q$, the initial marking and the goal marking are the same. Therefore, there is no need to fire any transition, i.e,

$$\mathcal{I} \xrightarrow{[\,]} \mathcal{I}.$$

Hence, there is a firing sequence $[\,]$ from $\mathcal{I}$ to $\mathcal{I}$ of $\mathcal{AN}_Q$.

I.H If there is an SLDENF-refutation of $\leftarrow causes(\mathcal{I}_j^{-I}, P_j, \mathcal{G}^{-I})$ with length j generating plan p_j, then there is a firing sequence p_j of $\mathcal{AN}_Q$ from $\mathcal{I}_j$ to $\mathcal{G}$.

I.C If there is an SLDENF-refutation of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ with length $j+1$ generating plan p, then there is a firing sequence p of $\mathcal{AN}_Q$ from $\mathcal{I}$ to $\mathcal{G}$.

I.S Assume there is an SLDENF-refutation of

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{4.25}$$

with length $j+1$ and generating plan $[a_1, ..., a_j]$. Hence, we can find an SLDENF-derivation from (4.25) to

$$\leftarrow \mathit{applicable}(\mathcal{I}^{-I}, A, S) \ \wedge \ \mathit{causes}(S, P', \mathcal{G}^{-I}) \tag{4.26}$$

as a result of the atom occurring in (4.25) and the head of a new variant

$$\mathit{causes}(I, [A \mid P'], G) \leftarrow \mathit{applicable}(I, A, S) \wedge \mathit{causes}(S, P', G)$$

of (3.3) with AC1-unifier

$$\{I \mapsto \mathcal{I}^{-I}, P \mapsto [A \mid P'], G \mapsto \mathcal{G}^{-I}\}.$$

The first atom occurring in (4.26) can only be AC1-unified with the head of a new variant

$$\begin{aligned} \mathit{applicable}(C \circ S', A', E \circ S') \leftarrow \quad & \mathit{action}(C, A', E) \wedge \mathit{precon}(R, A') \\ & \wedge\ R \circ R' \approx C \circ S' \wedge \neg \mathit{hinder}(C \circ S', A') \end{aligned}$$

of (4.2) with AC1-unifier

$$\{C \mapsto \mathcal{C}^{-I}, S' \mapsto \mathcal{I}'^{-I}, A' \mapsto A, S \mapsto E \circ \mathcal{I}'^{-I}\},$$

where $\mathcal{I}^{-I} \approx_{AC1} \mathcal{C}^{-I} \circ \mathcal{I}'^{-I}$. It yields an SLDENF-derivation of (4.26) to

$$\begin{aligned} \leftarrow \quad & \mathit{action}(\mathcal{C}^{-I}, A, E) \ \wedge \ \mathit{precon}(R, A) \ \wedge \ R \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \\ & \wedge\ \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, A) \ \wedge \ \mathit{causes}(\mathcal{I}'^{-I} \circ E, P', \mathcal{G}^{-I}). \end{aligned} \tag{4.27}$$

We can find a fact

$$\mathit{action}(\mathcal{C}^{-I}, a_1, \mathcal{E}^{-I}) \tag{4.28}$$

in $\mathcal{AFC}_Q$ which is AC1-unifiable with the first atom occurring in (4.27) with AC1-unifier

$$\{A \mapsto a_1, E \mapsto \mathcal{E}^{-I}\}.$$

It yields an SLDENF-derivation of (4.27) to

$$\begin{aligned} \leftarrow \quad & \mathit{precon}(R, a_1) \ \wedge \ R \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \\ & \wedge\ \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_1) \ \wedge \ \mathit{causes}(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}). \end{aligned} \tag{4.29}$$

We can, once again, find a fact

$$\mathit{precon}(\mathcal{R}^{-I}, a_1) \tag{4.30}$$

in $\mathcal{AFC}_Q$ which is AC1-unifiable with the first atom occurring in (4.29) with AC1-unifier

$$\{R \mapsto \mathcal{R}^{-I}\}.$$

It yields an SLDE-derivation from (4.29) to

$$\leftarrow \mathcal{R}^{-I} \circ R' \approx \mathcal{C}^{-I} \circ \mathcal{I}'^{-I} \wedge \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_1) \wedge \mathit{causes}(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}). \quad (4.31)$$

There is an SLDENF-derivation of (4.31) yielding

$$\leftarrow \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_1) \ \wedge \ \mathit{causes}(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}) \quad (4.32)$$

between the first atom occurring in (4.31) and a new variant

$$X \approx X$$

of $S \approx S$ in $\mathcal{K}_C$ and with substitution

$$\{X \mapsto \mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, R' \mapsto (\mathcal{I} \dot{\setminus} \mathcal{R})^{-I}\}$$

as AC1-unifier. For the first literal occurring in (4.32), there must be a finitely failed SLDENF-refutation of $\mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_1)$ such that the successful SLDENF-derivation of (4.32) leads to

$$\leftarrow \mathit{causes}(\mathcal{I}'^{-I} \circ \mathcal{E}^{-I}, P', \mathcal{G}^{-I}). \quad (4.33)$$

From (4.28) and (4.30) we may conclude that there must be an action a_1 with preconditions $\mathcal{R}$, conditions $\mathcal{C}$ and effects $\mathcal{E}$ in $\mathcal{Q}$. Therefore, there must be a transition a_1 with $\blacktriangle a_1 \doteq \mathcal{R}$, $\bullet a_1 \doteq \mathcal{C}$ and $a_1 \bullet \doteq \mathcal{E}$ in $\mathcal{AN}_Q$.

Because there is a derivation from (4.32) to (4.33), we can conclude that there is a finitely failed SLDENF-tree of $\mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_1)$. It means all the SLDENF-resolution proofs of $\mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_1)$ finitely fail. We can use Proposition 4.4 and learn that there are not enough obstacles in state $\mathcal{I}$ for the action a_1. Let assume action a_1 is of the form

$$a_1 : \mathcal{C} \xRightarrow{\mathcal{R},\mathcal{O}} \mathcal{E}$$

where $\mathcal{O}$ is multiset of simple fluents for obstacles. As $\mathcal{C} \dot{\subseteq} \mathcal{I}, \mathcal{R} \dot{\subseteq} \mathcal{I}$ and there are not enough obstacles, i.e., $\forall o \in_n \mathcal{O}\ (o \in_m \mathcal{I} \rightarrow n > m)$, we can conclude that action a_1 is applicable at $\mathcal{I}$ in $\mathcal{Q}$. Thus, transition a_1 is enabled and can be fired such that

$$\mathcal{I} \xrightarrow{a_1} (\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}. \quad (4.34)$$

We can apply (I.H) to (4.33) by setting $\mathcal{I}_j \doteq (\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$, $\mathcal{G}$ as it is, and $P_j = [a_2, \ldots, a_j]$ since the SLDENF-refutation only has length j. We find a firing sequence $[a_2, \ldots, a_j]$ from $(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$,i.e.,

$$(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E} \xrightarrow{[a_2,\ldots,a_j]} \mathcal{G}. \tag{4.35}$$

Combining (4.34) and (4.35), we obtain a firing sequence $[a_1, \ldots, a_j]$ such that

$$\mathcal{I} \xrightarrow{[a_1,\ldots,a_j]} \mathcal{G}.$$

Because there is a firing sequence p from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$, the claim is true. □

Lemma 4.8. *If p is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$ then p is a solution for $\mathcal{Q}$.*

Proof. We prove the lemma by induction on the length of firing sequence p.
I.B To show: If [ ] is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$ then [ ] is a solution for $\mathcal{Q}$. If [ ] is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$, i.e.,

$$\mathcal{I} \xrightarrow{[]} \mathcal{G}$$

then we can conclude that $\mathcal{I} \doteq \mathcal{G}$. Trivially, in $\mathcal{Q}$, we do not need to do any action since $\mathcal{I} \doteq \mathcal{G}$.

I.H If $[a_j, \ldots, a_1]$ is a firing sequence from $\mathcal{I}_j$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$ then $[a_j, \ldots, a_1]$ is a solution for $\mathcal{Q}_j = \langle \mathcal{I}_j, \mathcal{G}, \mathcal{A} \rangle$.
I.C If $[a_{j+1}, \ldots, a_1]$ is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$ then $[a_{j+1}, \ldots, a_1]$ is a solution for $\mathcal{Q}$.
I.S Suppose $[a_{j+1}, \ldots, a_1]$ is a firing sequence from $\mathcal{I}$ to $\mathcal{G}$ of $\mathcal{AN}_{\mathcal{Q}}$, i.e.,

$$\mathcal{I} \xrightarrow{[a_{j+1},\ldots,a_1]} \mathcal{G}.$$

a_{j+1} is the first enabled transition and assume that $\bullet a_{j+1} \doteq \mathcal{C}$, $a_{j+1}\bullet \doteq \mathcal{E}$, $\blacktriangle a_{j+1} \doteq \mathcal{R}$ and $\blacktriangledown a_{j+1} \doteq \mathcal{O}$. Hence, there is an action

$$a_{j+1} : \mathcal{C} \xRightarrow{\mathcal{R},\mathcal{O}} \mathcal{E}$$

in $\mathcal{Q}$. a_{j+1} is enabled at initial marking $\mathcal{I}$ and is fired leading to a new marking $(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$, i.e.,

$$\mathcal{I} \xrightarrow{a_{j+1}} (\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}.$$

Since the definitions of firing rule in advanced Petri nets and an applicable action in acpp are the same, we can conclude that a_{j+1} obeys the firing rule in $\mathcal{AN}_{\mathcal{Q}}$ and is applicable at state $\mathcal{I}$ in $\mathcal{Q}$. Thus, we can apply the action a_{j+1} to $\mathcal{I}$ yielding a new state $(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$.

By executing a_{j+1} at the initial marking $\mathcal{I}$, we have a firing sequence $[a_j, \ldots, a_1]$ such that

$$(\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E} \xrightarrow{[a_j,\ldots,a_1]} \mathcal{G} \tag{4.36}$$

Because this proof contains $[a_j, \ldots, a_1]$ and the goal marking in (I.H) is the same as in (4.36), we may conclude that $\mathcal{I}_j \doteq (\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E}$ and apply the induction hypothesis to learn that $[a_j, \ldots, a_1]$ is a solution for $\mathcal{Q}_j = \langle \mathcal{I}_j, \mathcal{G}, \mathcal{A} \rangle$.

$\mathcal{Q}_j$ with solution $[a_j, \ldots, a_1]$ has the initial state $(\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E}$. Applying a_{j+1} at state $\mathcal{I}$ in $\mathcal{Q}$ leads to state $(\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E}$ which leads acpp $\mathcal{Q}_j$. Thus, by combining this information, we learn that $[a_{j+1}, \ldots, a_1]$ is a solution for the acpp $\mathcal{Q}$. $\square$

Chapter 5

Advanced Planning Problems$^{\mathbb{R}}$

Up to now we defined conjunctive planning problems with the help of simple fluents. Simple fluents are an excellent representation of resources. Inspired by Petri nets as designed in [3] to use real values to model concurrent action, we would like to enhance our planning problems with information in the form of real values. To cover planning problems with real-valued information, we define conjunctive planning problems with fluents which can contain real-valued information. For that, we allow fluents with one argument which is an arithmetic expression. We introduce a set of *simple arithmetic expressions* as follows:

Definition 5.0.1. The set *Exp* of simple arithmetic expressions is the smallest set of expressions satisfying the following conditions:

1. Every variable is an arithmetic expression;
2. Every real number is an arithmetic expression;
3. If s is an arithmetic expression then so is $-s$; and
4. If s, t are arithmetic expressions then $s + t$, $s - t$, $s * t$, and s^t are arithmetic expressions.

Then fluents with arithmetic expressions are defined as follows:

Definition 5.0.2. $\mathcal{R}f$ is the smallest set of unary function symbols. *Real-valued fluents* are fluents of the form $r(v)$ at which $r \in \mathcal{R}f$ and $v \in Exp$.

To operate on real numbers, the usual arithmetic operations along with their standard interpretation accompany the real numbers. Values from arithmetic operations can be obtained by evaluating mathematical expressions inside real-valued fluents using a function *eval* defined as follows:

Definition 5.0.3. Let $eval : Exp \mapsto \mathbb{R}$ be a function which evaluates ground arithmetic expressions as usual and returns their value.

- $eval'$ is a mapping extending the function $eval$ such that

$$eval'(t) = \left\{\begin{array}{ll} r(eval(v)) & \text{if } t \text{ is a ground real-valued fluent of the form } r(v) \\ t & \text{otherwise} \end{array}\right\},$$

- $eval''$ is a mapping extending the function $eval'$ to multisets such that

$$eval'(t) \in_k eval''(\mathcal{M}) \text{ iff } t \in_k \mathcal{M} \ \wedge \ t \text{ is ground}$$

  where $\mathcal{M}$ is a multiset of fluents.

We drop the quote symbols from $eval'$ and $eval''$ whenever we perform arithmetic evaluations on fluent terms and multisets if the evaluations are clear from the context.

We define an extension of advanced conjunctive planning problems by taking real-valued fluents into consideration. We split the set of actions into two disjoint sets of actions. The first one is a set of actions that is intended to handle discrete resources which can be consumed and produced. The second one is a set of actions which should deal with real-valued fluents. Actions in the latter set do not consume values in their inputs.

Definition 5.0.4. Let $\mathcal{S}_s$ be a finite set of simple fluents and $\mathcal{S}_r$ be a finite set of real-valued fluents, and $\mathcal{S}_{sm}, \mathcal{S}_{rm}$ be finite multisets constructed from $\mathcal{S}_s$ and $\mathcal{S}_r$, respectively. *An advanced conjunctive planning problem with real values* (acpp$^{\mathbb{R}}$) is a tuple $\langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$, where

- $\mathcal{I} \dot{\subseteq} (\mathcal{S}_{sm} \dot{\cup} \mathcal{S}_{rm})$ s.t all fluents in $\mathcal{I}$ are ground;
- $\mathcal{G} \dot{\subseteq} (\mathcal{S}_{sm} \dot{\cup} \mathcal{S}_{rm})$ s.t all fluents in $\mathcal{G}$ are ground;
- For each $r \in \mathcal{R}f$, r appears at most once in $\mathcal{I}$ and in $\mathcal{G}$;
- $\mathcal{A}$ is a finite set of *concrete actions* of the form $A : \mathcal{C} \xRightarrow{\mathcal{R},\mathcal{O}} \mathcal{E}$ s.t $\mathcal{C} \dot{\subseteq} \mathcal{S}_{sm}$, $\mathcal{E} \dot{\subseteq} \mathcal{S}_{sm}$, $\mathcal{R} \dot{\subseteq} (\mathcal{S}_{sm} \dot{\cup} \mathcal{S}_{rm})$, $\mathcal{O} \dot{\subseteq} (\mathcal{S}_{sm} \dot{\cup} \mathcal{S}_{rm})$, and all fluents in $\mathcal{R} \dot{\cup} \mathcal{O}$ are ground;
- For each concrete action in $\mathcal{A}$ and for each $r \in \mathcal{R}f$, r appears at most once in $\mathcal{R}$ and $\mathcal{O}$;
- $\mathcal{A}_{real}$ is finite set of *theory actions* of the form $A_{real} : \mathcal{C}_{real} \xRightarrow{\mathcal{R}_{real},\mathcal{O}_{real}} \mathcal{E}_{real}$ s.t $\mathcal{C}_{real} \dot{\subseteq} \mathcal{S}_{rm}$ and $\mathcal{E}_{real} \dot{\subseteq} \mathcal{S}_{rm}$, $\mathcal{R}_{real} \dot{\subseteq} \mathcal{S}_{sm}$, and $\mathcal{O}_{real} \dot{\subseteq} \mathcal{S}_{sm}$;
- Each variable occurring in its effects must also occur in its conditions for each theory action in $\mathcal{A}_{real}$;

- For each $r \in \mathcal{R}f$, r appears at most once in $\mathcal{A}_{real}$; and
- All action names must be either in $\mathcal{A}$ or in $\mathcal{A}_{real}$ but not in both sets.

One can observe that an acpp$^{\mathbb{R}}$ $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ is an advanced cpp where $\mathcal{I}, \mathcal{G}$ only contain simple fluents and $\mathcal{A}_{real}$ is an empty set. We extend the definition of an applicable action in acpp$^{\mathbb{R}}$ to take real-valued fluents into consideration. We redefine the definition of applicable action as follows:

Definition 5.0.5. Let $\mathcal{S}$ be a multiset of simple and ground real-valued fluents, $\mathcal{R}_s, \mathcal{O}_s$ be multisets of simple fluents, $\mathcal{R}_r, \mathcal{O}_r$ be a multiset of ground real-valued fluents and $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ be an acpp$^{\mathbb{R}}$. A concrete action

$$A : \mathcal{C} \xRightarrow{\mathcal{R}_s \,\dot\cup\, \mathcal{R}_r, \mathcal{O}_s \,\dot\cup\, \mathcal{O}_r} \mathcal{E}$$

is *applicable* to $\mathcal{S}$ iff the following conditions

- $\mathcal{C} \dot\subseteq \mathcal{S}$; $\mathcal{R}_s \dot\subseteq \mathcal{S}$;
- $\forall o \in_n \mathcal{O}_s \ (o \in_m \mathcal{S} \rightarrow m < n)$;
- $\forall r(v) \in_1 \mathcal{O}_r \ \exists r(v') \in_1 \mathcal{S} \ (eval(v) > eval(v'))$; and
- $\forall r(v) \in_1 \mathcal{R}_r \ \exists r(v') \in_1 \mathcal{S} \ (eval(v) \leq eval(v'))$

are satisfied. The *application of a concrete action* leads to the multiset

$$(\mathcal{S} \,\dot\setminus\, \mathcal{C}) \,\dot\cup\, \mathcal{E}.$$

A theory action

$$A_{real} : \mathcal{C}_{real} \xRightarrow{\mathcal{R}_s, \mathcal{O}_s} \mathcal{E}_{real}$$

is *applicable* to $\mathcal{S}$ iff the following conditions

- there exists a substitution σ such that $\mathcal{C}\sigma \dot\subseteq \mathcal{S}$;
- $\mathcal{R}_s \dot\subseteq \mathcal{S}$; and
- $\forall o \in_n \mathcal{O}_s \ (o \in_m \mathcal{S} \rightarrow m < n)$

are satisfied. The *application of a theory action* to $\mathcal{S}$ leads to the multiset

$$(\mathcal{S} \,\dot\setminus\, \{r(v) \mid r(v) \in_1 \mathcal{S} \ \wedge \ \exists r(v') \in_1 \mathcal{E}_{real}\}) \,\dot\cup\, eval(\mathcal{E}_{real}\sigma).$$

We denote by $\mathcal{A}_{real}(\mathcal{S})$ a set of theory actions in $\mathcal{A}_{real}$ which are applicable in state $\mathcal{S}$.

An applicable concrete action in a state is an extension of applicable actions for extended actions in acpp. It is extended by the rules $\forall r(v) \in_1 \mathcal{O}_r\ \exists r(v') \in_1 \mathcal{S}\ (eval(v) > eval(v'))$ and $\forall r(v) \in_1 \mathcal{R}_r\ \exists r(v') \in_1 \mathcal{S}\ (eval(v) \leq eval(v'))$ to ensure that real-valued fluents are measured by their values if they appear either in the preconditions or the obstacles of the action.

A theory action has preconditions of simple fluents that must be met and the obstacles of simple fluents that must be avoided. However, to be applicable a theory action has to have a ground substitution for the conditions of the action such that they become part of the state. The application of a theory action replaces the current value of real-valued fluents in the state with the new one. One should note that the effects of an applicable theory action are ground as a result of having a ground substitution for the conditions of the action.

One should note that each real-valued fluents appears at most once in every state. This is the result of restricting the appearance of real-valued fluents in an initial state, conditions of a theory action, and effects of a theory action to at most once. Hence, when we mention states, we mean states in which each real-valued fluent appears at most once.

Definition 5.0.6. Let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ be an acpp$^{\mathbb{R}}$, and $\mathcal{S}, \mathcal{S}'$ be states. $\mathcal{S}'$ is the *closed state* of $\mathcal{S}$, denoted by $\mathcal{S} \downarrow$, iff $\mathcal{S}'$ is the closure of $\mathcal{S}$ under the theory actions in $\mathcal{A}_{real}$.

The following lemma shows that applying two applicable theory actions to a state in any order ends up in the same state.

Lemma 5.1. *Let $\mathcal{S}, \mathcal{S}'$ be states and a_1, a_2 be theory actions in $\mathcal{S}$. If $\mathcal{S} \xrightarrow{[a_1,a_2]} \mathcal{S}'$ then $\mathcal{S} \xrightarrow{[a_2,a_1]} \mathcal{S}'$.*

Proof. To show: $\mathcal{S} \xrightarrow{[a_2,a_1]} \mathcal{S}'$.
Assume that $\mathcal{S} \xrightarrow{[a_1,a_2]} \mathcal{S}'$ holds. Furthermore, assume a_1 is of the form

$$a_1 : \mathcal{C}_1 \overset{\mathcal{R}_1,\mathcal{O}_1}{\Longrightarrow} \mathcal{E}_1$$

and a_2 is of the form

$$a_2 : \mathcal{C}_2 \overset{\mathcal{R}_2,\mathcal{O}_2}{\Longrightarrow} \mathcal{E}_2$$

The restriction stating that "for each $r \in \mathcal{R}f$, r appears at most once in $\mathcal{A}_{real}$" implies that each $r \in \mathcal{R}f$ appears at most once either in $\mathcal{C}_1$, in $\mathcal{C}_2$, in $\mathcal{E}_1$ or in $\mathcal{E}_2$.

a_1 is the first executed theory action in $\mathcal{S}$. It means that there is a ground substitution σ such that $\mathcal{C}_1\sigma \dot{\subseteq} \mathcal{S}$. Hence, the state $\mathcal{S}$ is of the form $\mathcal{C}_1\sigma \dot{\cup} \mathcal{S}_1$. The result of applying

a_1 adds $\mathcal{E}_1\sigma$ to $(\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_1)$ where $\mathcal{E}'_1 = \{r(v) \mid r(v) \in_1 \mathcal{S}_1 \ \wedge \ \exists r(v') \in_1 \mathcal{E}_1\}$. In other words,

$$\mathcal{C}_1\sigma \dot{\cup} \mathcal{S}_1 \xrightarrow{a_1} \mathcal{C}_1\sigma \dot{\cup} ((\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_1) \dot{\cup} \mathcal{E}_1\sigma)$$

holds. When executing a_2 in $\mathcal{C}_1\sigma \dot{\cup} ((\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_1) \dot{\cup} \mathcal{E}_1\sigma)$, it satisfies that there exists a ground substitution σ' such that $\mathcal{C}_2\sigma' \dot{\subseteq} (\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_1)$. Hence, $(\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_1) = \mathcal{C}_2\sigma' \dot{\cup} \mathcal{S}_2$ is true. The result of applying a_2 adds $\mathcal{E}_2\sigma'$ specifically to $(\mathcal{S}_2 \dot{\setminus} \mathcal{E}'_2)$ where $\mathcal{E}'_2 = \{r(v) \mid r(v) \in_1 \mathcal{S}_2 \ \wedge \ \exists r(v') \in_1 \mathcal{E}_2\}$. In other words,

$$\mathcal{C}_1\sigma \dot{\cup} ((\mathcal{C}_2\sigma' \dot{\cup} \mathcal{S}_2) \xrightarrow{a_2} \mathcal{C}_1\sigma \dot{\cup} ((\mathcal{C}_2\sigma' \dot{\cup} ((\mathcal{S}_2 \dot{\setminus} \mathcal{E}'_2) \dot{\cup} \mathcal{E}_2\sigma')$$

holds where $\mathcal{S}' = ((\mathcal{C}_2\sigma' \dot{\cup} ((\mathcal{S}_2 \dot{\setminus} \mathcal{E}'_2) \dot{\cup} \mathcal{E}_2\sigma')$.

