A Reasoning Module for Long-Lived Cognitive Agents

by

Stavros Vassos

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Abstract

A Reasoning Module for Long-Lived Cognitive Agents

Stavros Vassos

Doctor of Philosophy

Graduate Department of Computer Science

University of Toronto

2009

In this thesis we study a reasoning module for agents that have cognitive abilities, such as memory, perception, action, and are expected to function autonomously for long periods of time. The module provides the ability to reason about action and change using the language of the situation calculus and variants of the basic action theories. The main focus of this thesis is on the logical problem of progressing an action theory.

First, we investigate the conjecture by Lin and Reiter that a practical first-order definition of progression is not appropriate for the general case. We show that Lin and Reiter were indeed correct in their intuitions by providing a proof for the conjecture, thus resolving the open question about the first-order definability of progression and justifying the need for a second-order definition.

Then we proceed to identify three cases where it is possible to obtain a first-order progression with the desired properties: i) we extend earlier work by Lin and Reiter and present a case where we restrict our attention to a practical class of queries that may only quantify over situations in a limited way; ii) we revisit the local-effect assumption of Liu and Levesque that requires that the effects of an action are fixed by the arguments of the action and show that in this case a first-order progression is suitable; iii) we investigate a way that the local-effect assumption can be relaxed and show that when the initial knowledge base is a database of possible closures and the effects of the actions are range-restricted then a first-order progression is also suitable under a just-in-time

assumption.

Finally, we examine a special case of the action theories with range-restricted effects and present an algorithm for computing a finite progression. We prove the correctness and the complexity of the algorithm, and show its application in a simple example that is inspired by video games.

Dedication

To my parents Elias and Marina

Στους γονείς μου Ηλία και Μαρίνα

Acknowledgements

Completing the requirements for the PhD degree in the University of Toronto is a long and lonely process, and it seems that it is only after it is over that one is able to realize how special this experience has been. What follows is an attempt to thank the people that made this memorable experience possible.

I should start by saying that I was very fortunate to have Professor Hector Levesque as my academic advisor. I am deeply grateful to Hector not only for his technical advice on the research problems we investigated for this thesis, but also for his continuous guidance and support as I begun shaping my intuitions on the field of artificial intelligence. Hector is an amazing teacher and I am more than happy that I had the opportunity investigate with him technical as well as philosophical matters about mathematical logic, human-level intelligence, and artificial intelligence.

I would also like to thank Professors Sheila McIlraith and Yves Lesperance, the other two members of my advisory committee, for their valuable feedback and support during my thesis work. Their comments, suggestions, and concerns helped me greatly in finding my way in the often obscure paths of research. Also, many thanks to my advisory committee and Michael Thielscher, my external examiner, for the care in reading my thesis and their helpful comments. In particular, I am especially thankful to Yves for his thorough reading of the thesis and the numerous detailed corrections and suggestions that helped me improve the quality of the final document.

I would like to thank Professor Gerhard Lakemeyer for being an important collaborator and supporter during my thesis work. The hours we spent over the whiteboard during his occasional visits to the University of Toronto influenced the technical results of this thesis to a great extend.

I would also like to thank Professor Stathis Zachos and Professor Pavlos Peppas who introduced me to the field of logic and artificial intelligence when I was an undergraduate student in the National Technical University of Athens. Furthermore, I want to thank

them and Professor Timos Sellis for encouraging me to pursue graduate studies far away from my home country.

It is no secret among people in academia that it is quite a difficult task to balance between trying to be a (successful) PhD student and having a (normal) life. I want to say a *big thanks* to Sotiris Droulias, Babis Samios, Niky Riga, and Penny Papargiropoulou for helping me keep my balance, each one in their own special way, even though they were not in Toronto. This thesis would not have been possible without their sincere love and support.

I would also like to thank Sebastian Sardina, Anastasia Bezerianos, George Katsirelos, Rozalia Christodoulopoulou, Stefanos Karterakis, and "The Candidates" Michael Mathioudakis, Sotirios Liaskos, Stratis Ioannides, and Andres Lagar Cavilla for the great time we had together in Toronto over board games, video editing sessions, music rehearsals, basketball games, and strategic plans for "Daltoween" parties, as well as all the Greek graduate students of 2002 who were always happy to help me get started in my first year in Toronto.

Last but not least, I would like to thank my parents Elias and Marina for always doing as much as they can to help me in every possible way. I am forever indebted to them for their understanding, endless patience, and encouragement when it was most required.

Contents

Chapter 1

Introduction

Knowledge representation and reasoning is the field of artificial intelligence that investigates the modeling and representation of various manifestations of knowledge and reasoning capabilities that are apparent in the behavior of humans. Within the field we are interested in providing the theoretical specifications for a particular knowledge or reasoning problem with respect to a formal system, as well as the necessary algorithmic procedures for effectively computing a solution to the problem. The desired goal is a clean theoretical framework that specifies the correctness of our procedures, and a practical implementation method so that our procedures can be used in a real system such as an embodied or software agent. The problem we examine in this thesis is described as follows.

The description of the problem

In this thesis we are interested in providing a reasoning module for agents that are *cognitive* in the sense that they have cognitive abilities such as memory, perception, action, problem solving, etc, and *long-lived* in the sense that they are expected to function autonomously for long periods of time scaling from few minutes to hours and days. The module we study accounts for a special kind of deliberation, namely *reasoning about*

action and change, and it is able to:

1 represent the current state of the world and its dynamics in a formal system;

2 answer queries about the current and possible future states of the world based on the representation;

3 update the representation correctly after the agent has performed an action in the world.

The module is intended to be used as part of the implementation of an agent in order to provide the decision making procedure of the agent with useful information about the current and possible future states of the world.

A motivating example

Consider a video game where the player is located in some dungeon and has to confront *non-player characters (NPCs)* such as hostile creatures, one of which is implemented as a cognitive agent equipped with a reasoning module as the one we described above. That is, there is a formal language that specifies the changing properties of the game-world, the available actions, and how they affect the game-world, and a simple interface to the module for asking queries about the current and future situations as well as informing the module about actions that the character performs.

The video game is implemented in the programming language C++ and each non-player character is an instance of the class NPC that (among other things) features a method Act() that is responsible for specifying the behavior of the character. At every time frame in the game-world the method Act() interacts with the game engine, i.e., the main method of the video game, as follows: it receives sensory data about the surroundings of the character and then specifies how the character should act based on the sensory

data and various other sources of information that may be available. In our case one such source of information is the reasoning module.

The method Act() typically uses information that is expressed as a set of rules that are activated when certain situations arise, such as the following:

> "If the sound of sword fighting is heard, then immediately take a battle stance";
>
> "If an unexpected attack occurs, then immediately turn toward the source of the attack".

The reasoning module we suggest can be used as another source of useful information that the method Act() may take into account in order to specify a more *proactive* behavior for the character. For example, in order to decide upon a course of action that will help the character go through the red door that is currently locked, the method Act() may perform queries to the reasoning module that refer to the current and future situations such as the following:

> "Is it true that the red door will be unlocked after the action β is performed?"
>
> "Is it true that there is no way to unlock the red door?".

Observe that the reasoning module may provide information that cannot be deduced by the sensory data that is available to the method Act() at every time frame. For instance, it may be the case that the character has no means of opening the red door because the only key that unlocks the door was destroyed when a bomb exploded in a previous time frame also hurting the character. In this case the sensory data may only imply that the red door is closed or if the character has unsuccessfully tried to open the red door that the door is locked.

The query answering functionality that we described corresponds to the second property that we listed in the specification of the reasoning module. In order to see why the third property is also important we need to examine in some more detail the way that

the queries are formed. First we assume that when the video game starts, that is, before the very first time frame has passed in the game-world, the reasoning module of the non-player character is initialized so that it accurately represents the available actions in the game-world and their effects, as well as the initial state of the game-world.

Suppose now that in the first time frame the method Act() processes various pieces of information and then informs the game-engine that the character performs the action α_1. Unless this information is passed on to the reasoning module, the internal representation of the game-world in the module still refers to the *initial state* and not the *current state* where the effects of α_1 may have possibly changed some of the properties of the game-world. This is not a major problem in the second time frame as the method Act() may perform a query about the current state using the fact that the action α_1 has already been performed. For instance the method Act() may perform the following query:

> "Is it true that the red door will be unlocked after the actions α_1, β are performed?",

which is intended to mean the following:

> "Is it true that the red door will be unlocked after the action β is performed in the current state of the game-world?".

Imagine now that the game evolves and the character has performed a hundred actions, which is normal for video games that last more than a few minutes. Then in order for the method Act() to perform the same query about the current state it needs to formalize the query with respect to the initial state and all the actions $\alpha_1, \ldots, \alpha_{100}$ that have been performed as follows:

> "Is it true that the red door will be unlocked after the actions $\alpha_1, \ldots, \alpha_{100}, \beta$ are performed?".

In practice this can be problematic as the action history becomes very large, which is often the case for long-lived agents like non-player characters in video games. The difficulty of this approach does not reside in that we need to construct large queries rather than in the way that the large queries are evaluated. The evaluation of such a query relies on processing the effects of every action that has been performed, either *regressing* the query to the initial state or *progressing* the relevant parts of the initial state to the current state. Unless special care is taken to the way each query is evaluated, the processing of the actions that have already been performed is repeated multiple times rendering the evaluation method impractical. The third property of the reasoning module we study in this thesis provides one way so that this problem can be avoided. The idea is that the internal representation is occasionally *permanently progressed*, i.e., updated, to reflect the current state.

Our approach

Our approach for providing a reasoning module with the properties that we listed in the beginning of the chapter is based on existing work in the *situation calculus* [McCarthy and Hayes, 1969] and the *basic action theories* of [Reiter, 2001]. The situation calculus is a predicate logic language that provides the vocabulary to formalize how properties of the world change under the effect of actions. A basic action theory is a special kind of logical theory built on the situation calculus vocabulary that consists mainly of two parts:

- a set of logical sentences of first-order predicate logic that represent the initial state of the world, which we will typically refer to as the *initial knowledge base*;
- a set of logical sentences of the first-order predicate logic and a limited form of the second-order predicate logic that represent the dynamics of the world, that is, how certain facts about the world change when actions are performed.

The first of the three properties of the module is realized by using the situation calculus vocabulary to construct a basic action theory in order to represent the current state of the world and its dynamics in a formal logical system. The other two properties correspond to finding an effective solution to two well-known problems in reasoning about action, namely the problem of performing *projection* [Reiter, 2001] and the problem of computing a *progression* of a knowledge base [Lin and Reiter, 1997]. We should note at this point that in the general case these two problems are far more complex than what the simple example of the previous section reveals. For instance, the initial knowledge base is essentially as expressive as a general first-order predicate logic theory which is semi-decidable and comes with many intractability results.

The problems of projection and progression have received attention in the context of the basic action theories and a lot of advancements have been made since they were first identified. As far as progression is concerned though there are still several aspects of the problem that remain unclear, including a question about the correct theoretical formalization of the problem. The main focus of this thesis is on the problem of progression, in particular toward the following two directions:

- resolve the open question about what is a correct formalization of progression and provide a clean logical specification for the reasoning module in the situation calculus;
- identify interesting cases that a practical solution to the problem of progression can be found and provide methods for computing the progression of a theory.

We now proceed to present the contributions of this thesis.

Technical results

In this thesis we present the following results with respect to the line of research we described in the previous section:

1. We investigate the conjecture by Lin and Reiter [1997] that a practical definition of progression based on the entailment relation of the first-order predicate logic is not correct in the general case. We show that Lin and Reiter were correct in their intuitions and give a proof for their conjecture, thus resolving the open question about the first-order definability of progression and justifying the need for a second-order definition.

2. Building on the result by Lin and Reiter [1997] that a form of first-order progression is always correct when we restrict our attention to queries that refer to a specific (future) situation only, we identify a more general class of queries that allows for some limited form of quantification over situations, and prove that a first-order progression is always correct with respect to this class as well.

3. We revisit the *local-effect* assumption of Liu and Levesque [2005] that requires that the effects of every action are specified exclusively by the action type and the arguments of the action term, and prove that under this restriction a first-order progression is always correct.

4. We further investigate the practicality of the local-effect assumption and present a method for computing a first-order progression that is correct provided that a slightly stronger assumption holds for the effects of the actions.

5. We investigate a way that the local-effect assumption can be relaxed so that a first-order progression is always correct for the more practical case where the effects of the actions are not necessarily specified by the action type and the arguments of the action term but may be specified also using information from the initial knowledge base. We show that when the initial knowledge base has a special syntactic structure and the effects of the actions are *range-restricted* then a first-order progression is always correct provided that a certain assumption holds about the information that is represented in the knowledge base.

6. We focus on a special case of the range-restricted theories and specify an algorithm for the core computational task that our progression method relies on. We prove the correctness and the complexity of the algorithm, and discuss the overall performance of the progression method.

We conclude the introduction with an outline for the rest of the document.

Thesis outline

The rest of the thesis is organized as follows. In Chapter 2 we review the relevant background literature. In Chapter 3 we lay the theoretical foundations for the reasoning module in the situation calculus. We give the preliminaries for the basic action theories, we study the logical problems that correspond to the functionality of the reasoning module, namely projection and progression, and we resolve an important open question about the definability of progression in first-order logic. In Chapter 4 we focus on the problem of progression and examine three cases where it is possible to obtain a first-order progression that is correct. First, we examine a case where we restrict our attention to a practical class of queries, and then we focus on two cases where the actions are restricted to local and range-restricted effects. In Chapter 5 we focus on a special case of the range-restricted theories and provide a method for computing a first-order progression. We prove the correctness of our method, discuss its complexity, and show its application in a simple example that is inspired from video games. Finally, in Chapter 6 we conclude with a summary of the thesis and a discussion on future work.

The main result of Chapter 3 that resolves the conjecture by Lin and Reiter and a preliminary version of Section 3.3 appear in [Vassos and Levesque, 2008]. The results of Chapter 4 have been published in [Vassos and Levesque, 2008; Vassos *et al.*, 2008; 2009] as follows: the main result of Section 4.1 about the first-order progression with respect to a practical class of queries and a preliminary version of the section appear

in [Vassos and Levesque, 2008]; the main result of Section 4.2 about the first-order progression of the local-effect theories appears in [Vassos *et al.*, 2008]; the main result of Section 4.3 about the first-order progression of the range-restricted theories appears in [Vassos *et al.*, 2009]. A preliminary version of the account of possible closures of Section 4.3 appears in [Vassos and Levesque, 2007; Sardina and Vassos, 2005]. Finally, a preliminary version of Chapter 5 appears in [Vassos *et al.*, 2009].

Chapter 2

Related Literature

In this chapter, we review the relevant background literature. We first give an overview of four logical approaches for knowledge representation and reasoning that account for reasoning about action and change, and illustrate how each approach handles the problems that correspond to the functionality of our module. Then, we review two popular design principles for building agents and the cognitive robotics approach that has been a major influence to our work.

2.1 Knowledge representation and reasoning

As far as the logical approaches for reasoning about action and change are concerned there have been a number of suggestions for the specifics of the representation and the logical dialect that should be used. Nonetheless, the main goal of these approaches is more or less common and amounts to providing a logical formalism that provides an effective solution to the following two problems:

1. the *frame* problem [McCarthy and Hayes, 1969];

2. the *projection* problem [Reiter, 2001].

The frame problem expresses the difficulty that arises in logic when one tries to represent the properties of the world that *change* due to the performance of an action, in which case it is also necessary to represent the fact that the rest of the properties of the world remain *unchanged*. Using an explicit representation for the so-called *non-effects* of actions that do not change is both unintuitive and problematic as it easily leads to a representation that is not practical. Various techniques have been suggested so that one is able to represent only the effects of the available actions and the non-effects are handled by a part of the logical formalism that is typically referred to as the *frame*.

Essentially, the frame problem captures the requirement that the dynamics of the world is represented in a formal system in a *concise* way. The projection problem then captures the requirement that the formal system is also correct in that it is able to successfully *predict* how the world looks like after some actions have occurred. Typically this is expressed with respect to the entailment relation of the logical language that is used.

A few other fundamental problems have also received attention such as the so-called *qualification* problem that refers to an effective representation of the executability of actions, and the so-called *ramification* problem that refers to successful formalization of the possible indirect effects of actions. Also, it is often assumed that the logical formalism is used to represent what an agent *knows* about the properties of the world instead of representing the actual state of the world. In this case *sensing* and *nondeterministic* actions become important, that is, actions that an agent may perform in order to receive information about the current state of the world, and actions that may result in the agent having less information about the state of the world respectively.

Note that the earlier work on logical approaches for reasoning about action and change typically focus more on the *representational* problem of having a concise logical theory that entails the correct theorems, and less on the *computational* problem of deciding the entailments of the theory. The more recent work then typically focuses on practical

classes of the logical theories such that the problem of projection has a tractable solution. To that end the problem of *progression*, which refers to updating the logical theory after an action occurs so that it is equivalent to the original theory with respect to predicting the future, has been traditionally considered as an mechanism for solving the projection problem.

We now proceed to review four logical formalisms for reasoning about action and change that are representative of the work that exists in the literature. For each of the approaches we shall illustrate how the frame, the projection, and the progression problem are handled. For our presentation we will often use examples with respect to a simple world that consists of a number of doors each of which can be opened or closed.

2.1.1 The situation calculus

The situation calculus [McCarthy and Hayes, 1969] is one of the most influential formalisms for representing dynamic worlds and reasoning about action and change. As presented in [Reiter, 2001] it is a special kind of a first-order predicate logic language with some limited second-order features.

In this language a *situation* represents a world history as a sequence of actions. The constant S_0 is used to denote the *initial situation* where no action has yet been performed in the world, and sequences of actions are built using the function *do* so that the term $do(a, s)$ denotes the situation resulting from performing the action a in the situation s. The changing properties of the world are represented as predicates whose truth value varies from situation to situation. These are called *fluents* and are denoted by predicate symbols that have a situation as their last argument. For example the fluent $IsOpen(x, s)$ may be used to represent that the door x is open in the situation s.

A logical theory of the situation calculus can then represent the state of the world in the initial situation using unrestricted first-order sentences that specify the truth value of the fluents in S_0, and the dynamics of the world using first-order sentences that relate

the truth value of $F(\vec{x}, s)$ and $F(\vec{x}, do(a, s))$ for every fluent symbol F. In particular the *basic action theories* [Reiter, 1991; 2001] provide a simple solution to the frame problem by using one *successor state axiom* for every fluent in the language.

The successor state axioms essentially formalize the intuitive assumption that the sufficient conditions for affecting the truth value of a fluent are also the necessary ones. For example the successor state axiom for the fluent $IsOpen(x, s)$ may be as follows:

$$IsOpen(x, do(a, s)) \equiv (a = open(x) \vee (IsOpen(x, s) \wedge a \neq close(x))).$$

According to this axiom the door x is open in the situation $do(a, s)$ iff either a is the action $open(x)$ that represents the opening of the door x or the door was open in the situation s and a is not the action $close(x)$ that represents the closing of the door x.

The language of the situation calculus can also be used to form sentences that express that a particular property holds in the world in a situation that lies in the future of S_0. This is formalized again using the fluents and the situations. For instance, the following sentence ϕ expresses that there is a door that is open after the action $open(door_1)$ occurs in the initial situation:

$$\exists x IsOpen(x, do(open(door_1), S_0).$$

In the context of a basic action theory $\mathcal{D}$ that represents the initial state of the world and its dynamics, the problem of projection then corresponds to deciding whether such a sentence ϕ is entailed by $\mathcal{D}$ in the regular sense of entailment in predicate logic.

Nonetheless, a basic action theory $\mathcal{D}$ also includes some necessary foundational axioms that formally characterize the legal situations, one of which is second-order. As a result, in the general case the projection problem corresponds to deciding whether a special kind of second-order theory entails a first-order sentence ϕ. As there is no complete inference system for the second-order logic it is then necessary to identify restrictions to the problem of projection under which a less general approach may be used. To that end

Pirri and Reiter [1999] show that when the sentence ϕ refers to only one situation in the future of S_0 then *regression*, a systematic way of reducing ϕ into an equivalent first-order formula about the initial situation, is a sound and complete way to solve the projection problem. Furthermore, Kelly and Pearce [2007] show that under certain conditions a sentence ϕ that refers to all possible future situations may also be treated in a similar way.

In the context of the basic action theories the problem of progression corresponds to updating the first-order part of the theory that represents the initial situation after an action occurs in the world, so that the new theory and the old theory are equivalent with respect to how they describe the future situations that come after the occurrence of the action. The fact that a basic action theory also contains a second-order axiom led Lin and Reiter [1997] to define the progressed theory based on second-order logic. Lin and Reiter also conjectured that a practical first-order definition of the progressed theory is incorrect in the general case. Liu and Levesque [2005] indeed adopted a weaker first-order definition that is logically sound but not complete, and presented a practical method for computing the progression of a special class of theories.

Finally, several variants of the basic action theories have been studied including extensions that account for knowledge and sensing, such as the work of De Giacomo and Levesque [1999b] on projection in the presence of limited incomplete information and sensing, the epistemic theories of Scherl and Levesque [2003], the epistemic fluents of Demolombe and Pozos Para [2000], the interval-valued fluents of Funge [1999], as well as extensions that account for nondeterministic actions, such as the work of Sardina on guarded action theories [2005] and the work of Lin [1996] on indeterminate actions.

2.1.2 The fluent calculus

The fluent calculus as presented by Thielscher [1999; 2001] is a novel version of an earlier logical formalism for reasoning about action and change by Hölldobler and Schneeberger

[1990]. The logical language and the theories that are presented by Thielscher are very similar in spirit to the language of the situation calculus and the basic action theories but are designed so that they are more compatible to a progression-based solution to projection.

Thielscher is motivated by the observation that when an agent uses a basic action theory to reason about the *current* situation, as the theory is not periodically updated, after functioning for a long period of time the history of actions becomes very long and a regression-based approach to projection becomes computationally inefficient. In order to account for this problem Thielscher first *reifies* the fluents of the situation calculus into first-order terms, and then defines a dual representation for the basic action theories based on the *state update axioms* that characterize how each action affects the state of the world so that the projection problem may be handled using progression. Before we review the details of the state update axioms we discuss a simple example.

Assume that the fluent $IsOpen(x)$ (without a situation argument) is used to represent that the door x is open. Then the following formula may be used to represent that the door x is open in the situation s:

$$Holds(IsOpen(x), s),$$

where the predicate $Holds(f, s)$ is a special predicate that is included in the language and is used to represent that the reified fluent atom f holds in the situation s. We can then construct axioms that relate the truth value of $Holds(f, s)$ and $Holds(f, do(A(\vec{x}), s))$ for a specific type of actions of the form $A(\vec{x})$. For instance, an axiom of this form for the action type $push(x)$ may be as follows:

$$Holds(f, do(open(x), s)) \equiv Holds(f, s) \vee f = IsOpen(x).$$

Observe that unlike the basic action theories, here only one axiom is needed to specify

what holds in the next situation. This is then a dual representation in the sense that in order to deal with the frame problem in the basic action theories we need to specify a successor state axiom for every fluent $F(\vec{x}, s)$ in the language, while here need to specify an axiom of this form for every action type $A(\vec{x})$ in the language.

However, this simple solution requires that there is complete information in the initial situation in order to be logically correct, which is a very strong assumption. For this reason Thielscher goes further and reifies the conjunction of fluent atoms and represents a state as a possibly incomplete list of conjuncted atoms. The logical conjunction is replaced by the binary function $\circ$ and a state is represented as a first-order term obtained by $\circ$-composition of other terms, each of which represents a fluent atom. For example, the following formula represents a state z with complete information where $door_1$ and $door_2$ are open and all the rest of the doors are closed:

$$z = IsOpen(door_1) \circ IsOpen(door_2).$$

A state with incomplete information can then be captured using variables that represent unknown sub-states. For instance, the following formula represents a state z where $door_1$ and $door_2$ are open but the rest of the fluent atoms are unknown:

$$z = IsOpen(door_1) \circ IsOpen(door_2) \circ z'.$$

The special function $State(s)$ is used to relate a situation s with the corresponding state in the previous sense, and $Holds(f, s)$ is defined as the following macro:

$$\exists z\ State(s) = f \circ z.$$

Similarly the binary functions $-$ and $+$ are defined as macros that specify the update of a state in a compact way. A *state update axiom* then specifies the direct effects of the

action type $A(\vec{x})$ in the situation s in terms of additions and deletions of fluent atoms to the state $State(s)$. For example, the state update axioms for the action types $open(x)$ and $close(x)$ may be as follows:

$$State(do(close(x), s)) = State(s) - IsOpen(x),$$

$$State(do(open(x), s)) = State(s) + IsOpen(x).$$

In the context of a fluent calculus theory $\mathcal{D}$ with axioms that represent the state of the world in the initial situation S_0, state update axioms that represent the dynamics of the world , and a set of foundational axioms that formally define the legal situations, the projection problem corresponds to deciding whether a sentence ϕ that refers to the future of $\mathcal{D}$ is entailed by $\mathcal{D}$. In the general case this corresponds to a similar problem of second-order entailment as in the situation calculus basic action theories. Nonetheless, with respect to the restricted projection problem where ϕ may only refer to one situation in the future of S_0, the fluent calculus theories offer an alternative for computing projection. The idea is that whenever an action occurs in the world there is exactly one state update axiom that applies which specifies the update that should be performed in the representation of the current state. In this way the projection with respect to a sentence ϕ that refers to the current situation S_c, which is in the future of S_0, can be handled efficiently by observing the updated representation instead of regressing to S_0.

There are a couple of subtle details about this approach. First, note that the state update axioms essentially provide the functionality for only *expressing the difference* between the old state and the new state. In the cases that the old state is a set of literals and as long as the update is a finite set of additions and deletions, then a (new) set of literals can indeed always be specified as the correct representation of the new state. In the general case that the old state may represent unrestricted incomplete information though, even if the update is a finite set of additions and deletions the representation of

the new state cannot be a list of facts about the new state rather than the representation of the old state plus a fluent calculus sentence that expresses the update. Observe that this is very similar in spirit to the regression-based projection. Moreover, observe that this account of first-order progression does not contradict with the intuitions of Lin and Reiter that the problem of progressing a basic action theory may not be defined in first-order logic. The observation of Lin and Reiter is exactly that in the general case the representation of the old state plus the sentence that expresses the update may not be definable as a set of first-order sentences that only refer to one (new) state.

Finally as far as other representational features are concerned, the fluent calculus theories are able to represent nondeterministic actions in a straightforward way by introducing disjunctions in the state update axioms. Furthermore, a few variants of the fluent calculus action theories have been studied including the work of Thielscher [2000] that provides a similar account for knowledge as the work of Scherl and Levesque [1993; 2003] in the situation calculus.

2.1.3 The event calculus

The event calculus [Kowalski and Sergot, 1986] is a logical framework for reasoning about action and change that is similar to the situation calculus but instead of situations and actions it is based on *time points* and *events*. Similar to the fluent calculus the fluent atoms are reified and several special predicates are included in language that are used to represent what fluent atoms hold at specific time points. Unlike both the situation and the fluent calculus, the event calculus relies on a *nonmonotonic* entailment relation in order to provide a solution to the frame problem.

In order to illustrate the details of this formalisms we consider the *simple event calculus* as presented in [Shanahan, 1999] that includes the following special predicates:

- $Initiates(a, f, t)$ is used to represent that the fluent atom f starts to hold after the action a at the time point t;

- $\mathit{Terminates}(a, f, t)$ is used to represent that the fluent atom f ceases to hold after the action a at the time point t;
- $\mathit{Initially}_P(f)$ is used to represent that the fluent atom f holds from the time point 0;
- $t_1 < t_2$ is used to represent that the time point t_1 is before the time point t_2;
- $\mathit{Happpens}(a, t)$ is used to represent that the action a occurs at the time point t;
- $\mathit{HoldsAt}(f, t)$ is used to represent that the fluent atom f holds at the time point t;
- $\mathit{Clipped}(t_1, f, t_2)$ is used to represent that the fluent atom f is terminated at some time point between the time points t_1 and t_2.

An event calculus theory then is a normal first-order theory that includes a series of foundational axioms that define the functionality of the special predicates, such as the following two that formally characterize $\mathit{HoldsAt}(f, t)$:

$$\mathit{Initially}_P(f) \wedge \neg \mathit{Clipped}(0, f, t) \supset \mathit{HoldsAt}(f, t),$$

$$\mathit{Happpens}(a, t_1) \wedge \mathit{Initiates}(a, f, t_1) \wedge t_1 < t_2 \wedge \neg \mathit{Clipped}(t_1, f, t_2) \supset \mathit{HoldsAt}(f, t_2)$$

These axioms essentially express that a fluent atom holds at the time point t if it held at time 0 and has not been terminated between the time points 0 and t, or if it was initiated at some time point before t and has not been terminated between then and t.

For example, consider a theory $\mathcal{D}$ that includes all the necessary foundational axioms and also the following sentences that represent that door_1 was open initially and that the action $\mathit{close}(x)$ results in the door x not being open:

$$\mathit{Initially}(\mathit{IsOpen}(\mathit{door}_1)),$$

$$\mathit{Terminates}(\mathit{close}(x), \mathit{IsOpen}(x), t).$$

Note that instead of specifying a set of successor state axioms for all the fluents in the language or a set of state update axioms for all the action types in the language, $\mathcal{D}$ only mentions a simple effect axiom that specifies the intuitive information that the action $close(x)$ results in the door x not being open. The fact that nothing else changes when this action occurs is a result of the *predicate minimization* property of the nonmonotonic entailment relation of the event calculus that is based on the notion of *circumscription* [McCarthy, 1980].

The special predicates of the event calculus are also used to form sentences that express that an action occurs at a specific time point, e.g., the next sentence ψ expresses that the action $close(door_1)$ occurs at the time point 5:

$$Happpens(close(door_1), 5),$$

and that a particular property holds in the world at a specific time point, e.g., the next sentence ϕ expresses that $door_1$ is closed at the time point 7:

$$\neg HoldsAt(IsOpen(door_1), 7).$$

In the context of a theory $\mathcal{D}$ of the event calculus that represents the initial state of the world and its dynamics, the problem of projection corresponds to augmenting the theory with the appropriate atoms that represent the occurrence of the actions in question at the specific time points, such as the sentence ψ above, and then use the nonmonotonic entailment relation of the event calculus to decide whether a sentence about the future, such as ϕ above, is entailed by $\mathcal{D}$.

So, the nonmonotonic entailment relation offers the ability to implicitly represent the frame in the semantics of the event calculus instead of relying on a stronger syntactic assumption for the axioms of the theory. Nonetheless, this also precludes the use of the theoretical and practical results in the literature about first-order theorem proving and

requires that the problem of entailment in the event calculus be treated by specialized theorem provers. Finally, we note that the problem of progression is typically not investigated in the event calculus, and that a similar approach is due to Sandewall [1995] who defines a logical framework for reasoning about action that is also based on a language of narratives and a nonmonotonic entailment relation.

2.1.4 The action description languages

The action description language $\mathcal{A}$ [Gelfond and Lifschitz, 1993] is a simple declarative propositional language for reasoning about action and change that is inspired by work in the field of logic programming using *answer-sets* [Gelfond and Lifschitz, 1991].

In this language the state of the world is represented as a set of propositional literals and the dynamics of the world is represented using the so-called *effect propositions* that describe the effect of an action on a particular literal when some preconditions hold. These have the following general form:

$$A \textbf{ causes } L \textbf{ if } P_1, \ldots, P_n,$$

where A is an action, L is a propositional literal and $P_1, \ldots, P_n$ are propositional atoms. For example, consider the propositions $IsOpen_1, IsOpen_2$, that represent that the door 1 and 2 is open, the actions A_1, A_2 that represent the actions of opening the door 1 and 2, the actions A_3, A_4 that represent the actions of closing the door 1 and 2, and the propositions P_1, P_2 that represent that the agent is near the door 1 and 2, respectively. The following effect propositions then may be used to capture the intuitive effect of the actions A_1, A_2, A_3, A_4:

$$A_1 \textbf{ causes } IsOpen_1 \textbf{ if } P_1,$$

$$A_2 \textbf{ causes } IsOpen_2 \textbf{ if } P_2.$$

$$A_3 \textbf{ causes } \neg IsOpen_1 \textbf{ if } P_1,$$

$$A_4 \textbf{ causes } \neg IsOpen_2 \textbf{ if } P_2.$$

In the context of the action description language $\mathcal{A}$ the projection problem is formalized using a *value proposition* of the following form:

$$L \textbf{ after } A_1, \ldots, A_m.$$

Intuitively this proposition is true iff the truth value of a propositional literal L is true after the occurrence of the actions $A_1, ..., A_n$. The interpretation of the value propositions is based on a form of *transitional semantics* that specify how each state evolves according to the effect propositions. Similar to the event calculus the semantics of $\mathcal{A}$ relies on a built-in *nonmonotonic* assumption so that the frame problem is solved. In particular, the semantics rely on the *default reasoning mechanism* of Reiter [1980] in order to formalize that the literals that are not forced by an effect proposition to get a particular truth value remain unaffected.

One important limitation of this approach is that it is essentially propositional. The only quantification allowed is the use of free variables in the effect propositions as a macro which corresponds to all the ground instances of the proposition with respect to all the objects in a finite domain. As is also the case with the event calculus, the nonmonotonic semantics precludes the use of normal propositional solvers for solving the projection problem. The semantics of $\mathcal{A}$ though are such that an action theory of $\mathcal{A}$ can be directly translated into an *extended logic program* [Gelfond and Lifschitz, 1991] for which solvers exist.

Several variants of the language $\mathcal{A}$ have been studied and a family of languages has been introduced [Baral and Gelfond, 2005] such as the language $\mathcal{L}_0$ that provides a situation-based syntax for specifying the domain description, $\mathcal{L}_1$ that extends $\mathcal{L}_0$ to

include situation-based value propositions which express hypothetical reasoning, $\mathcal{L}_2$ that handles concurrent actions, and $\mathcal{L}_3$ that handles nondeterministic actions. Finally, an epistemic variant is explored in [Son and Baral, 2001].

2.1.5 Other approaches

There are also other logical formalisms in the literature that account for reasoning about action and change that are not based on predicate logic. One line of research that should be noted is the work on modal logics. This includes approaches that are based on temporal logics as well as dynamic extensions of epistemic logics [Hintikka, 1962], such as the work of van Ditmarsch *et al.* on *dynamic epistemic logic* [2007b]. These formalisms are typically dealing with the propositional case and focus on regression, e.g., [van Ditmarsch *et al.*, 2007a].

2.2 Agent design principles

In this section we focus on three popular approaches for building robotic or software agents, namely planning agents, agent-oriented programming languages, and cognitive robotics. We go over the basic ideas behind each approach as an introduction to the work that is being done in this area.

2.2.1 Planning agents

A simple agent architecture that achieves a proactive behavior is one where the agent is required to find a *plan* for the fulfillment of a set of predefined *goals*. According to this approach the agent is equipped with a formal representation of the world and its dynamics, a set of goals that essentially specify the intended behavior of the agent, and a planning mechanism for computing a plan that will achieve the goals.

The problem of classical planning is to find a sequence of actions such that the corresponding projection problem ensures that the condition of the goal holds after the sequence of actions is executed. A plan need not necessarily be a linear sequence of actions though, but may be non-linear such as the conditional plans. Some early work on conditional planning includes work on universal plans [Schoppers, 1987], essentially a compact representation of every possible classical plan that achieves a certain goal. In this case the part of the universal plan that is actual executed depends on the state of the world at the execution time.

The idea of a universal plan can be seen as one of the first attempts to deal with the problem of planning in the presence of incomplete information and sensing. Levesque [1996] explores what would be a generic specification for the planning task in a domain that includes sensing actions, in a way that is neutral with respect to the choice of planning algorithm and the formalism for reasoning about action. Levesque suggests to reformulate the task of classical planning to the task of finding a *program* that achieves a goal that may contain conditionals or loops. A sequence of actions or a conditional plan is then a special case of these programs. Moreover, Levesque argues that we should look for a program that the agent *knows how* to execute it and suggests that we restrict our attention to programs built upon a simple but powerful programming language that ensures that the agent will always trivially know how to execute them. This is to be thought as an assembly language which includes only the basic constructs needed to produce non-linear iterative plans. Another approach toward planning for non-sequential plans is the work of Levesque [2005] on synthesizing plans with loops.

As far as practical planning systems are concerned, one of the earliest approaches and one of the most successful is the STRIPS planning system [Fikes and Nilsson, 1971]. In STRIPS the state of the world is represented as a regular database and the actions are represented as updates to the database in terms of add and delete lists. The simplicity and the practicality of this approach has inspired other approaches that rely on a similar

practical design. A similar approach is that of the PKS planning system [Petrick and Bacchus, 2002; 2004] that uses a set of four databases to represent what the agent knows about the state of the world and its dynamics. PKS is able to represent disjunctive information as well as sensing information that will be available at execution time. Based on this rich representation the PKS provides a tree-like conditional plan that specifies a plan for each possible outcome for the sensing actions that are relevant to the planning problem.

For a thorough overview of the work in automated planning the reader is referred to the book of Ghallab *et al.* [2004].

2.2.2 Agent-oriented programming

The *Belief-Desire-Intention* approach is based on ascribing mental qualities to agents such as beliefs, capabilities, choices, and commitments. Along these lines the *agent-oriented programming* paradigm supports a societal view of computation. According to Shoham [1993] agent-oriented programming languages provide a language for describing the mental attributes of an agent and a language for specifying how these attributes may result in action and a possible update to the mental state. Agents are usually defined in terms of beliefs, which correspond to the information the agent has about the world, desires, which correspond to the tasks the agent would like to achieve, and intentions, which correspond to desires that the agent has committed to achieve.

Among the agent-oriented programming languages in the literature there are approaches that are based on simple rules and correspond to a reactive approach to behavior. One such example is Agent-0, the first agent-oriented programming language that was introduced by Shoham. The PLACA programming language [Thomas, 1995] is an extension of Agent-0 that features some planning capabilities. Some more advanced languages are AgentSpeak(L) [Rao, 1996] and 3APL [Hindriks *et al.*, 1999].

For a thorough overview of the languages and platforms for multi-agent programming

the reader is referred to the book of Bordini *et al.* [2005].

2.2.3 Cognitive robotics

The *cognitive robotics* approach [Lesperance *et al.*, 1994; Levesque and Lakemeyer, 2007] is concerned with endowing robotic or software agents with higher-level cognitive abilities such as more sophisticated reasoning for task planning at run-time. According to this approach it is essential that the agent is equipped with the following two components:

1. a logical formalism for representing dynamic worlds similar to the ones we presented in Section 2.1;

2. a program of a high-level programming language that specifies the behavior of the agent using information from the logical formalism and the ability of the agent to reason about action and change.

To that end, a family of high-level programming languages for agents has been developed. We now proceed to discuss the main properties of two of these languages, Golog and Indigolog.

Golog [Levesque *et al.*, 1997] is similar to a regular imperative programming language except that the basic *statements* of the language correspond to *action execution* in an underlying basic action theory of the situation calculus, and the *expressions* of the language are formulas that are evaluated in the underlying theory by *projection*. Also, apart from the regular constructs like sequence, conditionals and loops, the language includes nondeterministic constructs such as nondeterministic choice between two actions, nondeterministic choice of action arguments, and nondeterministic iteration.

The intuition behind the development of Golog is that it aims for a middle ground solution to the problem of specifying the behavior of the agent. Instead of taking into account only the goal of the agent and then perform a search among all possible sequences of actions that might achieve the goal, a domain-dependent *high-level program* is specified

while $\exists x\ IsOpen(x)$
do $(\pi x, y)\ [(IsOpen(x) \wedge isInRoom(x, y))?\ ;$
$goto(x)\ ;$
$close(y)\ ],$
endWhile

Figure 2.1: A simple Golog program

so that it essentially *guides* the planning procedure to follow certain paths in trying to satisfy the goal. A high-level program can be variably non-deterministic and may be used as a sketch of a plan that gives strong clues or restrictions about the intended solution. This becomes very important because the search space may be drastically reduced and planning on complex domains can be performed with efficiency in such cases. Moreover, the classical planning approach can also be formulated as a special case of a Golog program and, similarly, a Golog program may be fully deterministic involving no searching at all.

For example Figure 2.1 shows a program in Golog that provides a high-level plan for achieving the simple goal of closing all the doors in a room. Note that $\pi\vec{x}\ a(\vec{x})$ denotes a non deterministic choice of arguments for the action $a(x)$, $\phi?$ denotes that the formula ϕ holds with respect to the underlying action theory and the action history so far, and $[a_1; a_2]$ denotes the execution of the sequence of actions a_1, a_2.

Note that the final plan to be executed by the agent is computed *offline*. So, the interpreter is required to look all the way until the last action of the program is processed before a single action can actually be executed in the world. In some cases this is not problematic as the success of the plan may depend strongly on the choices made in the very early steps of the program. Nonetheless, this is a serious problem when long-running programs are considered especially in the presence of incomplete information about the state of the world. Note that sensing actions may not help either as the information from any action may be received only after a plan is computed and the actions are executed

in the world.

In order to deal with this issue, De Giacomo and Levesque [1999a] propose a new *incremental way* of interpreting such high-level programs and a new construct for the language that extends the functionality of Golog. In particular they introduce IndiGolog [De Giacomo and Levesque, 1999a], an extension of Golog that works *online* and is based on the functionality of ConGolog [De Giacomo *et al.*, 1997] which accounts for concurrency.

For every step of a high-level program in IndiGolog the agent deliberates and commits to one or more actions that are actually executed in the world to account for the particular step of the program. Offline planning can still be invoked using a new *search operator* Σ. The statement $\Sigma\delta$ in a program forces the interpreter to search *offline* for a successful solution to the program δ before committing to any executing any actions. IndiGolog provides advanced functionality that is important especially in the presence of incomplete information and sensing actions that are intended to be executed online. It is then in the hands of the designer of the high-level program to exploit the language constructs so that offline planning and online execution of actions are combined appropriately.

Chapter 3

Theoretical foundations of the reasoning module

In this chapter we lay the theoretical foundations for the reasoning module we study in this thesis. We first present the *situation calculus* [McCarthy and Hayes, 1969], the logical language that we will be using, and the *basic action theories* [Reiter, 2001], a class of theories of this language that forms the basis of our representation. We give the formal definition of the reasoning problems that correspond to the functionality of the module we study in this thesis, namely *projection* [Reiter, 2001] and *progression* [Lin and Reiter, 1997], and we resolve an important open problem about the definability of progression in first-order logic.

3.1 Situation calculus

In this section we present the logical language of the situation calculus, a predicate logic language that provides the vocabulary to formalize how certain properties of the world change under the effect of actions.

3.1.1 Syntax and semantics

The language $\mathcal{L}$ of the situation calculus as presented by Reiter [2001] is a three-sorted first-order logic language with equality and some limited second-order features. The three sorts are the following: *action* for actions, *situation* for situations, and a catch-all sort *object* for everything else depending on the domain of application.

Similar to a normal one-sorted first-order language, $\mathcal{L}$ includes function and predicate symbols. In this case since there are three sorts, each of the symbols has a type that specifies the sorts for the arguments it takes. The situation calculus includes symbols only of certain types each of which has a special role in the representation of the world and its dynamics. One thing to note before moving to the formal details is that a situation is used to represent a world history as a sequence of actions, and the symbols that take arguments of sort *situation* are used to formalize the dynamics of the world.

The language of the situation calculus $\mathcal{L}$ includes the logical symbols $\neg, \wedge, \exists$, the symbol of equality $=$, and the following non-logical symbols:

- a countably infinite supply of variables for each of the three sorts, as well as a countably infinite supply of (second-order) predicate variables of all arities;
- a countably infinite number of constant symbols of sort *object*;
- for each $n \geq 1$, a finite number of *object function* symbols, or simply *function* symbols, of type $(action \cup object)^n \rightarrow object$;
- for each $n \geq 0$, a finite number of *action function* symbols of type $(action \cup object)^n \rightarrow action$;
- the special *situation function* symbol $do : action \times situation \rightarrow situation$ and the constant S_0;
- for each $n \geq 0$, a finite number of *predicate* symbols of type $(action \cup object)^n$;

- the special predicate symbols $Poss : action \times situation$ and $\sqsubset$: $situation \times situation$;

- for each $n \geq 0$, a finite number of *relational fluent* symbols of type $(action \cup object)^n \times situation$.

The terms of the language are defined inductively similarly to a normal one-sorted language but also respecting the type of each symbol with respect to the three different sorts. We adopt the following notations with subscripts and superscripts: α and a for terms and variables of sort *action*; σ and s for terms and variables of sort *situation*; t and x, y, z, w for terms and variables of sort *object*. Also, we will use A for action functions, F, G for relational fluents, and b, c, d, e for constants of sort *object*.

An action term or simply an *action* represents an atomic action that may be performed in the world. For example consider the action $move(x, y)$ that is intended to represent that item x is moved to location y. A situation term or simply a *situation* represents a world history as a sequence of actions. The constant S_0 is used to denote the *initial situation* where no actions have occurred. Sequences of actions are built using the function symbol do, such that $do(\alpha, \sigma)$ represents the successor situation resulting from performing action α in situation σ.

A *relational fluent* is a predicate whose last argument is a situation, and thus whose truth value can change from situation to situation. For example, $At(x, y, \sigma)$ is intended to represent that item x is at location y in situation σ. In order to simplify the analysis we have restricted the language $\mathcal{L}$ so that there are no functional fluent symbols in $\mathcal{L}$, that is, functions whose last argument is a situation. This is not a restriction on the expressiveness of $\mathcal{L}$ as functional fluents can be represented by relational fluents with a few extra axioms.

The normal predicates and functions that do not take arguments of sort *situation* are used to represent relations and functions that are *rigid* and remain the same for all situations. For example, $sqrt(x)$ is intended to be the function that returns the square

root of x, which is the same regardless of the actions that have been performed in the world.

Actions need not be executable in all situations, and the predicate $Poss(\alpha, \sigma)$ states that action α is executable in situation σ. For example, $Poss(move(x, y), \sigma)$ is intended to represent that the action $move(x, y)$ is possible in situation σ. Finally, the binary predicate symbol $\sqsubset$ provides an ordering on situations. The atom $\sigma \sqsubset \sigma'$ means that the action sequence σ' can be obtained from the sequence σ by performing one or more actions in σ. We will typically use the notation $\sigma \sqsubseteq \sigma'$ as a macro for $\sigma \sqsubset \sigma' \vee \sigma = \sigma'$.

The well-formed first-order formulas of $\mathcal{L}$ are defined inductively similarly to a normal one-sorted language but also respecting that each parameter has a unique sort. As far as the second-order formulas of $\mathcal{L}$ are concerned, only quantification over relations is allowed and the well-formed formulas are defined inductively similarly to a normal second-order language. The semantics of the situation calculus language is the standard model-theoretic Tarskian semantics. We assume that the reader is familiar with the notions of a *structure*, a *model*, *satisfaction* in a structure, and *entailment*. For the formal definitions the reader is referred to one of the standard textbooks for mathematical logic, such as [Enderton, 1972] and [Mendelson, 1997]. Finally, to avoid confusion we note that whenever we say that two formulas are logically equivalent we assume that the logical symbol $=$ is always interpreted as the true identity.

3.1.2 Restricting formulas with respect to situations

Often we need to restrict our attention to formulas in $\mathcal{L}$ that refer to a particular situation term σ. For example, as we will shortly see, the so-called initial knowledge base is a set of sentences in $\mathcal{L}$ that do not mention any situation terms except for S_0. For this purpose we define the formulas that are *uniform in* σ as follows.

Definition 3.1.1 (Lin and Reiter 1997, Reiter 2001). For any situation term σ, we define $\mathcal{L}_\sigma$ to be the subset of the well-formed formulas of $\mathcal{L}$ (both first-order and

second-order) that do not mention any other situation terms except for σ, do not mention *Poss*, and where σ is not used by any quantifier [Lin and Reiter, 1997]. When a formula $\phi(\sigma)$ is in $\mathcal{L}_\sigma$ we say that it is *uniform in* σ [Reiter, 2001]. ■

The following example illustrates the intuition behind Definition 3.1.1 for formulas that are first-order.

Example 3.1.2. Let $Open(x, s)$ be a fluent that represents that the object x is "open" in the situation s. Then the formula $Open(x, S_0)$ and the sentence $\exists x\, Open(x, S_0)$ are uniform in S_0, the formula $Open(x, do(\alpha, S_0))$ and the sentence $\exists x\, Open(x, do(\alpha, S_0))$ are uniform in $do(\alpha, S_0)$, while the formulas $\exists s(Open(x, s))$ and $Open(x, S_0) \wedge Open(x, do(\alpha, S_0))$ are not uniform in any situation term σ. Finally, the formulas $Open(x, s)$ and $\exists x\, Open(x, s)$ are both uniform in s. ■

We will also use notation similar to [Gabaldon, 2002] and [Reiter, 2001] to specify sequences of actions and situation terms that are *rooted* at some other situation term.

Definition 3.1.3 (Reiter 2001, Gabaldon 2002). Let σ be a situation term and δ be a (possibly empty) vector of action terms $\langle \alpha_1, \ldots, \alpha_n \rangle$. We use $do(\delta, \sigma)$ to denote the following situation:

$$do(\alpha_n, do(\alpha_{n-1}, \ldots do(\alpha_1, \sigma) \ldots)).$$

We say that a situation term κ is *rooted at* σ iff κ is syntactically the same term as $do(\delta, \sigma)$, for some vector of action terms δ. ■

The intuition is that a situation term κ is rooted at some other situation term σ iff κ can be obtained from σ by "adding" a sequence of actions using the function do.

Example 3.1.4. The situation terms $do(\alpha, S_0)$, $do(a, S_0)$, and $do(\alpha_2, do(\alpha_1, S_0))$ are all rooted at S_0. The latter is also rooted at $do(\alpha_1, S_0)$. Note that every situation term σ is always rooted at σ. For instance, the situation term $do(a, s)$ is rooted at s but also at $do(a, s)$. ■

We will also need to restrict our attention to formulas that only refer to some situation term σ and the possible futures of σ. For this purpose we introduce the next definition using the notion of rooted situation terms.

Definition 3.1.5. Let σ, κ be situation terms and ϕ a rectified[1] formula in $\mathcal{L}$. We say that κ *is in the future of* σ *in* ϕ[2] iff one of the following conditions holds:

- κ is σ, or
- κ is rooted at some situation term κ' that is in the future of σ in ϕ, or
- κ is a variable and $\forall\kappa(\kappa' \sqsubseteq \kappa \supset \beta)$ or $\exists\kappa(\kappa' \sqsubseteq \kappa \wedge \beta)$ is a sub-formula of ϕ, where κ' is a situation term that is in the future of σ in ϕ.

We say that the formula ϕ is *about the future of* σ iff the situation terms in ϕ that appear as arguments of *Poss* or some fluent or the equality predicate are all in the future of σ in ϕ. ■

As we will be focusing a lot on formulas about the future of $do(\alpha, S_0)$ for some action term α, for readability reasons we will typically use S_α to denote the situation term $do(\alpha, S_0)$. The intuition is that if a sentence is about the future of S_α then its truth depends only on S_α and situations that come after S_α. An example follows.

Example 3.1.6. Let $\phi(s)$ be a (first-order or second-order) formula uniform in s. Then the following sentence is about the future of S_α and expresses that $\phi(s)$ holds in all situations that are rooted at S_α:

$$\forall s(S_\alpha \sqsubseteq s \supset \phi(s)).$$

[1] A formula is rectified iff no variable occurs both bound and free, and all quantifiers in the formula refer to different variables.

[2] Strictly speaking we only care about future situations s that are *executable* in the sense of satisfying this formula: $\forall a.\forall s^*.do(a, s^*) \sqsubseteq s \supset Poss(a, s^*)$. A more refined definition would include this constraint.

For instance, let *gd* be an object constant that represents a green door in the world and $Open(x, s)$ a fluent that represents that object x is "open" in situation s. The following sentence expresses that after action α is performed in the initial situation, the green door will *always* remain open *in the future*:

$$\forall s(S_\alpha \sqsubseteq s \supset Open(gd, s)).$$

The third point in the previous definition specifies a way that sub-formulas about the future may be combined in an inductive way. For instance, the following sentence is about the future of S_α as well and expresses that for every situation s rooted at S_α there is a situation s' rooted at s such that $\phi(s')$ holds:

$$\forall s(S_\alpha \sqsubseteq s \supset \exists s'(s \sqsubseteq s' \wedge \phi(s'))).$$

Finally, note that neither of these three sentences we examined in this example are uniform in any situation term σ. ■

In some cases it will be useful to omit the situation argument from some formulas and restore it later. We will be using the following notation to do this when needed.

Definition 3.1.7. A *situation-suppressed* formula ϕ in $\mathcal{L}$ is one that does not mention any symbol that takes a situation argument except for the fluent symbols, does not quantify over situations, and has a special form such that all the situation arguments in the fluent atoms are suppressed. We use $\phi[\sigma]$ to denote the formula obtained from ϕ by restoring the situation argument σ into all the fluent atoms in ϕ. Similarly, for a set of situation-suppressed formulas Γ we use $\Gamma[\sigma]$ to denote the set $\{\phi[\sigma] \mid \phi \in \Gamma\}$. ■

Note that if ϕ is a situation-suppressed formula, then $\phi[\sigma]$ is uniform in σ. A simple example follows.

Example 3.1.8. Let ϕ be the formula $Open(x)$ and ψ be $\exists x Open(x)$. Then, ϕ and ψ are situation-suppressed formulas, $\phi[S_0]$ is the formula $Open(x, S_0)$ and ψ is the sentence $\exists x Open(x, S_0)$, and $\phi[S_0]$, $\psi[S_0]$ are uniform in S_0. ■

Now we move on to see the specifics of the situation calculus theories that we will be using to represent the world and its dynamics.

3.2 Basic action theories

The logical language of the situation calculus provides the vocabulary that is needed to represent how certain properties of the world change under the effect of actions: the changing properties are represented as fluents, which are conditioned on a situation argument, and the dynamics is represented using rules that specify how the truth value of the fluents changes from any situation s to $do(a, s)$, i.e., the situation after the action a has been performed.

This representation task is tricky and requires solving a few problems that have been examined extensively in the literature, such as the *qualification problem*, the *ramification problem*, and the *frame problem* [McCarthy and Hayes, 1969]. We will be dealing with situation calculus theories of a specific kind, the so-called *basic action theories* [Reiter, 2001], that provide an effective solution to the frame problem and a simple solution to the qualification problem that works for many practical scenarios. These theories consist mainly of two parts: i) a set of logical formulas that represent the initial state of the world and ii) a set of logical rules that represent how certain facts about the world change when actions are performed. Before moving to the formal definition of a basic action theory, note that for the sake of readability we will typically omit the leading universal quantifiers in the axioms of the theory.

Definition 3.2.1 (Reiter 2001). A *basic action theory* $\mathcal{D}$ is a theory of the situation

calculus language $\mathcal{L}$ of the form:[3]

$$\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_0 \cup \mathcal{D}_{fnd},$$

where each of the parts of $\mathcal{D}$ is as follows.

1. $\mathcal{D}_{ap}$ is a set of *action precondition axioms (APs)*, one for each action function symbol A, of the following form:

   $$Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s),$$

   where $\Pi_A(\vec{x}, s)$ is a first-order formula uniform in s that does not mention any free variable other than $\vec{x}, s$. The action precondition axiom defines the preconditions to the executability of an action in a given situation in terms of properties holding in that situation alone.

2. $\mathcal{D}_{ss}$ is a set of *successor state axioms (SSAs)*, one for each relational fluent symbol F, of the following form:

   $$F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a, s),$$

   where $\Phi_F(\vec{x}, a, s)$ is a first-order formula uniform in s that does not mention any free variable other than $\vec{x}, a, s$. The successor state axiom for the fluent symbol F characterizes the conditions under which F has a specific truth value for $\vec{x}$ in the situation $do(a, s)$ as a function of the situation s.

[3]We slightly deviate from the standard notation in [Reiter, 2001] as we use $\mathcal{D}_0$ instead of $\mathcal{D}_{S_0}$ and $\mathcal{D}_{fnd}$ instead of Σ. In the first case we use $\mathcal{D}_0$ in order to avoid using a symbol with a subscript (S_0) as a subscript of $\mathcal{D}$. In the second case we choose to use $\mathcal{D}_{fnd}$ for purposes of uniformity in the parts of $\mathcal{D}$. Finally, note that even though we do not use a different symbol like Reiter [2001], here it is also the case that $\mathcal{D}_{fnd}$ is identical in all basic action theories.

3. $\mathcal{D}_{una}$ is the set of unique-names axioms for actions: $A(\vec{x}) \neq A'(\vec{y})$, and $A(\vec{x}) = A(\vec{y}) \supset \vec{x}{=}\vec{y}$, for each pair of distinct action function symbols A and A'.