Because $(\mathcal{C}_2\sigma' \dot{\cup} \mathcal{S}_2) \dot{\subseteq} \mathcal{S}$, we can apply a_2 in $\mathcal{S}$. There is no problem with $\mathcal{R}_2$ since $\mathcal{R}_2$ is constructed from simple fluents and $\mathcal{S}_2$ are the possible multiset containing simple fluents. Hence, $\mathcal{R}_2 \dot{\subseteq} \mathcal{S}_2$ must hold. $\forall o \in_j \mathcal{O}_2 \ (o \in_m \mathcal{S}_2 \rightarrow m < j)$ is also satisfied based on our assumption that a_2 is executed in $\mathcal{C}_1\sigma \dot{\cup} ((\mathcal{C}_2\sigma' \dot{\cup} \mathcal{S}_2)$. Hence, the following transformation

$$\mathcal{S} \xrightarrow{a_2} \mathcal{C}_1\sigma \dot{\cup} ((\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_2) \dot{\cup} \mathcal{E}_2\sigma')$$

holds. The statements $\mathcal{R}_1 \dot{\subseteq} \mathcal{S}_2$ and $\forall o \in_j \mathcal{O}_1 \ (o \in_m \mathcal{S}_2 \rightarrow m < j)$ are true because $\mathcal{S}_2$ are the only possible multiset containing simple fluents and the fact that $\mathcal{S}_2$ is a submultiset of $\mathcal{C}_1\sigma \dot{\cup} ((\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_2) \dot{\cup} \mathcal{E}_2\sigma')$. Hence, a_1 can be applied in $\mathcal{C}_1\sigma \dot{\cup} ((\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_2) \dot{\cup} \mathcal{E}_2\sigma')$, i.e,

$$\mathcal{C}_1\sigma \dot{\cup} ((\mathcal{S}_1 \dot{\setminus} \mathcal{E}'_2) \dot{\cup} \mathcal{E}_2\sigma') \xrightarrow{a_1} \mathcal{S}'$$

Hence, $\mathcal{S} \xrightarrow{[a_2,a_1]} \mathcal{S}'$ is true. □

The following lemma shows that we do not need to apply an applicable theory action more than once because the resulting state does not change after the first application.

Lemma 5.2. *Let $\mathcal{S}, \mathcal{S}', \mathcal{S}''$ be states and a be a theory action. If $\mathcal{S} \xrightarrow{a} \mathcal{S}'$ and $\mathcal{S}' \xrightarrow{a} \mathcal{S}''$ then $\mathcal{S}' = \mathcal{S}''$.*

Proof. To show $\mathcal{S}' = \mathcal{S}''$.
Assume that $\mathcal{S} \xrightarrow{a} \mathcal{S}'$ and $\mathcal{S}' \xrightarrow{a} \mathcal{S}''$. Furthermore, assume that the theory action a is of the form

$$a : \mathcal{C} \xRightarrow{\mathcal{R},\mathcal{O}} \mathcal{E}$$

Since a is applicable both in $\mathcal{S}$ and $\mathcal{S}'$ then there must exists σ and σ' such that $\mathcal{C}\sigma \dot{\subseteq} \mathcal{S}$ and $\mathcal{C}\sigma' \dot{\subseteq} \mathcal{S}'$ respectively. Hence, the state $\mathcal{S}$ is of the form $\mathcal{C}\sigma \dot{\cup} \mathcal{S}_1$ and the state $\mathcal{S}'$ is of the form $\mathcal{C}\sigma' \dot{\cup} \mathcal{S}'_1$. Both $\mathcal{S}$ and $\mathcal{S}'$ must also satisfy the preconditions $\mathcal{R}$ and obstacles

$\mathcal{O}$. Executing a transforms $\mathcal{S}$ to $\mathcal{S}'$ and $\mathcal{S}'$ to $\mathcal{S}''$, i.e.,

$$\mathcal{C}\sigma \mathbin{\dot\cup} \mathcal{S}_1 \xrightarrow{a} \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}_1 \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$$

$$\mathcal{C}\sigma' \mathbin{\dot\cup} \mathcal{S}_1' \xrightarrow{a} \mathcal{C}\sigma' \mathbin{\dot\cup} ((\mathcal{S}_1' \mathbin{\dot\setminus} \mathcal{E}_1') \mathbin{\dot\cup} \mathcal{E}\sigma')$$

where $\mathcal{E}' = \{r(v) \mid r(v) \in_1 \mathcal{S}_1 \wedge \exists r(v') \in_1 \mathcal{E}\}$ and $\mathcal{E}_1' = \{r(v) \mid r(v) \in_1 \mathcal{S}_1' \wedge \exists r(v') \in_1 \mathcal{E}\}$. We can safely say that $\mathcal{S}' = \mathcal{C}\sigma' \mathbin{\dot\cup} \mathcal{S}_1' = \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}_1 \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$. We can conclude that $\sigma = \sigma'$ since each $r \in \mathcal{R}f$ in $\mathcal{C}$ appears exactly once in $\mathcal{S}'$.

$\mathcal{S}_1'$ is $\mathcal{S}_1$ where each $r \in \mathcal{R}f$ which appears both in $\mathcal{E}$ and in $\mathcal{S}_1$ is replaced by $\mathcal{E}\sigma$. Hence, we can safely say that $(\mathcal{S}_1 \mathbin{\dot\setminus} \mathcal{E}') = (\mathcal{S}_1' \mathbin{\dot\setminus} \mathcal{E}_1')$ is true. As a result

$$\mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}_1 \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma) \xrightarrow{a} \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}_1 \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$$

In other words, $\mathcal{S}' = \mathcal{S}''$. □

The following lemma shows that all applicable theory actions on state $\mathcal{S}$ are still applicable on state $\mathcal{S}'$ if state $\mathcal{S}'$ is the result of applying a sequence of theory actions on state $\mathcal{S}$.

Lemma 5.3. *Let $\mathcal{S}, \mathcal{S}'$ be states and a be a theory action. If $\mathcal{S} \xrightarrow{a} \mathcal{S}'$ then $\mathcal{A}_{real}(\mathcal{S}') = \mathcal{A}_{real}(\mathcal{S})$.*

Proof. To show: $\mathcal{A}_{real}(\mathcal{S}') = \mathcal{A}_{real}(\mathcal{S})$.
Assume that $\mathcal{S} \xrightarrow{a} \mathcal{S}'$. Furthermore, assume that $\mathcal{A}_{real}(\mathcal{S}) = \{a, a_1, \ldots, a_n\}$ and theory action a is of the form

$$a : \mathcal{C} \xrightarrow{\mathcal{R},\mathcal{O}} \mathcal{E}$$

Since a is executed in $\mathcal{S}$, there exists a substitution σ such that $\mathcal{C}\sigma \mathbin{\dot\subseteq} \mathcal{S}$. Hence, the state $\mathcal{S}$ is of the form $\mathcal{C}\sigma \mathbin{\dot\cup} \mathcal{S}''$. Executing a in $\mathcal{S}$ leads to $\mathcal{S}'$, i.e.,

$$\mathcal{C}\sigma \mathbin{\dot\cup} \mathcal{S}'' \xrightarrow{a} \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$$

where $\mathcal{S}' = \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$ and $\mathcal{E}' = \{r(v) \mid r(v) \in_1 \mathcal{S}'' \ \wedge \ \exists r(v') \in_1 \mathcal{E}\}$. $\mathcal{S}'$ is basically $\mathcal{S}$ by replacing each $r \in \mathcal{R}f$ which appears both in $\mathcal{S}$ and in $\mathcal{E}$ with the ones in $\mathcal{E}\sigma$.

$\mathcal{C}\sigma \mathbin{\dot\subseteq} \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$, and $\mathcal{R} \mathbin{\dot\subseteq} \mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$ are true. Furthermore, the statement $\forall o \in_j \mathcal{O}\ (o \in_m (\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \rightarrow m < j)$ is true because $(\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}')$ is the only possible submultiset containing simple fluents and $(\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\subseteq} \mathcal{S}$. Hence, a is applicable in $\mathcal{C}\sigma \mathbin{\dot\cup} ((\mathcal{S}'' \mathbin{\dot\setminus} \mathcal{E}') \mathbin{\dot\cup} \mathcal{E}\sigma)$.

For any $a_i, 1 \leq i \leq n$ of the form

$$a_i : \mathcal{C}_i \xRightarrow{\mathcal{R}_i, \mathcal{O}_i} \mathcal{E}_i$$

we know that $\mathcal{R}_i \dot{\subsetneq} (\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}')$ and $\forall o \in_j \mathcal{O}_i \ (o \in_m (\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}') \to m < j)$ are true because $(\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}') \dot{\subsetneq} \mathcal{S}$ is true. Hence, preconditions and obstacles are satisfied in the state $\mathcal{C}\sigma \mathbin{\dot{\cup}} ((\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}') \mathbin{\dot{\cup}} \mathcal{E}\sigma)$. Moreover, since a_i is applicable in state $\mathcal{S}$ and state $\mathcal{S}$ is of the form $\mathcal{C}\sigma \mathbin{\dot{\cup}} \mathcal{S}''$ then there must exist a substitution σ_i such that $\mathcal{C}_i\sigma_i \dot{\subsetneq} (\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}')$. The $\mathcal{C}_i\sigma_i$ is specifically a submultiset of $(\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}')$ because each $r \in \mathcal{R}f$ which appears in $\mathcal{C}_i$ can not appear in $\mathcal{C}$ or $\mathcal{E}$ due to the restriction 'for each $r \in \mathcal{R}f$, r appears at most once in $\mathcal{A}_{real}$". Hence, we can safely say that each $a_i, 1 \leq i \leq n$ is applicable in state $\mathcal{C}\sigma \mathbin{\dot{\cup}} ((\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}') \mathbin{\dot{\cup}} \mathcal{E}\sigma)$. With this, we can safely say that $\mathcal{A}_{real}(\mathcal{S}') \subseteq \mathcal{A}_{real}(\mathcal{S})$.

Now, we need to prove that $\mathcal{A}_{real}(\mathcal{S}) \subseteq \mathcal{A}_{real}(\mathcal{S}')$. Let assume, there is another theory action $a_k, k > n$ of the form

$$a_k : \mathcal{C}_k \xRightarrow{\mathcal{R}_k, \mathcal{O}_k} \mathcal{E}_k$$

To be applicable in state $\mathcal{C}\sigma \mathbin{\dot{\cup}} ((\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}') \mathbin{\dot{\cup}} \mathcal{E}\sigma)$, there must exist a substitution σ_k for a_k such that $\mathcal{C}_k\sigma_k \dot{\subsetneq} (\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}')$. With the restriction stating that "for each $r \in \mathcal{R}f$, r appears at most once in $\mathcal{A}_{real}$", any element of $\mathcal{C}_k$ can not appear in $\mathcal{C}$ or in $\mathcal{E}$. Since, $(\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}')$ is a submultiset of $\mathcal{S}$ and a_k is not applicable in $\mathcal{S}$, hence, a_k is not applicable in $\mathcal{C}\sigma \mathbin{\dot{\cup}} ((\mathcal{S}'' \mathbin{\dot{\setminus}} \mathcal{E}') \mathbin{\dot{\cup}} \mathcal{E}\sigma)$ either. Therefore, $\mathcal{A}_{real}(\mathcal{S}') = \mathcal{A}_{real}(\mathcal{S})$ is true. □

$\mathcal{S}{\downarrow}$ is a result of performing a sequence of theory actions to a state $\mathcal{S}$ where each applicable theory action must be applied exactly once. This is shown by the following corollary which is deducted from the three lemmas above.

Corollary 5.4. *$\mathcal{S}{\downarrow}$ is compiled by applying each element of $\mathcal{A}_{real}(\mathcal{S})$ exactly once and in any order.*

Proof. By Lemma 5.1, applying each element of $\mathcal{A}_{real}(\mathcal{S})$ on $\mathcal{S}$ in any order will bring to the same state. Lemma 5.2 ensures that applying an element of $\mathcal{A}_{real}(\mathcal{S})$ more than once does not bring to a different state. Lemma 5.3 ensures that there is no case that the successive state $\mathcal{S}'$ of state $\mathcal{S}$ by applying elements of $\mathcal{A}_{real}(\mathcal{S})$ introduces a new theory action that is not an element of $\mathcal{A}_{real}(\mathcal{S})$. Furthermore, what is applicable on $\mathcal{S}$ is still applicable in the successive state. Hence, we can apply each element of $\mathcal{A}_{real}(\mathcal{S})$ on $\mathcal{S}$ exactly once and in any order to obtain $\mathcal{S}{\downarrow}$. □

One should note that as long as a state is not closed, all applicable theory actions can be executed and yield the same result, i.e., to its closed state.

Definition 5.0.7. Let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ be an acpp$^{\mathbb{R}}$, and $\mathcal{S}, \mathcal{S}', \mathcal{S}''$ be states. Let l be a list of actions, t be a list of theory actions and a be a concrete action.

- A plan is defined as follows:
  - $\mathcal{S}\downarrow \xrightarrow{[]} \mathcal{S}\downarrow$.
  - If $\mathcal{S}\downarrow \xrightarrow{[a]} \mathcal{S}'$, $\mathcal{S}' \xrightarrow{t} \mathcal{S}'\downarrow$ and $\mathcal{S}'\downarrow \xrightarrow{l} \mathcal{S}''\downarrow$ then $\mathcal{S}\downarrow \xrightarrow{append([a|t],l)} \mathcal{S}''\downarrow$.
- A plan p is a solution for $\mathcal{Q}$ iff plan p transforms $\mathcal{I}\downarrow$ into $\mathcal{G}\downarrow$.

A plan in acpp$^{\mathbb{R}}$ is a sequence of actions transforming a closed state into another closed state. A solution for an acpp $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ is a plan transforming closed initial state $\mathcal{I}\downarrow$ into the closed goal state $\mathcal{G}\downarrow$. Without losing of generality, we consider the initial state $\mathcal{I} = \mathcal{I}\downarrow$ and the goal state $\mathcal{G} = \mathcal{G}\downarrow$.

In the following section, we will explain the fluent calculus with real-valued information to represent and solve acpp$^{\mathbb{R}}$. We mainly aim at an extension of the fluent calculus. We will not present any Petri nets which can correlate with acpp$^{\mathbb{R}}$ because there is none. Petri nets designed by Barret in [3] may be able to model acpp$^{\mathbb{R}}$. However, these Petri nets do no come with a well defined semantic. The only semantic is informal which can be found in [3].

5.1 ACPP$^{\mathbb{R}}$ in the Real-valued Fluent Calculus

The representation of ACPP$^{\mathbb{R}}$ in real-valued fluent calculus is basically a representation of ACPP in the advanced fluent calculus combined with real-valued information. The problem here is that how to represent real-valued fluents in the fluent calculus. As we know, each acpp$^{\mathbb{R}}$ may have a different set of real-valued fluents. We would like to have a general representation for each real-valued fluent so that we can have a general fluent calculus to represent and solve acpp$^{\mathbb{R}}$. However, taking real-valued fluents as they are will rise a problem especially when we want to model the application of a theory action in a state. It means that for each acpp$^{\mathbb{R}}$, we will have a different set of rules representing the application process of a theory action.

The general representation can be achieved by quantifying the function symbol $r \in \mathcal{R}f$ for each real-valued fluent. However, this is not the case since logic programs are a representation of first order logic. In order to achieve the general representation, we treat each function symbol $r \in \mathcal{R}f$ as a simple fluent. We introduce a binary function symbol $real/2$ to represent real-valued fluents where the first argument encodes the function symbol and the second argument encodes the arithmetic expression of the real-valued fluents.

Concrete actions are represented by the ternary predicate *action* where the conditions are encoded in the first argument, the name is encoded in the second argument, and the effects are encoded in the third argument. Theory actions are represented with the help of new predicate *taction*/3 where the conditions of the actions are encoded in the first argument, the name is encoded in the second argument, and the effects are encoded in the third argument. One should remember that the restriction in acpp$^{\mathbb{R}}$ stating that each variable occurring in the effects must also occur in the conditions is still present here. Moreover, one should notice that all *action* clauses are ground, whereas some *taction* clauses may not be ground.

Preconditions of concrete and theory actions are represented with the help of binary predicate *precon* where the first argument encodes the preconditions of the actions and the second argument encodes the name.

The representation for all simple fluents occurring in obstacles for both concrete and theory actions follows the representation that we have described in previous chapter for obstacles in an advanced cpp. We separate all real-valued fluents in obstacles for a concrete action from simple fluents. We use the same binary predicate *inhib* in the form

$$inhib(O, A)$$

where O encodes all real-valued fluents and A encodes the name of the concrete action.

To deal with real-valued information and arithmetic expressions $e \in Exp$, we add three binary predicates $val/2$, $\leq/2$, and $\geq/2$. val is used as fact clauses where the first argument encodes a ground arithmetic expression e and the second argument encodes the evaluation result of e. $\leq$ and $\geq$ are used as fact clauses where the first and the second argument are real values. $\leq$ and $\geq$ are interpreted as usual comparators. We further assume that there is a background knowledge $\mathcal{E}_{real}$ containing all ground fact clauses $val(e, eval(e))$, $eval(e) \leq eval(e')$, and $eval(e) \geq eval(e')$ where e, e' are ground arithmetic expressions.

Apart from $\mathcal{E}_{real}$, five new predicates *fulfil*/2, *member*/2, *replace*/3, *nonclosed*/1, and *reapplicable*/3 are introduced here. $fulfil/2$ is a predicate to give information whether the first argument (i.e., a state) fulfils the second argument (preconditions of an action). Clauses involving *fulfil* are of the form

$$fulfil(S, 1), \tag{5.1}$$

$$fulfil(P \circ S, P \circ Pr) \leftarrow fulfil(S, Pr), \tag{5.2}$$

$$\begin{aligned} fulfil(real(R, V) \circ S, real(R, Vl) \circ P) \leftarrow\ & val(V, Vv) \wedge val(Vl, Vvl) \\ & \wedge Vv \geq Vvl \wedge fulfil(S, P). \end{aligned} \tag{5.3}$$

The first clause tells that state S fulfils preconditions 1. The second clause tells that state $P \circ S$ fulfils preconditions $P \circ Pr$ if state S fulfils preconditions Pr. The third clause is read declaratively as follows:

> A state $real(R, V) \circ S$ fulfils preconditions $real(R, Vl) \circ P$ if the evaluation of V of the real-valued fluent $real(R, V)$ evaluates to Vv and the evaluation of Vl of the real-valued fluent $real(R, Vl)$ evaluates to Vvl and Vv is greater than or equal to Vvl and a state S fulfils preconditions P.

member is a predicate to check whether a particular real-valued fluent is part of a state. A clause involving *member* is of the form

$$member(real(R, V), real(R, Vl) \circ S). \tag{5.4}$$

replace is a predicate to replace some part of the first argument with the second argument where the result is put in the third argument. This predicate is intended for replacing real-valued fluents from a state with effects of a theory action. Clauses involving *replace* are of the form

$$\begin{aligned}
&replace(S, 1, S), && (5.5)\\
&replace(S, real(R, V) \circ E, real(R, Vl) \circ N) \leftarrow \neg member(real(R, V), S) && (5.6)\\
&\qquad\qquad\qquad\qquad\qquad\qquad \wedge\, val(V, Vl)\\
&\qquad\qquad\qquad\qquad\qquad\qquad \wedge\, replace(S, E, N),\\
&replace(real(R, V) \circ S, real(R, Vl) \circ E, real(R, Vvl) \circ N) \leftarrow val(Vl, Vvl) && (5.7)\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \wedge\, replace(S, E, N).
\end{aligned}$$

The first clause tells that there is nothing to be replaced in state S if the second argument is 1. The second clause is read declaratively as follows:

> A state S is replaced with a state $real(R, Vl) \circ N$ by effects $real(R, V) \circ E$ if $real(R, V)$ is not a part of state S and the evaluation of V gives Vl and the state S is replaced with a new state N by effects E.

The third clause is read declaratively as follows:

> A state $real(R, V) \circ S$ is replaced with a new state $real(R, Vvl) \circ N$ by effects $real(R, Vl) \circ E$ if the evaluation of Vl of the real-valued fluent $real(R, Vl)$ gives Vvl and there is a state replacement N for state S by E.

nonclosed is a predicate to check whether a state is a non-closed state or not. A clause involving *nonclosed* is of the form

$$\begin{aligned} nonclosed(C \circ S) \leftarrow\ & taction(C, A, E) \wedge precon(P, A) \\ & \wedge fulfil(C \circ S, P) \wedge \neg hinder(C \circ S, A) \\ & \wedge replace(C \circ S, E, N) \wedge C \circ S \not\approx N. \end{aligned} \tag{5.8}$$

The clause is read declaratively as follows:

> A state $C \circ S$ is not a closed state if there exists a theory action named A with conditions C, effects E, and preconditions P and $C \circ S$ fulfils P and there is no possible obstacle for state $C \circ S$ in A and the replacement of the state $C \circ S$ with E can give the new state N and and current state $C \circ S$ does not appear to be the same as the future state N.

reapplicable is a predicate to apply all possible theory actions after a state is transformed by a concrete action. Clauses involving *reapplicable* are of the form

$$reapplicable(S, [\,], S) \leftarrow \neg nonclosed(S), \tag{5.9}$$

$$\begin{aligned} reapplicable(C \circ S, [A \mid T], N) \leftarrow\ & taction(C, A, E) \wedge precon(P, A) \\ & \wedge fulfil(C \circ S, P) \wedge \neg hinder(C \circ S, A) \\ & \wedge replace(C \circ S, E, Z) \wedge C \circ S \not\approx Z \\ & \wedge reapplicable(Z, T, N). \end{aligned} \tag{5.10}$$

The first clause tells that if state S is a closed state then state S is transformed into S by $[\,]$. The second clause is read declaratively as follows:

> A series of theory actions $[A \mid T]$ transforms state $C \circ S$ into new closed state N if there exists a theory action named A with conditions C, effects E, and preconditions P and $C \circ S$ fulfils P and there is no possible obstacle for state $C \circ S$ in A and the replacement of the state $C \circ S$ with E can give new state Z and and current state $C \circ S$ does not appear to be the same as the future state Z and a series of theory actions T transforms state Z into new closed state N.

We add a new clause for predicate *hinder* to include the comparison for real-valued fluents.

$$\begin{aligned} hinder(real(R, V) \circ S, A) \leftarrow\ & inhib(real(R, Vl) \circ O, A) \wedge val(V, Vv) \\ & \wedge val(Vl, Vvl) \wedge Vv \geq Vvl \end{aligned} \tag{5.11}$$

The clause above tells that an action A is hindered in a state $real(R,V) \circ S$ if there exists an obstacle R with value Vl in a set of obstacles $real(R,Vl) \circ O$ of the action A where the evaluated value Vv of value V is greater than the evaluated value Vvl of value Vl.

Besides those new clauses, we modify the clause (4.2) such that an applicable concrete action can be represented and executed by ternary predicate *applicable*. The modification goes as follows:

$$\begin{aligned} applicable(C \circ S, A, E \circ S) \leftarrow\ & action(C,A,E) \wedge precon(P,A) \\ & \wedge fulfil(C \circ S, P) \wedge \neg hinder(C \circ S, A). \end{aligned} \tag{5.12}$$

The clause is read declaratively as follows:

> A concrete action A transforms state $C \circ S$ into state $E \circ S$ if there exists a concrete action named A with conditions C, effects E and preconditions P and the state $C \circ S$ fulfils P and there is no possible obstacle for state $C \circ S$ in T.

We also modify the clause (3.3) due to the introduction of new actions and closed states. The modification involves predicate *append* to append a series of theory actions applied to a new state with another series of actions leading to the goal state. The modification goes as follows:

$$\begin{aligned} causes(I, [A \mid L], G) \leftarrow\ & applicable(I,A,S) \wedge append(T,P,L) \\ & \wedge reapplicable(S,T,N) \wedge causes(N,P,G). \end{aligned} \tag{5.13}$$

The clause is read as follows:

> The execution of the plan $[A \mid L]$ transforms closed state I into closed state G if a concrete action A is applicable in state I and yields state S and appending a series of theory actions T with a plan P gives a plan L and a series of theory actions T transforms a state S into a closed state N and there is a plan P which transforms closed state N into a closed state G.

action, *taction*, *inhib* and *precon* are put in $\mathcal{K}_A$. Clauses (5.1), (5.2), (5.3), (5.4), (5.5), (5.6), and (5.7) are categorized under $\mathcal{K}_{Com}$. Clauses (4.1), (5.8), (5.9), (5.10), (5.11), and (5.12) are categorized under $\mathcal{K}_{App}$. The last group is $\mathcal{K}_C$ which only consists of (3.2) and (5.13).