4. $\mathcal{D}_0$ is a set of first-order sentences uniform in S_0 that describe the state of the world in the initial situation when no action has been performed. We will typically refer to this set as the *initial knowledge base (KB).*

5. $\mathcal{D}_{fnd}$ is the following set of domain independent foundational axioms that formally define the space of situations and the ordering $\sqsubset$:

$$do(a, s) = do(a', s') \supset (a = a' \wedge s = s')$$

$$\forall P\big(P(S_0) \wedge \forall a \forall s\big(P(s) \supset P(do(a, s))\big) \supset \forall s P(s)\big)$$

$$\neg s \sqsubset S_0$$

$$s \sqsubset do(a, s') \equiv s \sqsubset s' \vee s = s'.$$

■

Note that $\mathcal{D}_{fnd}$ is the only place where a second-order axiom is used, namely the third axiom that quantifies over the second-order predicate variable P. The purpose of this axiom is to ensure that the domain of situations is the smallest set that includes the initial situation and situations that are built using the function do, thus ensuring that when we quantify over situations we only refer to situations that are reachable from S_0 by a finite number of applications of the function do.

The following example shows a very simple basic action theory that will be our running example for presenting the results of this chapter.

Example 3.2.2 (The simple doors domain). Let $\mathcal{L}$ be the situation calculus language that consists of the standard logical symbols and the symbols $Poss, do, S_0$, the fluent $Open(x, s)$, the action function $push(x)$, and the object constant gd. Let $\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_0 \cup \mathcal{D}_{fnd}$ be the basic action theory of the *simple doors domain*, where

each of the parts of $\mathcal{D}$ is as follows.

1. $\mathcal{D}_{ap}$ consists of the following sentence:

$$Poss(push(x), s) \equiv true.$$

2. $\mathcal{D}_{ss}$ consists of the following sentence:

$$Open(x, do(a, s)) \equiv (a = push(x) \vee Open(x, s)).$$

3. $\mathcal{D}_{una}$ consists of the following sentence:

$$push(x) = push(y) \supset x = y.$$

4. $\mathcal{D}_0$ consists of the following sentence:

$$\neg Open(gd, S_0).$$

5. $\mathcal{D}_{fnd}$ is as in Definition 3.2.1.

In the simple doors domain the object domain represents doors each of which might be open or closed. The fluent $Open(x, s)$ represents that the object x, i.e., the door x, is open in the situation s. The object constant gd is a name for one specific door in the object domain, namely the green door. The action function $push(x)$ represents the action of pushing the door x open.

The sentence in $\mathcal{D}_{ap}$ represents that the action of pushing a door, which is the only action available in the language $\mathcal{L}$ of the simple doors domain, can be performed successfully at every situation. The successor state axiom in $\mathcal{D}_{ss}$ represents the dynamics of the simple doors domain which is simply that a door x is open in the situation $do(a, s)$

iff it is the case that $push(x)$ was just performed or the door was open in the situation s. Finally, the initial knowledge base $\mathcal{D}_0$ represents that in the initial situation S_0 the green door is closed. ■

3.2.1 The problem of projection

A fundamental problem in reasoning about action and change is to determine whether or not some condition holds after a given sequence of actions has been performed. In other words, we start in the initial situation S_0, we perform a sequence of actions $\langle \alpha_1, \ldots, \alpha_n \rangle$ that takes us to a new situation S_n, and we wish to know if some condition holds in S_n. There are in fact two versions of this problem. The special case where the condition refers only to the situation S_n is called the (simple) *projection problem* [Reiter, 2001].

Definition 3.2.3 (Reiter 2001). Let $\mathcal{D}$ be a basic action theory, δ a vector of ground action terms $\langle \alpha_1, \ldots, \alpha_n \rangle$, and $\phi(s)$ a first-order formula uniform in s whose only free variable is s. The problem of determining whether $\phi(s)$ holds in the situation $do(\delta, S_0)$ is an instance of the *simple projection problem*. In logical terms it is the problem of determining whether the following holds:

$$\mathcal{D} \models \phi(do(\delta, S_0)).$$

■

So, the simple projection problem is the problem of determining whether a first-order sentence ϕ uniform in some ground situation term σ is entailed by the basic action theory $\mathcal{D}$. We will typically refer to σ as the *query* of the projection problem. The more general case is when ϕ is not restricted to be uniform in some ground situation term σ. We introduce the following definition:

Definition 3.2.4. Let $\mathcal{D}$ be a basic action theory, δ a vector of ground action terms $\langle \alpha_1, \ldots, \alpha_n \rangle$, and $\phi(s)$ a first-order formula that is about the future of s whose only free variable is s. The problem of determining whether $\phi(s)$ holds in the situation $do(\delta, S_0)$

is an instance of the *generalized projection problem*. In logical terms it is the problem of determining whether the following holds:

$$\mathcal{D} \models \phi(do(\delta, S_0)).$$

■

The next example illustrates the important difference between the simple and the generalized projection problem.

Example 3.2.5. Let $\mathcal{D}$ be the basic action theory of the simple doors domain as in Example 3.2.2. The problem of determining whether the green door will be open after the action $push(gd)$ is performed is an instance of the simple projection problem. In logical terms it is the problem of determining whether the following holds:

$$\mathcal{D} \models Open(gd, do(push(gd), S_0)),$$

which is true and follows from the successor state axiom for the fluent *Open*.

If we wanted to determine whether the door will *always remain* open after this action is performed then this is an instance of the generalized projection problem. In particular, we want to determine whether $Open(gd, s)$ holds in all situations s that are in the future of $do(push(gd), S_0)$. In logical terms it is the problem of determining whether the following holds:

$$\mathcal{D} \models \forall s(do(push(gd), S_0) \sqsubseteq s \supset Open(gd, s)),$$

which is true and follows from the successor state axiom for *Open* and the foundational axioms that define legal situations. The intuition is that after $push(gd)$ is performed the green door is open, and as there is no condition listed in the successor state axiom that may change the truth value of *Open* from true to false, the door will always remain open in every situation that is built using the symbol *do*.

Finally, note that for the simple projection problem we examine whether a sentence

that is uniform in $do(push(gd), S_0)$ is entailed by $\mathcal{D}$. This is not the case for the generalized projection problem as it is easy to verify that the sentence we examined is not uniform in any situation term σ as it quantifies over situations. ■

The ability to *predict* how the world will be after performing a sequence of actions is the basis for other more complex reasoning problems such as automated planning, scheduling, web-service composition, high-level program execution [Reiter, 1993]. In such settings the simple projection problem refers to determining whether some condition holds in a specific point in the future, while the generalized projection problem refers to questions of *achievability*, i.e., "Is there a way to open all the doors in this room?", and *invariants*, i.e., "Will the green door remain closed forever after this action is performed?".

We now move on to the problem of progression for basic action theories.

3.2.2 The problem of progression

The problem of projection is central for many reasoning tasks that are performed *offline*, e.g, planning problems, where the reasoning process is completed before any action is actually executed in the world. In the case where we need to reason about action and change *online*, that is, while some selected actions are actually performed in the world and change its state, then the problem of *progression* also becomes important. This is the problem of *updating* the initial knowledge base of the basic action theory so that it reflects the current state of the world instead of the initial state of the world. In other words, in order to do a one-step progression of the basic action theory $\mathcal{D}$ with respect to the ground action α we need to replace $\mathcal{D}_0$ in $\mathcal{D}$ by a suitable set $\mathcal{D}_\alpha$ of sentences so that the original theory $\mathcal{D}$ and the theory $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha$ are equivalent with respect to how they describe the situation S_α and the situations in the future of S_α [Lin and Reiter, 1997]. We define the notion of a *correct progression* based on this requirement as follows.

Definition 3.2.6 (Correct progression). Let $\mathcal{D}$ be a basic action theory, α a ground action term, and $\mathcal{D}_\alpha$[4] a set of (first-order or second-order) sentences such that the following conditions hold:

1. just as $\mathcal{D}_0$ is a set of sentences that are uniform in S_0, the sentences of the new knowledge base $\mathcal{D}_\alpha$ are uniform in S_α;

2. for every first-order formula ϕ about the future of S_α,

$$\mathcal{D} \models \phi \text{ iff } (\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha \models \phi.$$

Whenever $\mathcal{D}_\alpha$ satisfies these two conditions we say that $\mathcal{D}_\alpha$ is a *correct progression* of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. ■

In a seminal paper Lin and Reiter [1997] gave a model-theoretic definition for $\mathcal{D}_\alpha$ which essentially requires that the models of $\mathcal{D}_\alpha$ should be identical to the models of $\mathcal{D}$ as far as the situation S_α is concerned. In the same paper they were able to prove that if $\mathcal{D}_\alpha$ follows their definition then it is always a correct progression. Instead of the original model-theoretic definition of $\mathcal{D}_\alpha$ by Lin and Reiter, we will be using one that is based on second-order logic and a technical result from [Lin and Reiter, 1997]. The intuition behind our definition is as follows.

Let $F_1, \ldots, F_n$ be the fluent symbols of $\mathcal{L}$, $\mathcal{D}$ be a basic action theory with a finite $\mathcal{D}_0$, and α a ground action term. Also assume that the successor state axiom for F_i is of the form $F_i(\vec{x}, do(a, s)) \equiv \Phi_i(\vec{x}, a, s)$. We want to find a set $\mathcal{D}_\alpha$ that successfully describes the situation S_α. Observe that $\mathcal{D}$ already tells us what is known about the situation S_α: $\mathcal{D}_0$ tells what is known about S_0, and the successor state axioms tell us how each fluent

[4]Similar to what we did in Definition 3.2.1 we slightly deviate from the notation of Lin and Reiter [1997] and use $\mathcal{D}_\alpha$ instead of $\mathcal{D}_{S_\alpha}$ in order to avoid using a symbol with a subscript as a subscript of $\mathcal{D}$.

changes in going from S_0 to S_α. So in a sense, the set

$$\mathcal{D}_0 \cup \{ \bigwedge_{i=1}^{n} \forall \vec{x}.F_i(\vec{x}, S_\alpha) \equiv \Phi_i(\vec{x}, \alpha, S_0) \}$$

qualifies as the set $\mathcal{D}_\alpha$ we are looking for, except for the fact that it also includes what is known about S_0, therefore is not uniform in S_α. The progression we propose removes the dependency on S_0 by using second-order quantification over predicates in order to express the information about S_0, instead of using the original set $\mathcal{D}_0$ as is. The resulting sentence is then uniform in S_α. More precisely, we introduce the following notation that is similar to the so-called second-order *lifting* of Lin and Reiter [1997].

Definition 3.2.7. Let $F_1, \ldots, F_n$ be relational fluent symbols, and $Q_1, \ldots, Q_n$ be second-order (non-fluent) predicate variables. For any first-order formula ϕ in $\mathcal{L}$, let $\phi\langle \vec{F}:\vec{Q} \rangle$ be the formula that results from replacing any fluent atom $F_i(t_1, \ldots, t_n, \sigma)$ in ϕ, where σ is a situation term, by $Q_i(t_1, \ldots, t_n)$. ■

We use this notation to define a second-order sentence uniform in S_α that will be the basis of our definition for progression.

Definition 3.2.8. Let $F_1, \ldots, F_n$ be all the relational fluent symbols of $\mathcal{L}$ and $\mathcal{D}$ a basic action theory, where $\mathcal{D}_0$ is a finite set of first-order sentences, ϕ is the conjunction of the sentences in $\mathcal{D}_0$, and for all i, $1 \leq i \leq n$, the successor state axiom for F_i has the form $F_i(\vec{x}, do(a, s)) \equiv \Phi_i(\vec{x}, a, s)$. Let α be a ground action term, and $Q_1, \ldots, Q_n$ be predicate variables. Then, $Pro(\mathcal{D}, \alpha)$ is the following second-order sentence uniform in S_α:

$$\exists \vec{Q}.\ \phi\langle \vec{F}:\vec{Q} \rangle \ \wedge \ \bigwedge_{i=1}^{n} \forall \vec{x}.F_i(\vec{x}, S_\alpha) \equiv (\Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F}:\vec{Q} \rangle).$$

■

Now we are ready to give the precise definition of *strong progression*.

Definition 3.2.9 (Strong progression). Let $\mathcal{D}$ be a basic action theory with a finite $\mathcal{D}_0$, α a ground action term, and $\mathcal{D}_\alpha$ a set of sentences uniform in S_α. The set $\mathcal{D}_\alpha$ is a *strong progression* of $\mathcal{D}_0$ wrt α and $\mathcal{D}$ iff $\mathcal{D}_\alpha \cup \mathcal{D}_{una}$ is logically equivalent[5] to $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una}$. In this case we also say that $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha$ is a strong progression of $\mathcal{D}$ wrt α. ■

Note that $\mathcal{D}_\alpha$ is unique up to logical equivalence *assuming that actions have unique-names*. The reason why we don't use the simpler condition that $\mathcal{D}_\alpha$ be logically equivalent to $Pro(\mathcal{D}, \alpha)$, is that $\mathcal{D}_{una}$ is needed in order to get a correct characterization of the situation S_α but since it is already included in $\mathcal{D}$ we don't want to assume that $\mathcal{D}_{una}$ is also included in $\mathcal{D}_\alpha$. A detailed example follows that illustrates the intuition behind this definition.

Example 3.2.10. Let $\mathcal{D}$ be the basic action theory of the simple doors domain that we introduced in Example 3.2.2, and α be the ground action $push(gd)$, i.e., the action of pushing the green door open. First we construct the second-order sentence $Pro(\mathcal{D}, \alpha)$. Recall that $\mathcal{D}_0$ consists only of the sentence

$$\neg Open(gd, S_0),$$

and the successor state axiom for $Open$ is the following:

$$Open(x, do(a, s)) \equiv (a = push(x) \vee Open(x, s)).$$

The instantiation of the successor state axiom for the ground action α and the situation S_0 is then the following:

$$Open(x, S_\alpha) \equiv (\alpha = push(x) \vee Open(x, S_0)).$$

[5] As we mentioned earlier, whenever we say that two formulas are logically equivalent we assume that the logical symbol = is always interpreted as the true identity.

The sentence $Pro(\mathcal{D}, \alpha)$ uses a unary second-order predicate variable Q in order to express the information about S_0, and the instantiated successor state axiom in order to describe what holds in the resulting situation S_α. The trick is that any occurrence of $Open(x, S_0)$ in the instantiated successor state axiom is replaced by $Q(x)$. $Pro(\mathcal{D}, \alpha)$ is the following second-order sentence:

$$\exists Q.\neg Q(gd) \wedge \forall x.Open(x, S_\alpha) \equiv (\alpha = push(x) \vee Q(x)).$$

Using the fact that α is the ground action $push(gd)$ and the uniqueness of names for actions, it is not too difficult to show that $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una}$ is logically equivalent to the first-order set $\{Open(gd, S_\alpha)\} \cup \mathcal{D}_{una}$. Therefore, the set $\mathcal{D}_\alpha$ that consists of the sentence

$$Open(gd, S_\alpha)$$

is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$.

In order to see why the set $\mathcal{D}_{una}$ is necessary in the definition of a strong progression of $\mathcal{D}_0$, consider the case where everything is the same as the example we just saw, except for $\mathcal{D}_0$ that consists of the following sentence:

$$\forall x \neg Open(x, S_0).$$

This sentence says that all doors are closed in S_0. Similar to the previous case $Pro(\mathcal{D}, \alpha)$ is the second-order sentence

$$\exists Q.\, \forall x \neg Q(x) \wedge \forall x.Open(x, S_\alpha) \equiv (\alpha = push(x) \vee Q(x)).$$

Let $\mathcal{D}_\alpha$ be the set that consists of the following sentence:

$$\forall x(Open(x, S_\alpha) \equiv x = gd).$$

This is the intended progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$, as it expresses that all doors remain closed except for the green door. Observe though that $Pro(\mathcal{D}, \alpha)$ is *not* logically equivalent to $\mathcal{D}_\alpha$. The reason is that $Pro(\mathcal{D}, \alpha)$ does not entail that the rest of the atoms other than $Open(gd, S_0)$ remain false. This is because there is a model where $push(gd) = push(x)$ is satisfied for some $x \neq gd$ and so $Open(x, S_\alpha)$ is true in this model. Nonetheless, it is not too difficult to show that $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una}$ and $\mathcal{D}_\alpha \cup \mathcal{D}_{una}$ are indeed logically equivalent, thus $\mathcal{D}_\alpha$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. ■

The fact that a strong progression of $\mathcal{D}$ is always a correct progression follows from the results in [Lin and Reiter, 1997]: the authors showed that when $\mathcal{D}_\alpha$ follows their model-theoretic definition it always satisfies the two conditions of Definition 3.2.6, and that a definition based on $Pro(\mathcal{D}, \alpha)$ is equivalent to the model-theoretic one whenever $\mathcal{D}_0$ is finite. Whenever $\mathcal{D}_0$ is infinite we can always revert to the model-theoretic definition but since we are interested in practical cases where $\mathcal{D}$ is a finite theory, Definition 3.2.9 will be sufficient. For a discussion about the differences between the definitions of progression as they appear in [Lin and Reiter, 1997], [Reiter, 2001], and in this thesis, as well as the results that prove that a strong progression is a correct progression, the reader is referred to Section A in the appendix. The following theorem states the important property of a strong progression that essentially implies that it is always a correct progression.

Theorem 3.2.11 (Lin and Reiter 1997). *Let $\mathcal{D}$ be a basic action theory with a finite $\mathcal{D}_0$, α a ground action term, and $\mathcal{D}_\alpha$ a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. Then, for every sentence ϕ uniform in S_α (first-order or second-order),*

$$\mathcal{D} \models \phi \text{ iff } \mathcal{D}_\alpha \cup \mathcal{D}_{una} \models \phi,$$

and for every sentence ϕ about the future of S_α (first-order or second-order),

$$\mathcal{D} \models \phi \text{ iff } (\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha \models \phi.$$ ■

Although the definition of strong progression is formulated in second-order logic, like Lin and Reiter we are concerned with progressions that are first-order. It is not hard to see that in simple cases that are used in practice (such as STRIPS planning [Fikes and Nilsson, 1971]), this definition does the right thing and remains within first-order logic. For instance consider Example 3.2.10 where the $\mathcal{D}_\alpha$ we specify is indeed first-order. However, as it was first shown by Lin and Reiter [1997], there are cases of basic action theories where no first-order strong progression exists. We postpone the discussion about the first-order definability of a correct progression until Section 3.3.

We close this section with a remark about the requirement that $\mathcal{D}_\alpha$ should be uniform in S_α. Strictly speaking the new theory $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha$ is not a basic action theory according to Definition 3.2.1 because the updated knowledge base $\mathcal{D}_\alpha$ is not uniform in S_0. Nonetheless, getting a basic action theory in the formal sense is a simple matter of replacing S_α by S_0 in all the sentences in $\mathcal{D}_\alpha$. The reason why $\mathcal{D}_\alpha$ is typically assumed in the literature to be uniform in S_α is that this simplifies the analysis, as we don't need to change our "S_0 point of reference" when we examine the original theory and the progressed theory.

3.2.3 Reasoning about action without the foundational axioms

In this section we review some important properties of the basic action theories with respect to entailment and satisfaction, and also prove two lemmas that are needed for some of the results of this and the next chapter.

The intuition is that the foundational axioms are only needed when a formula quantifies over the situation space, otherwise they can be omitted. This is an important property as the set $\mathcal{D}_{fnd}$ of the foundational axioms is the only place where a second-order axiom is used, therefore $\mathcal{D} - \mathcal{D}_{fnd}$ is a purely first-order theory. We first review the following lemma that appears in [Lin and Reiter, 1997] and extend it slightly for our purposes.

Lemma 3.2.12 (Lin and Reiter 1997). *Let $\mathcal{D}$ be a basic action theory. Given any model M^- of $\mathcal{D} - \mathcal{D}_{fnd}$, there is a model of $\mathcal{D}$ such that the following hold:*

1. *M^- and M have the same domains for sorts action and object, and interpret all situation independent predicates and functions the same;*

2. *for any ground situation term σ, any fluent F, and any variable assignment μ, $M, \mu \models F(\vec{x}, \sigma)$ iff $M^-, \mu \models F(\vec{x}, \sigma)$.* ■

This can be generalized to arbitrary formulas that are uniform in some ground situation term as follows.

Lemma 3.2.13. *Let $\mathcal{D}$ be a basic action theory. Given any model M^- of $\mathcal{D} - \mathcal{D}_{fnd}$, there is a model of $\mathcal{D}$ such that the following hold:*

1. *M^- and M have the same domains for sorts action and object, and interpret all situation independent predicates and functions the same;*

2. *for any ground situation term σ, any first-order formula ϕ uniform in σ, and any variable assignment μ, $M, \mu \models \phi$ iff $M^-, \mu \models \phi$.* ■

Proof. By Lemma 3.2.12 and induction on the construction of first-order formulas ϕ that are uniform in some situation term σ. □

This implies that when ϕ is uniform in some situation term σ, the set $\mathcal{D}_{fnd}$ is not needed to decide whether ϕ is entailed by the basic action theory.

Lemma 3.2.14. *Let $\mathcal{D}$ be a basic action theory. Then, for any ground situation term σ and any first-order formula ϕ uniform in σ, $\mathcal{D} \models \phi$ iff $(\mathcal{D} - \mathcal{D}_{fnd}) \models \phi$.* ■

In fact we can get an even stronger result using an important computational mechanism that is called *regression* [Reiter, 2001]. Essentially, it is a transformation that applies to a query formula in $\mathcal{L}$ such that the resulting formula is an equivalent way to

express the query and is also uniform in the initial situation S_0. We define the *regressable* sentences as follows.

Definition 3.2.15 (Reiter 2001). A formula ϕ in $\mathcal{L}$ is *regressable* iff the following conditions hold:

1. every situation term in ϕ is rooted at S_0;
2. for every atom of the form $Poss(\alpha, \sigma)$ that appears in ϕ, α has the form $A(\vec{t})$, where A is some n-ary action function symbol;
3. ϕ does not quantify over situations;
4. ϕ does not mention the predicate symbol $\sqsubset$ and it does not mention any equality atom built on situation terms. ■

Pirri and Reiter introduced a regression operator $\mathcal{R}$ that eliminates *Poss* atoms in favor of their definitions as given by $\mathcal{D}_{ap}$, and replaces fluent atoms about $do(\alpha, \sigma)$ by logically equivalent expressions about σ as given by the successor state axioms in $\mathcal{D}_{ss}$. After repeatedly doing this transformation to a regressable sentence ϕ we get a sentence $\mathcal{R}(\phi)$ that is uniform in S_0 such that $\mathcal{D} \models \phi \equiv \mathcal{R}(\phi)$. We omit the definition of the regression operation $\mathcal{R}$, which can be found in [Reiter, 2001], and only state the main theorem as it appears in [Pirri and Reiter, 1999]:

Theorem 3.2.16 (Pirri and Reiter 1999). *Let $\mathcal{D}$ be a basic action theory and ϕ a first-order sentence in $\mathcal{L}$ that is regressable. Then $\mathcal{R}(\phi)$ is a first-order sentence that is uniform in S_0. Moreover, $\mathcal{D} \models \phi$ iff $\mathcal{D}_0 \cup \mathcal{D}_{una} \models \mathcal{R}(\phi)$.* ■

We now present two lemmas that show a particular relation between the models of two first-order theories Σ_1, Σ_2 and their entailments with respect to a set of first-order sentences Γ. The intuition is that these lemmas also apply when Σ_1 is a basic action theory $\mathcal{D}$ without the set $\mathcal{D}_{fnd}$ of the foundational axioms and Σ_2 is a progression of

$\mathcal{D}_0$ according to some appropriate definition. First we present the following which is a straightforward result.

Lemma 3.2.17. *Let $\Sigma_1, \Sigma_2, \Gamma$ be sets of first-order or second-order sentences of the situation calculus language $\mathcal{L}$ such that for every model M_1 of Σ_1, there is a model M_2 of Σ_2 such that M_1 and M_2 satisfy the same set of sentences in Γ. Then for all ϕ in Γ, if $\Sigma_2 \models \phi$ then $\Sigma_1 \models \phi$.* ■

Proof. Let ϕ be an arbitrary sentence in Γ and assume that $\Sigma_2 \models \phi$. Let M_1 be an arbitrary model of Σ_1. By the hypothesis of the lemma it follows that there is a model M_2 of Σ_2 such that M_1 and M_2 satisfy the same set of sentences in Γ. By the assumption that $\Sigma_2 \models \phi$ it follows that $M_2 \models \phi$, and since ϕ is in Γ it follows that $M_1 \models \phi$. Since M_1 was arbitrary it follows that $\Sigma_1 \models \phi$. □

A similar lemma holds for the opposite direction for first-order sets: the following lemma shows that if Σ_1 and Σ_2 entail the same set of sentences in Γ then for every model of one theory we can always find a model of the other such that the two models satisfy the same set of sentences in Γ. Intuitively this might seem like an obvious fact but the proof is actually not straightforward and involves a non-constructive argument that makes use of the Compactness Theorem of first-order logic.

Lemma 3.2.18. *Let $\Sigma_1, \Sigma_2, \Gamma$ be sets of first-order sentences of the situation calculus language $\mathcal{L}$ such that the following two conditions hold:*

1. *Γ is closed under logical conjunction and negation;*

2. *for all ϕ in Γ, $\Sigma_1 \models \phi$ iff $\Sigma_2 \models \phi$.*

Then for every model M_1 of Σ_1, there is a model M_2 of Σ_2 such that for all ϕ in Γ, $M_1 \models \phi$ iff $M_2 \models \phi$. ■

Proof. We prove by contradiction as follows. Suppose otherwise. Then there is a model M_1 of Σ_1 such that there is no model M_2 of Σ_2 so that for all sentences ϕ in Γ, $M_1 \models \phi$ iff $M_2 \models \phi$. Equivalently, there is a model M_1 of Σ_1 such that for every model M_2 of Σ_2 there is some sentence ψ in Γ so that $M_1 \models \psi$ but $M_2 \not\models \psi$.[6] Let Δ be the following set:

$$\{\psi : \psi \in \Gamma \text{ and there is a model } M_2 \text{ of } \Sigma_2 \text{ such that } M_1 \models \psi \text{ but } M_2 \not\models \psi\}.$$

Clearly Δ is consistent as $M_1 \models \Delta$. Moreover there is no model M_2 of Σ_2 that satisfies Δ since for each model M_2 there is at least one sentence ψ in Δ such that $M' \not\models \psi$. Therefore $\Sigma_2 \cup \Delta \models \Box$. By the Compactness Theorem for first-order logic we get that there is a finite subset of Δ, Δ', such that $\Sigma_2 \cup \Delta' \models \Box$. Let γ be the conjunction of all sentences in Δ'. Then $\Sigma_2 \cup \{\gamma\} \models \Box$ which implies that $\Sigma_2 \models \neg\gamma$.

Since Γ is closed under logical conjunction and negation, $\gamma \in \Gamma$ as it is a conjunction of sentences in Γ and also $\neg\gamma \in \Gamma$. By point 2 in the hypothesis Σ_1 and Σ_2 entail the same set of sentences in Γ therefore we also get that $\Sigma_1 \models \neg\gamma$. Note that this yields a contradiction because M_1 is a model of Σ_1 and M_1 satisfies γ. $\Box$

Now we move on to discuss the first-order definability of progression and prove the main result of this chapter.

3.3 On the first-order definability of progression

Even though the definition of strong progression uses second-order logic, we are interested in finding a first-order set $\mathcal{D}_\alpha$ that qualifies as a strong progression of $\mathcal{D}_0$. In some cases this is feasible but as it was first shown by Lin and Reiter [1997] there are cases of basic action theories where no first-order strong progression exists. It has been unclear whether this is a problem of the definition of strong progression or if it is an inherent difficulty of

[6] Note that if instead of this there is a sentence ψ' in Γ so that $M_1 \not\models \psi'$ but $M_2 \models \psi'$ we can just take ψ to be $\neg\psi'$.

the problem of progression. In other words, it has been an open question whether there is an alternative (weaker) definition for $\mathcal{D}_\alpha$ according to which $\mathcal{D}_\alpha$ is always first-order definable and also qualifies as a correct progression according to Definition 3.2.6.

In fact there is a straightforward alternative definition for $\mathcal{D}_\alpha$ that is always first-order. The idea is to let $\mathcal{D}_\alpha$ be the infinite set of first-order sentences uniform in S_α that are entailed by $\mathcal{D}$ [Pednault, 1987]. The intuition is that if we replace $\mathcal{D}_0$ by a set $\mathcal{D}_\alpha$ that is strong enough to entail all the first-order sentences uniform in S_α that the original theory entails, then it should follow that $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{D}_\alpha$ also entails the same first-order sentences about the future of S_α as the original theory $\mathcal{D}$. We call this alternative definition of progression *weak progression* and in order to avoid confusion with strong progression we will be using $\mathcal{F}_\alpha$ to refer to a weak progression of $\mathcal{D}_0$.

Definition 3.3.1 (Weak progression). Let $\mathcal{D}$ be a basic action theory, α a ground action term, and $\mathcal{F}_\alpha$ a set of first-order sentences uniform in S_α. The set $\mathcal{F}_\alpha$ is a *weak progression* of $\mathcal{D}_0$ wrt α and $\mathcal{D}$ iff for all first-order sentences ϕ uniform in S_α, $\mathcal{F}_\alpha\cup\mathcal{D}_{una} \models \phi$ iff $\mathcal{D} \models \phi$. In this case we also say that $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha$ is a weak progression of $\mathcal{D}$ wrt α. ■

If we could prove that a weak progression of $\mathcal{D}_0$ is always a correct progression then the definition of weak progression would be the preferred option, as strong progression is much more cumbersome to work with and also comes with the strong negative result that second-order logic is necessary in the general case. Observe that a weak progression $\mathcal{F}_\alpha$ of $\mathcal{D}_0$ is a set of first-order sentences uniform in S_α therefore it is clear that the first condition of the definition of a correct progression, i.e., Definition 3.2.6, is always satisfied. Nonetheless, it has been open whether the second condition of Definition 3.2.6 is always satisfied, that is, whether for all first-order sentences ϕ about the future of S_α, $\mathcal{D} \models \phi$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \phi$. Following intuitions and results in [Peppas *et al.*, 1995] Lin and Reiter conjectured that there is a counter example that shows that this is not

always true. Here we state the conjecture in an equivalent way using the terminology that we introduced in this thesis.

Conjecture 3.3.2 (Lin and Reiter 1997). *There is a basic action theory $\mathcal{D}$, a ground action term α, a weak progression $\mathcal{F}_\alpha$ of $\mathcal{D}_0$ wrt α and $\mathcal{D}$, and a first-order sentence ϕ about the future of S_α such that $\mathcal{D} \models \phi$ but $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \not\models \phi$.* ■

The conjecture states that a weak progression is not a correct progression in the sense that in the general case $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha$ fails to entail a sentence ϕ about the future of S_α that is entailed by the original theory $\mathcal{D}$. In this section we give a proof of this conjecture by presenting a counter example of this form, thus resolving the open question whether a weak progression is a correct progression in the general case. The proof is based on the notion of *unnamed objects* that we will be defining shortly. We start by presenting the basic action theory that we will use for the proof.

Definition 3.3.3 (The infinite doors domain). Let $\mathcal{L}$ be the situation calculus language that consists of the standard logical symbols and the symbols $Poss, do, S_0$, the fluent $F(x, s)$, the action constants A, B, the object function $n(x)$, and the object constant 0. Let $\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_0 \cup \mathcal{D}_{fnd}$ be the basic action theory of the *infinite doors domain*, where each of the parts of $\mathcal{D}$ is as follows.

1. $\mathcal{D}_{ap}$ consists of following two sentences:

$$Poss(A, s) \equiv true, \tag{3.1}$$

$$Poss(B, s) \equiv true. \tag{3.2}$$

2. $\mathcal{D}_{ss}$ consists of the following sentence:

$$\begin{aligned} F(x, do(a, s)) \equiv\ & a = A \wedge x = 0 \vee \\ & a = B \wedge \neg F(x, s) \wedge \exists y(x = n(y) \wedge F(y, s)). \end{aligned} \tag{3.3}$$

3. $\mathcal{D}_{una}$ consists of the following sentence:

$$A \neq B. \tag{3.4}$$

4. $\mathcal{D}_0$ consists of the following sentences:

$$\forall a(a = A \vee a = B) \tag{3.5}$$

$$\forall x(x \neq 0 \equiv \exists y\ n(y) = x) \tag{3.6}$$

$$\forall x \forall y(n(x) = n(y) \supset x = y) \tag{3.7}$$

$$F(0, S_0) \wedge \forall x\,(F(x, S_0) \supset F(n(x), S_0)) \tag{3.8}$$

$$\exists x\ \neg F(x, S_0) \tag{3.9}$$

5. $\mathcal{D}_{fnd}$ is as in Definition 3.2.1. ■

The infinite doors domain is similar to the simple doors domain that we introduced in Example 3.2.2. In order to deal with formulas of manageable size, in this case we use short but less informative names for the symbols of the language. Similar to the simple doors domain, the object domain represents doors each of which may be open or closed. The fluent $F(x, s)$ represents that the object x, i.e., the door x, is open in the situation s. The object constant 0 is a name for one specific door in the object domain that we will refer to as the door zero. In the infinite doors domain there is an infinite number of doors that are located one next to the other forming an infinite chain of distinct doors. The door 0 is the first door in the chain and the function $n(x)$ denotes the door that is next to the door x.

The basic action theory $\mathcal{D}$ of the infinite doors domain was carefully defined so that all the models of $\mathcal{D}$ satisfy two properties that we will take advantage in the sequel.

Before we state the properties we need to introduce some notation. We introduce the notion of *named* and *unnamed* objects as follows.

Definition 3.3.4. Let $\mathcal{L}$ be the language of the infinite doors domain, GT the set of all the ground terms of sort object in $\mathcal{L}$, and M an $\mathcal{L}$-structure. For every q in the object domain of M we say that q is *named* iff there is a term t in GT such that t is interpreted as q in M, and *unnamed* otherwise. Also, we say that M is a *term structure* iff all the elements of the object domain of M are named. ■

The first property of $\mathcal{D}$ is due to the initial knowledge base $\mathcal{D}_0$ that can only be satisfied in models with at least one unnamed object. In particular, $\mathcal{D}_0$ is defined so that there exists a door that is different than any door in the infinite chain of doors that start from the door zero.

Lemma 3.3.5. *Let $\mathcal{D}$ be the basic action theory of the infinite doors domain. No model of $\mathcal{D}$ is a term structure.* ■

Proof. Observe that each of the ground terms of sort object in the language has the form $n^k(0)$, i.e., it is constructed by a finite number of applications of the function n to the constant 0. By induction on the construction of ground terms of sort object it follows that if M satisfies the sentence (3.8) then for all named objects q,

$$M, \mu_q^x \models F(x, S_0).$$

The sentence (3.9) on the other hand is satisfied only in a structure M that has an element q' in the object domain such that

$$M, \mu_{q'}^x \not\models F(x, S_0).$$

Therefore, the set $\{(3.8), (3.9)\}$ can only be satisfied in a structure that has an unnamed object. □

The second property of $\mathcal{D}$ is related to the effects of the actions A and B. The action A affects all the doors in the domain changing their status so that in the resulting situation exactly one of them is open, namely the door zero. Furthermore, every time a series of actions B is performed after A, in the resulting situation there is exactly one door that is open which is identified as follows.

Lemma 3.3.6. *Let $\mathcal{D}$ be the basic action theory of the infinite doors domain, M a model of $\mathcal{D}$, and S_A a macro for the situation term $do(A, S_0)$. For every action sequence δ, $M, \mu_q^x \models F(x, do(\delta, S_A))$ iff q is the denotation of $n^k(0)$, where k is*

- *the number zero, if the last action in δ is A;*
- *the number of B actions that appear after the last occurrence of the action A in $\langle A, \delta \rangle$, otherwise.* ■

Proof. First, observe that the sentences (3.6) and (3.7) in $\mathcal{D}_0$ ensure the uniqueness of names for the ground terms of sort object. We will prove the lemma by induction on the number k of B actions that appear after the last occurrence of the action A in $\langle A, \delta \rangle$. The base case is when $k = 0$, which means that A is the last action in $\langle A, \delta \rangle$. In this case it follows by the successor state axiom for F, i.e., sentence (3.3), that F is false in $do(\delta, S_A)$ for all the elements of the object domain except for the denotation of 0, therefore the lemma holds. For the induction step we assume that the lemma holds for k and prove for $k + 1$. By the induction hypothesis it follows that

$$M, \mu_q^x \models F(x, do(\delta, S_A))$$

iff q is the denotation of $n^k(0)$. It follows that

$$M, \mu_{qr}^{xy} \models \neg F(x, do(\delta, S_A)) \wedge \exists y(x = n(y) \wedge F(y, do(\delta, S_A)))$$

iff r is the denotation of $n^k(0)$ and q is the denotation of $n^{k+1}(0)$. Therefore, by the successor state axiom for F it follows that after one more B action, F is false in the resulting situation $do(B, do(\delta, S_A))$ for all the elements of the object domain except for the denotation of $n^{k+1}(0)$. Therefore the induction holds. □

We now present the sentence ϕ^* that we will use to prove the Conjecture 3.3.2.

Definition 3.3.7. Let $\mathcal{L}$ be the language of the infinite doors domain, and S_A a macro for the situation term $do(A, S_0)$. ϕ^* is the following first-order sentence of $\mathcal{L}$:

$$\exists x \forall s (S_A \sqsubseteq s \supset \neg F(x, s)).$$ ■

The sentence ϕ^* expresses that there is a door x such that after the action A is performed x will remain closed forever. First, we show that the basic action theory $\mathcal{D}$ of the infinite doors domain entails ϕ^*. The intuition is that x is the unnamed object that necessarily exists in all models of $\mathcal{D}$.

Lemma 3.3.8. *Let $\mathcal{D}$ be the basic action theory of the infinite doors domain and ϕ^* the first-order sentence as in Definition 3.3.7. Then, $\mathcal{D} \models \phi^*$.* ■

Proof. Let M be a model of $\mathcal{D}$. By Lemma 3.3.6 it follows that for every situation in the future of S_A there can only be named objects for which F is true. By Lemma 3.3.5 it follows that there exists at least one unnamed object q in the domain. Therefore there is an x, namely the unnamed object q, such that $F(x, s)$ is false in every situation in the future of S_A, which implies that $M \models \phi^*$. Since M was arbitrary the lemma follows. □

Note that the sentence ϕ^* is about the future of S_α, where α is the action A. Therefore, if $\mathcal{D}_\alpha$ is a strong progression of $\mathcal{D}_0$ wrt A and $\mathcal{D}$, it then follows that $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha \models \phi^*$, since we just showed that $\mathcal{D} \models \phi^*$. Nonetheless, we now proceed to show that this is not true for a weak progression. First we identify a weak progression $\mathcal{F}_\alpha$ of $\mathcal{D}_0$ wrt A and $\mathcal{D}$.

Lemma 3.3.9. *Let $\mathcal{D}$ be the basic action theory of the infinite doors domain of Definition 3.3.3 and $\mathcal{F}_\alpha$ the following set of first-order sentences:*

$$\{\forall x(x = 0 \equiv F(x, S_A)),\ (3.5),\ (3.6),\ (3.7)\}.$$

Then, $\mathcal{F}_\alpha$ is a weak progression of $\mathcal{D}_0$ wrt the ground action A and $\mathcal{D}$. ■

The proof is long and tedious and can be found in the appendix. It involves a series of model-theoretic constructions using properties of first-order logic such as the upward Lowenheim-Skolem Theorem of first-order logic.

The set $\mathcal{F}_\alpha$ is an updated version of the initial knowledge base of the basic action theory $\mathcal{D}$ of the infinite doors domain. The only difference is that the sentences about F are replaced by a sentence that expresses that there is exactly one door that is open, namely the door zero. Intuitively this is what summarizes the effects of the action A as far as first-order entailment is concerned, and we would expect that this is sufficient for $\mathcal{F}_\alpha$ to be a correct progression.

Nonetheless, there is a property of $\mathcal{D}_0$ that persists in S_A which $\mathcal{F}_\alpha$ fails to express, namely that there is an unnamed object in every model. Of course, this is not reflected in any first-order sentence uniform in S_A that is entailed by $\mathcal{D}$ and this is why $\mathcal{F}_\alpha$, which is defined based on the first-order entailments of $\mathcal{D}$, fails to express it. Apparently, this property is reflected though in a first-order sentence that is about the future of S_A, namely the sentence ϕ^* of Definition 3.3.7. The following lemma shows that $\mathcal{F}_\alpha$ indeed fails to express this property and as a result $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha$ fails to entail ϕ^*.

Lemma 3.3.10. *Let $\mathcal{D}$ be the basic action theory of the infinite doors domain, ϕ^* the first-order sentence as in Definition 3.3.7, and $\mathcal{F}_\alpha$ the set of first-order sentences as in Lemma 3.3.9. Then, $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \not\models \phi^*$.* ■

Proof. Consider a model M of $\mathcal{F}_\alpha$ that has the natural numbers as the domain for objects, and interprets the constant symbol 0 as the number zero and the object function

symbol n as the successor function. This is a term model where the ground term $n^k(0)$ is interpreted as the number $k \in \mathbb{N}$. Observe that the property about F that we showed in Lemma 3.3.6 also holds for all the models of $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha$. It follows that for every x in the object domain there is a sequence of actions after which $F(x, s)$ becomes true: x is the denotation of some term $n^k(0)$, therefore $F(x, do(\delta, S_A))$ is true when δ is a sequence of B actions of size k. It follows that

$$M \models \forall x \exists s (S_A \sqsubseteq s \wedge F(x, s))$$

which is equivalent to $M \models \neg\phi^*$. □

The next theorem establishes that the Conjecture 3.3.2 by Lin and Reiter is indeed true and thus closes the corresponding open question about the correctness of weak progression.

Theorem 3.3.11. *There is a basic action theory $\mathcal{D}$, a ground action term α, and a first-order sentence ϕ about the future of S_α such that $\mathcal{D} \models \phi$ but $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \not\models \phi$, where $\mathcal{F}_\alpha$ is a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$.* ■

Proof. Let $\mathcal{D}$ be the basic action theory of the infinite doors domain, α be the constant action A, ϕ be the first-order sentence ϕ^* of Definition 3.3.7, and $\mathcal{F}_\alpha$ be the set of first-order sentences of Lemma 3.3.9. The theorem follows from Lemma 3.3.8, Lemma 3.3.9, and Lemma 3.3.10. □

This result shows that weak progression is not a correct progression in the general case. Moreover, it shows that any definition of progression that is based on the first-order sentences uniform in S_α that are entailed by $\mathcal{D}$ is not a correct progression in the general case, as we can always do the same trick like the infinite doors domain.

3.4 Concluding remarks

In this chapter we presented the basic action theories of the situation calculus as the logical framework where we define the specifications for the reasoning module we study in this thesis. In this logical framework the functionality of our module corresponds to finding a solution to the problem of projection, i.e., determine whether some condition is true after a sequence of actions is performed, and the problem of progression, i.e., update the knowledge base of the theory so that it reflects to the current state of the world.

As far as the problem of projection is concerned there is a clear specification for the correct solution that is based on logical entailment. As far as the problem of progression is concerned though it has been unclear what is a practical formalization that has the intended properties. For this reason we examined two definitions that exist in the literature, namely the definition of a strong progression that is based on a second-order construct, and a weak progression that is based on the first-order entailments of the original theory.

We were able to prove a major result that resolves a problem that has been open since it was first identified by Lin and Reiter in [1997], namely that a weak progression does not have the intended properties, therefore is not a correct progression in the general case. The consequence of this result is that in the general case we cannot always find a first-order progression that is also a correct progression: a strong progression may necessarily be second-order while a weak progression may be incorrect. This is not a surprising result as every basic action theory includes a second-order inductive axiom that is necessary when we reason about all the possible future situations. What was not clear until now though is how this inductive axiom may be used implicitly, so that a first-order sentence that quantifies over situations captures a property that cannot be expressed in any first-order sentence that does not quantify over situations.

Nonetheless, in practice we are not interested in the most general case for the basic action theories. In the next chapter we proceed to examine a few practical cases where

a first-order progression is also a correct progression. In particular, we examine a case where a weak progression is always correct when we restrict our attention to a practical class of queries, and two cases where a strong progression is always first-order definable when the basic action theory is restricted to have actions that affect only a finite number of properties of the world.

Chapter 4

First-order progression for restricted action theories

In this chapter we focus on the problem of progression and examine three cases where it is possible to obtain a first-order progression that is correct. First, we examine a case where a weak progression is always correct when we restrict our attention to a practical class of queries. Then we focus on two cases where we can always find a strong progression that is first-order definable. In these cases unrestricted queries are allowed but we need to restrict the axioms in the basic action theory in certain ways.

4.1 Basic action theories with restricted queries

One of the properties of a strong progression $\mathcal{D}_\alpha$ is that the new theory $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{D}_\alpha$ and the original theory $\mathcal{D}$ entail the same set of sentences about the future of S_α. In other words, both theories are equivalent with respect to queries of the generalized projection problem. In the previous chapter we showed that a weak progression $\mathcal{F}_\alpha$ is not correct in this sense as it may fail to solve a query about the future of S_α correctly. Nonetheless, in this section we show that a weak progression is correct with respect to a smaller but non-trivial class of queries.

4.1.1 The generalized regression mechanism

First, we review the result by Lin and Reiter [1997] that a weak progression is correct with respect to queries of the simple projection problem.

Lemma 4.1.1 (Lin and Reiter 1997). *Let $\mathcal{D}$ be a basic action theory, α a ground action term, and $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt $\mathcal{D}$ and α. Then, for any first-order formula $\phi(s)$ uniform in s and any situation term σ that is rooted at S_α, $\mathcal{D} \models \phi(\sigma)$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \phi(\sigma)$.* ■

This result can be extended using the properties of regression. We define the following set which is a generalization of the regressable formulas.

Definition 4.1.2. A formula ϕ in $\mathcal{L}$ is *regressable* wrt the situation term σ iff the following conditions hold:

1. every term of sort situation mentioned in ϕ is rooted at σ;
2. for every atom of the form $Poss(\alpha, \kappa)$ that appears in ϕ, α has the form $A(\vec{t})$, where A is some n-ary action function symbol;
3. ϕ does not quantify over situations;
4. ϕ does not mention the predicate symbol $\sqsubset$ and it does not mention any equality atom built on situation terms. ■

The set of regressable formulas wrt S_0 is exactly the set of regressable formulas of Definition 3.2.15, while the set of regressable formulas wrt some ground term σ is the subset that can also be regressed down to σ.

Example 4.1.3. Consider the language of the infinite doors domain of Definition 3.3.3. The sentence $F(0, do(A, S_0)) \wedge F(0, do(B, S_0))$ is regressable wrt S_0 but it is not regressable wrt $do(A, S_0)$, while the sentence $F(0, do(A, S_0)) \wedge F(0, do(B, do(A, S_0)))$ is regressable wrt both S_0 and $do(A, S_0)$. ■

We introduce the generalized regression operator $\mathcal{R}_\sigma$ for formulas that are regressable wrt σ. This operator works exactly the same as the operator $\mathcal{R}$ regressing atoms according to the precondition and successor state axioms in $\mathcal{D}$, except that it only does so until a formula uniform in σ is obtained. Like Theorem 3.2.16, two similar results can be obtained for $\mathcal{R}_\sigma$.

Corollary 4.1.4. *Let $\mathcal{D}$ be a basic action theory and ϕ a first-order sentence that is regressable wrt the ground situation term σ. Then, $\mathcal{R}_\sigma(\phi)$ is a first-order sentence uniform in σ such that $\mathcal{D} \models \phi$ iff $\mathcal{D} \models \mathcal{R}_\sigma(\phi)$.* ■

Proof. The corollary follows by induction over a suitable well-founded ordering relation similar to the proof of Theorem 2 of Pirri and Reiter [1999]. The inductive argument and the details of the ordering relation are exactly the same except for the fact that the base case corresponds to a formula uniform in σ. □

Corollary 4.1.5. *Let $\mathcal{D}$ be a basic action theory, α a ground action term, σ the situation term S_α, $\mathcal{D}_\alpha$ a set of sentences uniform in S_α, and ϕ a first-order sentence that is regressable wrt S_α. Then, $\mathcal{R}_\sigma(\phi)$ is a first-order sentence uniform in S_α such that $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha \models \phi$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha \models \mathcal{R}_\sigma(\phi)$.* ■

Proof. The same proof method as the one for Corollary 4.1.4 applies. □

It is easy then to extend Lemma 4.1.1 and show that a weak progression is correct not only for reasoning about first-order sentences that are uniform in some ground situation term σ but also with respect to any first-order sentence that is regressable wrt S_α.

Lemma 4.1.6. *Let $\mathcal{D}$ be a basic action theory, α a ground action term, $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt $\mathcal{D}$ and α, and ϕ a first-order sentence that is regressable wrt S_α. Then, $\mathcal{D} \models \phi$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \phi$.* ■

Proof. By Corollary 4.1.4, the sentence $\mathcal{R}_{S_\alpha}(\phi)$ is uniform in S_α and $\mathcal{D} \models \phi$ iff $\mathcal{D} \models \mathcal{R}_{S_\alpha}(\phi)$. By Lemma 4.1.1, it follows that $\mathcal{D} \models \phi$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \mathcal{R}_{S_\alpha}(\phi)$. By

Corollary 4.1.5 and since $\mathcal{D}$ and $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$ share the same $\mathcal{D}_{ss}$, it follows that $\mathcal{D}\models\phi$ iff $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha\models\phi$. □

4.1.2 A first-order progression for theories restricted to a practical class of queries

We now proceed to show that a weak progression is correct with respect to a much wider class of sentences that may also quantify over situations. We define the set $\mathcal{Q}_\sigma$ of first-order sentences as follows:

Definition 4.1.7. Let σ be a ground situation term. Then, $\mathcal{Q}_\sigma$ is the smallest set such that the following conditions hold:

1. if the first-order formula $\phi(s)$ is regressable wrt s then the sentences $\phi(\sigma)$ and $\forall s(\sigma\sqsubseteq s\supset\phi(s))$ are in $\mathcal{Q}_\sigma$;
2. if the first-order sentences ϕ,ψ are in $\mathcal{Q}_\sigma$ then the sentences $\neg\phi$ and $\phi\wedge\psi$ are also in $\mathcal{Q}_\sigma$. ■

The set $\mathcal{Q}_\sigma$ is a subset of the first-order sentences about the future of σ such that a quantifier for a situation variable may only appear in a sub-formula of the form:

$$\forall s(\sigma\sqsubseteq s\supset\phi(s)),$$

where $\phi(s)$ does not have any free variables other than s. The following example shows a sentence about the future of σ that does not satisfy this restriction and as a result is not in $\mathcal{Q}_\sigma$.

Example 4.1.8. Consider the language of the infinite doors domain of Definition 3.3.3, and the sentence ϕ^* of Definition 3.3.7:

$$\exists x\forall s(S_A\sqsubseteq s\supset\neg F(x,s)),$$

and let σ be S_A. The sentence ϕ^* is an example of a sentence about the future of σ that is not in $\mathcal{Q}_\sigma$. ■

Nonetheless, $\mathcal{Q}_\sigma$ is quite large and includes many interesting cases such as sentences expressing state invariants of the form "after the execution of α it is ensured that $\phi(s)$ will always hold" or "after the execution of α there is no way to achieve $\phi(s)$", as well as boolean combinations of those.

Example 4.1.9. Consider the language of the infinite doors domain of Definition 3.3.3 and let σ be S_A and ϕ the following sentence:

$$\forall s(S_A \sqsubseteq s \supset F(0, s)).$$

The sentence ϕ expresses that the door 0 will always remain open in the future after action A is executed in the initial situation. Note that ϕ is about the future of σ and it is also in the set $\mathcal{Q}_\sigma$. The same holds for the $\neg\phi$. ■

The intuition is that a weak progression is correct with respect to the sentences in $\mathcal{Q}_\sigma$, where σ is the situation that $\mathcal{D}_0$ is progressed to. In other words, for every sentence ϕ in $\mathcal{Q}_\sigma$, $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \phi$ iff $\mathcal{D} \models \phi$, where σ is S_α. In order to prove this we first identify a sufficient condition that refers to the models of the two theories. This is stated formally in following lemma:

Lemma 4.1.10. *Let $\mathcal{D}$ be a basic action theory, α a ground action term, $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$, and Δ a set of first-order sentences in $\mathcal{L}$. Let M be a model of $\mathcal{D}$ and M' a model of $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha$. If the following holds for all M, M':*

> *for all first-order sentences ϕ uniform in S_α, $M \models \phi$ iff $M' \models \phi$, implies that*
> *for all ϕ in Δ, $M \models \phi$ iff $M' \models \phi$,*

then it follows that: for all $\phi \in \Delta$, $\mathcal{D} \models \phi$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \phi$. ■

Proof. For the ($\Leftarrow$) direction we proceed as follows. Since $\mathcal{F}_\alpha$ is a set of first-order sentences uniform in S_α and $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha \models \mathcal{F}_\alpha$, by Lemma 4.1.1 it follows that $\mathcal{D} \models \mathcal{F}_\alpha$. Therefore $\mathcal{D} \models (\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$ and as a result for all ϕ in Δ, if $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha \models \phi$, then $\mathcal{D} \models \phi$.

For the ($\Rightarrow$) direction we proceed as follows. Let Σ_1 be the theory $(\mathcal{D}-\mathcal{D}_0-\mathcal{D}_{fnd})\cup\mathcal{F}_\alpha$, Σ_2 the theory $\mathcal{D}-\mathcal{D}_{fnd}$, and Γ the set of all the first-order sentences uniform in S_α. By Lemma 4.1.1 it follows that for all ϕ in Γ, $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha \models \phi$ iff $\mathcal{D} \models \phi$ and by Lemma 3.2.14 it follows that for all ϕ in Γ, $\Sigma_1 \models \phi$ iff $\Sigma_2 \models \phi$. The set Γ is clearly closed under conjunction and negation, and so by Lemma 3.2.18 it follows that for every model M_1 of Σ_1, there is a model M_2 of Σ_2 such that for all ϕ in Γ, $M_1 \models \phi$ iff $M_2 \models \phi$. Since a model of $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$ must satisfy $(\mathcal{D}-\mathcal{D}_0-\mathcal{D}_{fnd})\cup\mathcal{F}_\alpha$ it follows that for every model M_1 of $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$, there is a model M_2 of Σ_2 such that for all ϕ in Γ, $M_1 \models \phi$ iff $M_2 \models \phi$. By Lemma 3.2.13 it follows that for every model M_1 of $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$, there is a model M_2 of $\mathcal{D}$ such that for all ϕ in Γ, $M_1 \models \phi$ iff $M_2 \models \phi$. By the assumption of the lemma about the the models of the theories it follows that for every model M_1 of $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$, there is a model M_2 of Σ_2 such that for all ϕ in Δ, $M_1 \models \phi$ iff $M_2 \models \phi$. Now, let Σ_1' be $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha$, Σ_2' to be $\mathcal{D}$. Then by Lemma 3.2.17 it follows that for all $\phi \in \Delta$, if $\mathcal{D} \models \phi$ then $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{F}_\alpha \models \phi$. □

This lemma is very important as it specifies a method for proving that a weak progression is correct with respect to a class of sentences Δ. Essentially, it reduces the question that the two theories entail the same set of sentences in Δ provided they entail the same set of sentences uniform in S_α, to a simpler question about the models of the theories, i.e., any two models of the theories satisfy the same set of sentences in Δ provided they satisfy the same set of sentences uniform in S_α. Finally, note that in the proof we had to use all the lemmas of Section 3.2.3 about basic action theories and their first-order part.

So, in order to show that a weak progression is correct with respect to $\mathcal{Q}_\sigma$ it suffices to prove the following lemma:

Lemma 4.1.11. *Let $\mathcal{D}$ be a basic action theory, α a ground action term, σ the situation term S_α and $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. Let M be a model of $\mathcal{D}$ and M' be a model of $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha$ such that for all sentences ϕ uniform S_α, $M \models \phi$ iff $M' \models \phi$. Then, for all ϕ in $\mathcal{Q}_\sigma$, $M \models \phi$ iff $M' \models \phi$.* ■

Proof. By induction on the construction of formulas ϕ in $\mathcal{Q}_\sigma$. The only interesting part is the base of the induction where we have two cases: i) ϕ is regressable wrt S_α; and ii) ϕ is $\forall s(S_\alpha \sqsubseteq s \supset \psi(s))$, where $\psi(s)$ is regressable wrt s. The first case follows from Lemma 4.1.6. For the second case we will use a trick to deal with the quantification over situations and reduce it to the first case. We show the ($\Rightarrow$) direction by contradiction and the other one follows similarly.