Once more, we need to guarantee that our fluent calculus is satisfiable. We denote by $\mathcal{AFC}\mathbb{R}^*_{\mathcal{Q}}$ a completion of $\mathcal{K}_A \cup \mathcal{K}_C \cup \mathcal{K}_{Com} \cup \mathcal{K}_{App} \cup \mathcal{E}_{AC1} \cup \mathcal{E}_{\approx} \cup \mathcal{E}_{real}$.

Proposition 5.5. *$\mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ is satisfiable.*

Proof. A model of $\mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ can be constructed as follows. From [2], $\mathcal{E}_{AC1}$ turns out to be satisfiable. Let

$$\mathcal{M}_{AC1} = \{s \approx t \mid \mathcal{E}^*_{AC1} \models s \approx t \ \wedge \ s, t \text{ are ground terms } \}.$$

Let

$$\begin{array}{llll}\mathcal{M}_{arit} & = \mathcal{M}_{AC1} & \cup \{val(arit, value) & \mid \ val(arit, value) \in \mathcal{E}_{real} \\ & & & \quad \wedge \ arit, value \text{ are ground terms } \} \\ & & \cup \{s \ X \ t & \mid \ s \ X \ t \in \mathcal{E}_{real} \ \wedge \ X \in \{\leq, \geq\} \\ & & & \quad \wedge \ s, t \text{ are ground terms } \}.\end{array}$$

Let

$$\begin{array}{llll}\mathcal{M}_{mem} & = \mathcal{M}_{arit} & \cup \{member(e, s) & \mid \ \exists r, v, vl, s' \ \{e \approx real(r, v) \\ & & & \quad \wedge \ s \approx real(r, vl) \circ s'\} \subseteq \mathcal{M}_{arit}\}, \\ \mathcal{M}_0 & = \mathcal{M}_{mem} & \cup \{fulfil(s, 1) & \mid \ s \text{ is a ground term } \} \\ & & \cup \{replace(s, 1, ns) & \mid \ s \approx ns \in \mathcal{M}_{mem}\}\end{array}$$

and for all $k \geq 1$ let

$$\begin{array}{llll}\mathcal{M}_k = & \mathcal{M}_{k-1} \cup & \{fulfil(s, p) & \mid \ (\exists p', s', pr \ \{s \approx p \circ s', p \approx p' \circ pr, \\ & & & \quad fulfil(s', pr)\} \subseteq \mathcal{M}_{k-1}) \ \vee \ (\exists r, v, vv, s', vl, vvl, p' \\ & & & \quad \{s \approx real(r, v) \circ s', p \approx real(r, vl) \circ p', val(v, vv), \\ & & & \quad val(vl, vvl), vv \geq vvl, fulfil(s', p')\} \subseteq \mathcal{M}_{k-1})\} \\ & & \cup \{replace(s, e, n) & \mid \ (\exists r, v, vl, e', n' \ \{e \approx real(r, v) \circ e', replace(s, e', n') \\ & & & \quad n \approx real(r, v) \circ n', val(v, vl)\} \subseteq \mathcal{M}_{k-1} \ \wedge \\ & & & \quad member(real(r, v), s) \notin \mathcal{M}_{k-1}\}) \ \vee \ (\exists r, v, s', vl, \\ & & & \quad vvl, e', n' \ \{s \approx real(r, v) \circ s', e \approx real(r, vl) \circ e', \\ & & & \quad val(vl, vvl), n \approx real(r, vvl) \circ n', \\ & & & \quad replace(s', e', n')\} \subseteq \mathcal{M}_{k-1}\}\end{array}$$

Then, by a straightforward induction on the length of the second argument of *fulfil* and *replace* we learn that $\mathcal{M} = \bigcup_{k \in \mathbb{N}} \mathcal{M}_k$ is a model of $\mathcal{K}_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ and, consequently, $\mathcal{M}$ is a model for $\mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$. □

Proposition 5.6. *$\mathcal{K}^*_{A} \cup \mathcal{K}^*_{App} \cup \mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ is satisfiable.*

Proof. Proposition 5.5 has proven that $\mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ is satisfiable. We just need to prove that $\mathcal{K}^*_{A} \cup \mathcal{K}^*_{App}$ is satisfiable. Let $\mathcal{M}_{com}$ be a model for $\mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$.

Let

$$\begin{array}{llll}
\mathcal{M}_{action} & = \mathcal{M}_{com} & \cup \{\ action(c,a,e) & \mid\ action(c,a,e) \in \mathcal{K}_A\}, \\
\mathcal{M}_{taction} & = \mathcal{M}_{action} & \cup \{\ taction(c,a,e) & \mid\ taction(c,a,e) \in \mathcal{K}_A\}, \\
\mathcal{M}_{inhib} & = \mathcal{M}_{taction} & \cup \{\ inhib(o,a) & \mid\ inhib(o,a) \in \mathcal{K}_A\}, \\
\mathcal{M}_{precon} & = \mathcal{M}_{inhib} & \cup \{\ precon(p,a) & \mid\ precon(p,a) \in \mathcal{K}_A\},
\end{array}$$

Let

$$\begin{array}{llll}
\mathcal{M}_{hin} & = \mathcal{M}_{precon} & \cup \{\ hinder(s,a) & \mid\ \exists o, s'\ \{inhib(o,a), \\
 & & & \quad o \circ s' \approx s\} \dot{\subseteq} \mathcal{M}_{precon}\}, \\
\mathcal{M}_{hincon} & = \mathcal{M}_{hin} & \cup \{\ hinder(s,a) & \mid\ \exists s', r, v, vv, vl, vvl, o \\
 & & & \quad \{inhib(real(r,vl) \circ o, a), v \geq vl \\
 & & & \quad s \approx s' \circ real(r,v), val(v,vv), \\
 & & & \quad val(vl,vvl)\} \subseteq \mathcal{M}_{hin}\} \\
\mathcal{M}_{clos} & = \mathcal{M}_{hincon} & \cup \{\ nonclosed(s) & \mid\ \exists c, a, e, p, s', n\{taction(c,a,e), \\
 & & & \quad precon(p,a), fulfil(s,p), s \approx c \circ s', \\
 & & & \quad replace(s,e,n)\} \dot{\subseteq} \mathcal{M}_{hincon} \wedge \\
 & & & \quad \{hinder(s,a), s \approx n\} \cap \mathcal{M}_{hincon} = \emptyset\} \\
\mathcal{M}_{app} & = \mathcal{M}_{clos} & \cup \{\ applicable(s,a,n) & \mid\ \exists c, s', e, p\ \{action(c,a,e), \\
 & & & \quad precon(p,a), fulfil(s,p), s \approx c \circ s', \\
 & & & \quad n \approx e \circ s'\} \subseteq \mathcal{M}_{clos} \wedge \\
 & & & \quad hinder(s,a) \notin \mathcal{M}_{clos}\} \\
\mathcal{M}_0 & = \mathcal{M}_{app} & \cup \{\ reapplicable(s,[\,],n) & \mid\ s \approx n \in \mathcal{M}_{app} \\
 & & & \quad nonclosed(s) \notin \mathcal{M}_{app}\}
\end{array}$$

and for all $k \geq 1$ let

$$\begin{array}{llll}
\mathcal{M}_k & = \mathcal{M}_{k-1} & \cup \{\ reapplicable(s,[a \mid t],n) & \mid\ \exists c, e, p, s', z \\
 & & & \quad \{hinder(s,a), s \approx z\} \cap \mathcal{M}_{k-1} = \emptyset \\
 & & & \quad \wedge \{taction(c,a,e), precon(p,a), \\
 & & & \quad fulfil(s,p), replace(s,e,z), \\
 & & & \quad reapplicable(z,t,n)\} \subseteq \mathcal{M}_{k-1}\}
\end{array}$$

Then, by a straightforward induction on the length of the second argument of *reapplicable* we learn that $\mathcal{M} = \bigcup_{k \in \mathbb{N}} \mathcal{M}_k$ is a model of $\mathcal{K}_A \cup \mathcal{K}_{App} \cup \mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ and, consequently, $\mathcal{M}$ is a model for $\mathcal{K}^*_A \cup \mathcal{K}^*_{App} \cup \mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$. □

Lemma 5.7. *$\mathcal{AFC}\mathbb{R}^*_Q$ is satisfiable.*

Proof. Proposition 5.6 has proven that $\mathcal{K}^*_A \cup \mathcal{K}^*_{App} \cup \mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ is satisfiable. We just need to prove that $\mathcal{K}^*_C$ is satisfiable. Let $\mathcal{M}_{app}$ be a model for $\mathcal{K}^*_A \cup \mathcal{K}^*_{App} \cup$

$\mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real} \cup \mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$. Let

$$\mathcal{M}_0 = \mathcal{M}_{app} \cup \{causes(i, [\,], g) \quad | \quad i \approx g \in \mathcal{M}_{app}\},$$

and for all $k \geq 1$ let

$$\begin{aligned}\mathcal{M}_k = \mathcal{M}_{k-1} \cup \ \{\ causes(i, [a \mid l], g) \mid \ & \exists s, t, p, n\ \{applicable(i, a, s), append(t, p, l), \\ & reapplicable(s, t, n), causes(n, p, g)\} \subseteq \mathcal{M}_{k-1}\}\end{aligned}$$

Then, by a straightforward induction on the length of the second argument of *causes* we learn that $\mathcal{M} = \bigcup_{k \in \mathbb{N}} \mathcal{M}_k$ is a model of $\mathcal{K}_C \cup \mathcal{K}^*_A \cup \mathcal{K}^*_{App} \cup \mathcal{K}^*_{Com} \cup \mathcal{E}^*_{AC1} \cup \mathcal{E}^*_{real}$ and, consequently, $\mathcal{M}$ is a model for $\mathcal{AFCR}^*_{\mathbb{Q}}$. □

Syntactical conditions which guarantee that no derivation flounders or is infinite need to be assured. They are important since we deal with SLDENF-resolution proofs. We focus on all recursive clauses such as clauses where predicate *fulfil*, *replace*, *reapplicable*, are the head of the clauses. We also need to reprove Lemma 4.3 since there is a modification in clauses where *causes* is the head. There is no need to worry about clauses (5.11), (5.8), and (5.12) because these clauses are not recursive.

Clauses (5.1), (5.2), and (5.3) will not have infinite derivations as long as the second argument is finite. This is shown by the following proposition.

Proposition 5.8. *Let $p_1, \ldots, p_n$ be ground fluent terms, and s be an arbitrary fluent term. No SLDENF-resolution proof of $\leftarrow fulfil(s, p_n \circ \ldots \circ p_1)$ is infinite.*

Proof. The proposition is proven by induction on the length of the second argument in *fulfil*.
I.B To show: No SLDENF-resolution proof of $\leftarrow fulfil(s, 1)$ is infinite.
In the case where $n = 0$, what is left in the second argument is 1 since $p_1 \circ \ldots \circ p_n \circ 1 \approx p_1 \circ \ldots \circ p_n$. In this case, $fulfil(s, 1)$ can only be AC1-unified with a new variant of (5.1). Thus, we obtain the empty clause or the derivation fails immediately.

I.H No SLDENF-resolution proof of $\leftarrow fulfil(sn, p_n \circ \ldots \circ p_1)$ is infinite.
I.C To show: No SLDENF-resolution proof of $\leftarrow fulfil(s, p_{n+1} \circ \ldots \circ p_1)$ is infinite.
I.S In the case $n + 1 > 0$, there are two possibilities for $fulfil(s, p_{n+1} \circ \ldots \circ p_1)$ to be AC1-unified with. The first is with the head of a new variant

$$fulfil(P \circ S, P \circ Pr) \leftarrow fulfil(S, Pr)$$

of (5.2) with AC1-unifier

$$\{P \mapsto p_j \circ \ldots \circ p_1, S \mapsto sn, Pr \mapsto p_{n+1} \circ \ldots \circ p_{j+1}\}$$

yielding

$$\leftarrow \mathit{fulfil}(sn, p_{n+1} \circ \ldots \circ p_{j+1}) \tag{5.14}$$

where $s \approx_{AC1} p_j \circ \ldots \circ p_1 \circ sn$, and $1 \leq j \leq n+1$. The second is with the head of a new variant

$$\begin{aligned} \mathit{fulfil}(\mathit{real}(R,V) \circ S, \mathit{real}(R,Vl) \circ P) \leftarrow \quad & \mathit{val}(V,Vv) \wedge \mathit{val}(Vl,Vvl) \\ & \wedge Vv \geq Vvl \wedge \mathit{fulfil}(S,P) \end{aligned}$$

of (5.3) with AC1-unifier

$$\{R \mapsto r, V \mapsto v, S \mapsto sn, Vl \mapsto vl, P \mapsto p_n \circ \ldots \circ p_1\}$$

yielding

$$\leftarrow \mathit{val}(v,Vv) \wedge \mathit{val}(vl,Vvl) \wedge Vv \geq Vvl \wedge \mathit{fulfil}(sn, p_n \circ \ldots \circ p_1) \tag{5.15}$$

where $s \approx_{AC1} \mathit{real}(r,v) \circ sn$, and $p_{n+1} \approx_{AC1} \mathit{real}(r,vl)$. At this state, (5.15) either fails or yields a successful derivation

$$\leftarrow \mathit{val}(vl,Vvl) \wedge vv \geq Vvl \wedge \mathit{fulfil}(sn, p_n \circ \ldots \circ p_1) \tag{5.16}$$

with the fact

$$\mathit{val}(v,vv)$$

and AC1-unifier

$$\{Vv \mapsto vv\}.$$

At this state, (5.16) either fails or there is an SLDENF-derivation between the first literal of (5.16) and the fact

$$\mathit{val}(vl,vvl)$$

in $\mathcal{E}_{real}$ yielding

$$\leftarrow vv \geq vvl \wedge \mathit{fulfil}(sn, p_n \circ \ldots \circ p_1) \tag{5.17}$$

with AC1-unifier

$$\{Vvl \mapsto vvl\}.$$

Again, at this point, (5.17) either fails or the first literal of (5.17) can be AC1-unified with the fact

$$vv \geq vvl$$

and AC1-unifier $\emptyset$ leading to

$$\leftarrow \mathit{fulfil}(sn, p_{n+1} \circ \ldots \circ p_1) \tag{5.18}$$

Both (5.14) and (5.18) have the length of the second argument less or equal to n. We can apply I.H and learn that no SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(sn, p_n \circ \ldots \circ p_1)$ is infinite. Combining this result and the derivations of $\leftarrow \mathit{fulfil}(s, p_{n+1} \circ \ldots \circ p_1)$, we can safely conclude that no SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(s, p_{n+1} \circ \ldots \circ p_1)$ is infinite. $\square$

Clauses (5.5), (5.6) and (5.7) will not lead to derivations which flounder or are infinite if the second argument is ground and finite and the first argument is ground. This is shown by the following proposition.

Proposition 5.9. *Let $s, e_1, \ldots, e_n$ be ground fluent terms, and ns be an arbitrary fluent term. Then, no SLDENF-resolution proof of $\leftarrow \mathit{replace}(s, e_1 \circ \ldots \circ e_n, ns)$ flounders or is infinite.*

Proof. The proposition is proven by induction on the length of the second argument in *replace*.

I.B To show: No SLDENF-resolution proof of $\leftarrow \mathit{replace}(s, 1, ns)$ flounders or is infinite. In the case where $n = 0$, what is left in the second argument is 1 since $e_n \circ \ldots \circ e_1 \circ 1 \approx e_n \circ \ldots \circ e_1$. $\mathit{replace}(s, 1, ns)$ can only be AC1-unified with a new variant of (5.5) and only if $s \approx ns$. Hence, we obtain the empty clause or the derivation fails immediately.

I.H No SLDENF-resolution proof of $\leftarrow \mathit{replace}(sn, e_n \circ \ldots \circ e_1, nsn)$ flounders or is infinite.

I.C No SLDENF-resolution proof of $\leftarrow \mathit{replace}(s, e_{n+1} \circ \ldots \circ e_1, ns)$ flounders or is infinite.

I.S In the case $n + 1 > 0$, $\mathit{replace}(s, e_{n+1} \circ \ldots \circ e_1, ns)$ can only be AC1-unified with either the head of a new variant

$$\begin{aligned} \mathit{replace}(\mathit{real}(R, V) \circ S, \mathit{real}(R, Vl) \circ E, \mathit{real}(R, Vvl) \circ N) \leftarrow \quad & \mathit{val}(Vl, Vvl) \\ & \wedge\ \mathit{replace}(S, E, N) \end{aligned}$$

of (5.7) or the head of a new variant

$$\begin{aligned} \mathit{replace}(S, \mathit{real}(R, V) \circ E, \mathit{real}(R, Vl) \circ N) \leftarrow \quad & \neg \mathit{member}(\mathit{real}(R, V), S) \wedge \mathit{val}(V, Vl) \\ & \wedge\ \mathit{replace}(S, E, N) \end{aligned}$$

of (5.6). If it is AC1-unified with the former then the successful derivation yields

$$\leftarrow \mathit{val}(vl, vvl) \wedge \mathit{replace}(sn, e_n \circ \ldots \circ e_1, nsn) \tag{5.19}$$

with AC1-unifier

$$\{R \mapsto r, V \mapsto v, S \mapsto sn, Vl \mapsto vl, Vvl \mapsto vvl, E \mapsto e_n \circ \ldots \circ e_1, N \mapsto nsn\}$$

where $s \approx_{AC1} real(r, v) \circ sn,\ e_{n+1} \approx_{AC1} real(r, vl),$ and $ns \approx_{AC1} real(r, vvl) \circ nsn.$ At this point, (5.19) either fails or there is an SLDENF-derivation between the first literal of (5.19) and the fact

$$val(vl, vvl)$$

in $\mathcal{E}_{real}$ with AC1-unifier $\emptyset$ yielding

$$\leftarrow replace(sn, e_n \circ \ldots \circ e_1, nsn) \tag{5.20}$$

If $replace(s, e_{n+1} \circ \ldots \circ e_1, ns)$ is AC1-unified with the latter then the successful derivation yields

$$\leftarrow \neg member(real(r, v), s)\ \wedge\ val(v, vl)\ \wedge\ replace(s, e_n \circ \ldots \circ e_1, nsn) \tag{5.21}$$

with AC1-unifier

$$\{S \mapsto s, R \mapsto r, V \mapsto v, Vl \mapsto vl, E \mapsto e_n \circ \ldots \circ e_1, N \mapsto nsn\}$$

where $e_{n+1} \approx_{AC1} real(r, v),$ and $ns \approx_{AC1} real(r, vl) \circ nsn.$ At this point, the derivation of (5.21) may either fail because there is an SLDENF-resolution proof of $member(real(c, r), cs)$ or continue with

$$\leftarrow val(v, vl)\ \wedge\ replace(s, e_n \circ \ldots \circ e_1, nsn) \tag{5.22}$$

Once more, at this point, the derivation of (5.22) may either fail or continue with

$$\leftarrow replace(s, e_n \circ \ldots \circ e_1, nsn) \tag{5.23}$$

As the second argument has the length n for both (5.20) and (5.23), we can apply I.H to (5.20) and (5.23) to conclude that both (5.20) and (5.23) has no infinite SLDENF-resolution proof and do not flounder. Combining this result with the derivation of $\leftarrow replace(s, e_{n+1} \circ \ldots \circ e_1, ns)$, we can conclude that $\leftarrow replace(s, e_{n+1} \circ \ldots \circ e_1, ns)$ has no either infinite or floundering SLDENF-resolution proof. □

From the above proposition, we can also conclude that if the first and the second argument of *replace* are ground fluent terms then so is the third argument. In the following, we will show that clauses involving *hinder* do not have infinite derivations.

Proposition 5.10. *Let s be a ground fluent term and a be a simple fluent term. Then, all SLDENF-resolution proofs of $\leftarrow hinder(s, a)$ are finite.*

Proof. Assume that $s \approx_{AC1} o \circ s'$, then the atom occurring

$$\leftarrow hinder(s, a)$$

can only be AC1-unifiable with either the head of a new variant

$$hinder(O \circ S, A) \leftarrow inhib(O, A)$$

of (4.1) where it yields to

$$\leftarrow inhib(o, a) \tag{5.24}$$

or the head of a new variant

$$\begin{aligned} hinder(real(R, V) \circ S, A) \leftarrow \quad & inhib(real(R, Vl) \circ O, A) \wedge val(V, Vv) \\ & \wedge\ val(Vl, Vvl) \wedge Vv \geq Vvl \end{aligned}$$

of (5.11) where it yields to

$$\leftarrow inhib(real(r, Vl) \circ O, a) \wedge val(v, Vv) \wedge val(Vl, Vvl) \wedge Vv \geq Vvl. \tag{5.25}$$

If $\leftarrow hinder(s, a)$ is AC1-unified with the head of the variant of (4.1) then the AC1-unifier is of the form

$$\sigma = \{O \mapsto o, S \mapsto s', A \mapsto a\}.$$

At this point, the derivation (5.24) either fails or the literal $inhib(o, a)$ can be solved with the fact

$$inhib(o', a)$$

in $\mathcal{K}_A$ to continue to the empty clause where $o \approx_{AC1} o'$.

If $\leftarrow hinder(s, a)$ is AC1-unified with the head of the variant of (5.11) then the AC1-unifier is of the form

$$\sigma = \{R \mapsto r, V \mapsto v, S \mapsto s', A \mapsto a\}$$

where $o \approx_{AC1} real(r, v)$. At this point, the derivation (5.25) either fails or the literal $inhib(real(r, Vl) \circ O, a)$ can be solved with the fact

$$inhib(real(r, vl) \circ o', a)$$

in $\mathcal{K}_A$ to continue to

$$\leftarrow val(v, Vv) \wedge val(vl, Vvl) \wedge Vv \geq Vvl \tag{5.26}$$

with AC1-unifier

$$\{Vl \mapsto vl, O \mapsto o'\}.$$

At this point, (5.26) either fails or there is an SLDENF-derivation between the first literal of (5.26) and the fact

$$val(v, vv)$$

in $\mathcal{E}_{real}$ yielding

$$\leftarrow val(vl, Vvl) \wedge vv \geq Vvl \tag{5.27}$$

with AC1-unifier

$$\{Vv \mapsto vv\}.$$

(5.27) either fails or there is an SLDENF-derivation between the first literal of (5.27) and the fact

$$val(vl, vvl)$$

in $\mathcal{E}_{real}$ yielding

$$\leftarrow vv \geq vvl \tag{5.28}$$

with AC1-unifier

$$\{Vvl \mapsto vvl\}.$$

Once more, at this point, the derivation (5.27) either fails or the literal $v \geq vl$ can be solved with the fact $v \geq vl$ in $\mathcal{E}_{real}$ to continue to the empty clause. □

Clause (5.8) does not cause floundering if the argument is ground. This is shown in the following proposition.

Proposition 5.11. *Let s be a ground fluent term. Then, no SLDENF-resolution proof of $\leftarrow nonclosed(s)$ flounders.*

Proof. The atom occurring in

$$\leftarrow nonclosed(s)$$

can only be AC1-unified with the head of a new variant

$$\begin{aligned} nonclosed(C \circ S) \leftarrow \quad & taction(C, A, E) \wedge precon(P, A) \\ & \wedge fulfil(C \circ S, P) \wedge \neg hinder(C \circ S, A) \\ & \wedge replace(C \circ S, E, N) \wedge C \circ S \not\approx N \end{aligned}$$

of (5.8) with AC1-unifier

$$\{C \mapsto c, S \mapsto s'\}$$

where $s \approx_{AC1} c \circ s'$. The derivation yields

$$\begin{aligned} \leftarrow \quad & \mathit{taction}(c, A, E) \wedge \mathit{precon}(P, A) \wedge \mathit{fulfil}(c \circ s', P) \wedge \neg \mathit{hinder}(c \circ s', A) \\ & \wedge\ \mathit{replace}(c \circ s', E, N) \wedge c \circ x \not\approx N \end{aligned} \tag{5.29}$$

At this point, the derivation (5.29) either fails or the literal $\mathit{taction}(c, A, E)$ can be solved with the fact

$$\mathit{taction}(c, a, e)$$

in $\mathcal{K}_A$ leading to

$$\begin{aligned} \leftarrow \quad & \mathit{precon}(P, a) \wedge \mathit{fulfil}(c \circ s', P) \wedge \neg \mathit{hinder}(c \circ s', a) \\ & \wedge\ \mathit{replace}(c \circ s', e, N) \wedge c \circ x \not\approx N \end{aligned} \tag{5.30}$$

with AC1-unifier

$$\{A \mapsto a, E \mapsto e\}.$$

Derivation (5.30) either fails or the literal $\mathit{precon}(P, a)$ can be solved with the fact

$$\mathit{precon}(p, a)$$

in $\mathcal{K}_A$ leading to

$$\begin{aligned} \leftarrow \quad & \mathit{fulfil}(c \circ s', p) \wedge \neg \mathit{hinder}(c \circ s', a) \\ & \wedge\ \mathit{replace}(c \circ s', e, N) \wedge c \circ x \not\approx N \end{aligned} \tag{5.31}$$

with AC1-unifier

$$\{P \mapsto p\}.$$

Proposition 5.8 ensures that SLDENF-resolution proofs of $\mathit{fulfil}(c \circ s', p)$ are finite. Hence, derivation (5.31) either fails because there is no SLDENF-resolution proof of $\mathit{fulfil}(c \circ s', p)$ or there is a finite SLDENF-resolution proof of $\mathit{fulfil}(c \circ s', p)$ an answer substitution $\emptyset$ leading to

$$\leftarrow \neg \mathit{hinder}(c \circ s', a) \wedge \mathit{replace}(c \circ s', e, N) \wedge c \circ s' \not\approx N \tag{5.32}$$

Proposition 5.10 ensures that SLDENF-resolution proofs of $\mathit{hinder}(c \circ s', a)$ are finite. Since all arguments of *hinder* are ground, derivation (5.32) either fails because there is an SLDENF-resolution proof of $\mathit{hinder}(c \circ s', a)$ or there is no finite SLDENF-resolution proof of $\mathit{hinder}(c \circ s', a)$. If the latter is the case, then there is an SLDENF-derivation from (5.32) to

$$\leftarrow \mathit{replace}(c \circ s', e, N) \wedge c \circ s' \not\approx N \tag{5.33}$$

Proposition 5.9 ensures that SLDENF-resolution proofs of $\mathit{replace}(c \circ s', e, N)$ are finite. Furthermore, the restriction stating that all variables in e must appear in c forces e to be ground because $c \circ s'$ is ground. Hence, derivation (5.33) either fails because there is no SLDENF-resolution proof of $\mathit{replace}(c \circ s', e, N)$ or there is a finite SLDENF-resolution proof of $\mathit{replace}(c \circ s', e, N)$ with an answer substitution $\{N \mapsto n\}$. If the latter is the case then there are SLDENF-derivations from (5.33) to

$$\leftarrow c \circ s' \not\approx n \tag{5.34}$$

where variable N is replaced by a ground term n in the process. At this point, the derivation either fails because there is a successful SLDENF-resolution proof of $c \circ s' \not\approx n$ or continues because there is no successful SLDENF-resolution proof of $c \circ s' \not\approx n$. If the latter is the case then there is a SLDENF-derivation from (5.34) to the empty clause. □

For clauses (5.9) and (5.10), we have to impose that the second argument of *reapplicable* is finite. Then we can be sure that the SLDENF-derivations are finite. On the other hand, floundering in SLDENF-derivations of *reapplicable* can be avoided as long as the first argument of *reapplicable* is a ground term.

Proposition 5.12. *Let s be a ground term, and ns be an arbitrary fluent term. No SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(s, [a_n, \ldots, a_1], ns)$ flounders or is infinite.*

Proof. The proposition is proven by induction on the length of the list of the second argument in *reapplicable*.
I.B To show: No SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(s, [\,], ns)$ flounders or is infinite.
In case where $n = 0$, i.e., the list is empty, $\mathit{reapplicable}(s, [\,], ns)$ can only be AC1-unified with a new variant

$$\mathit{reapplicable}(S, [\,], S) \leftarrow \neg \mathit{nonclosed}(S)$$

of (5.9) and only if $s \approx_{AC1} ns$. If $s \approx_{AC1} ns$ then the derivation yields

$$\leftarrow \neg \mathit{nonclosed}(s) \tag{5.35}$$

Since s is a ground term and Proposition 5.11 ensures that all derivations $\mathit{nonclosed}(s)$ do not flounder, then the derivation of (5.35) is either fail or continue to the empty clause.

I.H No SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(sn, [a_n, \ldots, a_1], ns)$ flounders or is infinite.
I.C To show: No SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(s, [a_{n+1}, \ldots, a_1], ns)$ flounders or is infinite.

I.S In the case $n+1 > 0$, $\mathit{reapplicable}(s, [a_{n+1}, \ldots, a_1], ns)$ can only be AC1-unified with the head of a new variant

$$\begin{aligned}\mathit{reapplicable}(C \circ S, [A \mid T], N) \leftarrow\ & \mathit{taction}(C, A, E) \wedge \mathit{precon}(P, A) \wedge \mathit{fulfil}(C \circ S, P) \\ & \wedge \neg \mathit{hinder}(C \circ S, A) \wedge \mathit{replace}(C \circ S, E, Z) \\ & \wedge\ C \circ S \not\approx Z \wedge \mathit{reapplicable}(Z, T, N)\end{aligned}$$

of (5.10) with AC1-unifier

$$\{C \mapsto c, S \mapsto s', A \mapsto a_{n+1}, T \mapsto [a_n, \ldots, a_1], N \mapsto ns\}$$

where $s \approx_{AC1} c \circ s'$. The derivation yields

$$\begin{aligned}\leftarrow\ & \mathit{taction}(c, a_{n+1}, E) \wedge \mathit{precon}(P, a_{n+1}) \wedge \mathit{fulfil}(c \circ s', P) \wedge \neg \mathit{hinder}(c \circ s', a_{n+1}) \\ & \wedge\ \mathit{replace}(c \circ s', E, Z) \wedge c \circ s' \not\approx Z \wedge \mathit{reapplicable}(Z, [a_n, \ldots, a_1], ns)\end{aligned} \tag{5.36}$$

At this point, the derivation (5.36) either fails or the literal $\mathit{taction}(c, a_{n+1}, E)$ can be solved with the fact

$$\mathit{taction}(c, a_{n+1}, e)$$

in $\mathcal{K}_A$ leading to

$$\begin{aligned}\leftarrow\ & \mathit{precon}(P, a_{n+1}) \wedge \mathit{fulfil}(c \circ s', P) \wedge \neg \mathit{hinder}(c \circ s', a_{n+1}) \\ & \wedge\ \mathit{replace}(c \circ s', e, Z) \wedge c \circ s' \not\approx Z \wedge \mathit{reapplicable}(Z, [a_n, \ldots, a_1], ns)\end{aligned} \tag{5.37}$$

with AC1-unifier

$$\{E \mapsto e\}.$$

Derivation (5.37) either fails or the literal $\mathit{precon}(P, a_{n+1})$ can be solved with the fact

$$\mathit{precon}(p, a_{n+1})$$

in $\mathcal{K}_A$ leading to

$$\begin{aligned}\leftarrow\ & \mathit{fulfil}(c \circ s', p) \wedge \neg \mathit{hinder}(c \circ s', a_{n+1}) \wedge \mathit{replace}(c \circ s', e, Z) \\ & \wedge\ c \circ s' \not\approx Z \wedge \mathit{reapplicable}(Z, [a_n, \ldots, a_1], ns)\end{aligned} \tag{5.38}$$

with AC1-unifier

$$\{P \mapsto p\}.$$

Proposition 5.8 ensures that SLDENF-resolution proofs of $\mathit{fulfil}(c \circ s', p)$ are finite. Hence, derivation (5.38) either fails because there is no SLDENF-resolution proof of $\mathit{fulfil}(c \circ s', p)$

or there is a finite SLDENF-resolution proof of $\mathit{fulfil}(c \circ s', p)$ leading to

$$\begin{aligned} \leftarrow \quad & \neg \mathit{hinder}(c \circ s', a_{n+1}) \wedge \mathit{replace}(c \circ s', e, Z) \\ & \wedge\ c \circ s' \not\approx Z \wedge \mathit{reapplicable}(Z, [a_n, \ldots, a_1], ns) \end{aligned} \tag{5.39}$$

Proposition 5.10 ensures that SLDENF-resolution proofs of $\mathit{hinder}(c \circ s', a_{n+1})$ are finite. Since all arguments of *hinder* are ground, derivation (5.39) either fails because there is an SLDENF-resolution proof of $\mathit{hinder}(c \circ s', a_{n+1})$ or there is no finite SLDENF-resolution proof of $\mathit{hinder}(c \circ s', a_{n+1})$. If the latter is the case, then there is an SLDENF-derivation from (5.39) to

$$\leftarrow \mathit{replace}(c \circ s', e, Z) \wedge c \circ s' \not\approx Z \wedge \mathit{reapplicable}(Z, [a_n, \ldots, a_1], ns) \tag{5.40}$$

Proposition 5.9 ensures that SLDENF-resolution proofs of $\mathit{replace}(c \circ s', e, Z)$ are finite. Furthermore, the restriction stating that all variables in e must appear in c forces e to be ground because $c \circ s'$ is ground. Hence, derivation (5.40) either fails because there is no SLDENF-resolution proof of $\mathit{replace}(c \circ s', e, Z)$ or there is a finite SLDENF-resolution proof of $\mathit{replace}(c \circ s', e, Z)$ with an answer substitution $\{Z \mapsto sn\}$. If the latter is the case then there are SLDENF-derivations from (5.40) to

$$\leftarrow c \circ s' \not\approx sn \wedge \mathit{reapplicable}(sn, [a_n, \ldots, a_1], ns) \tag{5.41}$$

where variable Z is replaced with a ground term sn in the process. $c \circ s'$ is known as a ground term, whereas sn is a ground term due to the previous derivation involving *replace*. Hence, at this point, the derivation either fails because there is a successful SLDENF-resolution proof of $c \circ s' \not\approx sn$ or continues because there is no successful SLDENF-resolution proof of $c \circ s' \not\approx sn$. If the latter is the case then there is a SLDENF-derivation from (5.41) to

$$\leftarrow \mathit{reapplicable}(sn, [a_n, \ldots, a_1], ns) \tag{5.42}$$

We can apply the induction hypothesis and learn that no SLDENF-resolution proof of (5.42) flounders or is infinite. Combining the derivations from $\leftarrow \mathit{reapplicable}(s, [a_{n+1}, \ldots, a_1], ns)$ to (5.42) with the result from induction hypothesis, we learn that no SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(s, [a_{n+1}, \ldots, a_1], ns)$ flounders or is infinite. □

The following proposition shows that clause (5.12) does not cause infinite SLDENF-derivations as long as the second argument has finite length. Furthermore, it does not cause floundering if the first argument is a ground fluent term.

Proposition 5.13. *Let s, a be ground fluent terms and ns be an arbitrary fluent term. No SLDENF-resolution proof of $\leftarrow \mathit{applicable}(s, a, ns)$ flounders or is infinite.*

Proof. The atom occurring in $applicable(s, a, ns)$ can only be AC1-unified with the head of a new variant

$$\begin{aligned} applicable(C \circ S, A, E \circ S) \leftarrow \quad & action(C, A, E) \wedge precon(P, A) \\ & \wedge fulfil(C \circ S, P) \wedge \neg hinder(C \circ S, A) \end{aligned}$$

of (5.12) with AC1-unifier

$$\{C \mapsto c, S \mapsto s', A \mapsto a, E \mapsto e\}$$

where $s \approx_{AC1} c \circ s'$ and $ns \approx_{AC1} e \circ s'$. The derivation yields

$$\leftarrow action(c, a, e) \wedge precon(P, a) \wedge fulfil(c \circ s', P) \wedge \neg hinder(c \circ s', a) \tag{5.43}$$

At this point, the derivation (5.43) either fails or the literal $action(c, a, e)$ can be solved with a fact in $\mathcal{K}_A$ and AC1-unifier $\emptyset$ leading to

$$\leftarrow precon(P, a) \wedge fulfil(c \circ s', P) \wedge \neg hinder(c \circ s', a) \tag{5.44}$$

Derivation (5.44) either fails or the literal $precon(P, a)$ can be solved with a fact

$$precon(p, a)$$

in $\mathcal{K}_A$ leading to

$$\leftarrow fulfil(c \circ s', P) \wedge \neg hinder(c \circ s', a) \tag{5.45}$$

with AC1-unifier

$$\{P \mapsto p\}.$$

Proposition 5.8 ensures that SLDENF-resolution proofs of $fulfil(c\circ s', p)$ are finite. Hence, derivation (5.45) either fails because there is no SLDENF-resolution proof of $fulfil(c\circ s', p)$ or there is a finite SLDENF-resolution proof of $fulfil(c \circ s', p)$ leading to

$$\leftarrow \neg hinder(c \circ s', a) \tag{5.46}$$

Proposition 5.10 ensures that SLDENF-resolution proofs of $hinder(c \circ s', a)$ are finite. Since all arguments of *hinder* are ground, derivation (5.46) either fails because there is an SLDENF-resolution proof of $hinder(c \circ s', a)$ or there is no finite SLDENF-resolution proof of $hinder(c \circ s', a)$. If the latter is the case, then there is an SLDENF-derivation from (5.46) to the empty clause. Therefore, combining all derivations starting from $\leftarrow applicable(s, a, ns)$ to (5.46), we conclude that no SLDENF-resolution proof of $\leftarrow applicable(s, a, ns)$ flounders or is infinite. □

The modification of advanced fluent calculus involving real values changes several clauses such as clauses with *applicable* and *causes* in their head. Hence, Proposition 4.3 is no longer valid for advanced fluent calculus with real values. Hence, we have to reprove Proposition 4.3. Let $\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ be an acpp$^{\mathbb{R}}$ and $\mathcal{I}_{fc}, \mathcal{G}_{fc}$ are the representation of $\mathcal{I}, \mathcal{G}$ in fluent calculus where real-valued fluents are represented in the form of $real(R, V)$.

Lemma 5.14. *No SLDENF-resolution proof of* $\leftarrow causes(\mathcal{I}_{fc}^{-I}, [a_1, \dots, a_n], \mathcal{G}_{fc}^{-I})$ *flounders or is infinite.*

Proof. The lemma is proven by induction on the length of the list of the second argument in *causes*.
I.B To show: No SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}_{fc}^{-I}, [\,], \mathcal{G}_{fc}^{-I})$ flounders or is infinite.
In case where $n = 0$, i.e., the list is empty, $causes(\mathcal{I}_{fc}^{-I}, [\,], \mathcal{G}_{fc}^{-I})$ can only be AC1-unified with a new variant

$$causes(S, [\,], S)$$

of (3.2) and only if $\mathcal{I}_{fc}^{-I} \approx_{AC1} \mathcal{G}_{fc}^{-I}$. Hence, we obtain the empty clause or the derivation fails immediately.

I.H No SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}_{fcn}^{-I}, [a_n, \dots, a_1], \mathcal{G}_{fc}^{-I})$ flounders or is infinite.
I.C To show: No SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}_{fc}^{-I}, [a_{n+1}, \dots, a_1], \mathcal{G}_{fc}^{-I})$ flounders or is infinite.
I.S In the case $n+1 > 0$, $causes(\mathcal{I}_{fc}^{-I}, [a_{n+1}, \dots, a_1], \mathcal{G}_{fc}^{-I})$ can only be AC1-unified with the head of a new variant

$$\begin{aligned} causes(I, [A \mid L], G) \leftarrow\ & applicable(I, A, S) \wedge append(T, P, L) \\ & \wedge\ reapplicable(S, T, N) \wedge causes(N, P, G) \end{aligned}$$

of (5.13) where the successful derivation yields

$$\begin{aligned} \leftarrow\ & applicable(\mathcal{I}_{fc}^{-I}, a_{n+1}, S) \wedge append(T, P, [a_n, \dots, a_1]) \\ & \wedge\ reapplicable(S, T, N) \wedge causes(N, P, \mathcal{G}_{fc}^{-I}) \end{aligned} \tag{5.47}$$

with AC1-unifier

$$\{I \mapsto \mathcal{I}_{fc}^{-I}, A \mapsto a_{n+1}, L \mapsto [a_n, \dots, a_1], G \mapsto \mathcal{G}_{fc}^{-I}\}.$$

Proposition 5.13 ensures that SLDENF-resolution proofs of $\leftarrow applicable(\mathcal{I}_{fc}^{-I}, a_{n+1}, S)$ are finite and do not cause floundering. It means SLDENF-derivations of (5.47) either fail or continue. If they continue, we can find the corresponding SLDENF-derivations from (5.47) to

$$\leftarrow \mathit{append}(T, P, [a_n, \dots, a_1]) \wedge \mathit{reapplicable}(s, T, N) \wedge \mathit{causes}(N, P, \mathcal{G}_{fc}^{-I}) \tag{5.48}$$

where variable S is replaced by an arbitrary fluent term s. Proposition 2.2 ensures that SLDENF-resolution proofs of $\leftarrow \mathit{append}(T, P, [a_n, \dots, a_1])$ are finite. It means SLDENF-derivations of (5.48) either fail or continue. If they continue, we can find the corresponding SLDENF-derivations from (5.48) to

$$\leftarrow \mathit{reapplicable}(s, [a_n, \dots, a_{j+1}], N) \wedge \mathit{causes}(N, [a_j, \dots, a_1], \mathcal{G}_{fc}^{-I}) \tag{5.49}$$

where along the derivation process, variables T and P are replaced by $[a_n, \dots, a_{j+1}]$ and $[a_j, \dots, a_1]$, respectively. Proposition 5.12 ensures that SLDENF-resolution proofs of $\leftarrow \mathit{reapplicable}(s, [a_n, \dots, a_{j+1}], N)$ are finite and do not flounder. It means SLDENF-derivations of (5.49) either fail or continue. If they continue, we can find the corresponding SLDENF-derivations from (5.49) to

$$\leftarrow \mathit{causes}(ns, [a_j, \dots, a_1], \mathcal{G}_{fc}^{-I}) \tag{5.50}$$

where variable N is replaced by an arbitrary fluent term ns. We can apply the induction hypothesis to (5.50) by assuming that $\mathcal{I}_{fcn}^{-I} \approx_{AC1} ns$ because the list in the second argument is shorter than the original one. We learn that there is no SLDENF-resolution proof of (5.50) flounders or is infinite. Combining all derivations from $\leftarrow \mathit{causes}(\mathcal{I}_{fc}^{-I}, [a_{n+1}, \dots, a_1], \mathcal{G}_{fc}^{-I})$ to (5.50) with the result of induction hypothesis, we can conclude that there is no SLDENF-resolution proof of $\leftarrow \mathit{causes}(\mathcal{I}_{fc}^{-I}, [a_{n+1}, \dots, a_1], \mathcal{G}_{fc}^{-I})$ flounders or is infinite. □

Proposition 5.14 shows that the length of the second argument of *causes* must be fixed in advanced to prevent from having possible infinite derivations. As the consequence, the question of whether there is a plan with length n solving an acpp$^{\mathbb{R}}$ with an initial state $\mathcal{I}$ and a goal state $\mathcal{G}$ is a question of whether

$$(\exists A_1, \dots, A_n)\mathit{causes}(\mathcal{I}_{fc}\downarrow^{-I}, [A_1, \dots, A_n], \mathcal{G}_{fc}\downarrow^{-I})$$

is a logical consequence of $\mathcal{AFC}\mathbb{R}^*_{\mathcal{Q}}$. We may change n to a greater number in the search for a solution of the planning problem.

One should notice that we only count the number of SLDENF-refutation steps involving subgoals *causes* and *reapplicable*. Therefore, the length of an SLDENF-resolution is the number of SLDENF-refutation steps involving these two subgoals.

5.2 An Instance of ACPP$^{\mathbb{R}}$

Once more, we modify the ill man problem as follows:

Suppose there was a man who had weight 1.0 quintal living in an apartment. He lived in the highest floor and there was no lift in this apartment. He was severely ill and needed to see a doctor. He could not go by himself; a helicopter was sent to help him. The helicopter could carry 2 quintal at maximum. The pilot who had weight 0.8 quintal helped him to the helicopter. Once the man was in the helicopter, the pilot brought him to hospital.

The problem above is considered as an advanced conjunctive planning problem with real values. The simple fluents are the apartment (apt), the helicopter (hel), and the hospital (hos), whereas the real-valued fluents are the ill man with weight 1.0 quintal ($ill(1.0)$), the pilot with weight 0.8 quintal ($pil(0.8)$), and the total weight ($wei(X)$). The concrete actions are *to help him to the helicopter* (h), and *to drive to the hospital* (d), whereas the theory action is *to calculate the weight of both the pilot and the ill man* (c). We may conclude that the only solution is by helping him to the helicopter, calculating the weight, and driving the helicopter to the hospital. Within the advanced conjunctive planning problem with real values we obtain

$$\begin{aligned}\mathcal{Q} = \langle \quad & \dot{\{}ill(1.0), pil(0.8), apt\dot{\}}, \dot{\{}ill(1.0), pil(0.8), wei(1.8), hos\dot{\}},\\ & \{h : \dot{\{}apt\dot{\}} \xRightarrow{\dot{\emptyset},\dot{\emptyset}} \dot{\{}hel\dot{\}}, d : \dot{\{}hel\dot{\}} \xRightarrow{\dot{\emptyset},\dot{\{}wei(2.0)\dot{\}}} \dot{\{}hos\dot{\}}\},\\ & \{c : \dot{\{}ill(X), pil(Y)\dot{\}} \xRightarrow{\dot{\{}hel\dot{\}},\dot{\emptyset}} \dot{\{}wei(X+Y)\dot{\}}\} \qquad \rangle\end{aligned}$$

We could start solving the problem by executing h, c, and d consecutively to the initial state, i.e,

$$\begin{array}{llll}\dot{\{}ill(1.0), pil(0.8), apt\dot{\}} & \xrightarrow{h} & \dot{\{}ill(1.0), pil(0.8), hel\dot{\}} & \xrightarrow{c}\\ \dot{\{}ill(1.0), pil(0.8), wei(1.8), hel\dot{\}} & \xrightarrow{d} & \dot{\{}ill(1.0), pil(0.8), wei(1.8), hos\dot{\}} & \end{array}$$

The obstacle $wei(2.0)$ in the concrete action d does not hinder the action from being executed because the threshold 2.0 for the real-valued fluent wei in the obstacles is greater than the value 1.8 possessed by the real-valued fluent wei in state $\dot{\{}ill(1.0), pil(0.8), wei(1.8), hel\dot{\}}$.

In the fluent calculus, the actions are represented by the set of clauses

$$\begin{aligned}\{ \quad & action(apt, h, hel), action(hel, d, hos),\\ & taction(real(ill, X) \circ real(pil, Y), c, real(wei, X+Y)) \quad \}\end{aligned}$$

$$\leftarrow causes(\dot{\{}real(ill,1.0), real(pil,0.8), apt\dot{\}}^{-I}, P, \dot{\{}real(ill,1.0), real(pil,0.8), real(wei,1.8), hos\dot{\}}^{-I}) \qquad causes(I,[A \mid L],G)$$

$$\sigma_1 = \{I \mapsto real(ill,1.0) \circ \ldots \circ apt, P \mapsto [A \mid L], G \mapsto real(ill,1.0) \circ \ldots \circ hos\}$$

$$\leftarrow applicable(real(ill,1.0) \circ \ldots \circ apt, A, S) \wedge \ldots \wedge causes(N, P_1, real(ill,1.0) \circ \ldots \circ hos)$$

$$\sigma_2 = \{A \mapsto h, S \mapsto real(ill,1.0) \circ real(pil,0.8) \circ hel\}$$

$$\leftarrow append(T, P_1, L) \wedge \ldots \wedge causes(N, P_1, real(ill,1.0) \circ \ldots \circ hos)$$

$$\sigma_3 = \{T \mapsto [T_1], P_1 \mapsto [P_2], L \mapsto [T_1, P2]\}$$

$$\leftarrow reapplicable(real(ill,1.0) \circ \ldots \circ hel, [T_1], N) \wedge causes(N, [P_2], real(ill,1.0) \circ \ldots \circ hos)$$

$$\sigma_4 = \{T_1 \mapsto c, N \mapsto real(ill,1.0) \circ real(pil,0.8) \circ real(wei,1.8) \circ hel\}$$

$$\leftarrow causes(real(ill,1.0) \circ \ldots \circ hel, [P_2], real(ill,1.0) \circ \ldots \circ hos)$$

$$\sigma_5 = \{P_2 \mapsto d\}$$

$$\Box$$

FIGURE 5.1: An SLDENF-resolution proof of $causes(\dot{\{}real(ill,1.0) \circ real(pil,0.8) \circ apt\dot{\}}^{-I}, P, \dot{\{}real(ill,1.0) \circ real(pil,0.8) \circ real(wei,1.8) \circ hos\dot{\}}^{-I})$.

inhibitors of action d are represented by the set of clauses

$$\{\ inhib(real(wei,2.0), d)\ \}$$

and the preconditions of action c are represented by the set of clauses

$$\{\ precon(hel, c)\ \}$$

If we ask, whether there exists a plan P such that its execution transforms state $\dot{\{}ill(1.0), pil(0.8), apt\dot{\}}$ into state $\dot{\{}ill(1.0), pil(0.8), wei(1.8), hos\dot{\}}$, i.e.

$$\begin{aligned}\exists P: \quad & causes(real(ill, 1.0) \circ real(pil, 0.8) \circ apt, P, \\ & real(ill, 1.0) \circ real(pil, 0.8) \circ real(wei, 1.8) \circ hos)\end{aligned}$$

then we obtain the refutation shown in Figure 5.1 yielding the answer substitution

$$\{P \mapsto [h, [c, [d, []]]]\}$$

Hence, the ill man has to be helped to the helicopter (h) first, the calculation of the weight follows after (c), and the pilot drives the helicopter to hospital (d) afterwards, and they have reached the goal ([ ]).