Let $M \models \forall s(S_\alpha \sqsubseteq s \supset \psi(s))$ where $\psi(s)$ is regressable wrt s and suppose that $M' \not\models \forall s(S_\alpha \sqsubseteq s \supset \psi(s))$. It follows that there is an element q of the situation domain such that $M', \mu^s_q \models S_\alpha \sqsubseteq s \wedge \neg\psi(s)$. Since M' satisfies the foundational axioms $\mathcal{D}_{fnd}$, this element q is reachable from the denotation of S_α by a finite number of applications of the function *do*. In particular let $e_1, \ldots, e_n$ be elements of the action domain such that $do^{M'}(\langle e_1, \ldots, e_n\rangle, S^{M'}_\alpha) = q$. It follows that $M' \models \gamma$, where γ is the following sentence:

$$\exists a_1 \cdots \exists a_n \; \neg\psi(do(\langle a_1, \ldots, a_n\rangle, S_\alpha)).$$

By the hypothesis $\psi(s)$ is regressable wrt s and so γ is regressable wrt S_α. By case i) it follows that $M \models \gamma$. Since M satisfies the foundational axioms $\mathcal{D}_{fnd}$, it follows that $M \models \exists s(S_\alpha \sqsubseteq s \wedge \neg\psi(s))$ or equivalently $M \not\models \forall s(S_\alpha \sqsubseteq s \supset \psi(s))$ which is a contradiction. Thus our assumption is wrong and $M' \models \forall s(S_\alpha \sqsubseteq s \supset \psi(s))$. □

Now we are ready to state and prove the main theorem of the section, namely that a weak progression is correct with respect to sentences in $\mathcal{Q}_\sigma$.

Theorem 4.1.12. *Let $\mathcal{D}$ be a basic action theory, α a ground action term, σ the situation*

term S_α, $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$, and ϕ a (first-order) sentence in $\mathcal{Q}_\sigma$. Then $\mathcal{D} \models \phi$ iff $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \models \phi$. ■

Proof. Let Δ be the set $\mathcal{Q}_\sigma$. The theorem follows by Lemma 4.1.10 and Lemma 4.1.11.□

Note that this result does *not* imply that a weak progression also qualifies as a strong progression. A strong progression is always correct with respect to all sentences about the future of S_α, unlike a weak progression that is not correct in the general case: there are sentences ϕ that are not in $\mathcal{Q}_\sigma$, like the sentence ϕ^* of Definition 3.3.7, such that $\mathcal{D} \models \phi$ but $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{F}_\alpha \not\models \phi$. Nonetheless, we are interested in identifying restrictions on the basic action theories such that we can always find a *first-order* strong progression. The following lemma shows that whenever a first-order strong progression exists, it is actually logically equivalent to a weak progression.

Lemma 4.1.13. *Let $\mathcal{D}$ be a basic action theory with a finite $\mathcal{D}_0$, α a ground action term, $\mathcal{D}_\alpha$ a set of first-order sentences in $\mathcal{L}$ that qualifies as a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$, and $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. Then $\mathcal{D}_\alpha \cup \mathcal{D}_{una}$ is logically equivalent to $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$.* ■

Proof. By Definition 3.3.1 it follows that $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$ is a set of first-order sentences uniform in S_α and $\mathcal{D} \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}$. By Theorem 3.2.11 it then follows that

$$\mathcal{D}_\alpha \cup \mathcal{D}_{una} \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}.$$

Similarly, by Theorem 3.2.11 again and since $\mathcal{D}_\alpha$ is a set of first-order sentences uniform in S_α it follows that $\mathcal{D} \models \mathcal{D}_\alpha \cup \mathcal{D}_{una}$. By Definition 3.3.1 it follows that

$$\mathcal{F}_\alpha \cup \mathcal{D}_{una} \models \mathcal{D}_\alpha \cup \mathcal{D}_{una}.$$

□

In the next two sections we shall examine restrictions on the basic action theories under which we can always find a first-order strong progression. In other words we shall examine cases where a weak progression qualifies as a strong progression.

4.2 Basic action theories with local effects

In this section we focus on a less general class of basic action theories and prove two major results about the first-order definability of progression. First, we show that under a locality assumption for the effects of the actions a weak progression always qualifies as a strong progression, and second, we show that under a slightly stronger assumption we can provide a method for computing a strong progression that is first-order and finite. Note that unlike the previous chapter where for the more general theories a weak progression is correct only with respect to a restricted class of queries, here a weak progression is always correct.

4.2.1 Local-effect theories

We now restrict the language $\mathcal{L}$ of the situation calculus so that only the symbols *do* and *Poss* take an argument sort action. Also, in order to simplify the analysis we will be further restricting the language $\mathcal{L}$ so that there are no (non-fluent) predicate symbols and no object function symbols except for constants. Note that this is not a restriction as far as the expressiveness of $\mathcal{L}$ is concerned, as object functions and (non-fluent) predicates can be represented as (relational) fluents that do not change over time. So, the language of the situation calculus $\mathcal{L}$ that we will be dealing with includes the logical symbols $\neg, \wedge, \exists$, the symbol of equality $=$, and the following non-logical symbols:

- a countably infinite supply of variables for each of the three sorts, as well as a countably infinite supply of (second-order) predicate variables of all arities;
- a countably infinite number of constants of sort *object*;

- for each $n \geq 0$, a finite number of *action function* symbols of type $object^n \rightarrow action$;
- the special situation function symbol $do : action \times situation \rightarrow situation$ and the constant S_0;
- the special predicate symbols $Poss : action \times situation$ and $\sqsubset$: $situation \times situation$;
- for each $n \geq 0$, a finite number of *relational fluent* symbols of type $object^n \times situation$.

Moreover, similarly to [Liu and Levesque, 2005] we restrict our attention to basic action theories with *local effects*. For these theories we will show that we can always find a first-order strong progression, and for a special case we can construct one that is also finite. In particular, we consider theories where $\mathcal{D}_0$ is unrestricted but the successor state axioms in $\mathcal{D}_{ss}$ are such that if an action $A(\vec{e})$ changes the truth value of the fluent $F(\vec{c}, s)$ then $\vec{c}$ must be contained in $\vec{e}$. The formal definition of a local-effect basic action theory follows.[1]

Definition 4.2.1. Let the successor state axiom for the fluent symbol F have the following form:

$$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee (F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s)),$$

where $\gamma_F^+(\vec{x}, a, s)$ is called the *positive effects formula* of F and $\gamma_F^-(\vec{x}, a, s)$ is called the *negative effects formula* of F. The successor state axiom for F is *local-effect* iff both $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^-(\vec{x}, a, s)$ are disjunctions of formulas of the following form:

$$\exists\vec{z}(a = A(\vec{y}) \wedge \phi(\vec{y}, s)),$$

where A is an action function, $\vec{y}$ contains $\vec{x}$, $\vec{z}$ corresponds to the remaining variables of $\vec{y}$, and $\phi(\vec{y}, s)$, which is called a *context formula*, is a first-order formula uniform in

[1] See the concluding remarks in Section 4.4 for a short discussion on variant definitions.

s. A basic action theory $\mathcal{D}$ is *local-effect* iff each of the successor state axioms in $\mathcal{D}_{ss}$ is local-effect. ■

A detailed example of a local-effect basic action theory follows.

Example 4.2.2 (The simple levers domain). Let $\mathcal{L}$ be the situation calculus language that consists of the standard logical symbols and the symbols $Poss, do, S_0$, the fluents $Up(x, s)$ and $Unlocked(s)$, the action functions $push(x)$ and $pull(x)$, the action constant *press*, and the object constants *gl* and *rl*. Let $\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_0 \cup \mathcal{D}_{fnd}$ be the basic action theory of the *simple levers domain*, where each of the parts of $\mathcal{D}$ is as follows.

1. $\mathcal{D}_{ap}$ consists of the following three sentences:

$$Poss(push(x), s) \equiv true,$$

$$Poss(pull(x), s) \equiv true,$$

$$Poss(press, s) \equiv true.$$

2. $\mathcal{D}_{ss}$ consists of the following two sentences:

$$Up(x, do(a, s)) \equiv a = push(x) \vee (Up(x, s) \wedge a \neq pull(x)),$$

$$Unlocked(do(a, s)) \equiv (a = press \wedge \forall w\, Up(w, s)) \vee Unlocked(s).$$

3. $\mathcal{D}_{una}$ consists of the following five sentences:

$$push(x) = push(y) \supset x = y,$$

$$pull(x) = pull(y) \supset x = y,$$

$$push(x) \neq pull(y),$$

$$push(x) \neq press,$$

$$pull(x) \neq press.$$

4. $\mathcal{D}_0$ consists of the following two sentences:

$$Up(rl, S_0),$$

$$\neg Unlocked(S_0).$$

5. $\mathcal{D}_{fnd}$ is as in Definition 3.2.1.

The simple levers domain represents a room in which there are various levers, a button, and a door that needs to be unlocked. The object domain represents levers that may be in "up" or "down" position. The fluent $Up(x, s)$ represents that the object x, i.e., the lever x, is up in s, and the fluent $Unlocked(s)$ represents that the door is unlocked in s. The object constant gl is a name for one of the levers, namely the green lever, and similarly rl is a name for the red lever. The action functions $push(x)$ and $pull(x)$ represent the actions of pushing and pulling the lever x to change its position from down to up and vice versa. The action *press* represents the action of pressing the button in the room, which unlocks the door provided that all levers are in up position. Finally, $\mathcal{D}_0$ describes the initial situation where the red lever is in up position and the door is locked. Note that $\mathcal{D}_0$ does not imply anything about the status of the green lever.

Both of the successor state axioms in $\mathcal{D}_{ss}$ are local-effect according to Definition 4.2.1, therefore the basic action theory of the simple levers domain is local-effect. In particular for the successor state axiom for Up, the positive effects formula $\gamma^+_{Up}(x, a, s)$ is the formula

$$a = push(x)$$

and the negative effects formula $\gamma^-_{Up}(x, a, s)$ is the formula

$$a = pull(x).$$

Note that $\gamma^+_{Up}(x, a, s)$ is a disjunction consisting of a single disjunct that has no context formula and such that the vector $\vec{z}$ of Definition 4.2.1 is empty. The same is true for $\gamma^-_{Up}(x, a, s)$. Finally, as far as the successor state axiom for *Unlocked* is concerned, the positive effects formula $\gamma^+_{Unlocked}(a, s)$ is a disjunction that consists only of the disjunct

$$a = press \wedge \phi(s),$$

where $\phi(s)$ is the context formula

$$\forall w\, Up(w, s).$$

The negative effects formula $\gamma^-_{Unlocked}(a, s)$ is the empty disjunction, i.e, it is logically equivalent to *false*. ■

We now proceed to introduce some necessary notation and define a set of formulas whose interpretation remains *unaffected* when the action α is performed, and a transformation that simplifies a formula based on a partial model. Based on these we will then show that a weak progression always qualifies as a strong progression for action theories with local effects. As a weak progression is first-order by definition, this implies that for such theories we can always find a first-order strong progression.

4.2.2 A set of formulas that remain unaffected wrt an action

The idea behind local effects is that a ground action α may only affect a finite number of ground fluent atoms of F that are explicitly specified by α. Assuming that all successor state axioms are local-effect, we can then identify a subset of the first-order formulas

uniform in σ whose interpretation remains unaffected when the action α is performed in σ. First, similarly to [Liu and Levesque, 2005], we show that we can use a ground action term α to simplify a positive or negative effects formula so that it explicitly lists a finite number of ground fluent atoms that may be affected by α.

Lemma 4.2.3. *Let α be the ground action term $A(\vec{e})$, where $\vec{e}$ is a vector of constants in $\mathcal{L}$, and suppose that the successor state axiom for F is local-effect. Then, there exists a formula $\beta(\vec{x}, s)$ of the form*

$$(\vec{x} = \vec{c}_1 \wedge \phi_1(s)) \vee \cdots \vee (\vec{x} = \vec{c}_n \wedge \phi_n(s)),$$

where each $\vec{c}_i$ is a distinct vector of constants contained in $\vec{e}$ and each $\phi_i(s)$ is a first-order formula uniform in s, such that for every model M of $\mathcal{D}_{una}$ and every variable assignment μ the following holds:

$$M, \mu \models \gamma_F^+(\vec{x}, \alpha, s) \equiv \beta(\vec{x}, s).$$

Similarly for $\gamma_F^-(\vec{x}, \alpha, s)$. ■

Proof. Let M be an arbitrary model of $\mathcal{D}_{una}$ and μ an arbitrary variable assignment. Since the successor state axiom for F is local-effect, the formula $\gamma_F^+(\vec{x}, a, s)$ is a disjunction of formulas of a certain form as described in Definition 4.2.1. Let one of the disjuncts be the formula $\exists \vec{z}(a = A(\vec{y}) \wedge \phi(\vec{y}, s))$, where $\vec{y}$ contains $\vec{x}$ and $\vec{z}$ is the remaining variables of $\vec{y}$. By the uniqueness of names for actions it follows that $M, \mu \models \exists \vec{z}(A(\vec{e}) = A(\vec{y}) \wedge \phi(\vec{y}, s))$ iff $M, \mu \models \vec{x} = \vec{c} \wedge \phi(\vec{e}, s)$, where $\vec{x} = \vec{c}$ is contained in $\vec{y} = \vec{e}$. The lemma follows using the same argument for each of the disjuncts of $\gamma_F^+(\vec{x}, a, s)$. □

We will typically use this lemma to obtain a simplified version of the positive (negative) effect formula that explicitly shows the positive (negative) effects of a ground action α. Note that in general this simplified version is different for every ground action α.

Example 4.2.4. Consider the basic action theory $\mathcal{D}$ of the simple levers domain of Example 4.2.2 and let α be the ground action $push(gl)$. Then assuming the uniqueness of names for actions, $\gamma^{+}_{Up}(x, \alpha, s)$ is logically equivalent to the formula $x = gl$. Similarly, $\gamma^{-}_{Up}(x, \alpha, s)$ is logically equivalent to the empty disjunction, i.e., *false*. One way to interpret this is that for every situation s, the action $push(gl)$ may only affect the fluent atom $Up(gl, s)$. For example, if the action $push(gl)$ is performed in S_0 it may only affect the ground fluent atom $Up(gl, S_0)$. More precisely, in any model M of $\mathcal{D}$ the action $push(gl)$ may only affect the atoms $Up(x, S_0)$ such that x is the same as gl with respect to equality. For example, there is a model M of $\mathcal{D}$ where $M \models gl = rl$ and as a result both $Up(gl, S_0)$ and $Up(rl, S_0)$ are affected by the action $push(gl)$. ■

As it follows from Lemma 4.2.3, the parameters of the simplified version of an effects formula wrt a ground action α do not depend on the situation that α is performed. The following definition then introduces the notation that we use to characterize the ground fluent atoms that may be affected by a ground action α using the notion of the situation-suppressed formulas of Definition 3.1.7.

Definition 4.2.5. Let $\mathcal{D}$ be a basic action theory that is local-effect and α a ground action term. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma^{+}_{F}(\vec{x}, \alpha, s)$ and $\gamma^{-}_{F}(\vec{x}, \alpha, s)$ of the successor state axiom for F have the simplified form that Lemma 4.2.3 implies. We define the *argument set* of F wrt α as the following set $\mathcal{C}_F$ of object constant vectors:

$$\mathcal{C}_F = \{\vec{c} \mid \vec{x} = \vec{c} \text{ appears in } \gamma^{+}_{F}(\vec{x}, \alpha, s) \text{ or } \gamma^{-}_{F}(\vec{x}, \alpha, s)\}.$$

We define the *characteristic set* of α as the following set $\mathcal{G}$ of ground fluent atoms that are situation-suppressed:

$$\mathcal{G} = \{F(\vec{c}) \mid \vec{c} \text{ in } \mathcal{C}_F \text{ for some } F \text{ in } \mathcal{L}\}.$$

■

Note that since there are only finitely many fluent symbols in $\mathcal{L}$ it follows that the sets $\mathcal{C}_F$ and $\mathcal{G}$ are *always finite*. A simple example follows.

Example 4.2.6. Consider the basic action theory of the simple levers domain of Example 4.2.2 and let α be the ground action $push(gl)$. The argument set of Up wrt α is the singleton set $\{\langle gl \rangle\}$, and the argument set of $Unlocked$ is the empty set. The characteristic set of α is then the singleton set $\{Up(gl)\}$. ■

The following is a straightforward property of the argument set $\mathcal{C}_F$ and the characteristic set $\mathcal{G}$ which shows that whenever we simplify a successor state axiom using Lemma 4.2.3 we are guaranteed to always get the same characteristic set.

Lemma 4.2.7. *Let $\mathcal{D}$ be a basic action theory that is local-effect, α a ground action term, and F a fluent symbol in $\mathcal{L}$. Then the argument set $\mathcal{C}_F$ of F wrt α, and the characteristic set $\mathcal{G}$ of α are unique.* ■

Proof. By Lemma 4.2.3 it is easy to show by contradiction that the argument set $\mathcal{C}_F$ of F wrt α is unique. Then it follows that the characteristic set $\mathcal{G}$ of α is also unique. □

The intuition behind the definition of the argument set $\mathcal{C}_F$ is that for any fluent atom $F(\vec{x}, \sigma)$ such that $\vec{x}$ is not equal to some $\vec{c}$ in $\mathcal{C}_F$, it follows that its truth value remains the same after action α is performed in the situation σ. This is made precise in the following lemma:

Lemma 4.2.8. *Let $\mathcal{D}$ be a basic action theory that is local-effect, α a ground action term, σ a situation term, $\mathcal{C}_F$ the argument set of F wrt α, M a model of $\mathcal{D}$, and μ a variable assignment. If for all $\vec{c}$ in $\mathcal{C}_F$, $M, \mu \models \vec{x} \neq \vec{c}$, then*

$$M, \mu \models F(\vec{x}, do(\alpha, \sigma)) \text{ iff } M, \mu \models F(\vec{x}, \sigma).$$

■

Proof. Without loss of generality we assume that the formulas $\gamma_F^+(\vec{x}, \alpha, s)$ and $\gamma_F^-(\vec{x}, \alpha, s)$ have the simplified form that Lemma 4.2.3 implies. It follows that $M, \mu \models \gamma_F^+(\vec{x}, \alpha, \sigma)$ only if there is some $\vec{c}$ in $\mathcal{C}_F$ such that $M, \mu \models \vec{x} = \vec{c}$. Similarly, $M, \mu \models \gamma_F^-(\vec{x}, \alpha, \sigma)$ only if there is some $\vec{c}$ in $\mathcal{C}_F$ such that $M, \mu \models \vec{x} = \vec{c}$. Assume that for all $\vec{c}$ in $\mathcal{C}_F$, $M, \mu \not\models \vec{x} = \vec{c}$. It follows that $M, \mu \not\models \gamma_F^+(\vec{x}, \alpha, \sigma)$ and $M, \mu \not\models \gamma_F^-(\vec{x}, \alpha, \sigma)$. Since M models the successor state axiom for F it follows that $M, \mu \models F(\vec{x}, do(\alpha, \sigma)) \equiv F(\vec{x}, \sigma)$. □

Based on Lemma 4.2.8 we now define a subset of the situation-suppressed first-order formulas ϕ that remain *unaffected* wrt a ground action α in the sense that for any situation term σ, $\phi[\sigma]$ is satisfied in a model of $\mathcal{D}$ iff $\phi[do(\alpha, \sigma)]$ is satisfied.

Definition 4.2.9. Let $\mathcal{D}$ be a basic action theory that is local-effect and α a ground action term. We define the set $\mathcal{U}$ of situation-suppressed first-order formulas as the smallest set such that the following conditions hold:

1. all the (situation-independent) equality atoms and their negation are in $\mathcal{U}$;
2. the formulas $\bigwedge_{i=1}^n \vec{x} \neq \vec{c}_i \wedge F(\vec{x})$ and $\bigwedge_{i=1}^n \vec{x} \neq \vec{c}_i \wedge \neg F(\vec{x})$ are in $\mathcal{U}$, where $\{\vec{c}_1, \ldots, \vec{c}_n\}$ is the argument set of F wrt α;
3. $\mathcal{U}$ is closed under the logical operators $\wedge, \vee$, and universal and existential quantification over objects.

When a formula ϕ is in $\mathcal{U}$ we say that ϕ is *unaffected* wrt α. ■

The following lemma states formally the main property of a formula ϕ that is unaffected wrt α.

Lemma 4.2.10. *Let $\mathcal{D}$ be a basic action theory that is local-effect, α a ground action term, σ a situation term, ϕ a situation-suppressed formula that is unaffected wrt α, M a model of $\mathcal{D}$, and μ a variable assignment. Then,*

$$M, \mu \models \phi[\sigma] \text{ iff } M, \mu \models \phi[do(\alpha, \sigma)].$$

■

Proof. By induction on the construction of the situation-suppressed formulas ϕ that are unaffected wrt α. Following Definition 4.2.9 there are the following base cases:

1. $\phi[\sigma]$ is a situation-independent atom. Then $\phi[\sigma]$ and $\phi[do(\alpha, \sigma)]$ coincide and the lemma follows.

2. $\phi[\sigma]$ is a conjunction of an equality sub-formula of the form $\bigwedge_{i=1}^{n} \vec{x} \neq \vec{c}_i$ and a fluent literal of the form $F(\vec{x}, \sigma)$ or $\neg F(\vec{x}, \sigma)$, where $\{c_1, \ldots, c_n\}$ is the argument set of F wrt α. We consider two cases as follows.

   - For all $i = 1, \ldots, n$, $M, \mu \models \vec{x} \neq \vec{c}_i$. Then by Lemma 4.2.8 it follows that $M, \mu \models F(\vec{x}, \sigma)$ iff $M, \mu \models F(\vec{x}, do(\alpha, \sigma))$ and the lemma follows.
   - There is some i, $1 \leq i \leq n$, such that $M, \mu \models \vec{x} = \vec{c}_i$. Then the equality sub-formula does not hold and the lemma follows.

The induction step is straightforward as we are dealing with models. □

We now proceed to present a transformation that simplifies a formula with respect to a partial model that specifies the truth value for a finite number of ground fluent atoms. The idea is similar to the well-known technique of *unit-propagation* of propositional logic. Here we apply this idea in the first-order language of the situation calculus according to our needs. The intuition is that we will use this transformation to "unit-propagate" with respect to all the atoms in the characteristic set $\mathcal{G}$ of a ground action α so that we get formulas that are unaffected wrt α.

4.2.3 A transformation that simplifies a formula wrt a partial model

Without loss of generality we assume that the formulas we will be dealing with have a particular syntactic form that is similar to the *negation normal form*.

Definition 4.2.11. We say that a first-order formula ϕ in $\mathcal{L}$ is in *negation normal form plus (NNF$^+$)* iff the following two conditions hold:

1. ϕ is in negation normal form, i.e., negation is only applied to atoms in ϕ;

2. every fluent atom in ϕ has the form $F(\vec{x}, \sigma)$, where $\vec{x}$ is some vector of variables, and σ some situation term in $\mathcal{L}$.

When ϕ is situation-suppressed the definition is similar. ■

The assumptions of the NNF$^+$ do not restrict the expressiveness of ϕ as it is well-known that for every first-order formula there is a logically equivalent one in negation normal form, and it is also straightforward to show that any atom of the form $F(\vec{t}, \sigma)$, where $\vec{t}$ is any set of object terms, is logically equivalent to the formula $\exists \vec{x}(\vec{x} = \vec{t} \wedge F(\vec{x}, \sigma))$.

The following definition introduces the notion of an H-model θ and the operator $\mathcal{V}(\theta, \phi)$ that simplifies the situation-suppressed formula ϕ based on θ.

Definition 4.2.12. Let ϕ be a situation-suppressed first-order formula that is also in NNF$^+$, and H a finite set of situation-suppressed ground fluent atoms. The formula θ is an *H-model* iff it is a conjunction of literals such that for every $F(\vec{c})$ in H there is exactly one occurrence of $F(\vec{c})$ or $\neg F(\vec{c})$ in θ, all atoms that appear in θ are in H, and they appear in θ in lexicographical order. The operator $\mathcal{V}(\theta, \phi)$ is then defined inductively as follows:

1. if ϕ is an equality atom or the negation of one, then $\mathcal{V}(\theta, \phi) = \phi$;

2. if ϕ is the positive literal $F(\vec{x})$ then

$$\mathcal{V}(\theta, \phi) = \bigvee_{i=1}^{n} (\vec{x} = \vec{c}_i \wedge v_i) \vee (\bigwedge_{i=1}^{n} (\vec{x} \neq \vec{c}_i) \wedge F(\vec{x})),$$

where $\{\vec{c}_1, \ldots, \vec{c}_n\}$ is the set of all constant vectors $\vec{c}$ such that $F(\vec{c})$ is in H, and

$$v_i = \begin{cases} \mathit{true}, & \text{if } F(\vec{c}_i) \text{ is a conjunct of } \theta; \\ \mathit{false}, & \text{otherwise.} \end{cases}$$

In the case that F has no arguments $\mathcal{V}(\theta, \phi)$ is *true* if F is a conjunct of θ, *false* if $\neg F$ is a conjunct of θ, and F otherwise.

3. if ϕ is the negative literal $\neg F(\vec{x})$ then $\mathcal{V}(\theta, \phi)$ is the same as above except that $F(\vec{x})$ and $F(\vec{c}_i)$ are replaced by the corresponding negative literals;

4. for the case of $\phi \vee \psi$, $\phi \wedge \psi$, $\exists x\phi$, and $\forall x\phi$, the operator inductively goes into the sub-formulas until it reaches a literal, for example $\mathcal{V}(\theta, \exists x\phi) = \exists x\mathcal{V}(\theta, \phi)$. ■

An H-model θ essentially specifies a truth value for each of the fluent atoms in H. The operator $\mathcal{V}(\theta, \phi)$ then performs something similar to the process of *unit propagation* in propositional logic: assuming that the partial model θ holds, it simplifies ϕ and eliminates any dependency of ϕ on any of the atoms in H. As the characteristic set $\mathcal{G}$ of α lists the fluent atoms that may be affected by α, the intended property of $\mathcal{V}$ is that when we let H be a superset of $\mathcal{G}$ and θ be an H-model then the simplified formula $\mathcal{V}(\theta, \phi)$ is unaffected wrt α. This is stated formally in the following lemma:

Lemma 4.2.13. *Let $\mathcal{D}$ be a basic action theory that is local-effect, α a ground action term, $\mathcal{G}$ the characteristic set of α, H a finite set of situation-suppressed ground fluent atoms that contains $\mathcal{G}$, θ an H-model, and ϕ a situation-suppressed first-order formula that is also in NNF$^+$. Then $\mathcal{V}(\theta, \phi)$ is a formula that is unaffected wrt α.* ■

Proof. By induction on the construction of the situation-suppressed first-order formulas ϕ in $\mathcal{L}$. As we have assumed that ϕ is also in NNF$^+$, we need consider the following base cases:

1. ϕ is an equality atom. Then $\mathcal{V}(\theta, \phi) = \phi$ and by Definition 4.2.9 $\mathcal{V}(\theta, \phi)$ is unaffected wrt α.

2. ϕ is a positive fluent literal of the form $F(\vec{x})$. Then $\mathcal{V}(\theta, \phi)$ is

$$\bigvee_{i=1}^{n}(\vec{x} = \vec{c}_i \wedge v_i) \vee (\bigwedge_{i=1}^{n}(\vec{x} \neq \vec{c}_i) \wedge F(\vec{x})),$$

   where $\{\vec{c}_1, \ldots, \vec{c}_n\}$ is the set of all constant vectors $\vec{c}$ such that $F(\vec{c}, \sigma)$ is in H, and v_i is *true* or *false*. The first disjunct of $\mathcal{V}(\theta, \phi)$ consists of equality atoms and so by Definition 4.2.9 it is unaffected wrt α. For the second disjunct note that since H is a superset of $\mathcal{G}$ it follows that $\{\vec{c}_1, \ldots, \vec{c}_n\}$ is a superset of the argument set of F wrt α. The second disjunct then of $\mathcal{V}(\theta, \phi)$ is also unaffected wrt α as it corresponds to the Case 2 of Definition 4.2.9 conjuncted with some extra inequality atoms. As the set of unaffected formulas is closed under disjunction if follows that $\mathcal{V}(\theta, \phi)$ is unaffected wrt α.

3. ϕ is negative fluent literal $\neg F(\vec{x})$. This case is similar to the case of a positive literal.

The induction step for the cases of $\phi \vee \psi$, $\phi \wedge \psi$, $\exists x\phi$, and $\forall x\phi$, is immediate from the Case 3 of Definition 4.2.9. □

Note that typically we will let H be a particular set that we have already specified elsewhere. For instance, in the following example we use the characteristic set $\mathcal{G}$ and a $\mathcal{G}$-model θ.