5.3 Sound and Completeness of the Real-valued Fluent Calculus

In the previous section, we modeled the modified ill man problem with advanced conjunctive planning problems with real values. We also showed that this problem can be represented and solved within the real-valued fluent calculus. In this section, we will show that the real-valued fluentcalculus is sound and complete in solving real-valued conjunctive planning problems. We assume that whenever a multiset containing simple and real-valued fluents is transformed into a fluent term, the process also transforms real-valued fluents to their representation in the real-valued fluent calculus and vice versa.

Throughout this section let $\mathcal{I}_s, \mathcal{G}_s$ denote finite multisets of simple fluents, $\mathcal{I}_c, \mathcal{G}_c$ denote finite multisets of ground real-valued fluents, $\mathcal{I} \doteq \mathcal{I}_s \dot\cup \mathcal{I}_c$, $\mathcal{G} \doteq \mathcal{G}_s \dot\cup \mathcal{G}_c$, and
$\mathcal{Q} = \langle \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ denotes a real-valued conjunctive planning problem. Furthermore, let $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$ denotes its presentation in the real-valued fluent calculus.

Proposition 5.15. *There are obstacles at a state $\mathcal{S}$ to hinder an action a in $\mathcal{Q}$ iff there is an SLDENF-resolution proof of $\leftarrow hinder(\mathcal{S}^{-I}, a)$ in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.*

Proof. The action a can be either a concrete action or a theory action. Both types of actions have the same form and have the same rules for obstacles. However, theory actions can only have obstacles containing simple fluents, whereas concrete actions can have obstacles containing both simple and real-valued fluents. Hence, considering a as a concrete action is enough to show that the proposition holds.

$\Rightarrow$ To show: there is an SLDENF-resolution proof of $\leftarrow hinder(\mathcal{S}^{-I}, a)$.
Assume there are obstacles at a state $\mathcal{S}$ to hinder a concrete action a. Assume that the action a is of the form

$$a : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$$

There are two possibilities for obstacles to hinder an action. The first is the obstacles in $\mathcal{O}$ are simple fluents which appear in state $\mathcal{S}$. Let $\mathcal{O}' \dot{\subseteq} \mathcal{S}$ contain only fluents o with multiplicity n where o is the obstacle. Hence, we have $o \in_m \mathcal{S}$, $o \in_n \mathcal{O}$ and $m \geq n$. Hence, there must exist a corresponding fact

$$inhib(\mathcal{O}'^{-I}, a) \tag{5.51}$$

in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$. Let σ

$$\{O \mapsto \mathcal{O}'^{-I}, S \mapsto (\mathcal{S} \dot{\setminus} \mathcal{O}')^{-I}, A \mapsto a\}$$

be a substitution. σ is the AC1-unifier for the atom occurring in

$$\leftarrow hinder(\mathcal{S}^{-I}, a) \tag{5.52}$$

and the head of a new variant

$$hinder(O \circ S, A) \leftarrow inhib(O, A)$$

of (4.1). Therefore, there is an SLDENF-derivation from (5.52) to

$$\leftarrow inhib(\mathcal{O}'^{-I}, a). \tag{5.53}$$

We can do an SLDENF-derivation using (5.51) and the empty substitution to (5.53) to reach the empty clause. Hence, there is an SLDENF-resolution proof of (5.52).

The second possibility is if the obstacles in $\mathcal{O}$ are real-valued fluents which appear in state $\mathcal{S}$. Let $r(v) \in_1 \mathcal{O}$ be the obstacle that hinder the action a in $\mathcal{S}$. It infers that there is $r(v') \in_1 \mathcal{S}$ such that $eval(v') \geq eval(v)$. Furthermore, there must exist the corresponding fact

$$inhib(real(r, v) \circ o, a) \tag{5.54}$$

containing all real-valued fluents appearing in $\mathcal{O}$ in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$ and we can also find facts

$$val(v', vv) \tag{5.55}$$

and

$$val(v, vvl) \tag{5.56}$$

in $\mathcal{E}_{real}$ reflecting the evaluation function on v' and v. Let

$$\sigma' = \{R \mapsto r, V \mapsto v', S \mapsto \mathcal{S}'^{-I}, A \mapsto a\}$$

be a substitution. σ' is the AC1-unifier for the atom occurring in

$$\leftarrow hinder(real(r, v') \circ \mathcal{S}'^{-I}, a) \tag{5.57}$$

and the head of a new variant

$$\begin{aligned} hinder(real(R, V) \circ S, A) \leftarrow \quad & inhib(real(R, Vl) \circ O, A) \wedge val(V, Vv) \\ & \wedge\, val(Vl, Vvl) \wedge Vv \geq Vvl \end{aligned}$$

of (5.11) where $\mathcal{S}^{-I} \approx_{AC1} real(r, v') \circ \mathcal{S}'^{-I}$. Therefore, there is an SLDENF-derivation from (5.57) to

$$\leftarrow inhib(real(r, Vl) \circ O, a) \;\wedge\; val(v', Vv) \;\wedge\; val(Vl, Vvl) \;\wedge\; Vv \geq Vvl \tag{5.58}$$

We can do an SLDENF-derivation from (5.58) using (5.54) and AC1-unifier

$$\{O \mapsto o, Vl \mapsto v\}$$

to obtain

$$\leftarrow val(v', Vv) \;\wedge\; val(v, Vvl) \;\wedge\; Vv \geq Vvl. \tag{5.59}$$

There is an SLDENF-derivation of (5.59) between the first atom occurring in (5.59) and (5.55) yielding

$$\leftarrow val(v, Vvl) \;\wedge\; vv \geq Vvl \tag{5.60}$$

with AC1-unifier

$$\{Vv \mapsto vv\}.$$

There is an SLDENF-derivation of (5.60) between the first atom occurring in (5.60) and (5.56) yielding

$$\leftarrow vv \geq vvl \tag{5.61}$$

with AC1-unifier

$$\{Vvl \mapsto vvl\}.$$

From our assumption we know that $vv \geq vvl$, hence, there must exist the fact $vv \geq vvl$ in $\mathcal{E}_{real}$ which can be AC1-unified with the atom occurring in (5.61) with AC1-unifier $\emptyset$ yielding the empty clause.

$\Leftarrow$ To show: There are obstacles at state $\mathcal{S}$ to hinder a discrete action a in $\mathcal{O}$. Assume there is an SLDENF-resolution proof of a goal clause

$$\leftarrow hinder(\mathcal{S}^{-I}, a). \tag{5.62}$$

It means there are two possible SLDENF-derivations of (5.62). The first possibility is the SLDENF-derivation from (5.62) to

$$\leftarrow inhib(\mathcal{O}'^{-I}, a) \tag{5.63}$$

using a new variant

$$hinder(O \circ S, A) \leftarrow inhib(O, A)$$

of (4.1) and AC1-unifier

$$\{O \mapsto \mathcal{O}'^{-I}, S \mapsto \mathcal{S}'^{-I}, A \mapsto a\}$$

where $\mathcal{S} \doteq \mathcal{O}' \,\dot\cup\, \mathcal{S}'$. We can find the fact

$$inhib(\mathcal{O}'^{-I}, a) \tag{5.64}$$

in $\mathcal{AFC}\mathbb{R}_Q$ to have an SLDENF-derivation from (5.63) to the empty clause using the empty substitution and (5.64).

The second possibility is the SLDENF-derivation from (5.62) to

$$\leftarrow inhib(real(r, Vl) \circ O, a) \ \wedge \ val(v, Vv) \ \wedge \ val(Vl, Vvl) \ \wedge \ Vv \geq Vvl \tag{5.65}$$

using a new variant

$$\begin{aligned} hinder(real(R, V) \circ S, A) \leftarrow \quad & inhib(real(R, Vl) \circ O, A) \wedge val(V, Vv) \\ & \wedge val(Vl, Vvl) \wedge Vv \geq Vvl \end{aligned}$$

of (5.11) and AC1-unifier

$$\{R \mapsto r, V \mapsto v, S \mapsto \mathcal{S}'^{-I}, A \mapsto a\}$$

where $\mathcal{S}^{-I} \approx_{AC1} real(r, v) \circ \mathcal{S}'^{-I}$. We can find the fact

$$inhib(real(r, v') \circ o, a) \tag{5.66}$$

in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$ to have an SLDENF-derivation from (5.65) to

$$\leftarrow val(v, Vv) \ \wedge \ val(v', Vvl) \ \wedge \ Vv \geq Vvl \tag{5.67}$$

with AC1-unifier

$$\{O \mapsto o, Vl \mapsto v'\}.$$

The first atom occurring in (5.67) and the fact

$$val(v, vv)$$

in $\mathcal{E}_{real}$ are AC1-unifiable with AC1-unifier

$$\{Vv \mapsto vv\}$$

yielding

$$\leftarrow val(v', Vvl) \ \wedge \ vv \geq Vvl \tag{5.68}$$

The first atom occurring in (5.68) and the fact

$$val(v', vvl)$$

with AC1-unifier

$$\{Vvl \mapsto vvl\}$$

yielding

$$\leftarrow vv \geq vvl \tag{5.69}$$

Once more, the atom occurring in (5.69) and the fact

$$vv \geq vvl$$

with AC1-unifier $\emptyset$ yielding the empty clause.

From (5.64) and (5.66), we know that there is a concrete action a in $\mathcal{Q}$. Assume that the action a is of the form

$$a : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$$

The derivation from (5.62), (5.63) to the empty clause suggests that $\mathcal{O}' \dot{\subseteq} \mathcal{O}$. The fact (5.64) contains either only simple fluents $o \circ \ldots \circ o$ or all ground real-valued fluents in $\mathcal{O}$. If the former is the case then we obtain $o \in_n \mathcal{O}'$, therefore, $o \in_n \mathcal{O}$. As $\mathcal{O}' \dot{\subseteq} \mathcal{S}$, we may conclude that $o \in_m \mathcal{S}$ where $m \geq n$. Hence, there are obstacles o at $\mathcal{S}$ to hinder the action a.

If the latter is the case then all ground real-valued fluents in $\mathcal{O}$ are in $\mathcal{S}$. It infers that $\forall o(m) \in_1 \mathcal{O}\ \exists o(n) \in_1 \mathcal{S}\ (eval(n) = eval(m))$ is true. Hence, all of these real-valued fluents are obstacles for the concrete action a in state $\mathcal{S}$.

The other derivation from (5.62) to the empty clause suggests that $\forall o(m) \in_1 \mathcal{O}\ \exists o(n) \in_1 \mathcal{S}\ (eval(n) \geq eval(m))$ is true. Hence, this particular real-valued fluent is the obstacle for action a in $\mathcal{S}$. $\square$

The following lemma shows the correlation between state transformations when a theory action is applied in acpp$^{\mathbb{R}}$ and the corresponding SLDENF-resolution proofs in the real-valued fluent calculus. One should recall that all variables occurring in the effects must also occur in the conditions of the real-valued action. Moreover, an initial state as well as all other states only contain ground fluents. As a result, whenever a real-valued fluent is applied to a state, the corresponding effects will only contain ground fluents.

Lemma 5.16. *The a theory action with ground effects $\mathcal{E}$ transforms $\mathcal{S}$ into state $\mathcal{S}'$ iff there is an SLDENF-resolution proof of $\leftarrow replace(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{Q}$.*

Proof. The lemma is proven by Proposition 5.17 and 5.18.

Proposition 5.17. *If a theory action with ground effects $\mathcal{E}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$ then there is an SLDENF-resolution proof of $\leftarrow replace(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{Q}$.*

Proof. The proposition is proven by induction on the number of real-valued fluents in $\mathcal{E}$.

I.B To show: If a theory action with ground effects $\dot{\emptyset}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$ then there is an SLDENF-resolution proof of $\leftarrow replace(\mathcal{S}^{-I}, 1, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{Q}$.
If state $\mathcal{S}$ is transformed into state $\mathcal{S}'$ by a theory action with effects $\dot{\emptyset}$, then it is clear that $\mathcal{S} \doteq \mathcal{S}'$.
Let $\sigma = \{S \mapsto \mathcal{S}^{-I}\}$ be a substitution. σ is the AC1-unifier for the atom occurring in

$$\leftarrow replace(\mathcal{S}^{-I}, 1, \mathcal{S}^{-I}) \tag{5.70}$$

and a new variant

$$replace(S, 1, S)$$

of (5.5). Hence, there is an SLDENF-derivation from (5.70) to the empty clause using σ.

I.H If a theory action with ground effects $\mathcal{E}_n = \dot{\{}e_1, \ldots, e_n\dot{\}}$ transforms state $\mathcal{S}_n$ into state $\mathcal{S}'_n$ then there is an SLDENF-resolution proof of $\leftarrow replace(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I})$ in

$\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.

I.C To show: If a theory action with ground effects $\mathcal{E} = \dot{\{}e_1, \ldots, e_{n+1}\dot{\}}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$ then there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.

I.S Suppose state $\mathcal{S}$ is transformed into state $\mathcal{S}'$ by a theory action with effects $\mathcal{E} = \dot{\{}e_1, \ldots, e_{n+1}\dot{\}}$. The transformation consists of two options for each element in $\mathcal{E}$. Lets assume $e_{n+1} = r(v)$ is the first element either to replace one element in or to be added in $\mathcal{S}$.

The first option is that the real-valued fluent r appears in $\mathcal{S}$. Hence, the transformation is done by replacing the real value v' of r in $\mathcal{S}$ with $\mathit{eval}(v)$. Let

$$\sigma = \{R \mapsto r, V \mapsto v', S \mapsto \mathcal{S}_n^{-I}, Vl \mapsto v, Vvl \mapsto vl, E \mapsto \mathcal{E}_n, N \mapsto \mathcal{S}_n'^{-I}\}$$

be a substitution. σ is the AC1-unifier for the atom occurring in

$$\leftarrow \mathit{replace}(\mathit{real}(r, v') \circ \mathcal{S}_n^{-I}, \mathit{real}(r, v) \circ \mathcal{E}_n^{-I}, \mathit{real}(r, vl) \circ \mathcal{S}_n'^{-I}) \tag{5.71}$$

and the head of a new variant

$$\begin{aligned} \mathit{replace}(\mathit{real}(R, V) \circ S, \mathit{real}(R, Vl) \circ E, \mathit{real}(R, Vvl) \circ N) \leftarrow \quad & \mathit{val}(Vl, Vvl) \\ & \wedge\ \mathit{replace}(S, E, N) \end{aligned}$$

of (5.7). Hence, there is an SLDENF-derivation of (5.71) yielding

$$\leftarrow \mathit{val}(v, vl) \wedge \mathit{replace}(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.72}$$

where $\mathcal{S} \doteq \dot{\{}r(v')\dot{\}} \dot\cup \mathcal{S}_n, \mathcal{E} = \dot{\{}r(v)\dot{\}} \dot\cup \mathcal{E}_n,$ and $\mathcal{S}_n' \doteq \dot{\{}r(vl)\dot{\}} \dot\cup \mathcal{S}_n'$. We can find the fact

$$\mathit{val}(v, vl) \tag{5.73}$$

in $\mathcal{E}_{real}$ because vl is the value of $r(vl)$ in $\mathcal{S}'$ and vl is the result of $\mathit{eval}(v)$. Hence, there is an SLDENF-derivation of (5.72) to

$$\leftarrow \mathit{replace}(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.74}$$

using (5.73) and AC1-unifier $\emptyset$.

The second option is that the real-valued fluent r does not appear in $\mathcal{S}$. Hence, the transformation is done by adding the real-valued fluent $r(\mathit{eval}(v))$ into $\mathcal{S}$. Let

$$\sigma' = \{S \mapsto \mathcal{S}^{-I}, R \mapsto r, V \mapsto v, Vl \mapsto vl, E \mapsto \mathcal{E}_n^{-I}, N \mapsto \mathcal{S}_n'^{-I}\}$$

be a substitution. σ is the AC1-unifier for the atom occurring in

$$\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathit{real}(r,v) \circ \mathcal{E}_n^{-I}, \mathit{real}(r,vl) \circ \mathcal{S}_n'^{-I}) \tag{5.75}$$

and the head of a new variant

$$\begin{aligned} \mathit{replace}(S, \mathit{real}(R,V) \circ E, \mathit{real}(R,Vl) \circ N) \leftarrow \quad & \neg \mathit{member}(\mathit{real}(R,V), S) \wedge \mathit{val}(V,Vl) \\ & \wedge\ \mathit{replace}(S,E,N) \end{aligned}$$

of (5.6). Hence, there is an SLDENF-derivation of (5.75) yielding

$$\leftarrow \neg \mathit{member}(\mathit{real}(r,v), \mathcal{S}^{-I}) \ \wedge\ \mathit{val}(v,vl) \ \wedge\ \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.76}$$

where $\mathcal{E} = \dot{\{}r(v)\dot{\}} \,\dot{\cup}\, \mathcal{E}_n$, and $\mathcal{S}'_n \doteq \dot{\{}r(vl)\dot{\}} \,\dot{\cup}\, \mathcal{S}'_n$. Since the real-valued fluent r does not appear in $\mathcal{S}$, we may safely conclude that there is no successful SLDENF-resolution proof of $\mathit{member}(\mathit{real}(r,v), \mathcal{S}^{-I})$ due to non existence of AC1-unifiers. Therefore, there is an SLDENF-derivation of (5.76) yielding

$$\leftarrow \mathit{val}(v,vl) \ \wedge\ \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.77}$$

vl is the evaluation of v using *eval* since vl is in $r(vl)$ of state $\mathcal{S}'$. Hence, there must exist the corresponding fact

$$\mathit{val}(v,vl)$$

in $\mathcal{E}_{real}$ which is AC1-unifiable with the first atom occurring in (5.77). Therefore, there is an SLDENF-derivation from (5.77) to

$$\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.78}$$

with AC1-unifier $\emptyset$. We can apply induction hypothesis to (5.74) and (5.78) by assuming that $\mathcal{S}^{-I} \doteq \mathcal{S}_n^{-I}$ for (5.78) to learn that there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$. Combining this derivation with either the ones starting from (5.71) to (5.74) or the ones starting from (5.75) to (5.78), we may conclude that there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$. $\Box$

Proposition 5.18. *If there is an SLDENF-resolution proof of* $\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ *in* $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$ *then a theory action with ground effects* $\mathcal{E}$ *transforms* $\mathcal{S}$ *into state* $\mathcal{S}'$.

Proof. The proposition is an immediate consequence of the following claim.

If there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$ with n subgoals of *replace* then a theory action with ground effects $\mathcal{E}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$.

The claim is proven by induction on the number of subgoals of *replace* in the SLDENF-resolution proof.

I.B To show: If there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_Q$ with only one subgoal of *replace* then a theory action with ground effects $\mathcal{E}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$.

If subgoal *replace* only appears once then the atom occurring in

$$\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$$

can only be AC1-unified with a new variant

$$\mathit{replace}(S, 1, S)$$

of (5.5) with an AC1-unifier of the form

$$\sigma = \{S \mapsto S^{-I}\}$$

yielding the empty clause. This happens only if $\mathcal{E} \doteq \dot{\emptyset}$ and $\mathcal{S} \doteq \mathcal{S}'$. Hence, the theory action with ground effect $\dot{\emptyset}$ transforms state $\mathcal{S}$ into $\mathcal{S}$.

I.H If there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I})$ in $\mathcal{AFC}\mathbb{R}_Q$ with n subgoals of *replace* then a theory action with ground effects $\mathcal{E}_n$ transforms state $\mathcal{S}_n$ into state $\mathcal{S}'_n$.

I.C If there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I})$ in $\mathcal{AFC}\mathbb{R}_Q$ with $n+1$ subgoals of *replace* then a theory action with ground effects $\mathcal{E}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$.

I.S Suppose there is an SLDENF-resolution proof of

$$\leftarrow \mathit{replace}(\mathcal{S}^{-I}, \mathcal{E}^{-I}, \mathcal{S}'^{-I}) \tag{5.79}$$

with $n+1$ subgoals of *replace*. Then, the atom occurring in (5.79) can only be AC1-unified with either the head of a new variant

$$\begin{aligned} \mathit{replace}(\mathit{real}(R, V) \circ S, \mathit{real}(R, Vl) \circ E, \mathit{real}(R, Vvl) \circ N) \leftarrow \quad & \mathit{val}(Vl, Vvl) \\ & \wedge\ \mathit{replace}(S, E, N) \end{aligned}$$

of (5.7) or the head of a new variant

$$\begin{aligned} \mathit{replace}(S, \mathit{real}(R, V) \circ E, \mathit{real}(R, Vl) \circ N) \leftarrow \quad & \neg \mathit{member}(\mathit{real}(R, V), S) \wedge \mathit{val}(V, Vl) \\ & \wedge\ \mathit{replace}(S, E, N) \end{aligned}$$

of (5.6). If the former is the case then the successful derivation yields

$$\leftarrow val(vl, vvl) \ \wedge \ replace(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.80}$$

with AC1-unifier

$$\sigma = \{R \mapsto r, V \mapsto v, S \mapsto \mathcal{S}_n^{-I}, Vl \mapsto vl, E \mapsto \mathcal{E}_n^{-I}, Vvl \mapsto vvl, N \mapsto \mathcal{S}_n'^{-I}\}$$

where $\mathcal{S}^{-I} \approx_{AC1} real(r, v) \circ \mathcal{S}_n^{-I}$, $\mathcal{E}^{-I} \approx_{AC1} real(r, vl) \circ \mathcal{E}_n^{-I}$ and $\mathcal{S}'^{-I} \approx_{AC1} real(r, vvl) \circ \mathcal{S}_n'^{-I}$. The first atom occurring in (5.80) and the fact

$$val(vl, vvl)$$

in $\mathcal{E}_{real}$ are AC1-unifiable with AC1-unifier $\emptyset$. The successful SLDENF-derivation yields

$$\leftarrow replace(\mathcal{S}_n^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.81}$$

If the latter is the case then the successful derivation yields

$$\leftarrow \neg member(real(r, v), \mathcal{S}^{-I}) \ \wedge \ val(v, vl) \ \wedge \ replace(\mathcal{S}^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.82}$$

with AC1-unifier

$$\sigma' = \{R \mapsto r, V \mapsto v, S \mapsto \mathcal{S}^{-I}, E \mapsto \mathcal{E}_n^{-I}, Vl \mapsto vl, N \mapsto \mathcal{S}_n'^{-I}\}$$

where $\mathcal{E}^{-I} \approx_{AC1} real(r, v) \circ \mathcal{E}_n^{-I}$ and $\mathcal{S}'^{-I} \approx_{AC1} real(r, vl) \circ \mathcal{S}_n'^{-I}$. There is an SLDENF-derivation from (5.82) to

$$\leftarrow val(v, vl) \ \wedge \ replace(\mathcal{S}^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}). \tag{5.83}$$

The reason is that there is no successful SLDENF-resolution proof of $member(real(r, v),$ $\mathcal{S}^{-I})$. there must exist an SLDENF-derivation between the first atom occurring in (5.83) and the fact

$$val(v, vl)$$

in $\mathcal{E}_{real}$ yielding

$$\leftarrow replace(\mathcal{S}^{-I}, \mathcal{E}_n^{-I}, \mathcal{S}_n'^{-I}) \tag{5.84}$$

with AC1-unifier $\emptyset$.

Based on the definition of application of a theory action, the real value v possessed by r in $\mathcal{S}$ must be replaced by $eval(vl)$ iff $r(vl)$ is in $\mathcal{E}$. The SLDENF-derivations from (5.79) to (5.81) show that the real-valued fluent r appears both in $\mathcal{S}$ and $\mathcal{E}$. The real value v possessed by r in $\mathcal{S}$ is replaced by the arithmetic evaluation of vl which is vvl in $\mathcal{S}'$. By applying induction hypothesis on (5.81) we learn that a theory action with ground effects $\mathcal{E}_n$ transforms state $\mathcal{S}_n$ into state $\mathcal{S}'_n$. Since $\mathcal{S}^{-I} \approx_{AC1} real(r,v) \circ \mathcal{S}_n^{-I}, \mathcal{E}^{-I} \approx_{AC1} real(r,vl) \circ \mathcal{E}_n^{-I}$ and $\mathcal{S}'^{-I} \approx_{AC1} real(r,vvl) \circ \mathcal{S}_n'^{-I}$, we conclude that the theory action with ground effects $\mathcal{E}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$.

On the other hand, if the real-valued fluent r does not appear in $\mathcal{S}$ then $r(v)$ in $\mathcal{E}$ is added into $\mathcal{S}$. The SLDENF-derivations from (5.79) to (5.82) to (5.84) show that the real-valued fluent r possessed by $\mathcal{E}$ does not appear in $\mathcal{S}$. As the result, the real-valued fluent r is added to $\mathcal{S}$. By applying induction hypothesis on (5.84) we learn that a theory action with ground effects $\mathcal{E}_n$ transforms state $\mathcal{S}_n$ into state $\mathcal{S}'_n$. Since $\mathcal{S}^{-I} \approx_{AC1} \mathcal{S}_n^{-I}, \mathcal{E}^{-I} \approx_{AC1} real(r,vl) \circ \mathcal{E}_n^{-I}$ and $\mathcal{S}'^{-I} \approx_{AC1} real(r,vvl) \circ \mathcal{S}_n'^{-I}$, we conclude that the theory action with ground effects $\mathcal{E}$ transforms state $\mathcal{S}$ into state $\mathcal{S}'$. $\square$

Lemma 5.19. *State $\mathcal{S}$ fulfils preconditions $\mathcal{R}$ iff there is an SLDENF-resolution proof of $\leftarrow fulfil(\mathcal{S}^{-I}, \mathcal{R}^{-I})$*

Proof. This lemma is proven by Proposition 5.20 and 5.21.

Proposition 5.20. *If state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$ then there is an SLDENF-resolution proof of $\leftarrow fulfil(\mathcal{S}^{-I}, \mathcal{R}^{-I})$.*

Proof. This proposition is proven on the number of elements of $\mathcal{R}$.
I.B To show: If state $\mathcal{S}$ fulfils preconditions $\dot{\emptyset}$ then there is an SLDENF-resolution proof of $\leftarrow fulfil(\mathcal{S}^{-I}, 1)$.
If the preconditions are an empty multiset, i.e, $\mathcal{R} \doteq \dot{\emptyset}$, then the following conditions

- $\mathcal{R} \dot{\subseteq} \mathcal{S}$;
- $\forall c(r) \in_1 \mathcal{R}\ \exists c(r') \in_1 \mathcal{S}\ (r' \geq r)$

are satisfied trivially. Let

$$\sigma = \{S \mapsto \mathcal{S}^{-I}\}$$

be a substitution. σ is the AC1-unifier between the atom occurring in

$$\leftarrow fulfil(\mathcal{S}^{-I}, 1) \tag{5.85}$$

and a new variant

$$fulfil(S, 1)$$

of (5.1). Hence, there is an SLDENF-resolution proof of (5.85) to the empty clause.

I.H If state $\mathcal{S}_n$ fulfils preconditions $\mathcal{R}_n$ then there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}_n^{-I}, \mathcal{R}_n^{-I})$.
I.C To show: If state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$ then there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I})$.
I.S Suppose state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$. Preconditions consist of simple fluents and continuous fluents. Hence, we can split $\mathcal{R}$ into $\mathcal{R}_s$ for all simple fluents and $\mathcal{R}_c$ for all real-valued fluents appearing in $\mathcal{R}$. It means that the following conditions

- $\mathcal{R}_s \dot{\subseteq} \mathcal{S}$;
- $\forall c(r) \in_1 \mathcal{R}_c \; \exists c(r') \in_1 \mathcal{S} \; (r' \geq r)$

are satisfied. Let r be a fluent in $\mathcal{R}$, then there are two possibilities. The first one is that p is a simple fluent and it belongs to $\mathcal{R}_s$. Hence, p satisfies $\dot{\{}p\dot{\}} \dot{\subseteq} \mathcal{S}$. Let

$$\sigma = \{P \mapsto p, S \mapsto (\mathcal{S} \dot{\setminus} \dot{\{}p\dot{\}})^{-I}, Pr \mapsto (\mathcal{R} \dot{\setminus} \dot{\{}p\dot{\}})^{-I}\}$$

be a substitution. σ is the AC1-unifier between the atom occurring in

$$\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I}) \tag{5.86}$$

and the head of a new variant

$$\mathit{fulfil}(P \circ S, P \circ Pr) \leftarrow \mathit{fulfil}(S, Pr)$$

of (5.2). Hence, there is an SLDENF-derivation of (5.86) yielding

$$\leftarrow \mathit{fulfil}((\mathcal{S} \dot{\setminus} \dot{\{}p\dot{\}})^{-I}, (\mathcal{R} \dot{\setminus} \dot{\{}p\dot{\}})^{-I}) \tag{5.87}$$

The second one is that p is a ground real-valued fluent of the form $r(v')$ and it belongs to $\mathcal{R}_c$. Hence, $r(v')$ satisfies $r(v') \in_1 \mathcal{R}_c \; \exists r(v) \in_1 \mathcal{S} \; (eval(v) \geq eval(v'))$. Let

$$\sigma' = \{R \mapsto r, V \mapsto v', S \mapsto (\mathcal{S} \dot{\setminus} \dot{\{}c(o')\dot{\}})^{-I}, Vl \mapsto v, P \mapsto (\mathcal{R} \dot{\setminus} \dot{\{}c(o)\dot{\}})^{-I}\}$$

be a substitution. σ is the AC1-unifier between the atom occurring in (5.86) and the head of a new variant

$$\begin{aligned} \mathit{fulfil}(real(R,V) \circ S, real(R,Vl) \circ P) \leftarrow \quad & val(V,Vv) \wedge val(Vl,Vvl) \\ & \wedge Vv \geq Vvl \wedge \mathit{fulfil}(S,P) \end{aligned}$$

of (5.3). Hence, there is an SLDENF-derivation of (5.86) yielding

$$\leftarrow val(v, Vv) \ \wedge\ val(v', Vvl) \ \wedge\ Vv \geq Vvl \ \wedge\ fulfil((\mathcal{S}\dot{\setminus}\dot{\{}p\dot{\}})^{-I}, (\mathcal{R}\dot{\setminus}\dot{\{}p\dot{\}})^{-I}) \tag{5.88}$$

The arithmetic evaluation on v provides a real value. Let assume $eval(v)$ is vv. there must exist the corresponding fact

$$val(v, vv)$$

in $\mathcal{E}_{real}$ which is AC1-unifiable with the first atom occurring in (5.88) with AC1-unifier

$$\{Vv \mapsto vv\}.$$

Hence, there is an SLDENF-derivation from (5.88) to

$$\leftarrow val(v', Vvl) \ \wedge\ vv \geq Vvl \ \wedge\ fulfil((\mathcal{S}\dot{\setminus}\dot{\{}p\dot{\}})^{-I}, (\mathcal{R}\dot{\setminus}\dot{\{}p\dot{\}})^{-I}). \tag{5.89}$$

The arithmetic evaluation on v' provides a real value. Let assume $eval(v')$ is vvl. there must exist the corresponding fact

$$val(v', vvl)$$

in $\mathcal{E}_{real}$ which is AC1-unifiable with the first atom occurring in (5.89) with AC1-unifier

$$\{Vvl \mapsto vvl\}.$$

Hence, there is an SLDENF-derivation from (5.89) to

$$\leftarrow vv \geq vvl \ \wedge\ fulfil((\mathcal{S}\dot{\setminus}\dot{\{}p\dot{\}})^{-I}, (\mathcal{R}\dot{\setminus}\dot{\{}p\dot{\}})^{-I}). \tag{5.90}$$

Because of the fact $(eval(v) \geq eval(v'))$, there must exist the corresponding fact

$$vv \geq vvl$$

in $\mathcal{E}_{real}$ which is AC1-unifiable with the first atom occurring in (5.90) with AC1-unifier $\emptyset$. Hence, there is an SLDENF-derivation from (5.90) to (5.87).

We can apply the induction hypothesis by assigning $\mathcal{R}_n \doteq \mathcal{R}\dot{\setminus}\dot{\{}r\dot{\}}$ and $\mathcal{S}_n \doteq \mathcal{S}\dot{\setminus}\dot{\{}r\dot{\}}$ and dropping r since it is already processed. We learn that there is an SLDENF-resolution proof of (5.87) with a computed answer substitution $\emptyset$. Combining either SLDENF-derivation of (5.86) or SLDENF-derivations of (5.86) with the SLDENF-derivation of (5.87), we obtain that there is an SLDENF-resolution proof of (5.86). $\square$

Proposition 5.21. *If there is an SLDENF-resolution proof of* $\leftarrow fulfil(\mathcal{S}^{-I}, \mathcal{R}^{-I})$ *then state* $\mathcal{S}$ *fulfils preconditions* $\mathcal{R}$*.*

Proof. The proposition is an immediate consequence of the following claim.

If there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I})$ with n subgoals of *fulfil* then state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$.

The claim is proven on the number of subgoals of *fulfil* in the SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I})$.

I.B To show: If there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I})$ with one subgoal of *fulfil* then state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$.

If there is only one subgoal for *fulfil* then the atom occurring in

$$\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I}) \tag{5.91}$$

can only be AC1-unified with a new variant

$$\mathit{fulfil}(S, 1)$$

of (5.1) with AC1-unifier

$$\{S \mapsto \mathcal{S}^{-I}\}.$$

From this derivation, we conclude $\mathcal{R} \doteq \dot{\emptyset}$. Hence, state $\mathcal{S}$ fulfils preconditions $\dot{\emptyset}$.

I.H If there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}_n^{-I}, \mathcal{R}_n^{-I})$ with n subgoals of *fulfil* then state $\mathcal{S}_n$ fulfils preconditions $\mathcal{R}$.

I.C To show: If there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I})$ with $n+1$ subgoals of *fulfil* then state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$.

I.S Assume that there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I})$ with $n+1$ subgoals of *fulfil*. Then there is an SLDENF-derivation of

$$\leftarrow \mathit{fulfil}(\mathcal{S}^{-I}, \mathcal{R}^{-I}) \tag{5.92}$$

leading either to

$$\leftarrow \mathit{fulfil}(\mathcal{S}_n^{-I}, \mathcal{R}_n^{-I}) \tag{5.93}$$

with AC1-unifier

$$\{P \mapsto p, S \mapsto \mathcal{S}_n^{-I}, Pr \mapsto \mathcal{R}_n^{-I}\}$$

between the atom occurring in (5.92) and the head of a new variant

$$\mathit{fulfil}(P \circ S, P \circ Pr) \leftarrow \mathit{fulfil}(S, Pr)$$

of (5.2) where $\mathcal{S}^{-I} \approx_{AC1} p \circ \mathcal{S}_n^{-I}$ and $\mathcal{R}^{-I} \approx_{AC1} p \circ \mathcal{R}_n^{-I}$ or to

$$\leftarrow val(v, Vv) \ \wedge \ val(vl, Vvl) \ \wedge \ Vv \geq Vvl \ \wedge \ qualified(\mathcal{S}_n^{-I}, \mathcal{R}_n^{-I}) \tag{5.94}$$

with AC1-unifier

$$\{R \mapsto r, V \mapsto v, S \mapsto \mathcal{S}_n^{-I}, Vl \mapsto vl, P \mapsto \mathcal{R}_n^{-I}\}$$

between the atom occurring in (5.92) and the head of a new variant

$$\begin{aligned} \mathit{fulfil}(real(R,V) \circ S, real(R,Vl) \circ P) \leftarrow \quad & val(V,Vv) \wedge val(Vl,Vvl) \\ & \wedge Vv \geq Vvl \wedge \mathit{fulfil}(S,P) \end{aligned}$$

of (5.3) where $\mathcal{S}^{-I} \approx_{AC1} real(r,v) \circ \mathcal{S}_n^{-I}$ and $\mathcal{R}^{-I} \approx_{AC1} real(r,vl) \circ \mathcal{R}_n^{-I}$. If the latter is the case then there must exist an SLDENF-derivation of (5.94) yielding

$$\leftarrow val(vl,Vvl) \;\wedge\; vv \geq Vvl \;\wedge\; qualified(\mathcal{S}_n^{-I}, \mathcal{R}_n^{-I}) \tag{5.95}$$

with AC1-unifier

$$\{Vv \mapsto vv\}$$

between the first atom occurring in(5.94) and the fact

$$val(v,vv)$$

in $\mathcal{E}_{real}$. there must exist another SLDENF-derivation of (5.95) yielding

$$\leftarrow vv \geq vvl \;\wedge\; qualified(\mathcal{S}_n^{-I}, \mathcal{R}_n^{-I}) \tag{5.96}$$

with AC1-unifier

$$\{Vvl \mapsto vvl\}$$

between the first atom occurring in (5.95) and the fact

$$val(vl,vvl)$$

in $\mathcal{E}_{real}$. There is another SLDENF-derivation of (5.96) yielding (5.93) with AC1-unifier $\emptyset$ between the first atom occurring in (5.96) and the fact

$$vv \geq vvl$$

in $\mathcal{E}_{real}$.

For preconditions $\mathcal{R}$ to be satisfied by state $\mathcal{S}$, then the following conditions

- $\mathcal{R}_s \dot{\subseteq} \mathcal{S}$;
- $\forall c(r) \in_1 \mathcal{R}_c \; \exists c(r') \in_1 \mathcal{S} \; (r' \geq r)$

must be met where $\mathcal{R}_s$ contains all simple fluents in $\mathcal{R}$ and $\mathcal{R}_c$ contains all ground continuous fluents in $\mathcal{R}$.

The SLDENF-derivations of (5.92) suggest that a fluent term p is part of $\mathcal{S}^{-I}$ and $\mathcal{R}^{-I}$. Because p is part of them, then p is constructed from simple fluents and/or ground real-valued fluents. If p is constructed from simple fluents then $p^I \dot{\subseteq} \mathcal{S}$ is satisfied. If p is constructed from ground real-valued fluents then $\forall r(v) \in_1 p \; \exists r(v') \in_1 \mathcal{S} \; (v' = v)$ is satisfied. Since p satisfies both conditions for preconditions to be met in any ways then we can drop p from $\mathcal{R}$ and $\mathcal{S}$. We just need to verify whether $\mathcal{R}_n$ are satisfied by $\mathcal{S}_n$.

The SLDENF-derivations of (5.92) suggest that there exists a fluent term $real(r, v)$ as part of the fluent term $\mathcal{S}^{-I}$ and a fluent term $real(r, vl)$ as part of the fluent term $\mathcal{R}^{-I}$. Furthermore, the derivations tell that $vv \geq vvl$ is satisfied where vv is the evaluation value of v and vvl is the evaluation value of vl. It means that the real-valued fluent $r(v)$ in $\mathcal{S}$ satisfies the real-valued fluent $r(vl)$ in $\mathcal{R}$. Therefore, we can drop $real(r, v)$ from $\mathcal{R}$ and $real(r, vl)$ from $\mathcal{S}$ and work only with $\mathcal{S}_n$ and $\mathcal{R}_n$.

Since the SLDENF-derivations of (5.93) only contain n subgoals of *fulfil*, then we can apply induction hypothesis and learn state $\mathcal{S}_n$ fulfils preconditions $\mathcal{R}_n$. Since $\mathcal{R}$ is constructed by $\mathcal{R}_n$ and either p or $r(vl)$ and $\mathcal{S}$ is constructed by $\mathcal{S}_n$ and either p or $r(v)$ then we can safely conclude that state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$. □

The following lemma shows that a state which is not closed can be transformed into another state by applying a theory action using SLDENF-resolution proof.

Lemma 5.22. *A state $\mathcal{S}$ is not a closed state iff there is an SLDENF-resolution proof of $\leftarrow nonclosed(\mathcal{S}^{-I})$.*

Proof. This lemma is proven by Proposition 5.23 and 5.24.

Proposition 5.23. *If a state $\mathcal{S}$ is not a closed state then there is an SLDENF-resolution proof of $\leftarrow nonclosed(\mathcal{S}^{-I})$.*

Proof. Assume that a state $\mathcal{S}$ is not a closed state. Then there must exist a theory action a which is applicable in $\mathcal{S}$. Lets assume that the theory action a is of the form

$$a : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}.$$

Hence, we can find the corresponding clauses

$$taction(\mathcal{C}^{-I}, a, \mathcal{E}^{-I}) \tag{5.97}$$

and

$$precon(\mathcal{R}^{-I}, a) \tag{5.98}$$

in $\mathcal{AFC}\mathbb{R}_Q$. Since a is applicable in $\mathcal{S}$, it means that there is a substitution σ such that $\mathcal{C}\sigma \dot{\subseteq} \mathcal{S}$. Furthermore, $\mathcal{R} \dot{\subseteq} \mathcal{S}$ and $\forall o \in_n \mathcal{O}\ (o \in_m \mathcal{S} \rightarrow m > n)$ are satisfied as well. We can apply the substitution σ to (5.97) to obtain the corresponding ground clause

$$taction(\mathcal{C}^{-I}\sigma, a, \mathcal{E}^{-I}\sigma) \tag{5.99}$$

of (5.97). If this theory action is applied then the action transforms the state $\mathcal{S}$ into another state $\mathcal{S}''$, i.e.,

$$\mathcal{S} \xrightarrow{a} \mathcal{S}''$$

where $\mathcal{S} \not\doteq \mathcal{S}''$. Let

$$\sigma' = \{C \mapsto \mathcal{C}', S \mapsto \mathcal{S}'\}$$

be a substitution. σ' is the AC1-unifier between the atom occurring in

$$\leftarrow nonclosed(\mathcal{S}^{-I}) \tag{5.100}$$

and the head of a new variant

$$\begin{aligned} nonclosed(C \circ S) \leftarrow\ & taction(C, A, E) \wedge precon(P, A) \\ & \wedge fulfil(C \circ S, P) \wedge \neg hinder(C \circ S, A) \\ & \wedge replace(C \circ S, E, N) \wedge C \circ S \not\approx N \end{aligned}$$

of (5.8) where $\mathcal{S} \doteq \mathcal{C}' \,\dot{\cup}\, \mathcal{S}'$. Then there is an SLDENF-derivation from (5.100) to

$$\begin{aligned} \leftarrow\ & taction(\mathcal{C}'^{-I}, A, E) \wedge precon(P, A) \wedge fulfil(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, P) \\ & \wedge \neg hinder(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, A) \wedge replace(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, E, N) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx N \end{aligned} \tag{5.101}$$

using the variant of (5.8) and the AC1-unifier σ'. Let

$$\sigma'' = \{E \mapsto \mathcal{E}^{-I}\sigma, A \mapsto a\}$$

be a substitution. σ'' is the AC1-unifier between the first atom occurring in (5.101) and (5.99). Hence, there is an SLDENF-derivation from (5.101) to

$$\begin{aligned} \leftarrow\ & precon(P, a) \wedge fulfil(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, P) \wedge \neg hinder(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a) \\ & \wedge replace(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, N) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx N \end{aligned} \tag{5.102}$$

using (5.99) and the AC1-unifier σ''. There is an SLDENF-derivation from (5.102) to

$$\begin{aligned}\leftarrow \quad & \mathit{fulfil}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{R}^{-I}) \wedge \neg \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a) \\ & \wedge\ \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, N) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx N\end{aligned} \tag{5.103}$$

between the first atom occurring in (5.102) and (5.98) with an AC1-unifier of the form

$$\{P \mapsto \mathcal{R}^{-I}\}.$$

As state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$, we can apply Lemma 5.19 and learn that there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{R}^{-I})$ with an answer substitution $\emptyset$. Hence, we can find the corresponding derivations from (5.103) to

$$\leftarrow \neg \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a) \wedge \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, N) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx N \tag{5.104}$$

As $\forall o \in_n \mathcal{O}\ (o \in_m \mathcal{S} \to m > n)$ is satisfied, we can apply Lemma 5.15 and learn that there is no SLDENF-resolution proof of $\leftarrow \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a)$. We can construct a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a)$. Hence, there is an SLDENF-derivation from (5.104) to

$$\leftarrow \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, N) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx N \tag{5.105}$$

We can apply Lemma 5.16 to $\mathcal{S}$, $\mathcal{E}\sigma$ and $\mathcal{S}''$ and learn that there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, N)$ with an answer substitution $\{N \mapsto \mathcal{S}''^{-I}\}$. Hence, we can find the corresponding derivations from (5.105) to

$$\leftarrow \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx \mathcal{S}''^{-I} \tag{5.106}$$

where variable N is replaced by $\mathcal{S}''^{-I}$ in the process. Because the new state $\mathcal{S}''$ is not equal to state $\mathcal{S}$, we can conclude that $\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx_{AC1} \mathcal{S}''^{-I}$. Hence, we can construct a finitely failed SLDENF-tree of $\leftarrow \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \approx \mathcal{S}''^{-I}$. Therefore, there is an SLDENF-derivation from (5.106) to the empty clause. $\square$

Proposition 5.24. *If there is an SLDENF-resolution proof of* $\leftarrow \mathit{nonclosed}(\mathcal{S}^{-I})$ *then a state* $\mathcal{S}$ *is not a closed state.*

Proof. Assume that there is an SLDENF-resolution proof of

$$\leftarrow \mathit{nonclosed}(\mathcal{S}^{-I}) \tag{5.107}$$

Hence, the atom occurring in (5.107) can only be AC1-unified with the head of a new variant

$$\begin{aligned} \mathit{nonclosed}(C \circ S) \leftarrow\ & \mathit{taction}(C, A, E) \wedge \mathit{precon}(P, A) \\ & \wedge \mathit{fulfil}(C \circ S, P) \wedge \neg \mathit{hinder}(C \circ S, A) \\ & \wedge \mathit{replace}(C \circ S, E, N) \wedge C \circ S \not\approx N \end{aligned}$$

of (5.8) with AC1-unifier

$$\{C \mapsto c, S \mapsto s\}$$

where $\mathcal{S}^{-I} \approx_{AC1} c \circ s$. Then the successful derivation of (5.107) yields

$$\begin{aligned} \leftarrow\ & \mathit{taction}(c, A, E) \wedge \mathit{precon}(P, A) \wedge \mathit{fulfil}(c \circ s, P) \\ & \wedge \neg \mathit{hinder}(c \circ s, A) \wedge \mathit{replace}(c \circ s, E, N) \wedge c \circ s \not\approx N \end{aligned} \tag{5.108}$$

There must exist an AC1-unifier σ between the first atom occurring in (5.108) and a new variant of the fact

$$\mathit{taction}(c', a, e') \tag{5.109}$$

such that $c \approx_{AC1} c'\sigma$, variable A maps to a and $E\sigma \approx_{AC1} e'\sigma$. The successful derivation of (5.108) yields

$$\begin{aligned} \leftarrow\ & \mathit{precon}(P, a) \wedge \mathit{fulfil}(c \circ s, P) \\ & \wedge \neg \mathit{hinder}(c \circ s, a) \wedge \mathit{replace}(c \circ s, e'\sigma, N) \wedge c \circ s \not\approx N \end{aligned} \tag{5.110}$$

There must exist an AC1-unifier

$$\sigma' = \{P \mapsto p\}$$

between the first atom occurring in (5.110) and the fact

$$\mathit{precon}(p, a). \tag{5.111}$$

The successful derivation of (5.110) yields

$$\leftarrow \mathit{fulfil}(c \circ s, p) \wedge \neg \mathit{hinder}(c \circ s, a) \wedge \mathit{replace}(c \circ s, e'\sigma, N) \wedge c \circ s \not\approx N \tag{5.112}$$

There must exist a series of derivations from (5.112) to

$$\leftarrow \neg \mathit{hinder}(c \circ s, a) \wedge \mathit{replace}(c \circ s, e'\sigma, N) \wedge c \circ s \not\approx N \tag{5.113}$$

There is a derivation from (5.113) to

$$\leftarrow \mathit{replace}(c \circ s, e'\sigma, N) \wedge c \circ s \not\approx N \tag{5.114}$$

Hence, there must exist a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(c \circ s, a)$. There is a series of derivations from (5.114) to

$$\leftarrow c \circ s \not\approx n \tag{5.115}$$

where variable N is replaced by a ground fluent term n on the process. There is a derivation from (5.115) to the empty clause. Therefore we can conclude that there is a finitely failed SLDENF-tree of $\leftarrow c \circ s \approx n$.