Example 4.2.14. Consider the basic action theory of the simple levers domain of Example 4.2.2 and let α be the ground action $push(gl)$. As shown in Example 4.2.6 the characteristic set $\mathcal{G}$ of α is the singleton set $\{Up(gl)\}$. Let θ be the $\mathcal{G}$-model

$$\neg Up(gl)$$

and ϕ the situation-suppressed formula

$$\exists x\, Up(x) \wedge Unlocked.$$

The operator $\mathcal{V}$ simplifies ϕ according to θ by splitting cases for each of the fluent literals in ϕ: if the literal unifies with some atom in θ then it replaces the literal by its truth value in θ; otherwise if it does not unify with any of the atoms in θ then it keeps the literal as it is. For example, $\mathcal{V}(\theta, Up(x))$ is the following (situation-suppressed) formula:

$$(x = gl \wedge false) \vee (x \neq gl \wedge Up(x)).$$

Similarly, $\mathcal{V}(\theta, Unlocked) = Unlocked$ and $\mathcal{V}(\theta, \phi)$ then is the following formula:

$$\exists x\big((x = gl \wedge false) \vee (x \neq gl \wedge Up(x))\big) \wedge Unlocked.$$

Finally, observe that the sentence we obtain is unaffected wrt α. Intuitively, this implies for instance that if $\mathcal{D} \models \mathcal{V}(\theta, \phi)[S_0]$, then this will be preserved after the action α is performed in S_0, i.e., $\mathcal{D} \models \mathcal{V}(\theta, \phi)[S_\alpha]$ ■

As Lemma 4.2.13 implies, the formula $\mathcal{V}(\theta, \phi)[\sigma]$ is a simplified version of $\phi[\sigma]$ but it is not logically equivalent to $\phi[\sigma]$. Nonetheless, the two formulas are equivalent with respect to models that satisfy $\theta[\sigma]$. This is stated formally in the following lemma:

Lemma 4.2.15. *Let $\mathcal{D}$ be a basic action theory that is local-effect, α a ground action term, σ a situation term, H a finite set of situation-suppressed ground fluent atoms, and ϕ a situation-suppressed first-order formula that is also in NNF$^+$. Let θ be an H-model, M a structure of $\mathcal{L}$ and μ a variable assignment such that $M, \mu \models \theta[\sigma]$. Then,*

$$M, \mu \models \phi[\sigma] \equiv \mathcal{V}(\theta, \phi)[\sigma].$$

■

Proof. By induction on the construction of the situation-suppressed first-order formulas ϕ in $\mathcal{L}$. As we have assumed that ϕ is also in NNF$^+$, we need consider the following base cases:

1. ϕ is an equality atom. Then $\mathcal{V}(\theta, \phi) = \phi$ and the lemma follows.

2. ϕ is a positive fluent literal of the form $F(\vec{x})$. Let $\{\vec{c}_1, \ldots, \vec{c}_n\}$ be the set of all constant vectors $\vec{c}$ such that $F(\vec{c})$ is in H. Recall that

$$\mathcal{V}(\theta, \phi) = \bigvee_{i=1}^{n} (\vec{x} = \vec{c}_i \wedge v_i) \vee (\bigwedge_{i=1}^{n} (\vec{x} \neq \vec{c}_i) \wedge F(\vec{x})),$$

   where v_i is either *true* or *false* depending on θ as in the definition of $\mathcal{V}$.

   For the ($\Rightarrow$) direction let $M, \mu \models \phi[\sigma]$, i.e., $M, \mu \models F(\vec{x}, \sigma)$. We take two cases.

   - For all $i = 1, \ldots, n$, $M, \mu \models \vec{x} \neq \vec{c}_i$. Then $M, \mu \models \bigwedge_{i=1}^{n} (\vec{x} \neq \vec{c}_i) \wedge F(\vec{x}, \sigma)$ and so $M, \mu \models \mathcal{V}(\theta, \phi)[\sigma]$.
   - There is some i, $1 \leq i \leq n$, such that $M, \mu \models \vec{x} = \vec{c}_i$. Since $M, \mu \models F(\vec{x}, \sigma)$ it follows that $M, \mu \models F(\vec{c}_i, \sigma)$. By the definition of an H-model and since $M, \mu \models \theta[\sigma]$ it follows that $F(\vec{c}_i)$ is a conjunct of θ and so by the definition of $\mathcal{V}$ it follows that v_i is *true*. It follows that $M, \mu \models \vec{x} = \vec{c}_i \wedge v_i$ and so $M, \mu \models \mathcal{V}(\theta, \phi)[\sigma]$.

   The ($\Leftarrow$) direction is similar.

3. ϕ is a negative fluent literal $\neg F(\vec{x})$. This case is similar to the case of a positive literal.

For the induction step assume that the lemma holds for ϕ. We show that the lemma also holds for $\exists x \phi$, and the other cases for $\phi \vee \psi$, $\phi \wedge \psi$, and $\forall x \phi$ are similar.
By the definition of $\mathcal{V}$ it follows that $M, \mu \models \mathcal{V}(\theta, \exists x \phi)[\sigma]$ iff

$$M, \mu \models (\exists x \mathcal{V}(\theta, \phi))[\sigma],$$

which by the induction hypothesis holds iff

$$M, \mu \models (\exists x \phi)[\sigma].$$

□

We now have everything we need in order to present the first result about the progression of local-effect basic action theories.

4.2.4 A first-order progression for local-effect theories

For basic action theories with local-effects we are able to show that we can always find a first-order strong progression. We prove this by showing that for such theories a weak progression always qualifies as a strong progression.

Theorem 4.2.16. *Let $\mathcal{D}$ be a basic action theory that is local-effect and has a finite $\mathcal{D}_0$, α a ground action term, and $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. Then $\mathcal{F}_\alpha$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$.* ■

The formal proof relies on Lemma 4.2.3 which implies that we can always rewrite the successor state axioms in $\mathcal{D}_{ss}$ into a form that explicitly specifies the ground fluent atoms that may be affected by α, and the two properties of the $\mathcal{V}$ operator that we showed in Lemmas 4.2.13 and 4.2.15. In particular, $\mathcal{V}$ is used to show that any model of $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$ satisfies a simplified form of the successor state axioms that corresponds to the unaffected fluent atoms wrt α. The proof is long and can be found in Section B.2 of the appendix.

A simple example of the application of Theorem 4.2.16 follows.

Example 4.2.17. Consider the basic action theory $\mathcal{D}$ of the simple levers domain of Example 4.2.2 and let α be the ground action $push(gl)$. Let $\mathcal{F}_\alpha$ be the following set of sentences uniform in S_α:

$$\{\neg Unlocked(S_0),\ \ Up(gl, S_0),\ \ Up(rl, S_0)\}.$$

Then $\mathcal{F}_\alpha$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. ■

Theorem 4.2.16 is a positive result as it implies that we can always find a strong progression that is first-order by simply letting $\mathcal{D}_\alpha$ be the set of first-order sentences uniform in S_α that are entailed by $\mathcal{D}$. Nonetheless this is not practical because there is no guarantee that we can always find a *finite* representation of this infinite set of entailments. We now proceed to present a method for computing a first-order strong progression that is also finite for a slightly more restricted class of theories.

4.2.5 Strictly local-effect theories

In Section 4.2.4 we showed that for local-effect theories a weak progression qualifies as a strong progression. We now turn our attention to a slightly more restricted class of theories, called *strictly local-effect*, for which we can compute a *finite* set of first-order sentences that qualifies as a strong progression. Similar to the local-effect theories, we require that if a ground action $A(\vec{e})$ changes the truth value of the fluent $F(\vec{c}, s)$ then $\vec{c}$ is contained in $\vec{e}$, but we also require that if the change depends on the fluent $G(\vec{d}, s)$ then $\vec{d}$ is also contained in $\vec{e}$.

Definition 4.2.18. Let the successor state axiom for the fluent symbol F have the following form:

$$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee (F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s)).$$

The successor state axiom for F is *strictly local-effect* iff both $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^-(\vec{x}, a, s)$ are disjunctions of formulas of the following form:

$$\exists\vec{z}(a = A(\vec{y}) \wedge \phi(\vec{y}, s)),$$

where A is an action function, $\vec{y}$ contains $\vec{x}$, $\vec{z}$ corresponds to the remaining variables of $\vec{y}$,

and the context formula $\phi(\vec{y}, s)$ is quantifier-free and uniform in s, and does not mention any other free variable other than $\vec{y}, s$. A basic action theory $\mathcal{D}$ is *strictly local-effect* iff each of the successor state axioms in $\mathcal{D}_{ss}$ is strictly local-effect. ■

The definition of a strictly local-effect successor state axiom is exactly the same with that of a local-effect axiom, except for the restriction that the context formulas are also quantifier-free and do not mention any free variable other than $\vec{y}, s$. An example follows that illustrates this difference.

Example 4.2.19 (The two levers domain). Consider the basic action theory of the simple levers domain of Example 4.2.2 and the successor state axiom for the fluent *Unlocked*:

$$Unlocked(do(a, s)) \equiv (a = press \wedge \forall w\, Up(w, s)) \vee Unlocked(s).$$

The positive effects formula $\gamma^{+}_{Unlocked}(a, s)$ is a disjunction that consists only of the disjunct

$$a = press \wedge \phi(s),$$

where $\phi(s)$ is the context formula

$$\forall w\, Up(w, s).$$

The negative effects formula $\gamma^{-}_{Unlocked}(a, s)$ is the empty disjunction, i.e, it is logically equivalent to *false*. Observe that this axiom is local-effect but not strictly local-effect as the context formula $\phi(s)$ quantifies over the variable w. We could transform this axiom into a strictly local-effect axiom if we could propositionalize the context formula $\phi(s)$. For example, assuming that $\mathcal{D}_0$ includes a sentence stating that the only levers in the

domain are the green lever and the red lever, i.e.,

$$\forall x(x = gl \vee x = rl),$$

then the following is a strictly local-effect version of the successor state axiom for *Unlocked*:

$$Unlocked(do(a, s)) \equiv (a = press \wedge Up(gl, s) \wedge Up(rl, s)) \vee Unlocked(s).$$

Similarly, consider the successor state axiom for the fluent *Up*:

$$Up(x, do(a, s)) \equiv a = push(x) \vee (Up(x, s) \wedge a \neq pull(x)).$$

This axiom is trivially strictly local-effect as it is local-effect and the context formulas are empty. Finally, we will refer to this modification of the basic action theory of the simple levers domain as the theory of the *two levers domain*. ■

The instantiation of a strictly local-effect successor state axiom on a ground action α can be simplified so that all fluent atoms that appear in the positive and the negative effects formula have object arguments that are ground and, similarly to the local-effect restriction, a finite number of fluent atoms that may be affected by α are explicitly listed. This follows by the same argument that we used to prove Lemma 4.2.3 for local-effect theories and the fact that the context formulas are quantifier-free.

Lemma 4.2.20. *Let α be the ground action term $A(\vec{e})$, where $\vec{e}$ is a vector of constants in $\mathcal{L}$, and suppose that the successor state axiom for F is strictly local-effect. Then, there exists a formula $\beta(\vec{x}, s)$ of the form*

$$(\vec{x} = \vec{c}_1 \wedge \phi_1(s)) \vee \cdots \vee (\vec{x} = \vec{c}_n \wedge \phi_n(s)),$$

where each $\vec{c}_i$ is a distinct vector of constants contained in $\vec{e}$ and each $\phi_i(s)$ is a first-order

formula uniform in s such that all fluent atoms in $\phi_i(s)$ have ground arguments of sort object that are contained in $\vec{e}$, and for every model M of $\mathcal{D}_{una}$ and variable assignment μ the following holds:

$$M, \mu \models \gamma_F^+(\vec{x}, \alpha, s) \equiv \beta(\vec{x}, s).$$

Similarly for $\gamma_F^-(\vec{x}, \alpha, s)$. ■

We will typically use this lemma to obtain a simplified version of the positive (negative) effect formula that explicitly shows the positive (negative) effects of a ground action α, as well as the ground fluent atoms on which the effects are conditioned.

Example 4.2.21. Consider the basic action theory of the two levers domain of Example 4.2.19, and let α be the ground action $push(gl)$. Then assuming the uniqueness of names for actions, $\gamma_{Up}^+(x, \alpha, s)$ is logically equivalent to the formula

$$x = gl,$$

while $\gamma_{Up}^-(x, \alpha, s)$, $\gamma_{Unlocked}^+(\alpha, s)$, and $\gamma_{Unlocked}^-(\alpha, s)$ are all logically equivalent to the empty disjunction, i.e., *false*. Now let α be the ground action *press*. Then assuming the uniqueness of names for actions, $\gamma_{Unlocked}^+(\alpha, s)$ is logically equivalent to the formula

$$Up(gl, s) \wedge Up(rl, s),$$

while $\gamma_{Unlocked}^-(\alpha, s)$, $\gamma_{Up}^+(x, \alpha, s)$, and $\gamma_{Up}^-(x, \alpha, s)$ are all logically equivalent to *false*. ■

We now define the *context set* of α as an extension of the characteristic set $\mathcal{G}$ of α.

Definition 4.2.22. Let $\mathcal{D}$ be a basic action theory that is strictly local-effect and α a ground action term. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, s)$ and $\gamma_F^-(\vec{x}, \alpha, s)$ of the successor state axiom for F have the simplified form that Lemma 4.2.20 implies. Let $\mathcal{G}$ be the characteristic set of α. We

define the *context set* of α as the following set $\mathcal{J}$ of situation-suppressed fluent atoms:

$$\mathcal{J} = \mathcal{G} \cup \{F(\vec{c}) \mid F(\vec{c}, s) \text{ appears in } \gamma_F^+(\vec{x}, \alpha, s) \text{ or } \gamma_F^-(\vec{x}, \alpha, s) \text{ for some } F \text{ in } \mathcal{L}\}. \quad \blacksquare$$

Similar to the characteristic set, a context set $\mathcal{J}$ is always finite and there are finitely many $\mathcal{J}$-models. A simple example follows.

Example 4.2.23. Consider the basic action theory of the two levers domain of Example 4.2.19, and let α be the ground action $push(gl)$. As shown in Example 4.2.21, assuming that actions have unique names the formula $\gamma_{Up}^+(x, \alpha, s)$ can be simplified to the formula $x = gl$, while the rest of the instantiated effects formulas can be simplified to *false*. Therefore using the simplified version of these formulas the characteristic set $\mathcal{G}$ of α is the set

$$\{Up(gl)\},$$

and the context set of α coincides with the set $\mathcal{G}$ as the simplified version of each of the formulas does not mention any fluent atoms.

Now let α be the ground action *press*. As shown in Example 4.2.21, assuming that actions have unique names the formula $\gamma_{Unlocked}^+(\alpha, s)$ can be simplified to the formula $Up(gl, s) \wedge Up(rl, s)$, while $\gamma_{Unlocked}^-(\alpha, s)$, $\gamma_{Up}^+(x, \alpha, s)$, and $\gamma_{Up}^-(x, \alpha, s)$ can all be simplified to *false*. Therefore using the simplified version of these formulas, the characteristic set $\mathcal{G}$ of α is the singleton set

$$\{Unlocked\},$$

and the context set of α is the set

$$\{Unlocked, Up(gl), Up(rl)\}. \quad \blacksquare$$

Unlike the characteristic set the context set $\mathcal{J}$ is not unique. An example follows that illustrates this.

Example 4.2.24. Consider the basic action theory of the two levers domain of Example 4.2.19, and let α be the ground action $push(gl)$. The formula

$$x = gl \wedge (Up(rl, s) \vee \neg Up(rl, s))$$

also qualifies as a simplification of $\gamma^{+}_{Up}(x, \alpha, s)$ according to Lemma 4.2.20 as it is logically equivalent to the formula $x = gl$ of Example 4.2.23. Note that in this case the characteristic set is again the set $\{Up(gl)\}$, but the context set is the set

$$\{Up(gl), Up(rl)\},$$

which is not the same with the one we obtain when $\gamma^{+}_{Up}(x, \alpha, s)$ is simplified to the formula $x = gl$. ■

The fact that the context set is not unique is not a problem. In practice we would like to identify a simplification for each of the formulas in the successor state axioms that corresponds to a minimal context set. In fact this is what a straightforward algorithm based on unification would provide but as far as our method for progression is concerned any context set is appropriate. Note that a $\mathcal{J}$-model contains all the information that is needed in order to decide whether a fluent atom may change its truth value after action α is performed. The idea then behind our progression mechanism is that we only need to progress these partial models that correspond to the atoms in $\mathcal{J}$. We now proceed to introduce a transformation for formulas that we need for this task.

4.2.6 A transformation for formulas that separates the dependency on a set of atoms

In Section 4.2.3 we introduced the operator $\mathcal{V}(\theta, \phi)$ that simplifies a formula ϕ based on an H-model θ. Since H is a finite set, it is possible then to start with a formula ϕ, split

cases with respect to all the finitely many H-models, and use the operator $\mathcal{V}$ to simplify the formula ϕ according to each one, so that in the end we obtain a logically equivalent formula. We now define the operator $\mathcal{T}(H, \phi)$ which transforms a situation-suppressed formula based on this intuition.

Definition 4.2.25. Let H be a finite set of situation-suppressed ground fluent atoms, $\{\theta_1, \ldots, \theta_m\}$ the set of all the H-models, and ϕ a situation-suppressed first-order formula that is also in NNF$^+$. We define $\mathcal{T}(\mathcal{G}, \phi)$ as follows:

$$\mathcal{T}(H, \phi) = \bigvee_{j=1}^{m} \Big(\theta_j \wedge \mathcal{V}(\theta_j, \phi)\Big).$$

■

The formula we obtain from $\mathcal{T}(H, \phi)$ is a disjunction of formulas of the form $\theta_j \wedge \psi_j$, where θ_j is a partial model that corresponds to the ground atoms in H and ψ_j is a formula such that any dependency on the atoms in H has been removed. An example of the application of $\mathcal{T}$ follows.

Example 4.2.26. Consider the basic action theory of the simple levers domain of Example 4.2.2, and let α be the ground action $push(gl)$ and ϕ the situation-suppressed sentence of Example 4.2.14,

$$\exists x\, Up(x) \wedge Unlocked.$$

Let H be the singleton set $\{Up(gl)\}$, therefore the set of all the H-models is simply the following set:

$$\{Up(gl), \neg Up(gl)\}.$$

$\mathcal{T}(H, \phi)$ is then as follows:

$$Up(gl) \wedge \mathcal{V}(Up(gl), \phi) \vee$$
$$\neg Up(gl) \wedge \mathcal{V}(\neg Up(gl), \phi),$$

which according to the definition of $\mathcal{V}$ is the following sentence:

$$Up(gl) \wedge \Big[\exists x\big((x = gl \wedge true) \vee (x \neq gl \wedge Up(x))\big) \wedge Unlocked\Big] \vee$$
$$\neg Up(gl) \wedge \Big[\exists x\big((x = gl \wedge false) \vee (x \neq gl \wedge Up(x))\big) \wedge Unlocked\Big].$$

Note that $\mathcal{T}(H, \phi)$ separates the part of ϕ whose interpretation depends on the interpretation of the atom $Up(gl)$. Moreover, as shown in Example 4.2.6, the set H coincides with the characteristic set of α and as it follows from Lemma 4.2.13 the sub-formulas in brackets are unaffected wrt α. ■

Finally, before we proceed to the progression of a strictly local-effect theory we need to show that $\mathcal{T}$ preserves logical equivalence.

Lemma 4.2.27. *Let H be a finite set of situation-suppressed ground fluent atoms, ϕ a situation-suppressed first-order formula that is also in NNF$^+$. Let M be a structure of $\mathcal{L}$, μ a variable assignment, and σ be a situation term. Then,*

$$M, \mu \models \phi[\sigma] \equiv \mathcal{T}(H, \phi)[\sigma].$$ ■

Proof. By the definition of an H-model and the fact that $\{\theta_1, \ldots, \theta_m\}$ is the set of all the H-models, it follows that there is exactly one k, $1 \leq k \leq m$, such that $M \models \theta_k[\sigma]$. Therefore in order to prove the lemma it suffices to show that

$$M, \mu \models \phi[\sigma] \equiv \mathcal{V}(\theta_k, \phi)[\sigma],$$

where $M, \mu \models \theta_k[\sigma]$. This follows directly from Lemma 4.2.15. □

We now proceed to present our second result about the progression of local-effect basic action theories.

4.2.7 A finite first-order progression for strictly local-effect theories

For basic action theories with strictly local-effects we are able to show that we can always find a first-order strong progression that is also finite. We prove this by presenting a method that updates the initial knowledge base $\mathcal{D}_0$ appropriately so that it qualifies as a strong progression. Our method first separates the part of $\mathcal{D}_0$ that is related to the changes due to a ground action α and then updates this part accordingly while it keeps the other part unaltered. In particular, the intuition is that we only need to progress each of the $\mathcal{J}$-models separately, where $\mathcal{J}$ is the context set of α. The following definition shows how a $\mathcal{J}$-model is progressed wrt α.

Definition 4.2.28. Let $\mathcal{D}$ be a basic action theory that is strictly local-effect and α a ground action term. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, s)$ and $\gamma_F^-(\vec{x}, \alpha, s)$ of the successor state axiom for F have the simplified form that Lemma 4.2.20 implies, and that the formula $\Phi_F(\vec{x}, \alpha, s)$ consists of these simplified formulas. Let $\mathcal{G}$ be the characteristic set and $\mathcal{J}$ the context set of α, and θ a $\mathcal{J}$-model. We define the *progression* of θ wrt α to be the $\mathcal{J}$-model θ^* such that:

- for each atom $F(\vec{c})$ in $\mathcal{G}$, $F(\vec{c})$ appears positive in θ^* iff

$$\theta[s] \models \Phi_F(\vec{c}, \alpha, s),$$

  and negated otherwise;

- for each atom $F(\vec{c})$ in $\mathcal{J}$ that is not in $\mathcal{G}$, $F(\vec{c})$ appears in θ^* in the same polarity as in θ. ■

The following example shows how the progression of the $\mathcal{J}$-models works in the two levers domain.

Example 4.2.29. Consider the basic action theory of the two levers domain of Example 4.2.19, and let α be the ground action *press*. As shown in Example 4.2.23, the characteristic set $\mathcal{G}$ of α is the singleton set

$$\{Unlocked\},$$

and the context set of α is the set

$$\{Unlocked, Up(gl), Up(rl)\}.$$

Let θ be the following $\mathcal{J}$-model:

$$\neg Unlocked \wedge Up(gl) \wedge Up(rl).$$

Then, the progression of θ^* wrt α is the following $\mathcal{J}$-model:

$$Unlocked \wedge Up(gl) \wedge Up(rl).$$

Intuitively, assuming that $\theta[S_0]$ is the initial knowledge base then $\theta^*[do(\alpha, S_0)]$ is the progression of $\theta[S_0]$ wrt α and $\mathcal{D}$. So in this example, after the button is pressed the door will become unlocked since both levers are in up position according to θ. Therefore, the progression of the partial model θ is one that specifies that *Unlocked* changes truth value in θ^* and the rest of the fluent atoms that are involved remain unchanged. Note that since only *Unlocked* is in the characteristic set $\mathcal{G}$, it is the only fluent atom that may change in θ. The fluent atoms that are in $\mathcal{J}$ but not in $\mathcal{G}$ do not change but are needed nonetheless because the change is *conditioned* on their truth value.

Similarly, the progression of the $\mathcal{J}$-model

$$\neg Unlocked \wedge \neg Up(gl) \wedge Up(rl)$$

is the same $\mathcal{J}$-model

$$\neg \mathit{Unlocked} \wedge \neg \mathit{Up}(gl) \wedge \mathit{Up}(rl),$$

as in this case α does not change the truth value of *Unlocked*. ■

Now we are ready to present the second result about local-effect theories. Our progression method essentially works in two steps: first, we use the operator $\mathcal{T}$ to separate the $\mathcal{J}$-models from the initial knowledge base $\mathcal{D}_0$, and second, we progress each of the $\mathcal{J}$-models accordingly.

Theorem 4.2.30. *Let $\mathcal{D}$ be a basic action theory that is strictly local-effect and has a finite $\mathcal{D}_0$, and ϕ a situation-suppressed sentence such that $\phi[S_0]$ is the conjunction of all the sentences in $\mathcal{D}_0$. Let α be a ground action, $\mathcal{J}$ the context set of α, $\{\theta_1, \ldots, \theta_m\}$ the set of all the $\mathcal{J}$-models, and ϕ^* the following situation-suppressed sentence:*

$$\bigvee_{j=1}^{m} \Big(\theta_j^* \wedge \mathcal{V}\big(\theta_j, \phi\big)\Big),$$

where θ_j^* is the progression of θ_j wrt α. Then, $\phi^*[S_\alpha]$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. ■

The proof of this result relies on Lemma 4.2.20 which shows that we can always rewrite the successor state axioms in $\mathcal{D}_{ss}$ so that they explicitly specify the ground fluent atoms that may be affected by α as well as the ground fluent atoms that the change depends on. The set of all these ground fluent atoms forms the context set $\mathcal{J}$. The result then follows from Lemma 4.2.27 that shows that we can transform ϕ into $\bigvee_{j=1}^{m} \Big(\theta_j \wedge \mathcal{V}\big(\theta_j, \phi\big)\Big)$ and the fact that for every $\mathcal{J}$-model θ_j, $\theta_j^*[S_\alpha]$ is a correct progression of $\theta_j[S_0]$. Note that for all j, $\mathcal{V}\big(\theta_j, \phi\big)$ is a simplified version of ϕ whose truth value does not depend on θ_j and as a result we only need to update θ_j in order to progress. The formal proof is similar to the one for Theorem 4.2.16 and can be found in Section B.3 of the appendix.

A simple example of the application of Theorem 4.2.30 follows.

Example 4.2.31. Consider the basic action theory $\mathcal{D}$ of the two levers domain of Example 4.2.19, and α be the ground action *press*. As shown in Example 4.2.23, the context set of α is the set

$$\{Unlocked, Up(gl), Up(rl)\}.$$

Let ϕ be a situation-suppressed sentence in NNF$^+$ such that $\phi[S_0]$ is logically equivalent to the conjunction of the two sentences in $\mathcal{D}_0$, i.e., ϕ is

$$\exists x(x = rl \wedge Up(x)) \wedge \neg Unlocked.$$

First we apply the $\mathcal{T}(H, \phi)$ operator to separate the $\mathcal{J}$-models from ϕ, where we use the set $\mathcal{J}$ as H. The resulting sentence ϕ' is a disjunction that consists of eight disjuncts, one for each of the eight different $\mathcal{J}$-models that exist:

$$\begin{aligned}
&\big[Unlocked \wedge Up(gl) \wedge Up(rl) \\
&\qquad \exists x\big(x = rl \ \wedge (x = gl \wedge true \vee x = rl \wedge true \vee x \neq gl \wedge x \neq rl \wedge Up(x))\big) \wedge false\big] \ \vee \\
&\qquad \ldots \\
&\big[\neg Unlocked \wedge Up(gl) \wedge Up(rl) \wedge \\
&\qquad \exists x\big(x = rl \ \wedge (x = gl \wedge true \vee x = rl \wedge true \vee x \neq gl \wedge x \neq rl \wedge Up(x))\big) \wedge true\big] \ \vee \\
&\big[\neg Unlocked \wedge Up(gl) \wedge \neg Up(rl) \wedge \\
&\qquad \exists x\big(x = rl \ \wedge (x = gl \wedge true \vee x = rl \wedge false \vee x \neq gl \wedge x \neq rl \wedge Up(x))\big) \wedge true\big] \ \vee \\
&\big[\neg Unlocked \wedge \neg Up(gl) \wedge Up(rl) \wedge \\
&\qquad \exists x\big(x = rl \ \wedge (x = gl \wedge false \vee x = rl \wedge true \vee x \neq gl \wedge x \neq rl \wedge Up(x))\big) \wedge true\big] \ \vee \\
&\big[\neg Unlocked \wedge \neg Up(gl) \wedge \neg Up(rl) \wedge \\
&\qquad \exists x\big(x = rl \ \wedge (x = gl \wedge false \vee x = rl \wedge false \vee x \neq gl \wedge x \neq rl \wedge Up(x))\big) \wedge true\big].
\end{aligned}$$

Using straightforward properties of equality the sentence ϕ' can be simplified to the

following intuitive sentence ϕ'':

$$\neg Unlocked \wedge Up(gl) \wedge Up(rl) \ \vee$$
$$\neg Unlocked \wedge \neg Up(gl) \wedge Up(rl).$$

The sentence ϕ'' essentially consists of only two of the disjuncts of ϕ' that are also simplified, as the rest of the disjuncts of ϕ' are not consistent. Note also that $\phi[S_0]$, $\phi'[S_0]$, and $\phi''[S_0]$ are all logical equivalent.