From (5.109) and (5.111), we may conclude that there is theory action a in the form of

$$a : c'^{I} \xrightarrow{p^I, \mathcal{O}} e'^{I}$$

The derivation from (5.108) to (5.110) shows that there exists substitution σ such that $(c'\sigma)^I \dot{\subseteq} \mathcal{S}$. Because there is a series of derivations from (5.112) to (5.113), there must exist a successful SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(c \circ s, p)$. We can apply Lemma 5.19 and learn that preconditions p^I are fulfilled by $\mathcal{S}$. There is a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(c \circ s, a)$. We can apply Lemma 5.15 and learn that there is no obstacle from $\mathcal{O}$ which can hinder the theory action a from being applied in $\mathcal{S}$. Hence, we may conclude that a is applicable in $\mathcal{S}$.

Because there is a series of derivations from (5.114) to (5.115) where variable N is replaced by a ground fluent term n, there must exist a successful SLDENF-resolution proof of $\leftarrow \mathit{replace}(c \circ s, e'\sigma, N)$ with an answer substitution $\{N \mapsto n\}$. We can apply Lemma 5.16 and learn that the application of a with ground effects $e'\sigma^I$ in $\mathcal{S}$ transforms $\mathcal{S}$ into a new state n^I. There is a finitely failed SLDENF-tree of $\leftarrow c \circ s \not\approx n$. We can conclude that $c \circ s \not\approx_{AC1} n$ is true and so is $\mathcal{S} \dot{\neq} n^I$. Therefore, state $\mathcal{S}$ is not closed. $\Box$

The following lemma shows that a sequence of theory actions applied to a state to obtain a closed state can be generated by an SLDENF-resolution proof.

Lemma 5.25. *p is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}{\downarrow}$ iff p can be generated by an SLDENF-resolution proof of*
$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}{\downarrow}^{-I})$.

Proof. This lemma is proven by Proposition 5.26 and 5.27.

Proposition 5.26. *If p is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}{\downarrow}$ then p can be generated by an SLDENF-resolution proof of*
$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}{\downarrow}^{-I})$.

Proof. This proposition is proven on the length of the sequence p.

I.B To show: If [] is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed

state $\mathcal{S}\!\downarrow$ then $[\,]$ can be generated by an SLDENF-resolution proof of
$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}\!\downarrow^{-I})$.
Assume that $[\,]$ is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}\!\downarrow$. It means that $\mathcal{S} \doteq \mathcal{S}\!\downarrow$. For each $a \in \mathcal{A}_{real}(\mathcal{S})$ does not transform $\mathcal{S}$ into a different state. Let

$$\sigma = \{S \mapsto \mathcal{S}^{-I}, P \mapsto [\,]\}$$

be a substitution. σ is the AC1-unifier between the head of a new variant

$$\mathit{reapplicable}(S, [\,], S) \leftarrow \neg \mathit{nonclosed}(S)$$

of (5.9) and

$$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}^{-I}) \tag{5.116}$$

The successful derivation of (5.116) yields

$$\leftarrow \neg \mathit{nonclosed}(\mathcal{S}^{-I}) \tag{5.117}$$

We can apply Lemma 5.22 from the fact that $\mathcal{S}$ is a closed state to obtain that there is no successful SLDENF-resolution proof of $\leftarrow \mathit{nonclosed}(\mathcal{S}^{-I})$. Hence, there is an SLDENF-derivation from (5.117) to the empty clause. Therefore $[\,]$ is generated by an SLDENF-resolution proof of (5.116).

I.H If $[a_j, \ldots, a_1]$ is a sequence of theory actions transforming a state $\mathcal{S}_j$ into a closed state $\mathcal{S}_j\!\downarrow$ then $[a_j, \ldots, a_1]$ can be generated by an SLDENF-resolution proof of
$\leftarrow \mathit{reapplicable}(\mathcal{S}_j^{-I}, P_j, \mathcal{S}_j\!\downarrow^{-I})$.
I.C To show: If $[a_{j+1}, \ldots, a_1]$ is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}\!\downarrow$ then $[a_{j+1}, \ldots, a_1]$ can be generated by an SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}\!\downarrow^{-I})$.
I.S Suppose $[a_{j+1}, \ldots, a_1]$ is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}\!\downarrow$. a_{j+1} is the first theory action applied to $\mathcal{S}$. Lets assume that a_{j+1} is of the form

$$a_{j+1} : \mathcal{C} \overset{\mathcal{R},\mathcal{O}}{\Longrightarrow} \mathcal{E}$$

Hence, there are corresponding clauses

$$\mathit{taction}(\mathcal{C}^{-I}, a_{j+1}, \mathcal{E}^{-I}) \tag{5.118}$$

and

$$\mathit{precon}(\mathcal{R}^{-I}, a_{j+1}) \tag{5.119}$$

in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$. Since a_{j+1} is applicable in $\mathcal{S}$, it means that there is a substitution σ such that $\mathcal{C}\sigma \dot{\subseteq} \mathcal{S}$. Furthermore, $\mathcal{R} \dot{\subseteq} \mathcal{S}$ and $\forall o \in_n \mathcal{O}\ (o \in_m \mathcal{S} \rightarrow m > n)$ are satisfied as well. We can apply the substitution σ to (5.118) to obtain the corresponding ground clause

$$\mathit{taction}(\mathcal{C}^{-I}\sigma, a_{j+1}, \mathcal{E}^{-I}\sigma) \tag{5.120}$$

of (5.118). The theory action a_{j+1} transforms the state $\mathcal{S}$ into another state $\mathcal{S}''$, i.e.,

$$\mathcal{S} \xrightarrow{a_{j+1}} \mathcal{S}''$$

where $\mathcal{S} \not\doteq \mathcal{S}''$. Let

$$\sigma' = \{C \mapsto \mathcal{C}', S \mapsto \mathcal{S}', N \mapsto \mathcal{S}{\downarrow}^{-I}, P \mapsto [A \mid T]\}$$

be a substitution. σ' is the AC1-unifier between the atom occurring in

$$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}{\downarrow}^{-I}) \tag{5.121}$$

and the head of a new variant

$$\begin{aligned} \mathit{reapplicable}(C \circ S, [A \mid T], N) \leftarrow\ & \mathit{taction}(C, A, E) \wedge \mathit{precon}(P, A) \\ & \wedge \mathit{fulfil}(C \circ S, P) \wedge \neg \mathit{hinder}(C \circ S, A) \\ & \wedge \mathit{replace}(C \circ S, E, Z) \wedge C \circ S \not\approx Z \\ & \wedge \mathit{reapplicable}(Z, T, N) \end{aligned}$$

of (5.10) where $\mathcal{S} \doteq \mathcal{C}' \dot{\cup} \mathcal{S}'$. Then there is an SLDENF-derivation from (5.121) to

$$\begin{aligned} \leftarrow\ & \mathit{taction}(\mathcal{C}'^{-I}, A, E) \wedge \mathit{precon}(P, A) \wedge \mathit{fulfil}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, P) \\ & \wedge \neg \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, A) \wedge \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, E, Z) \\ & \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx Z \wedge \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.122}$$

using the variant of (5.8) and the AC1-unifier σ'. Let

$$\sigma'' = \{E \mapsto \mathcal{E}^{-I}\sigma, A \mapsto a_{j+1}\}$$

be a substitution. σ'' is the AC1-unifier between the first atom occurring in (5.122) and (5.120). Hence, there is an SLDENF-derivation from (5.122) to

$$\begin{aligned} \leftarrow\ & \mathit{precon}(P, a_{j+1}) \wedge \mathit{fulfil}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, P) \wedge \neg \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a_{j+1}) \\ & \wedge \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, Z) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx Z \\ & \wedge \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.123}$$

using (5.120) and the AC1-unifier σ''. There is an SLDENF-derivation from (5.123) to

$$\begin{aligned} \leftarrow \quad & \mathit{fulfil}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{R}^{-I}) \wedge \neg \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a_{j+1}) \\ & \wedge\ \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, Z) \wedge \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx Z \\ & \wedge\ \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.124}$$

between the first atom occurring in (5.123) and (5.119) with an AC1-unifier of the form

$$\{P \mapsto \mathcal{R}^{-I}\}.$$

As state $\mathcal{S}$ fulfils preconditions $\mathcal{R}$, we can apply Lemma 5.19 and learn that there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{R}^{-I})$ with an answer substitution $\emptyset$. Hence, we can find the corresponding derivations from (5.124) to

$$\begin{aligned} \leftarrow \quad & \neg \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a_{j+1}) \wedge \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, Z) \\ & \wedge\ \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx Z \wedge \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.125}$$

As $\forall o \in_n \mathcal{O}\ (o \in_m \mathcal{S} \rightarrow m > n)$ is satisfied, we can apply Lemma 5.15 and learn that there is no SLDENF-resolution proof of $\leftarrow \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a_{j+1})$. We can construct a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, a_{j+1})$. Hence, there is an SLDENF-derivation from (5.125) to

$$\begin{aligned} \leftarrow \quad & \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, Z) \wedge\ \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx Z \\ & \wedge\ \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.126}$$

We can apply Lemma 5.16 to $\mathcal{S}$, $\mathcal{E}\sigma$ and $\mathcal{S}''$ and learn that there is an SLDENF-resolution proof of $\leftarrow \mathit{replace}(\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I}, \mathcal{E}^{-I}\sigma, Z)$ with an answer substitution

$$\{Z \mapsto \mathcal{S}''^{-I}\}.$$

Hence, we can find the corresponding derivations from (5.126) to

$$\leftarrow \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx \mathcal{S}''^{-I} \wedge \mathit{reapplicable}(\mathcal{S}'^{-I}, T, \mathcal{S}{\downarrow}^{-I}) \tag{5.127}$$

where variable Z is replaced by $\mathcal{S}''^{-I}$ in the process. Because the new state $\mathcal{S}''$ is not equal to state $\mathcal{S}$, we can conclude that $\mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \not\approx_{AC1} \mathcal{S}''^{-I}$. Hence, we can construct a finitely failed SLDENF-tree of $\leftarrow \mathcal{C}'^{-I} \circ \mathcal{S}'^{-I} \approx \mathcal{S}''^{-I}$. Therefore, there is an SLDENF-derivation from (5.127) to

$$\leftarrow \mathit{reapplicable}(\mathcal{S}''^{-I}, T, \mathcal{S}{\downarrow}^{-I}) \tag{5.128}$$

Because this proof contains $[a_j, \ldots, a_1]$, we may conclude that $\mathcal{S}{\downarrow}^{-I} = \mathcal{S}_j{\downarrow}^{-I}$ and $\mathcal{S}''^{-I} =$

$\mathcal{S}_j^{-I}$. We can apply the induction hypothesis and learn that there is an SLDENF-resolution proof of (5.128) with an answer substitution

$$\{T \mapsto [a_j, \dots, a_1]\}.$$

Combining this refutation with derivations from (5.121) to (5.128) yields an SLDENF-derivation of (5.121) with an answer substitution $\{P \mapsto [a_{j+1}, \dots, a_1]\}$. Hence a sequence $[a_{j+1}, \dots, a_1]$ of theory actions is generated. □

As we mentioned earlier in Subsection 5.1, we only count the number of SLDENF-refutation steps involving subgoals *causes* and *reapplicable*. In the following proposition, the length of an SLDENF-resolution is the number of SLDENF-refutation steps involving subgoal *reapplicable*.

Proposition 5.27. *If p can be generated by an SLDENF-resolution proof of $\leftarrow$ reapplicable$(\mathcal{S}^{-I}, P, \mathcal{S}\downarrow^{-I})$ then p is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}\downarrow$.*

Proof. The proposition is proven by induction on the length j of the SLDENF-resolution proof.
I.B To show: If p can be generated by an SLDENF-resolution proof of $\leftarrow$ *reapplicable*$(\mathcal{S}^{-I}, P, \mathcal{S}\downarrow^{-I})$ with length 1 then p is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}\downarrow$.
Assume that there is an SLDENF-resolution proof of $\leftarrow$ *reapplicable*$(\mathcal{S}^{-I}, P, \mathcal{S}\downarrow^{-I})$ with length 1. Hence, the atom occurring in

$$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}\downarrow^{-I}) \tag{5.129}$$

can only be AC1-unified with the head of a new variant

$$\mathit{reapplicable}(S, [\,], S) \leftarrow \neg \mathit{nonclosed}(S)$$

of (5.9) only if $\mathcal{S} \doteq \mathcal{S}\downarrow$. The AC1-unifier is of the form

$$\sigma = \{S \mapsto \mathcal{S}^{-I}, P \mapsto [\,]\}.$$

The derivation of (5.129) yields

$$\leftarrow \neg \mathit{nonclosed}(\mathcal{S}^{-I}) \tag{5.130}$$

there must exist a finitely failed SLDENF-tree of $\leftarrow$ *nonclosed*$(\mathcal{S}^{-I})$. Hence, there is an SLDENF-derivation from (5.130) to the empty clause.

We can apply Lemma 5.22 from the fact that there is no successful SLDENF-resolution proof of $\leftarrow \mathit{nonclosed}(\mathcal{S}^{-I})$. We learn that $\mathcal{S}$ is a closed state. No theory action which is applicable in $\mathcal{S}$ can bring $\mathcal{S}$ into a different state. Hence, $[\,]$ is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}{\downarrow}$.

I.H If p_j can be generated by an SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(\mathcal{S}_j^{-I}, P_j, \mathcal{S}_j{\downarrow}^{-I})$ with length j then p_j is a sequence of theory actions transforming a state $\mathcal{S}_j$ into a closed state $\mathcal{S}_j{\downarrow}$.
I.C To show: If p can be generated by an SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}{\downarrow}^{-I})$ with length $j+1$ then p is a sequence of theory actions transforming a state $\mathcal{S}$ into a closed state $\mathcal{S}{\downarrow}$.
I.S Suppose there is an SLDENF-resolution proof of

$$\leftarrow \mathit{reapplicable}(\mathcal{S}^{-I}, P, \mathcal{S}{\downarrow}^{-I}) \tag{5.131}$$

with length $j+1$ and generating $[a_j, \ldots, a_1]$. The first atom occurring in (5.131) can only be AC1-unified with the head of a new variant

$$\begin{aligned} \mathit{reapplicable}(C \circ S, [A \mid T], N) \leftarrow\ & \mathit{taction}(C, A, E) \wedge \mathit{precon}(P', A) \\ & \wedge \mathit{fulfil}(C \circ S, P') \wedge \neg \mathit{hinder}(C \circ S, A) \\ & \wedge \mathit{replace}(C \circ S, E, Z) \wedge C \circ S \not\approx Z \\ & \wedge \mathit{reapplicable}(Z, T, N) \end{aligned}$$

of (5.10) with an AC1-unifier of the form

$$\{C \mapsto c, S \mapsto s, P \mapsto [A \mid T], N \mapsto \mathcal{S}{\downarrow}^{-I}\}$$

where $\mathcal{S}^{-I} \approx_{AC1} c \circ s$. The successful derivation yields

$$\begin{aligned} \leftarrow\ & \mathit{taction}(c, A, E) \wedge \mathit{precon}(P', A) \wedge \mathit{fulfil}(c \circ s, P') \wedge \neg \mathit{hinder}(c \circ s, A) \\ & \wedge \mathit{replace}(c \circ s, E, Z) \wedge c \circ s \not\approx Z \wedge \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.132}$$

there must exist a fact

$$\mathit{taction}(c', a_j, e') \tag{5.133}$$

which can be AC1-unified with the first atom occurring in (5.132). Let σ be a substitution such that $c \approx_{AC1} c'\sigma$, $E\sigma \approx_{AC1} e'\sigma$ and $A \mapsto a_j$. σ is the AC1-unifier between the first atom occurring in (5.132) and (5.133). Hence, there is an SLDENF-derivation from (5.132) to

$$\begin{aligned} \leftarrow\ & \mathit{precon}(P', a_j) \wedge \mathit{fulfil}(c \circ s, P') \wedge \neg \mathit{hinder}(c \circ s, a_j) \\ & \wedge \mathit{replace}(c \circ s, e'\sigma, Z) \wedge c \circ s \not\approx Z \wedge \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.134}$$

there must exist a fact

$$\mathit{precon}(p, a_j) \tag{5.135}$$

which can be AC1-unified with the first atom occurring in (5.134). Let

$$\sigma' = \{P' \mapsto p\}$$

be a substitution. σ' is the AC1-unifier between the first atom occurring in (5.134) and (5.135). Hence, there is an SLDENF-derivation from (5.134) to

$$\begin{aligned} \leftarrow \quad & \mathit{fulfil}(c \circ s, p) \wedge \neg \mathit{hinder}(c \circ s, a_j) \\ & \wedge\ \mathit{replace}(c \circ s, e'\sigma, Z) \wedge c \circ s \not\approx Z \\ & \wedge\ \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.136}$$

There must exist an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(c \circ s, p)$ with an answer substitution $\emptyset$. Hence, we can find the corresponding SLDENF-derivations from (5.136) to

$$\begin{aligned} \leftarrow \quad & \neg \mathit{hinder}(c \circ s, a_j) \wedge \mathit{replace}(c \circ s, e'\sigma, Z) \\ & \wedge\ c \circ s \not\approx Z \wedge \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.137}$$

There must exist a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(c \circ s, a_j)$. Hence, there is an SLDENF-derivation from (5.137) to

$$\begin{aligned} \leftarrow \quad & \mathit{replace}(c \circ s, e'\sigma, Z) \wedge c \circ s \not\approx Z \\ & \wedge\ \mathit{reapplicable}(Z, T, \mathcal{S}{\downarrow}^{-I}) \end{aligned} \tag{5.138}$$

There must exist an SLDENF-resolution proof of $\leftarrow \mathit{replace}(c \circ s, e'\sigma, Z)$ with an answer substitution $\{Z \mapsto z\}$. Hence, we can find the corresponding SLDENF-derivations from (5.138) to

$$\leftarrow c \circ s \not\approx z \wedge \mathit{reapplicable}(z, T, \mathcal{S}{\downarrow}^{-I}) \tag{5.139}$$

where variable Z is replaced by a ground fluent term z in the process. There must exist a finitely failed SLDENF-tree of $\leftarrow c \circ s \approx z$. Hence, there is an SLDENF-derivation from (5.139) to

$$\leftarrow \mathit{reapplicable}(z, T, \mathcal{S}{\downarrow}^{-I}) \tag{5.140}$$

From (5.133) and (5.135), we may conclude that there must exist a theory action a_j of the form

$$a_j : c'^I \xRightarrow{p^I, \mathcal{O}} e'^I$$

in $\mathcal{Q}$. The derivation from (5.132) to (5.134) shows that there exists a substitution σ such that $c'\sigma^I \dot{\subseteq} \mathcal{S}$. We can apply Lemma 5.19 because the existence of the SDLENF-resolution proof of $\leftarrow \mathit{fulfil}(c \circ s, p)$. We learn that state $\mathcal{S}$ fulfils the preconditions p^I.

We can apply Lemma 5.15 because there is a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(c \circ s, a_j)$. We learn that there are obstacles at a state $\mathcal{S}$ to hinder the theory action a_j in $\mathcal{Q}$. Hence, we may conclude that a_j is applicable in $\mathcal{S}$.

We can apply Lemma 5.16 because the existence of the SLDENF-resolution proof of $\leftarrow \mathit{replace}(c \circ s, e'\sigma, z)$. We learn that the application of a_j to state $\mathcal{S}$ transforms state $\mathcal{S}$ into a new state z^I. There is a finitely failed SLDENF-tree of $\leftarrow c \circ s \approx z$. We may conclude that $c \circ s \not\approx_{AC1} z$ is true and so is $\mathcal{S} \not\doteq z^I$.

We can apply (I.H) to (5.140) since the SLDENF-resolution proof only has length j. We learn that $[a_{j-1}, \ldots, a_1]$ is a sequence of theory actions transforming state z^I to a closed state $\mathcal{S}\downarrow$. Combining the transformation from $\mathcal{S}$ to z^I by the theory action a_j and the result from (I.H), we conclude that $[a_j, \ldots, a_1]$ is a sequence of theory actions transforming state $\mathcal{S}$ to a closed state $\mathcal{S}\downarrow$. □

In the following theorem, we show that a solution for an acpp$^{\mathbb{R}}$ can be generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.

Theorem 5.28. *p is a solution for $\mathcal{Q}$ iff p is generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.*

Proof. The theorem is proven by Lemma 5.29 and 5.30.

Lemma 5.29. *If p is a solution for $\mathcal{Q}$ then p is generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.*

Proof. We prove it by induction on length n of solution p.
I.B To show: If $[\,]$ is a solution for $\mathcal{Q}$ then $[\,]$ is generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.
If $[\,]$ is a solution for $\mathcal{Q}$ then it means $\mathcal{I} \doteq \mathcal{G}$. There is no need to do any action. Let

$$\sigma = \{S \mapsto \mathcal{I}^{-I}, P \mapsto [\,]\}$$

be a substitution. σ is the AC1-unifier for the atom occurring in

$$\leftarrow \mathit{causes}(\mathcal{I}^{-I}, [\,], \mathcal{G}^{-I}) \tag{5.141}$$

and the variant

$$\mathit{causes}(S, [\,], S)$$

of clause (3.2). Thus, there exists an SLDENF-derivation from (5.141) to the empty clause generating the plan $[\,]$.