Now that the $\mathcal{J}$-models are separated from ϕ we can progress ϕ by progressing each of the $\mathcal{J}$-models in ϕ'' separately as we did in Example 4.2.29. Let ϕ^* be the following sentence:

$$Unlocked \wedge Up(gl) \wedge Up(rl) \ \vee$$
$$\neg Unlocked \wedge \neg Up(gl) \wedge Up(rl).$$

Then the sentence $\phi^*[S_\alpha]$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. ■

Theorem 4.2.30 is a more practical result than Theorem 4.2.16 as it shows how to construct a strong progression that is first-order and also finite. Moreover, unlike the case of Theorem 4.2.16 where we need to find and prove that a set $\mathcal{F}_\alpha$ is a weak progression which is often a non-trivial task, in the case of Theorem 4.2.30 we only need to do a few straightforward reasoning tasks that are similar to the process of unification, and a simple syntactic transformation.

We now proceed to we examine a different class of theories where the local-effect assumption is relaxed but a certain structure needs also to be assumed for the initial knowledge base.

4.3 Basic action theories with range-restricted effects

In Section 4.2 we examined theories that are unrestricted as far as the initial knowledge base is concerned but the successor state axioms are restricted so that actions may have only effects that are local. In this section we focus on a different class of theories that are more general in that they allow for non-local effects but assume a certain structure for the initial knowledge base. For such theories we show a method for computing a strong progression that is first-order and finite using similar techniques as the ones of Section 4.2. Moreover, we explore a case of sensing actions and show how to extend our method for progression in the presence of online sensing results.

4.3.1 A database of possible closures

We assume the same situation calculus language $\mathcal{L}$ that we used in Section 4.2 that has no (non-fluent) predicate symbols and no object function symbols except for constants. Moreover, we assume that the basic action theories we study also include the set $\mathcal{E}$ of the (infinitely many) unique-names axioms for the constants of sort object:

$$\{c \neq d \mid c, d \text{ distinct constants of sort object in } \mathcal{L}\}.$$

Unlike the theories we have seen so far where the initial knowledge base $\mathcal{D}_0$ was unrestricted, here we assume that $\mathcal{D}_0$ may express only a certain kind of disjunctive information. Intuitively we treat each fluent as a *multi-valued function*, where the last argument of sort object is considered as the *output* and the rest of the arguments of sort object as the *input* of the function. We require then that $\mathcal{D}_0$ is set of *possible closures axioms* that express disjunctive information about the output of fluents. First, we need to define the notion of a *closure* of a fluent atom with a ground input.

Definition 4.3.1. Let $V = \{e_1, \ldots, e_m\}$ be a set of constants and τ a fluent atom of the form $F(\vec{c}, w, S_0)$, where $\vec{c}$ is a vector of constants in $\mathcal{L}$ and w a variable. We say that τ is an atom of F that has the *ground input* $\vec{c}$ and the *output* w. The *atomic closure* χ of τ on $\{e_1, \ldots, e_m\}$ is the following sentence:

$$\forall w.F(\vec{c}, w, S_0) \equiv (w = e_1 \vee \cdots \vee w = e_m).$$

We use $\vec{\tau}$ to denote a vector of distinct fluent atoms with a ground input and the variable w as the output. The *closure* χ of $\vec{\tau}$ on a vector $\vec{V}$ of sets of constants of the same size as $\vec{\tau}$ is then the conjunction of each of the atomic closures of τ_i on V_i. ■

Assuming that the constants have unique names, an atomic closure χ of $F(\vec{c}, w, S_0)$ on a set of constants V implies complete information for all the ground atoms of $F(\vec{c}, w, S_0)$. In particular χ entails all the (finitely many) ground atoms where w is replaced by a constant in V, and for all the rest of the (infinitely many) ground atoms their negation is entailed. The fluent F is a multi-valued function in the sense that when the input to F is $\vec{c}$, then the closure χ defines the output to be the set of constants V. A non-atomic closure has the same effect for more than one fluent atom with a ground input. A simple example follows.

Example 4.3.2. Let $Near(x, y, s)$ be a fluent that represents that the object y is lying near the object x in the situation s, and $agent$, box_1, box_2, $bomb$ four constants of sort object. The atomic closure χ_1 of the fluent atom $Near(bomb, w, S_0)$ on the set $\{agent, box_1\}$ is the following sentence:

$$\forall w.Near(bomb, w, S_0) \equiv (w = agent \vee w = box_1).$$

Assuming that objects have unique names, the atomic closure χ_1 implies complete information for all the ground instances of $Near(bomb, w, S_0)$: there are *exactly* two ground

atoms that are entailed by χ_1, namely $Near(bomb, agent, S_0)$ and $Near(bomb, box_1, S_0)$, and for all the rest their negation is entailed. Similarly, the sentence χ_2,

$$\forall w.Near(bomb, w, S_0) \equiv (w = agent \vee w = box_2),$$

is a different atomic closure of $Near(bomb, w, S_0)$, i.e., the closure of $Near(bomb, w, S_0)$ on the set $\{agent, box_2\}$. Finally, the sentence χ_3,

$$\forall w.Near(agent, w, S_0) \equiv w = bomb,$$

is the atomic closure of $Near(agent, w, S_0)$ on $\{bomb\}$. Assuming that objects have unique names, χ_3 implies that there is *exactly* one object that is near the agent in S_0, i.e., the bomb, and all the rest of the objects are far.

A non-atomic closure is a conjunction of atomic closures such that each one refers to a different atom of the form $F(\vec{c}, w, S_0)$, where w and S_0 are fixed, and F and $\vec{c}$ vary. Therefore, the sentence $\chi_1 \wedge \chi_2$ does not qualify as a closure according to Definition 4.3.1 and note also that it is not consistent. On the other hand the sentence $\chi_1 \wedge \chi_3$ is indeed a closure, in particular the closure of the vector

$$\langle Near(bomb, w, S_0), Near(agent, w, S_0)\rangle$$

of distinct fluent atoms with a ground input on the vector

$$\langle \{agent, box_1\}, \{bomb\}\rangle$$

of sets of constants. ■

The following is a straightforward property of closures that follows from the fact that they imply complete information for a set of distinct fluent atoms with a ground input.

Lemma 4.3.3. *Let ϕ be the closure of $\vec{\tau}$ and ψ be a closure of $\vec{\pi}$ on some appropriate vectors and assume that the constants have unique names. Then $\phi \wedge \psi$ is consistent iff for every i, j such that $\tau_i = \pi_j$, the atomic closure of τ_i in ϕ is a closure on the same set of constants as the one of π_j in ψ. If $\phi \wedge \psi$ is consistent then it is straightforward to construct a closure χ such that χ and $(\phi \wedge \psi)$ are logically equivalent.* ■

We now introduce the notion of a *possible closure* that we use for representing disjunctive information about the output of fluents.

Definition 4.3.4. Let $\vec{\tau}$ be a vector of distinct fluent atoms with a ground input and the variable w as the output. A *possible closures axiom (PCA)* ϕ for $\vec{\tau}$ is a sentence of the form $\bigvee_{i=1}^{n} \chi_i$, where each χ_i is a closure of $\vec{\tau}$ on an appropriate vector of sets of constants $\vec{V}_i$, such that the vectors $\vec{V}_i, \ldots, \vec{V}_n$ are distinct. We say that each χ_i is a *possible closure* of $\vec{\tau}$ wrt ϕ, and each atomic closure in χ_i is a *possible atomic closure* wrt ϕ. ■

A closure χ of $F(\vec{c}, w, S_0)$ on V expresses complete information in the sense that when the input to F is $\vec{c}$ then the output is the set V. Along the same lines, a possible closures axiom ϕ for $F(\vec{c}, w, S_0)$ expresses disjunctive information in the sense that the output may be exactly one of the finitely many sets that are listed in ϕ. The same is true for a vector $\vec{\tau}$ of fluent atoms with ground input: in this case a possible closures axiom for $\vec{\tau}$ expresses disjunctive information about the combination of the outputs of the fluent atoms in $\vec{\tau}$. A simple example follows.

Example 4.3.5. Consider the atomic closures χ_1, χ_2, χ_3 of Example 4.3.2. The sentence

$$\chi_1 \vee \chi_2$$

is a possible closures axiom for $Near(bomb, w, S_0)$. This axiom expresses that the output of the fluent *Near* for the input *bomb* in S_0 is either $\{agent, box_1\}$ or $\{agent, box_2\}$. Assuming that objects have unique names then $\chi_1 \vee \chi_2$ implies that there are exactly two

objects near the bomb, one being the agent and the other one being either box_1 or box_2. Similarly, the sentence

$$(\chi_1 \wedge \chi_3) \vee (\chi_2 \wedge \chi_3)$$

is a possible closures axiom for the vector

$$\langle Near(bomb, w, S_0), Near(agent, w, S_0)\rangle$$

expressing the same information as $\chi_1 \vee \chi_2$ plus that the only object near the agent is the bomb. In general though when we use a possible closures axiom for a vector $\vec{\tau}$ it is because we cannot decompose it to a simpler set of axioms, unlike here where the axiom is logically equivalent to the set $\{\chi_1 \vee \chi_2, \chi_3\}$. ■

We can now state formally our restriction on the initial knowledge base $\mathcal{D}_0$. We require that $\mathcal{D}_0$ be a *database of possible closures* in the following sense:

Definition 4.3.6. The set $\mathcal{D}_0$ is a *database of possible closures (DBPC)* iff $\mathcal{D}_0$ is a set of possible closures axioms such that there is no fluent atom with a ground input that appears in more than one axiom. We say that the closure χ is a *possible closure* wrt $\mathcal{D}_0$ iff χ is a possible closure wrt some axiom in $\mathcal{D}_0$. Similarly, χ is a *possible atomic closure* wrt $\mathcal{D}_0$ iff χ is a possible atomic closure wrt some axiom in $\mathcal{D}_0$. ■

The definition of a database of possible closures implies that for every fluent symbol F and every ground input $\vec{c}$, either the output of $F(\vec{c}, w, S_0)$ is completely unknown or there is a finite list of possible closures for $F(\vec{c}, w, S_0)$ that are explicitly listed in exactly one axiom in $\mathcal{D}_0$. The example that follows shows a simple database of possible closures.

Example 4.3.7. Let $Near(x, y, s)$ be a fluent that represents that the object y is lying near the object x in the situation s, $Status(x, y, s)$ a fluent that represents that the object x has the status y in s, and *agent*, box_1, box_2, *bomb*, *closed*, *broken*, *ready* seven constants of sort object. Let χ_1, χ_2, χ_3 be the atomic closures of Example 4.3.2, χ_4 the atomic

closure of $Status(box_1, w, S_0)$ on $\{closed\}$, χ_5 the atomic closure of $Status(box_2, w, S_0)$ on $\{closed\}$, and χ_6 the atomic closure of $Status(agent, w, S_0)$ on $\{ready\}$. Then the following set $\mathcal{D}_0$ is a database of possible closures:

$$\{\chi_1 \vee \chi_2, \chi_3, \chi_4, \chi_5, \chi_6\},$$

Note that each sentence in $\mathcal{D}_0$ is a possible closures axiom: $\chi_1 \vee \chi_2$ expresses disjunctive information as it lists two possible closures for $Near(bomb, w, S_0)$, while $\chi_3, \chi_4, \chi_5, \chi_6$ express complete information as they list exactly one possible closure. Finally, each of the sentences $\chi_1, \chi_2, \chi_3, \chi_4, \chi_5, \chi_6$ is a possible closure wrt $\mathcal{D}_0$. ■

We now proceed to present the set of *possible answers* to a query formula γ with respect to a database of possible closures $\mathcal{D}_0$, and explore the conditions under which this set is finite. As we will see later this forms the basis for relaxing the local-effect assumption and providing a method for progressing $\mathcal{D}_0$.

4.3.2 Possible answers to a query wrt a database of possible closures

As we saw in Section 4.3.1 a database of possible closures $\mathcal{D}_0$ expresses a special kind of disjunctive information. In particular, for every fluent atom with a ground input $F(\vec{c}, w, S_0)$ that is mentioned in $\mathcal{D}_0$ there is a finite number of constants e such that $\mathcal{D}_0 \cup \mathcal{E} \models F(\vec{c}, e, S_0)$, and for all the rest of the constants in the language the negated literal is entailed. The intuition is that a similar property holds for the free variables of certain non-atomic formulas. First we define the notion of a *possible answer* to a formula γ wrt $\mathcal{D}_0$ and then we investigate the conditions under which the set of possible answers is finite.

Definition 4.3.8. Let $\mathcal{D}_0$ be a database of possible closures and $\gamma(\vec{x})$ a first-order formula uniform in S_0 whose only free variables are the ones in $\vec{x}$. The *possible answers* to

γ wrt $\mathcal{D}_0$, denoted as $\mathsf{pans}(\gamma, \mathcal{D}_0)$, is the smallest set of pairs $(\vec{c}, \chi)$ such that:

- $\vec{c}$ is a vector of constants of the same size as $\vec{x}$ and χ is a closure;
- $\{\chi\} \cup \mathcal{E} \models \gamma(\vec{c})$;
- χ is consistent with $\mathcal{D}_0$ and minimal in the sense that every atomic closure in χ is necessary. ■

Intuitively, the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$ characterizes the cases where the formula $\gamma(\vec{x})$ is satisfied in a model of $\mathcal{D}_0 \cup \mathcal{E}$ for some ground instantiation of $\vec{x}$. Moreover, the closure χ of the possible answer $(\vec{c}, \chi)$ specifies a minimal partial model of $\mathcal{D}_0$ in which $\gamma(\vec{c})$ is satisfied. The role of χ is similar to the notion of a prime implicant in propositional logic but here instead of a conjunction of propositions we are interested in getting a minimal closure χ that is consistent with $\mathcal{D}_0$. Note also that Definition 4.3.8 is well-defined as it is straightforward that the set of possible answers implied by the definition is unique. An example follows.

Example 4.3.9. Let $\mathcal{D}_0$ be the database of possible closures of Example 4.3.7 and $\gamma(x)$ the formula $\mathit{Near}(\mathit{bomb}, x, S_0)$. Then, $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is the set

$$\{(\mathit{agent}, \chi_1), (\mathit{box}_1, \chi_1), (\mathit{agent}, \chi_2), (\mathit{box}_2, \chi_2)\}.$$

For instance, the possible answer (agent, χ_1) in the previous set is due to the closure χ_1 which is a possible closure wrt $\mathcal{D}_0$:

$$\{\chi_1\} \cup \mathcal{E} \models \mathit{Near}(\mathit{bomb}, \mathit{agent}, S_0).$$ ■

It is important to observe that the possible answers to a query may be *infinite* in general. Consider the following example.

Example 4.3.10. Let $\mathcal{D}_0$ be the database of possible closures of Example 4.3.7 and let the language also include the constant box_3 of sort object. Let $\gamma_1(x)$ be the formula

$$Near(box_3, x, S_0).$$

Since nothing is said about the objects near box_3 in $\mathcal{D}_0$, every closure of $Near(box_3, w, S_0)$ on any set of constants is consistent with $\mathcal{D}_0$. This implies that there are infinitely many possible answers to $\gamma_1(x)$ wrt $\mathcal{D}_0$. For example for every constant c in $\mathcal{L}$,

$$(c, \chi_c) \in \mathsf{pans}(\gamma_1, \mathcal{D}_0),$$

where χ_c is the closure of $Near(box_3, w, S_0)$ on $\{c\}$. In other words, for every constant c in $\mathcal{L}$ there is always a model of $\mathcal{D}_0 \cup \mathcal{E}$ in which the object c is the only one near the object box_3. Now let $\gamma_2(x)$ be the formula

$$\neg Near(bomb, x, S_0).$$

Then, there are infinitely many possible answers to $\gamma_2(x)$ wrt $\mathcal{D}_0$ but this time it is not because $Near(bomb, w, S_0)$ is not mentioned in any axiom in $\mathcal{D}_0$ rather than because the database is queried for negative literals. For instance, $\mathsf{pans}(\gamma_2, \mathcal{D}_0)$ includes the infinite set

$$\{(c, \chi_1) \mid c \in \mathcal{L}, c \neq agent, c \neq box_1\},$$

since all the objects except for *agent* and box_1 are far when the possible closure χ_1 is assumed. ■

Following the intuitions from the previous example we distinguish two potential sources of an infinite number of possible answers to a formula γ: first, when γ includes a fluent atom of the form $F(\vec{c}, w, S_0)$ that is not mentioned in $\mathcal{D}_0$, and second, when γ

queries $\mathcal{D}_0$ for negative literals. Our objective is to identify a class of formulas γ such that these cases are avoided and $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is always a finite set. To that end we introduce the formulas that are *just-in-time* wrt $\mathcal{D}_0$ and the *range-restricted* formulas as follows.

Definition 4.3.11. Let $\mathcal{D}_0$ be a database of possible closures and $\gamma(\vec{x})$ a first-order formula uniform in S_0 that does not mention any free variable other than those in $\vec{x}$. Then $\gamma(\vec{x})$ is *just-in-time* wrt $\mathcal{D}_0$ iff for every vector of constants $\vec{c}$ that has the same size as $\vec{x}$, $\gamma(\vec{c})$ is consistent with $\mathcal{D}_0 \cup \mathcal{E}$ iff there exists a closure χ such that $\{\chi\} \cup \mathcal{E} \models \gamma(\vec{c})$, where χ is a conjunction of possible closures wrt $\mathcal{D}_0$. ■

Definition 4.3.11 implies that each of the possible answers to γ wrt $\mathcal{D}_0$ consists of information that is listed explicitly in $\mathcal{D}_0$. This provides a way to avoid the cases that are similar to the case of γ_1 of Example 4.3.10 (note that γ_1 is not just-in-time wrt the $\mathcal{D}_0$ of the example). Nonetheless, assuming that a formula γ is just-in-time wrt $\mathcal{D}_0$ is not enough to avoid an infinite set of possible answers (note that the formula γ_2 of Example 4.3.7 is just-in-time wrt the $\mathcal{D}_0$ of the example but $\mathsf{pans}(\gamma_2, \mathcal{D}_0)$ is still an infinite set). We also need to ensure that γ is *range-restricted* in the following sense:

Definition 4.3.12. Let γ be a first-order formula and X a set of variables in $\mathcal{L}$. Then γ is *safe-range* wrt X according to the following rules:

1. let $\vec{t}$ be a vector of variables and constants, c a constant, and x a variable of sort object, then:

   - $x = c$ is safe-range wrt $\{x\}$;
   - $F(\vec{t}, c, S_0)$ is safe-rage wrt $\{\}$;
   - $F(\vec{t}, x, S_0)$ is safe-range wrt $\{x\}$ if x is not included in $\vec{t}$, and safe-range wrt $\{\}$ otherwise;

2. if ϕ is safe-range wrt X_ϕ and ψ is safe-range wrt X_ψ, then:

- $\phi \vee \psi$ is safe-range wrt $X_\phi \cap X_\psi$;
- $\phi \wedge \psi$ is safe-range wrt $X_\phi \cup X_\psi$;
- $\neg\phi$ is safe-range wrt $\{\}$;
- $\exists x\phi$ is safe-range wrt $X/\{x\}$ provided that $x \in X$;

3. no other formula is safe-range.

A formula γ is said to be *range-restricted* iff it is safe-range wrt the set of all its free variables. ■

Definition 4.3.12 specifies a class of formulas γ such that γ may query $\mathcal{D}_0$ for negative literals in a way that avoids the cases that are similar to γ_2 of Example 4.3.7 (note that the formula γ_2 of Example 4.3.7 is not range-restricted). The intuition is that when a formula γ is both just-in-time wrt $\mathcal{D}_0$ and range-restricted then the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is always finite. This is stated formally in the following lemma:

Lemma 4.3.13. *Let $\mathcal{D}_0$ be a database of possible closures and $\gamma(\vec{x})$ a first-order formula that is range-restricted, just-in-time wrt $\mathcal{D}_0$, and does not mention any free variable other than those in $\vec{x}$. Then, $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is a finite set $\{(\vec{c}_1, \chi_1), \ldots, (\vec{c}_n, \chi_n)\}$ such that every χ_i is a conjunction of possible atomic closures wrt $\mathcal{D}_0$ and every c_i consists of constants in $\mathcal{D}_0$ and $\gamma(\vec{x})$, and the following holds:*

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}.\gamma(\vec{x}) \equiv \bigvee_{i=1}^{n} (\vec{x} = \vec{c}_i \wedge \chi_i).$$

■

The proof is based on a stronger lemma for safe-range formulas. Even though the ideas are straightforward the actual proof is tedious and can be found in Section B.4 of the appendix. Lemma 4.3.13 implies that whenever γ is range-restricted and just-in-time wrt $\mathcal{D}_0$, we can use the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$ to specify a logically equivalent representation for γ. An example follows that shows the possible answers to a formula γ of the appropriate form and the application of Lemma 4.3.13.

Example 4.3.14. Let $\mathcal{D}_0$ and the fluents $\mathit{Near}(x, y, s)$ and $\mathit{Status}(x, y, s)$ be as in Example 4.3.7. According to the base case of Definition 4.3.12 the formula $\mathit{Near}(x, y, S_0)$ is safe-range wrt $\{y\}$ but since it is not safe-range wrt $\{x, y\}$ it is not range-restricted. Similarly, the formula $\mathit{Status}(y, z, S_0)$ is safe-range wrt $\{z\}$ but not range-restricted. On the other hand the formula $\mathit{Near}(\mathit{bomb}, y, S_0)$ is safe-range wrt $\{y\}$ therefore range-restricted. Now let $\gamma(y, z)$ be the following formula:

$$\mathit{Near}(\mathit{bomb}, y, S_0) \wedge \mathit{Status}(y, z, S_0).$$

According to the rule for conjunction in Definition 4.3.12 it follows that the formula $\gamma(y, z)$ is safe-range wrt $\{y\} \cup \{z\}$, i.e., $\{y, z\}$, therefore is range-restricted. Note that the range-restricted requirement captures the intuitive idea that we can use the output of a fluent atom with a ground input, here $\mathit{Near}(\mathit{bomb}, y, S_0)$, in order to determine the input to another fluent, here $\mathit{Status}(y, z, S_0)$, so that each of the possible answers to the whole formula is composed by the possible answers to fluent atoms with a ground input.

Moreover, it is not too difficult to show that $\gamma(y, z)$ is also just-in-time wrt $\mathcal{D}_0$. We need to show that for every pair of constants c, d, the sentence $\gamma(c, d)$ is consistent with $\mathcal{D}_0 \cup \mathcal{E}$ only if $\gamma(c, d)$ is implied by a combination of possible closures wrt $\mathcal{D}_0$ and the set $\mathcal{E}$. This is true because the possible closures axiom $\chi_1 \vee \chi_2$ specifies the objects that may be near the bomb, which are one of the constants $\mathit{box}_1, \mathit{box}_2, \mathit{agent}$, and then for each of these objects one of the possible closures axioms χ_4, χ_5, χ_6 specifies its status.

Now we specify $\mathsf{pans}(\gamma(y, z), \mathcal{D}_0)$ which is the following set:

$$\big\{(\langle \mathit{box}_1, \mathit{closed}\rangle, \chi_1 \wedge \chi_4),\ (\langle \mathit{box}_2, \mathit{closed}\rangle, \chi_2 \wedge \chi_5),$$
$$(\langle \mathit{agent}, \mathit{ready}\rangle, \chi_1 \wedge \chi_6),\ (\langle \mathit{agent}, \mathit{ready}\rangle, \chi_2 \wedge \chi_6)\big\}.$$

The set $\mathsf{pans}(\gamma(y, z), \mathcal{D}_0)$ specifies all the pairs c, d such that $\gamma(c, d)$ may be consistent with $\mathcal{D}_0 \cup \mathcal{E}$. Note that the set $\mathsf{pans}(\gamma(y, z), \mathcal{D}_0)$ is finite as Lemma 4.3.13 implies. Moreover,

the lemma also implies something stronger about this set, namely that $\gamma(y, z)$ may be satisfied in a model of $\mathcal{D}_0 \cup \mathcal{E}$ only if x and y are interpreted as constants that correspond to a possible answer in $\mathsf{pans}(\gamma(y, z), \mathcal{D}_0)$. By Lemma 4.3.13 the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \gamma(y, z) \equiv$$
$$(y = \mathit{box}_1 \wedge z = \mathit{closed} \wedge \chi_1 \wedge \chi_4) \vee (y = \mathit{box}_2 \wedge z = \mathit{closed} \wedge \chi_2 \wedge \chi_5) \vee$$
$$(y = \mathit{agent} \wedge z = \mathit{ready} \wedge \chi_1 \wedge \chi_6) \vee (y = \mathit{agent} \wedge z = \mathit{ready} \wedge \chi_2 \wedge \chi_6).$$

So, the set $\mathsf{pans}(\gamma(y, z), \mathcal{D}_0)$ essentially provides an alternative representation for $\gamma(y, z)$ when $\mathcal{D}_0 \cup \mathcal{E}$ is assumed to hold. ■

We now proceed to define a class of basic action theories where the initial knowledge base $\mathcal{D}_0$ is a database of possible closures and the successor state axioms are such that the local-effect assumption is relaxed. The idea is that the positive and negative effects formulas of the successor state axioms are required to satisfy a safe-range restriction.

4.3.3 Range-restricted theories

In Section 4.2 we studied basic action theories where a ground action may only affect a fixed set of ground fluents that are directly specified by the arguments of the action. Now we introduce a different class of theories where this condition is relaxed and a ground action may also affect ground fluents that are specified using information from a database of possible closures. We define the *range-restricted* successor state axioms as follows.

Definition 4.3.15. Let the successor state axiom for the fluent symbol F have the following form:

$$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee (F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s)).$$

The successor state axiom for F is *range-restricted* iff both $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^-(\vec{x}, a, s)$ are

disjunctions of formulas of the following form:

$$\exists \vec{z}(a = A(\vec{y}) \wedge \phi(\vec{x}, \vec{z}, s)),$$

where A is an action function, $\vec{y}$ may contain some variables from $\vec{x}$, $\vec{z}$ corresponds to the remaining variables of $\vec{y}$, and the context formula ϕ is uniform in s, does not mention any other free variable other than $\vec{x}, \vec{z}, s$, and is such that $\phi(\vec{x}, \vec{z}, S_0)$ is safe-range wrt the variables in $\vec{x}$ that are not in $\vec{y}$. A basic action theory $\mathcal{D}$ is *range-restricted* iff all the successor state axioms in $\mathcal{D}_{ss}$ are range-restricted in the previous sense, $\mathcal{D}_0$ is a database of possible closures, and the theory also includes the set $\mathcal{E}$ of the unique-names axioms for objects. ■

A range-restricted successor state axiom is similar to a local-effect axiom of Definition 4.2.1. The main difference is that $\vec{y}$ is not required to include *all* the variables from $\vec{x}$, therefore the arguments of the affected fluent atoms $F(\vec{x}, do(\alpha, s))$ are not necessarily fixed by the arguments of the action α and may be specified also by the context formula ϕ. Nonetheless, ϕ needs to be formalized in such way so that it is safe-range wrt the rest of the variables in $\vec{x}$ that are not mentioned in $\vec{y}$. The intuition behind this requirement is that after a is replaced by a ground action α then the positive and negative effects formulas can be rewritten into a range-restricted form, i.e., a form that is safe-range wrt all the variables in $\vec{x}$. First we present a detailed example of a range-restricted basic action theory and then we make this intuition precise by proving the corresponding lemma.

Example 4.3.16 (The simple bomb domain). Let $\mathcal{L}$ be the situation calculus language that consists of the standard logical symbols and the symbols $Poss, do, S_0$, the fluents $Near(x_1, x_2, s)$ and $Status(x_1, x_2, s)$, the action constant *explode*, and the object constants *agent*, box_1, box_2, *bomb*, *ready*, *closed*, *broken*.

The simple bomb domain represents a room where several things are located including the agent, two boxes, and a bomb. The fluent $Near(x_1, x_2, s)$ represents that the object

x_2 is lying near the object x_1 in the situation s, and the fluent $Status(x_1, x_2, s)$ represents that the object x_1 has the status x_2 in s. The action constant *explode* represents the explosion of the bomb which has the effect of setting the status *broken* to all objects that are near the bomb and removing any other status that they may have.

Let $\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_0 \cup \mathcal{E} \cup \mathcal{D}_{fnd}$ be the basic action theory of the *simple bomb domain*, where each of the parts of $\mathcal{D}$ is as follows.

1. $\mathcal{D}_{ap}$ consists of the following sentence:

$$Poss(explode, s) \equiv true.$$

2. $\mathcal{D}_{ss}$ consists of the following two sentences:

$$Near(x_1, x_2, do(a, s)) \equiv Near(x_1, x_2, s)$$

$$Status(x_1, x_2, do(a, s)) \equiv \gamma^{+}_{Status}(x_1, x_2, a, s) \vee (Status(x_1, x_2, s) \wedge \neg\gamma^{-}_{Status}(x_1, x_2, a, s)),$$

where $\gamma^{+}_{Status}(x_1, x_2, a, s)$ is the following formula:

$$a = explode \wedge \ Near(bomb, x_1, s) \wedge x_2 = broken,$$

and $\gamma^{-}_{Status}(x_1, x_2, a, s)$ is the following formula:

$$a = explode \wedge \ Near(bomb, x_1, s) \wedge Status(x_1, x_2, s) \wedge x_2 \neq broken.$$

3. $\mathcal{D}_{una}$ is the empty set.