I.H If $[a_j, \ldots, a_1]$ is a solution for $\mathcal{Q}_j = \langle \mathcal{I}_j, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$ then $[a_j, \ldots, a_1]$ is generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.
I.C To show: If $[a_{j+1}, \ldots, a_1]$ is a solution for $\mathcal{Q}$ then $[a_{j+1}, \ldots, a_1]$ is generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$.
I.S Suppose $[a_{j+1}, \ldots, a_1]$ is a solution for $\mathcal{Q}$. a_{j+1} is the first concrete action taken and is followed by a sequence $[a_j, \ldots, a_n]$ of theory actions where $1 \leq n \leq j$. Lets assume a_{j+1} is of the form

$$a_{j+1} : \mathcal{C} \xrightarrow{\mathcal{R},\mathcal{O}} \mathcal{E}$$

We can find the corresponding clauses

$$action(\mathcal{C}^{-I}, a_{j+1}, \mathcal{E}^{-I}) \tag{5.142}$$

and

$$precon(\mathcal{R}^{-I}, a_{j+1}) \tag{5.143}$$

in $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$. Since a_{j+1} is applied to the initial state $\mathcal{I}$, then the conditions $\mathcal{C}$ must be a submultiset of $\mathcal{I}$, the preconditions $\mathcal{R}$ are fulfilled by $\mathcal{I}$, and there is not enough obstacle in $\mathcal{I}$ to hinder the action a_{j+1} from being taken. The application of a_{j+1} leads to a new state $(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$, i.e.,

$$\mathcal{I} \xrightarrow{a_{j+1}} (\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$$

From state $(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E}$, the sequence $[a_j, \ldots, a_n]$ is applied leading to a closed state $((\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E})\downarrow$, i.e.,

$$(\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E} \xrightarrow{[a_j,\ldots,a_n]} ((\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E})\downarrow \tag{5.144}$$

From the closed state $((\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E})\downarrow$, a sequence $[a_{n-1}, \ldots, a_1]$ of actions is applied to transform it to the goal state $\mathcal{G}$, i.e.,

$$((\mathcal{I} \dot{\setminus} \mathcal{C}) \dot{\cup} \mathcal{E})\downarrow \xrightarrow{[a_{n-1},\ldots,a_1]} \mathcal{G} \tag{5.145}$$

With function *append*, we can combine list $[a_{j+1} \mid [a_j, \ldots, a_n]]$ and $[a_{n+1}, \ldots, a_1]$ to obtain the plan $[a_{j+1}, \ldots, a_1]$. Let

$$\sigma = \{I \mapsto \mathcal{I}^{-I}, G \mapsto \mathcal{G}^{-I}, P \mapsto [A \mid L]\}$$

σ is the AC1-unifier between the first atom occurring in

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{5.146}$$

and the head of a new variant

$$\begin{aligned} causes(I,[A \mid L],G) \leftarrow\ & applicable(I,A,S) \wedge append(T,P'',L) \\ & \wedge\ reapplicable(S,T,N) \wedge causes(N,P'',G) \end{aligned}$$

of clause (5.13). Hence, there is an SLDENF-derivation from (5.146) to

$$\begin{aligned} \leftarrow\ & applicable(\mathcal{I}^{-I},A,S) \wedge append(T,P'',L) \\ & \wedge\ reapplicable(S,T,N) \wedge causes(N,P'',\mathcal{G}^{-I}) \end{aligned} \tag{5.147}$$

using the new variant of clause (5.13) and the AC1-unifier σ. Let $\mathcal{I}$ be equal to $\mathcal{I}' \mathbin{\dot\cup} \mathcal{C}$ and

$$\sigma' = \{C \mapsto \mathcal{C}^{-I}, S' \mapsto \mathcal{I}'^{-I}, S \mapsto E \circ \mathcal{I}'^{-I}\}$$

be a substitution. σ' is the AC1-unifier between the head of a new variant

$$\begin{aligned} applicable(C \circ S',A,E \circ S') \leftarrow\ & action(C,A,E) \wedge precon(P',A) \\ & \wedge\ fulfil(C \circ S',P') \wedge \neg hinder(C \circ S',A) \end{aligned}$$

of clause (5.12) and the first atom occurring in (5.147). The successful derivation yields

$$\begin{aligned} \leftarrow\ & action(\mathcal{C}^{-I},A,E) \wedge precon(P',A) \wedge fulfil(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I},P') \\ & \wedge\ \neg hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I},A) \wedge append(T,P'',L) \\ & \wedge\ reapplicable(E \circ \mathcal{I}'^{-I},T,N) \wedge causes(N,P'',\mathcal{G}^{-I}) \end{aligned} \tag{5.148}$$

There is a successful derivation from (5.148) to

$$\begin{aligned} \leftarrow\ & precon(P',a_{j+1}) \wedge fulfil(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I},P') \\ & \wedge\ \neg hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I},a_{j+1}) \wedge append(T,P'',L) \\ & \wedge\ reapplicable(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I},T,N) \wedge causes(N,P'',\mathcal{G}^{-I}) \end{aligned} \tag{5.149}$$

between the first atom occurring in (5.149) and (5.142) with an AC1-unifier of the form

$$\{E \mapsto \mathcal{E}^{-I}, A \mapsto a_{j+1}\}$$

There is a successful derivation from (5.149) to

$$\begin{aligned} \leftarrow\ & fulfil(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I},\mathcal{R}^{-I}) \\ & \wedge\ \neg hinder(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I},a_{j+1}) \wedge append(T,P'',L) \\ & \wedge\ reapplicable(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I},T,N) \wedge causes(N,P'',\mathcal{G}^{-I}) \end{aligned} \tag{5.150}$$

between the first atom occurring in (5.149) and (5.143) with an AC1-unifier of the form

$$\{P' \mapsto \mathcal{R}^{-I}\}$$

We can apply Lemma 5.19 from the fact that $\mathcal{S}$ fulfils $\mathcal{R}$ and obtain that there is an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, \mathcal{R}^{-I})$ with an answer substitution $\emptyset$. Hence, there must exist the corresponding derivations from (5.150) to

$$\begin{aligned} \leftarrow \quad & \neg \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1}) \wedge \mathit{append}(T, P'', L) \\ & \wedge\, \mathit{reapplicable}(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, T, N) \wedge \mathit{causes}(N, P'', \mathcal{G}^{-I}) \end{aligned} \tag{5.151}$$

We can apply Lemma 5.15 from the fact that there is not enough obstacle in $\mathcal{S}$ to hinder the action a_{j+1} from being taken. We learn that there is a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(\mathcal{C}^{-I} \circ \mathcal{I}'^{-I}, a_{j+1})$. Hence, there is an SLDENF-derivation from (5.151) to

$$\leftarrow \mathit{append}(T, P'', L) \wedge \mathit{reapplicable}(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, T, N) \wedge \mathit{causes}(N, P'', \mathcal{G}^{-I}) \tag{5.152}$$

From the fact that the function *append* used in $\mathcal{Q}$ combines list $[a_{j+1} \mid [a_j, \ldots, a_n]]$ and $[a_{n+1}, \ldots, a_1]$ to obtain the plan $[a_{j+1}, \ldots, a_1]$, we can conclude that there is an SLDENF-resolution proof of $\leftarrow \mathit{append}(T, P'', L)$ with an answer substitution

$$\sigma'' = \{T \mapsto [A_j, \ldots, A_n], P'' \mapsto [A_{n-1}, \ldots, A_1], L \mapsto [A_j, \ldots, A_1]\}$$

Hence, there must exist the corresponding SLDENF-derivations from (5.152) to

$$\leftarrow \mathit{reapplicable}(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, [A_j, \ldots, A_n], N) \wedge \mathit{causes}(N, [A_{n-1}, \ldots, A_1], \mathcal{G}^{-I}) \tag{5.153}$$

where variable T and P'' are replaced by $[A_j, \ldots, A_n]$ and $[A_{n-1}, \ldots, A_1]$, respectively. We can apply Lemma 5.25 from the fact shown in (5.144). We learn that there is an SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(\mathcal{E}^{-I} \circ \mathcal{I}'^{-I}, [A_j, \ldots, A_n], N)$ with an answer substitution

$$\sigma''' = \{N \mapsto ((\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E})\!\downarrow, A_j \mapsto a_j, \ldots, A_n \mapsto a_n\}.$$

Hence, there must exist the corresponding SLDENF-derivations from (5.153) to

$$\leftarrow \mathit{causes}(((\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E})\!\downarrow, [A_{n-1}, \ldots, A_1], \mathcal{G}^{-I}) \tag{5.154}$$

where variable N is replaced by $((\mathcal{I} \mathbin{\dot{\setminus}} \mathcal{C}) \mathbin{\dot{\cup}} \mathcal{E})\!\downarrow$ in the process.

We can apply (I.H) to (5.154) because the length of plan $[a_{n-1}, \ldots, a_1]$ is smaller than j and the goal marking in (I.H) is the same as in (5.154). We learn that there is an SLDENF-resolution proof of (5.154) with an answer substitution

$$\{A_{n-1} \mapsto a_{n-1}, \ldots, A_1 \mapsto a_1\}.$$

Combining the derivations from (5.146) to (5.154) and the derivations of (5.154), we can conclude that there is an SLDENF-resolution proof of (5.146) with an answer substitution $\{P \mapsto [a_{j+1}, \ldots, a_1]\}$. □

Lemma 5.30. *If p is generated by an SLDENF-resolution proof of $\mathcal{AFC}\mathbb{R}_{\mathcal{Q}}$ then p is a solution for $\mathcal{Q}$.*

Proof. The lemma is an immediate consequence of the following claim

> If there is an SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ generating plan p, then p is a solution for $\mathcal{Q}$.

The claim is proven on the length j of the SLDENF-resolution proof.
I.B To show: If there is an SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ with length 1 generating plan p, then p is a solution for $\mathcal{Q}$.
If the length of the SLDENF-resolution proof is 1, then the atom occurring in

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{5.155}$$

and a variant

$$causes(S, [\,], S)$$

of (3.2) must be AC1-unifiable and only if $\mathcal{I} \doteq \mathcal{G}$ with an AC1-unifier of the form

$$\{S \mapsto \mathcal{I}^{-I}, P \mapsto [\,]\}.$$

The successful SLDENF-derivation yields the empty clause. Hence, the SLDENF-resolution proof of (5.155) generates an empty plan. Since $\mathcal{I} \doteq \mathcal{G}$, there is no need to apply any action. Therefore, $[\,]$ is a solution for $\mathcal{Q}$.

I.B If there is an SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}_j^{-I}, P_j, \mathcal{G}^{-I})$ with length j generating plan p_j, then p_j is a solution for $\mathcal{Q}_j = \langle \mathcal{I}_j, \mathcal{G}, \mathcal{A}, \mathcal{A}_{real} \rangle$.
I.C To show: If there is an SLDENF-resolution proof of $\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I})$ with length $j+1$ generating plan p, then p is a solution for $\mathcal{Q}$.
I.S Suppose there is an SLDENF-resolution proof of

$$\leftarrow causes(\mathcal{I}^{-I}, P, \mathcal{G}^{-I}) \tag{5.156}$$

with length $j+1$ generating plan $[a_k, \ldots, a_1]$, $1 \leq k \leq j$. The first atom occurring in (5.156) can only be AC1-unified with the head of a new variant

$$\begin{aligned} causes(I, [A \mid L], G) \leftarrow \quad & applicable(I, A, S) \wedge append(T, P', L) \\ & \wedge\ reapplicable(S, T, N) \wedge causes(N, P', G) \end{aligned}$$

of (5.13) with an AC1-unifier of the form

$$\{I \mapsto \mathcal{I}^{-I}, G \mapsto \mathcal{G}^{-I}, P \mapsto [A \mid L]\}.$$

The successful derivation of (5.156) yields

$$\begin{aligned} \leftarrow \quad & \mathit{applicable}(\mathcal{I}^{-I}, A, S) \wedge \mathit{append}(T, P', L) \\ & \wedge\ \mathit{reapplicable}(S, T, N) \wedge \mathit{causes}(N, P', \mathcal{G}^{-I}) \end{aligned} \tag{5.157}$$

The first atom occurring in (5.157) can only be AC1-unified with the head of a new variant

$$\begin{aligned} \mathit{applicable}(C \circ S', A, E \circ S') \leftarrow \quad & \mathit{action}(C, A, E) \wedge \mathit{precon}(P'', A) \\ & \wedge\ \mathit{fulfil}(C \circ S', P'') \wedge \neg \mathit{hinder}(C \circ S', A) \end{aligned}$$

of (5.12) with an AC1-unifier of the form

$$\{C \mapsto c, S' \mapsto s, S \mapsto E \circ s\}$$

where $\mathcal{I}^{-I} = c \circ s$. The successful derivation of (5.157) yields

$$\begin{aligned} \leftarrow \quad & \mathit{action}(c, A, E) \wedge \mathit{precon}(P'', A) \wedge \mathit{fulfil}(c \circ s, P'') \\ & \wedge\ \neg \mathit{hinder}(c \circ s, A) \wedge \mathit{append}(T, P', L) \\ & \wedge\ \mathit{reapplicable}(e \circ s, T, N) \wedge \mathit{causes}(N, P', \mathcal{G}^{-I}) \end{aligned} \tag{5.158}$$

there must exist a clause of the form

$$\mathit{action}(c, a_k, e) \tag{5.159}$$

which can be AC1-unified with the first atom occurring in (5.158). The successful derivation of (5.158) yields

$$\begin{aligned} \leftarrow \quad & \mathit{precon}(P'', a_k) \wedge \mathit{fulfil}(c \circ s, P'') \\ & \wedge\ \neg \mathit{hinder}(c \circ s, a_k) \wedge \mathit{append}(T, P', L) \\ & \wedge\ \mathit{reapplicable}(e \circ s, T, N) \wedge \mathit{causes}(N, P', \mathcal{G}^{-I}) \end{aligned} \tag{5.160}$$

with an AC1-unifier of the form

$$\{A \mapsto a_k, E \mapsto e\}.$$

there must exist a clause of the form

$$\mathit{precon}(p, a_k) \tag{5.161}$$

which can be AC1-unified with the first atom occurring in (5.160). The successful derivation of (5.160) yields

$$\begin{array}{rl}\leftarrow & \mathit{fulfil}(c \circ s, p) \wedge \neg \mathit{hinder}(c \circ s, a_k) \wedge \mathit{append}(T, P', L) \\ & \wedge\, \mathit{reapplicable}(e \circ s, T, N) \wedge \mathit{causes}(N, P', \mathcal{G}^{-I})\end{array} \tag{5.162}$$

with an AC1-unifier of the form

$$\{P'' \mapsto p\}.$$

There must exist an SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(c \circ s, p)$ with an answer substitution $\emptyset$. Hence, we can find the corresponding derivations from (5.162) to

$$\begin{array}{rl}\leftarrow & \neg \mathit{hinder}(c \circ s, a_k) \wedge \mathit{append}(T, P', L) \\ & \wedge\, \mathit{reapplicable}(e \circ s, T, N) \wedge \mathit{causes}(N, P', \mathcal{G}^{-I})\end{array} \tag{5.163}$$

There must exist a finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(c \circ s, a_k)$. Hence, there is an SLDENF-derivation from (5.163) to

$$\leftarrow \mathit{append}(T, P', L) \wedge \mathit{reapplicable}(e \circ s, T, N) \wedge \mathit{causes}(N, P', G) \tag{5.164}$$

There must exist an SLDENF-resolution proof of $\leftarrow \mathit{append}(T, P', L)$ with an answer substitution

$$\{T \mapsto [A_{k-1}, \ldots, A_n], P' \mapsto [A_{n-1}, \ldots, A_1], L \mapsto [A_{k-1}, \ldots, A_1]\}$$

Hence, we can find the corresponding derivations from (5.164) to

$$\leftarrow \mathit{reapplicable}(e \circ s, [A_{k-1}, \ldots, A_n], N) \wedge \mathit{causes}(N, [A_{n-1}, \ldots, A_1], \mathcal{G}^{-I}) \tag{5.165}$$

where variable T and P' are replaced by $[A_{k-1}, \ldots, A_n]$ and $[A_{n-1}, \ldots, A_1]$, respectively. There must exist an SLDENF-resolution proof of $\leftarrow \mathit{reapplicable}(e \circ s, [A_{k-1}, \ldots, A_n], N)$ with an answer substitution

$$\sigma = \{N \mapsto n, A_{k-1} \mapsto a_{k-1}, \ldots, A_n \mapsto a_n\}.$$

Hence, we can find the corresponding derivations from (5.165) to

$$\leftarrow \mathit{causes}(n, [A_{n-1}, \ldots, A_1], \mathcal{G}^{-I}) \tag{5.166}$$

where variable N is replaced by ground fluent term n.

From (5.159) and (5.161), we may conclude that there is a concrete action a_k of the form

$$a_k : c^I \overset{p^I,\mathcal{O}}{\Longrightarrow} e^I.$$

The derivation from (5.158) to (5.160) shows that $c^I \dot{\subseteq} \mathcal{I}$. We can apply Lemma 5.19 due to the existence of SLDENF-resolution proof of $\leftarrow \mathit{fulfil}(c \circ s, p)$. We learn that the initial state $\mathcal{I}$ fulfils the preconditions p^I of the concrete action a_k. We can apply Lemma 5.15 due to the existence of the finitely failed SLDENF-tree of $\leftarrow \mathit{hinder}(c \circ s, a_k)$. We learn that there is not enough obstacle in the initial state $\mathcal{I}$ to hinder the action a_k from being taken. Hence, we can conclude that the concrete action a_k is applicable in the initial state $\mathcal{I}$. The application of a_k to $\mathcal{I}$ leads to $e^I \dot{\cup} s^I$, i.e.,

$$\mathcal{I} \xrightarrow{a_k} e^I \dot{\cup} s^I \tag{5.167}$$

We can apply Lemma 5.22 due to the existence of the SLDENF-resolution of
$\leftarrow \mathit{reapplicable}(e \mapsto s, [A_{k-1}, \ldots, A_n], N)$ with the answer substitution σ. We learn that a sequence $[a_{k-1}, \ldots, a_n]$ of theory actions applied to $e^I \dot{\cup} s^I$ transforms $e^I \dot{\cup} s^I$ to a closed state n^I, i.e.,

$$e^I \dot{\cup} s^I \xrightarrow{[a_{k-1},\ldots,a_n]} n^I \tag{5.168}$$

We can apply (I.H) to (5.166) since the SLDENF-refutation has length $j - ((k-n)+1)$ where $(k-n)+1$ is the length of the SLDENF-resolution of
$\leftarrow \mathit{reapplicable}(e \mapsto s, [A_{k-1}, \ldots, A_n], N)$. We learn a plan $[a_{n-1}, \ldots, a_1]$ applied to n^I to reach the goal state $\mathcal{G}$, i.e.,

$$n^I \xrightarrow{[a_{n-1},\ldots,a_1]} \mathcal{G} \tag{5.169}$$

Combining (5.167), (5.168) and (5.169), we obtain a plan $[a_k, \ldots, a_1]$ such that

$$\mathcal{I} \xrightarrow{[a_k,\ldots,a_1]} \mathcal{G}$$

Because the plan $[a_k, \ldots, a_1]$ transforms the initial state into the goal state, the claim is true. □

Chapter 6

Conclusion and Discussion

In this thesis, we have elaborated the relationship between Petri networks and the fluent calculus including some of Petri networks and the fluent calculus extensions in solving conjunctive planning problems and their variations. The simple fluent calculus is capable of solving simple conjunctive planning problems as presented in [5]. Furthermore, we have shown that ordinary Petri nets are capable as well. Ordinary Petri nets and the simple fluent calculus are proven to be equivalent. Whereas in simple fluent calculus, SLDE-resolution proofs are used to find solutions for simple conjunctive planning problems, in Petri nets finding the correspondence firing sequences is the way to find solutions.

Two modifications of conjunctive planning problems were introduced: advanced conjunctive planning problems and advanced conjunctive planning problems with real values. In advanced conjunctive planning problems, two new features for actions are given. Preconditions are a set of fluents which are only tested when an action is executed but are not consumed. Obstacles are a set of simple fluents preventing an action from being executed even if its conditions are satisfied. We have shown that advanced conjunctive planning problems can be represented and solved elegantly by advanced Petri nets and the advanced fluent calculus. Advanced Petri nets are ordinary Petri nets equipped with inhibitor arcs to represent obstacles and test arcs to represent preconditions. In the advanced fluent calculus several new rules are added to catch up with advanced conjunctive planning problems. Furthermore, SLDENF-resolution proofs instead of SLDE-resolution proofs have to be chosen caused by the introduction of negation in the body of the rules. We have shown that these two extensions are equivalent in solving advanced conjunctive planning problems.

The final modification of conjunctive plannning problems was carried out by adding real-valued fluents into advanced conjunctive planning problems. Moreover, we introduced two kinds of actions: concrete actions and theory actions. Concrete actions are actions to deal with simple fluents with preconditions and obstacles attached, while theory actions are actions to deal with real-valued fluents. Several new rules and predicates are proposed to match with the new features in advanced conjunctive planning problem. SLDENF-resolution proofs are still chosen due to negations occurring in the bodies of the rules. Finally, we showed that the new fluent calculus is able to represent and solve the advanced conjunctive planning problems with real-valued information.

In [3], it was shown that Petri nets can be combined with Bayesian networks via real-valued information. Now, it is possible to replace Petri networks with fluent calculus to be combined with Bayesian networks. However, several considerations must be observed. First, the fact that there is no formal definition for Petri nets combining advanced Petri nets and Petri nets calculating mathematical expressions makes it harder to justify the equivalence between our fluent calculus and Barret's Petri nets. The possible justification is via examples given by Barret in his report. Secondly, Petri networks proposed in [3] includes timed Petri nets. Hence, to match completely with this kind of Petri nets, a time feature must be added to and thoroughly investigated in the fluent calculus.

In this work, SLDE- and SLDENF-resolution proofs are used to compute answer substitutions in the fluent calculus. We would like to see the corresponding fixpoint characterization of the fluent calculus in the next investigation. With fixpoint characterization, a combination between the human reasoning approach proposed by Hölldobler et al in [24, 25] and reasoning about actions and causality in the fluent calculus can be built.

Bibliography

[1] Michael Thielscher. The concurrent, continuous fluent calculus. *Studia Logica*, 67(3):315–331, 2001.

[2] Steffen Hölldobler and Michael Thielscher. Computing change and specificity with equational logic programs. *Annals of Mathematics and Artificial Intelligence*, 14(1):99–133, 1995.

[3] Leon Rubin Barrett. *An architecture for structured, concurrent, real-time action.* PhD thesis, University of California, Berkeley, CA, USA, 2010.

[4] Gerd Große, Steffen Hölldobler, Josef Schneeberger, Ute Cornelia Sigmund, and Michael Thielscher. Equational logic programming actions, and change. In *Joint International Conference and Symposium on Logic Programming*, pages 177–191, 1992.

[5] Gerd Große, Steffen Hölldobler, and Josef Schneeberger. Linear deductive planning. *Journal of Logic and Computation*, 6(2):233–262, 1996.

[6] Steffen Hölldobler and Josef Schneeberger. A new deductive approach to planning. *New Generation Computing*, 8(3):225–244, 1990.

[7] Michael Thielscher. Introduction to the fluent calculus. *Electronic Transactions on Artificial Intelligence (http://www. etaij. org)*, 3, 1998.

[8] CA Petri. Kommunikation mit automaten. schriften des iim 2, institut für instrumentelle mathematik, bonn, 1962. Technical report, English translation: Technical Report RADCTR-65-377, 1966.

[9] Wolfgang Reisig. *Petri nets: an introduction.* Springer-Verlag New York, Inc., 1985.

[10] Luca Bernardinello and Fiorella De Cindio. A survey of basic net models and modular net classes. In *Advances in Petri Nets 1992*, pages 304–351. Springer, 1992.

[11] Tadao Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[12] Jörg Desel and Wolfgang Reisig. Place/transition Petri nets. In *Lectures on Petri Nets I: Basic Models*, pages 122–173. Springer, 1998.

[13] John C. Shepherdson. SLDNF-resolution with equality. *Journal of Automated Reasoning*, 8(2):297–306, 1992.

[14] Steffen Hölldobler. Equational logic programming. In *Symposium on Logic Programming*, pages 335–346, 1987.

[15] Steffen Hölldobler, Josef Schneeberger, and Michael Thielscher. AC1-unification/matching in linear logic programming. In *Proceedings of Sixth International Workshop on Unification Schloss Dagstuhl, Germany*, page 49, 1992.

[16] Steffen Hölldobler. *Foundations of equational logic programming*, volume 353. Springer-Verlag Berlin-Heidelberg-New York, 1989.

[17] Keith L Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and data bases*, pages 293–322. Springer, 1978.

[18] Jean H Gallier and Stan Raatz. SLD-resolution methods for Horn clauses with equality based on e-unification. In *Symposium on Logic Programming*, volume 86, pages 168–179, 1986.

[19] Gordon Plotkin. Building-in equational theories. *Machine intelligence*, 7(73-90):433, 1972.

[20] Michael Thielscher. On the completeness of SLDENF-resolution. *Journal of Automated Reasoning*, 17(2):199–214, 1996.

[21] J.W. Lloyd. Foundations of logic programming. Springer-Verlag New York Inc., New York, NY, 1987.

[22] Hans-Peter Störr and Michael Thielscher. A new equational foundation for the fluent calculus. In J. Lloyd, editor, *International Conference on Computational Logic*, pages 733–746. Springer, 2000.

[23] René David and Hassane Alla. On hybrid Petri nets. *Discrete Event Dynamic Systems*, 11(1-2):9–40, 2001.

[24] Emmanuelle-Anna Dietz, Steffen Hölldobler, and Marco Ragni. A computational logic approach to the suppression task. In D. Peebles N. Miyake and Eds. Austin R. P. Cooper, editors, *Proceedings of the 34th annual conference of the cognitive science society*, pages 1500–1505. Cognitive Science Society, 2012.

[25] Emmanuelle-Anna Dietz, Steffen Hölldobler, and Marco Ragni. A computational logic approach to the abstract and the social case of the selection task. In *Proceedings of the 11th International Symposium on Logical Formalizations of Commonsense Reasoning, COMMONSENSE*, 2013. http://www.wv.inf.tu-dresden.de/Publications/2013/report-13-03.pdf.