4. $\mathcal{D}_0$ is the database of possible closures that we specified in Example 4.3.7:

$$\{\chi_1 \vee \chi_2, \chi_3, \chi_4, \chi_5, \chi_6\},$$

where $\chi_1, \ldots, \chi_6$ are as follows:

$$\begin{aligned}
\chi_1: &\quad \forall w.Near(bomb, w, S_0) \equiv (w = agent \vee w = box_1)\\
\chi_2: &\quad \forall w.Near(bomb, w, S_0) \equiv (w = agent \vee w = box_2)\\
\chi_3: &\quad \forall w.Near(agent, w, S_0) \equiv w = bomb\\
\chi_4: &\quad \forall w.Status(box_1, w, S_0) \equiv w = closed\\
\chi_5: &\quad \forall w.Status(box_2, w, S_0) \equiv w = closed\\
\chi_6: &\quad \forall w.Status(agent, w, S_0) \equiv w = ready
\end{aligned}$$

5. $\mathcal{E}$ is the set of unique-names axioms for the constants *agent*, box_1, box_2, *bomb*, *ready*, *closed*, and *broken*.

6. $\mathcal{D}_{fnd}$ is as in Definition 3.2.1.

The successor state axiom for *Near* is range-restricted in a trivial way as both the negative and the positive effects formulas are the empty disjunction, i.e, are logically equivalent to *false*. For the successor state axiom for *Status* the safe-range requirement of Definition 4.3.15 reduces to the requirement that the context formulas:

$$Near(bomb, x_1, S_0) \wedge x_2 = broken,$$

and

$$Near(bomb, x_1, S_0) \wedge Status(x_1, x_2, S_0) \wedge x_2 \neq broken,$$

are safe-range wrt $\{x_1, x_2\}$. This follows easily from Definition 4.3.12 using arguments similar the ones we used in Example 4.3.14 to show that the formula $Near(bomb, y, S_0) \wedge$

Status(y, z, S_0) is safe-range wrt $\{y, z\}$. Therefore all the successor state axioms in $\mathcal{D}_{ss}$ are range-restricted according to Definition 4.3.15 and so the basic action theory of the simple bomb domain is range-restricted. Finally, note that the successor state axiom for *Status* mentions only one action function, i.e, *explode*, which is a constant and has no arguments. The axiom then is clearly not local-effect because x_1, x_2 are not included in the arguments of the action as Definition 4.2.1 requires. ■

The safe-range requirement for the context formulas of a range-restricted successor state axiom implies that the axiom can be simplified wrt a ground action α so that it consists of range-restricted formulas.

Lemma 4.3.17. *Let F be a fluent symbol of $\mathcal{L}$ and α a ground action term. If the successor state axiom for F is range-restricted in the sense of Definition 4.3.15, then assuming that actions have unique names the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ can be simplified so that they are range-restricted in the sense of Definition 4.3.12.* ■

Proof. Let α be the ground term $A(\vec{c})$ and assume that $\gamma_F^+(\vec{x}, a, s)$ is the following singleton disjunction:

$$\exists \vec{z}(a = A(\vec{y}) \wedge \phi(\vec{x}, \vec{z}, s)).$$

Then $\gamma_F^+(\vec{x}, A(\vec{c}), S_0)$ is the following formula:

$$\exists \vec{z}(A(\vec{c}) = A(\vec{y}) \wedge \phi(\vec{x}, \vec{z}, S_0)).$$

By the uniqueness of names for actions it follows that this can be simplified to the following formula:

$$\exists \vec{z}(\vec{c} = \vec{y} \wedge \phi(\vec{x}, \vec{z}, S_0)).$$

By Definition 4.3.15 it follows that $\phi(\vec{x}, \vec{z}, S_0)$ is safe-range wrt the variables in $\vec{x}$ that are not in $\vec{y}$, and by the rule of Definition 4.3.12 for the atomic case it follows that $\vec{c} = \vec{y}$ is safe-range wrt the variables in $\vec{y}$. Therefore by the rule for conjunction it follows that

the formula $\vec{c} = \vec{y} \wedge \phi(\vec{x}, \vec{z}, S_0)$ is safe-range wrt the variables in $\vec{x}, \vec{y}$ which is the same set as the variables in $\vec{x}, \vec{z}$. Finally, by the rule for existential quantification it follows that $\gamma_F^+(\vec{x}, A(\vec{c}), S_0)$ is safe-range wrt the variables in $\vec{x}$ therefore is range-restricted. Observe that if $\gamma_F^+(\vec{x}, a, s)$ is a disjunction of more than one formulas of the appropriate form, then the lemma follows by the rule of Definition 4.3.12 for disjunctions. □

In practice the safe-range requirement for the context formulas captures the intuitive idea that we can use the output of a fluent atom with a ground input in order to determine the input to another fluent and recursively obtain a set of affected fluents. An example follows that illustrates the way in which the range-restricted axioms are more expressive than the local-effect successor state axioms.

Example 4.3.18. Consider the basic action theory $\mathcal{D}$ of the simple bomb domain of Example 4.3.16. Observe that the instances of $Status(x_1, x_2, do(explode, S_0))$ that may be affected by the action *explode* are not specified by the arguments of the action, instead they are specified by information that is explicitly represented in the possible closures wrt $\mathcal{D}_0$. For example consider the context formula of $\gamma^-_{Status}(x_1, x_2, \alpha, S_0)$ after we simplify based on the uniqueness of names for actions, i.e.,

$$Near(bomb, x_1, S_0) \wedge Status(x_1, x_2, S_0) \wedge x_2 \neq broken,$$

and a model of $\mathcal{D}_0 \cup \mathcal{E}$ that satisfies the possible closures $\chi_1, \chi_3, \chi_4, \chi_5, \chi_6$: the fluent atom $Near(bomb, x_1, S_0)$ specifies all the objects x_1 that are near the bomb, namely *agent* and box_1, and then for each object x_1 the fluent atom $Status(x_1, x_2, S_0)$ and the equality atom $x_2 \neq broken$ specify all the objects x_2 such that the truth value of the atom $Status(x_1, x_2, do(explode, S_0))$ is necessarily false due to the action *explode*. These are the following: $x_2 = closed$ for $x_1 = box_1$ and $x_2 = ready$ for $x_1 = agent$.

Essentially, the arguments of the ground fluent atoms that may be affected by *explode* are recursively traced back to the output of a fluent atom with a ground input, in a logic

programming manner. In this way the action *explode* may affect in general any finite number of ground fluent atoms that are specified by the information in $\mathcal{D}_0$. This cannot be the case for a local-effect successor state axiom as the arguments of the ground fluent atoms that may be affected by *explode* are required to be included in the arguments of the action. ■

We now proceed to present a method for progressing a range-restricted theory provided that the just-in-time assumption holds for the context formulas of the successor state axioms.

4.3.4 Just-in-time progression for range-restricted theories

As we showed in Lemma 4.3.17, the range-restricted theories are defined in such way so that for any ground action α and any fluent F, the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ can be simplified to a form that is range-restricted. When the simplified formulas are also just-in-time wrt $\mathcal{D}_0$ then we can use the possible answers to these formulas wrt $\mathcal{D}_0$ and Lemma 4.3.13 to further simplify each one in a convenient form that is suitable for specifying the *characteristic set* and the *context set* of α, and progress $\mathcal{D}_0$ using similar ideas as those in Section 4.2. First, we state formally the condition under which our method for progression is logically correct and then we introduce the normal form we assume for $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$.

Definition 4.3.19. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a ground action term. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ of the successor state axiom for F have been simplified to the range-restricted form that Lemma 4.3.13 implies. Then, $\mathcal{D}$ is *just-in-time* wrt α iff for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ are just-in-time wrt $\mathcal{D}_0$ and for all constant vectors $\vec{c}$ in the argument set of F, $F(\vec{c}, w, S_0)$ is mentioned in $\mathcal{D}_0$. ■

Note that this is a strong condition as it requires that the fluent atoms with a ground input that may be affected by α as well as the fluent atoms with a ground input on which the change may be conditioned, are all mentioned in $\mathcal{D}_0$. This does not imply that $\mathcal{D}_0$ has complete information for the output of these fluent atoms, but it is required that there is at least some disjunctive information in the form of possible closures axioms.

When a theory is range-restricted and just-in-time wrt a ground action α we will typically assume that the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ are in *context normal form* in the following sense:

Definition 4.3.20. Let $\mathcal{D}$ be a basic action theory that is range-restricted, α a ground action term such that $\mathcal{D}$ is just-in-time wrt α, and $\gamma(\vec{x})$ the formula $\gamma_F^+(\vec{x}, \alpha, S_0)$ or $\gamma_F^-(\vec{x}, \alpha, S_0)$ of the successor state axiom for F. Let $\gamma'(\vec{x})$ be the range-restricted version of $\gamma(\vec{x})$ as Lemma 4.3.17 implies. By Lemma 4.3.13 and the fact that $\mathcal{D}$ is just-in-time wrt α it follows that $\mathsf{pans}(\gamma'(\vec{x}), \mathcal{D}_0)$ is a finite set $\{(\vec{c}_1, \chi_1), \ldots, (\vec{c}_n, \chi_n)\}$ such that every χ_i is a conjunction of possible atomic closures wrt $\mathcal{D}_0$ and every c_i consists of constants in $\mathcal{D}_0$ and $\gamma'(\vec{x})$. Now let $\mathcal{K}$ be the smallest set of pairs such that for every $(\vec{c}, \chi) \in \mathsf{pans}(\gamma'(\vec{x}), \mathcal{D}_0)$, $\mathcal{K}$ includes all pairs $(\vec{c}, \phi)$, where ϕ is a minimal conjunction of possible closures wrt $\mathcal{D}_0$ such that the atomic closures of χ are mentioned in ϕ. It follows that $\mathcal{K}$ is a finite set of the form $\{(\vec{d_1}, \phi_1), \ldots, (\vec{d_m}, \phi_m)\}$. The *context normal form* for $\gamma(\vec{x})$ is then following formula:

$$\bigvee_{i=1}^{m} (\vec{x} = \vec{d_i} \wedge \phi_i).$$

■

The context normal form for a formula $\gamma_F^+(\vec{x}, \alpha, S_0)$ or $\gamma_F^-(\vec{x}, \alpha, S_0)$ is essentially obtained in three steps: first we obtain a range-restricted version of the formula based on Lemma 4.3.17, then we simplify the resulting formula based on the set $\mathsf{pans}(\gamma_F^+, \mathcal{D}_0)$ according to Lemma 4.3.13, and finally we augment the resulting formula so that each of the sub-formulas ϕ_i is a conjunction of possible closures wrt $\mathcal{D}_0$. An example follows.

Example 4.3.21. Consider the basic action theory of the simple bomb domain of Example 4.2.2 and let α be the ground action *explode*. Let $\gamma'(x_1, x_2)$ be the formula we obtain after we simplify $\gamma^{-}_{Status}(x_1, x_2, \alpha, S_0)$ based on the uniqueness of names for actions, i.e.,

$$Near(bomb, x_1, S_0) \wedge Status(x_1, x_2, S_0) \wedge x_2 \neq broken.$$

As we argued in Example 4.3.16 $\gamma'(x_1, x_2)$ is range-restricted. Moreover, using a similar argument as in Example 4.3.14 it is easy to show that is is also just-in-time wrt $\mathcal{D}_0$. As expected then by Lemma 4.3.13 the set of possible answers to $\gamma'(x_1, x_2)$ wrt $\mathcal{D}_0$ is the following finite set:

$$\big\{(\langle box_1, closed\rangle, \chi_1 \wedge \chi_4),\ (\langle box_2, closed\rangle, \chi_2 \wedge \chi_5),$$
$$(\langle agent, ready\rangle, \chi_1 \wedge \chi_6),\ (\langle agent, ready\rangle, \chi_2 \wedge \chi_6)\big\}.$$

In this simple case the set $\mathcal{K}$ of Definition 4.3.20 coincides with $\mathsf{pans}(\gamma'(x_1, x_2), \mathcal{D}_0)$ and so the context normal form for $\gamma^{-}_{Status}(x_1, x_2, \alpha, S_0)$ is the following formula:

$$(x_1 = box_1 \wedge x_2 = closed \wedge \chi_1 \wedge \chi_4) \vee (x_1 = box_2 \wedge x_2 = closed \wedge \chi_2 \wedge \chi_5) \vee$$
$$(x_1 = agent \wedge x_2 = ready \wedge \chi_1 \wedge \chi_6) \vee (x_1 = agent \wedge x_2 = ready \wedge \chi_2 \wedge \chi_6).$$

In the general case though $\mathcal{K}$ is a different set. For instance if χ_4 was mentioned in $\mathcal{D}_0$ as part of the non-atomic possible closure $\chi_4 \wedge \chi_7$, then $\mathcal{K}$ would include the pair $(\langle box_1, closed\rangle, \chi_1 \wedge \chi_4 \wedge \chi_7)$ instead of the pair $(\langle box_1, closed\rangle, \chi_1 \wedge \chi_4)$. Note also that in the general case $\mathcal{K}$ may have more elements than $\mathsf{pans}(\gamma'(x_1, x_2), \mathcal{D}_0)$.

Finally, using a similar argument it is easy to show that the context normal form for $\gamma^{+}_{Status}(x_1, x_2, \alpha, S_0)$ is the following formula:

$$(x_1 = box_1 \wedge x_2 = broken \wedge \chi_1) \vee (x_1 = agent \wedge x_2 = broken \wedge \chi_1) \vee$$
$$(x_1 = box_2 \wedge x_2 = broken \wedge \chi_2) \vee (x_1 = agent \wedge x_2 = broken \wedge \chi_2).$$

■

The next lemma shows that in the context of a theory that is range-restricted and just-in-time wrt α it is safe to assume without loss of generality that all the formulas of the form $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ are in context normal form.

Lemma 4.3.22. *Let $\mathcal{D}$ be a basic action theory that is range-restricted, α a ground action term such that $\mathcal{D}$ is just-in-time wrt α, and $\gamma(\vec{x})$ the formula $\gamma_F^+(\vec{x}, \alpha, S_0)$ or $\gamma_F^-(\vec{x}, \alpha, S_0)$ of the successor state axiom for F. Let the following formula be the context normal form for $\gamma(\vec{x})$:*

$$\bigvee_{i=1}^{m}(\vec{x} = \vec{d_i} \wedge \phi_i).$$

Then the following holds:

$$\mathcal{D}_0 \cup \mathcal{D}_{una} \cup \mathcal{E} \models \forall\vec{x}.\gamma(\vec{x}) \equiv \bigvee_{i=1}^{m}(\vec{x} = \vec{d_i} \wedge \phi_i).$$

■

Proof. Let $\gamma'(\vec{x})$ be the range-restricted version of $\gamma(\vec{x})$ as Lemma 4.3.17 implies. It follows that

$$\mathcal{D}_{una} \models \forall\vec{x}.\gamma(\vec{x}) \equiv \gamma'(\vec{x}).$$

By Lemma 4.3.13 it follows that $\mathsf{pans}(\gamma'(\vec{x}), \mathcal{D}_0)$ is a finite set $\{(\vec{c}_1, \chi_1), \ldots, (\vec{c}_n, \chi_n)\}$ such that every χ_i is a conjunction of possible atomic closures wrt $\mathcal{D}_0$ and every c_i consists of constants in $\mathcal{D}_0$, and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall\vec{x}.\gamma'(\vec{x}) \equiv \bigvee_{i=1}^{n}(\vec{x} = \vec{c_i} \wedge \chi_i).$$

Therefore it follows that

$$\mathcal{D}_0 \cup \mathcal{D}_{una} \cup \mathcal{E} \models \forall\vec{x}.\gamma(\vec{x}) \equiv \bigvee_{i=1}^{n}(\vec{x} = \vec{c_i} \wedge \chi_i).$$

Now let $\mathcal{K}$ be the set $\{(\vec{d_1}, \phi_1), \ldots, (\vec{d_m}, \phi_m)\}$ of Definition 4.3.20. It suffices to show that

$$\mathcal{D}_0 \cup \mathcal{D}_{una} \cup \mathcal{E} \models \forall \vec{x}. \bigvee_{i=1}^{n} (\vec{x} = \vec{c_i} \wedge \chi_i) \equiv \bigvee_{i=1}^{m} (\vec{x} = \vec{d_i} \wedge \phi_i).$$

Let M be an arbitrary model of $\mathcal{D}_0 \cup \mathcal{D}_{una} \cup \mathcal{E}$. For the ($\Leftarrow$) direction observe that for every pair $(\vec{d_i}, \phi_i)$ there is a pair $(\vec{c_k}, \chi_k)$ such that $\vec{c_k} = \vec{d_i}$ and χ_k is a sub-formula of ϕ_i. For the ($\Rightarrow$) let $M \models \chi_k$. Observe that there is a possible closure ϕ wrt $\mathcal{D}_0$ such that $M \models \phi$ and either ϕ is χ_k or χ_k is a sub-formula of ϕ. By the way we defined $\mathcal{K}$ it follows that $(\vec{c_k}, \phi)$ is in $\mathcal{K}$. □

A simple application of the previous lemma follows.

Example 4.3.23. Consider the basic action theory of the simple bomb domain of Example 4.2.2 and let α be the ground action *explode*. Consider the context normal form for the formula $\gamma^{-}_{Status}(x_1, x_2, \alpha, S_0)$ as we specified in Example 4.3.21. By Lemma 4.3.22 the following holds:

$$\begin{aligned}\mathcal{D}_0 \cup \mathcal{D}_{una} \cup \mathcal{E} \models\ & \gamma^{-}_{Status}(x_1, x_2, \mathit{explode}, S_0) \equiv \\ & (x_1 = \mathit{box}_1 \wedge x_2 = \mathit{closed} \wedge \chi_1 \wedge \chi_4) \vee (x_1 = \mathit{box}_2 \wedge x_2 = \mathit{closed} \wedge \chi_2 \wedge \chi_5) \vee \\ & (x_1 = \mathit{agent} \wedge x_2 = \mathit{ready} \wedge \chi_1 \wedge \chi_6) \vee (x_1 = \mathit{agent} \wedge x_2 = \mathit{ready} \wedge \chi_2 \wedge \chi_6).\end{aligned}$$

A similar condition holds for $\gamma^{+}_{Status}(x_1, x_2, \alpha, S_0)$ and the context normal form that we specified in Example 4.3.21. ■

Now we proceed to specify the *characteristic set* of α in almost the exact same way as we did in Section 4.2 for the local-effect theories. The only difference is that we rely on a different lemma to transform the successor state axioms into a convenient form and that we do not use the situation-suppressed notation.

Definition 4.3.24. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a ground action term such that $\mathcal{D}$ is just-in-time wrt α. Without loss of generality we

assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ of the successor state axiom for F are in context normal form. We define the *argument set* of F wrt α as the following set $\mathcal{C}_F$ of object constant vectors:

$$\mathcal{C}_F = \{\vec{c} \mid \vec{x} = \vec{c} \text{ appears in } \gamma_F^+(\vec{x}, \alpha, S_0) \text{ or } \gamma_F^-(\vec{x}, \alpha, S_0)\}.$$

We define the *characteristic set* of α as the following set $\mathcal{G}$ of ground fluent atoms:

$$\mathcal{G} = \{F(\vec{c}, S_0) \mid \vec{c} \text{ in } \mathcal{C}_F \text{ for some } F \text{ in } \mathcal{L}\}.$$ ■

The characteristic set $\mathcal{G}$ of α specifies the ground fluent atoms that may be affected by the ground action α. Since there are only finitely many fluent symbols in $\mathcal{L}$ it follows that $\mathcal{G}$ is always finite. Moreover, by the fact that the set of possible answers to a formula is unique and the way the context normal form is defined it follows that $\mathcal{G}$ is also unique. A simple example follows.

Example 4.3.25. Consider the basic action theory of the simple bomb domain of Example 4.2.2 and let α be the ground action *explode*. We assume that the formulas $\gamma_{Status}^-(x_1, x_2, \alpha, S_0)$ and $\gamma_{Status}^+(x_1, x_2, \alpha, S_0)$ are in context normal form as we specified in Example 4.3.21. By Definition 4.3.24 it follows that the argument set of *Near* wrt α is the empty set, and the argument set of *Status* wrt α is the following set:

$$\{\langle box_1, closed\rangle, \langle box_1, broken\rangle, \langle box_2, closed\rangle,$$
$$\langle box_2, broken\rangle, \langle agent, ready\rangle, \langle agent, broken\rangle\}.$$

The characteristic set of α is then the following set:

$$\{Status(box_1, closed), Status(box_1, broken), Status(box_2, closed),$$
$$Status(box_2, broken), Status(agent, ready), Status(agent, broken)\}.$$

This set lists all the ground fluent atoms whose truth value may change when the action *explode* is performed in S_0. ■

Based on the characteristic set of α we can define the set of *unaffected* formulas wrt α in exactly the same way as we did for the strictly local-effect theories in Definition 4.2.9. We include the definition for clarity.

Definition 4.3.26. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a ground action term. We define the set $\mathcal{U}$ of situation-suppressed first-order formulas as the smallest set such that the following conditions hold:

1. all the (situation-independent) equality atoms and their negation are in $\mathcal{U}$;
2. the formulas $\bigwedge_{i=1}^{n} \vec{x} \neq \vec{c}_i \wedge F(\vec{x})$ and $\bigwedge_{i=1}^{n} \vec{x} \neq \vec{c}_i \wedge \neg F(\vec{x})$ are in $\mathcal{U}$, where $\{\vec{c}_1, \ldots, \vec{c}_n\}$ is the argument set of F wrt α;
3. $\mathcal{U}$ is closed under the logical operators $\wedge, \vee$, and universal and existential quantification over objects.

When a formula ϕ is in $\mathcal{U}$ we say that ϕ is *unaffected* wrt α. ■

The next lemma follows easily using the same proof method as the one for Lemma 4.2.10.

Lemma 4.3.27. *Let $\mathcal{D}$ be a basic action theory that is range-restricted, α a ground action term, ϕ a situation-suppressed formula that is unaffected wrt α, M a model of $\mathcal{D}$, and μ a variable assignment. Then,*

$$M, \mu \models \phi[S_0] \text{ iff } M, \mu \models \phi[do(\alpha, S_0)].$$

■

A simple example follows.

Example 4.3.28. Consider the sentence χ_3 of the database of possible closures $\mathcal{D}_0$ of the simple bomb domain that we specified in Example 4.3.16 and let ϕ be the situation-suppressed version of χ_3:

$$\forall w.Near(agent, w) \equiv w = bomb.$$

It is straightforward that $\phi[S_0]$ is logically equivalent to $\phi'[S_0]$, where ϕ' is the following situation-suppressed sentence:

$$\forall x(x = agent \supset \forall w(Near(x, w) \equiv w = bomb)).$$

Let α be the ground action *explode*. As it was shown in Example 4.3.25 the argument set of *Near* wrt α is the empty set, therefore it follows easily by Definition 4.3.26 that ϕ' is unaffected wrt α. By Lemma 4.3.27 it then follows that $\mathcal{D} \models \phi[S_\alpha]$. ■

The intuition is that for every possible closures axiom $\phi[S_0]$ for $\vec{\tau}$ in $\mathcal{D}_0$ such that the characteristic $\mathcal{G}$ wrt α does not include any ground instance of any of the atoms in $\vec{\tau}$, it follows that ϕ is unaffected wrt α. These axioms correspond to the part of $\mathcal{D}_0$ that is preserved when progressing $\mathcal{D}_0$ wrt α.

Now we turn our attention to the part of $\mathcal{D}_0$ that needs updating and the *context set* of α. Note that the context normal form we assume for the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ is a disjunction where each disjunct is a conjunction of an equality atom of the form $\vec{x} = \vec{c}_i$ and a conjunction ϕ_i of possible closures wrt $\mathcal{D}_0$. Essentially ϕ_i specifies the condition that a model of $\mathcal{D}$ needs to satisfy so that the truth value of the fluent atom $F(\vec{c}_i, S_0)$ changes when α is performed in S_0. In order to specify the context set of α we need to collect all the fluent atoms with a ground input that are mentioned in the sentences ϕ_i.

Definition 4.3.29. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a

ground action term such that $\mathcal{D}$ is just-in-time wrt α. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, s)$ and $\gamma_F^-(\vec{x}, \alpha, s)$ in $\mathcal{D}_{ss}$ are in context normal form. Let $\mathcal{G}$ be the characteristic set of α. We define the *context set* $\mathcal{J}$ of α as the smallest set that includes all the fluent atoms with a ground input $F(\vec{c}, w, S_0)$ such that one of the following holds:

- $F(\vec{c}, e, S_0)$ is in $\mathcal{G}$ for some constant e in $\mathcal{L}$;
- $F(\vec{c}, w, S_0)$ appears in $\gamma_F^+(\vec{x}, \alpha, S_0)$ or $\gamma_F^-(\vec{x}, \alpha, S_0)$ for some F in $\mathcal{L}$. ■

Similar to the characteristic set, the context set $\mathcal{J}$ of α is always finite and it is unique. A simple example follows.

Example 4.3.30. Consider the basic action theory of the simple bomb domain of Example 4.3.16, and let α be the ground action *explode*. We assume that the formulas $\gamma_{Status}^-(x_1, x_2, \alpha, S_0)$ and $\gamma_{Status}^+(x_1, x_2, \alpha, S_0)$ are in context normal form as we specified in Example 4.3.21. It follows that the context set $\mathcal{J}$ of α is the following:

$$\{Status(box_1, w, S_0), Status(box_2, w, S_0), Status(agent, w, S_0), Near(bomb, w, S_0)\}.$$ ■

As in the strictly local-effect theories a $\mathcal{J}$-model is a partial model that specifies an interpretation for all the ground atoms in $\mathcal{J}$. In the case of the strictly local-effect theories a $\mathcal{J}$-model is simply a conjunction of literals. Here, a $\mathcal{J}$-model is a closure of the atoms in $\mathcal{J}$.

Definition 4.3.31. Let $\mathcal{D}$ be a basic action theory that is range-restricted, α a ground action term such that $\mathcal{D}$ is just-in-time wrt α, and $\mathcal{J} = \{\tau_1, \ldots, \tau_n\}$ the context set of α, where $\tau_1, \ldots, \tau_n$ appear in lexicographical order. A *$\mathcal{J}$-model* θ is a closure of the vector $\langle \tau_1, \ldots, \tau_n \rangle$ such that for every i, $1 \leq i \leq n$, the atomic closure of τ_i in χ is a possible atomic closure wrt $\mathcal{D}_0$. ■

A simple example follows.

Example 4.3.32. Consider the basic action theory of the simple bomb domain of Example 4.3.16, and let α be the ground action *explode* and $\mathcal{J}$ the context set of α as in Example 4.3.30. Let $\vec{\tau}$ be the following vector of fluent atoms with a ground input that corresponds to the atoms in the context set $\mathcal{J}$:

$$\langle Status(box_1, w, S_0), Status(box_2, w, S_0), Status(agent, w, S_0), Near(bomb, w, S_0)\rangle.$$

Then the closure θ_1 of $\vec{\tau}$ on the vector

$$\langle\{closed\}, \{closed\}, \{ready\}, \{agent, box_1\}\rangle$$

is a $\mathcal{J}$-model. Note that θ_1 is a conjunction of the atomic closures $\chi_1, \chi_4, \chi_5, \chi_6$, each of which is a possible closure wrt $\mathcal{D}_0$. Similarly, the closure θ_2 of $\vec{\tau}$ on the vector

$$\langle\{closed\}, \{closed\}, \{ready\}, \{agent, box_2\}\rangle$$

is also a $\mathcal{J}$-model. Finally, note that there is no other $\mathcal{J}$-model other than θ_1 and θ_2. ■

Since $\mathcal{J}$ is finite and a $\mathcal{J}$-model consists of possible closures wrt $\mathcal{D}_0$ it follows that there are always finitely many $\mathcal{J}$-models. The disjunction ϕ then of all the $\mathcal{J}$-models is a possible closures axiom that corresponds to the "cross-product" of several axioms in $\mathcal{D}_0$. The intuition is that we can progress $\mathcal{D}_0$ by replacing the axioms of $\mathcal{D}_0$ that mention any of the atoms in $\mathcal{J}$ by the larger possible closures axiom that consists of the $\mathcal{J}$-models, and progressing each of the $\mathcal{J}$-models accordingly.

Definition 4.3.33. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a ground action term such that $\mathcal{D}$ is just-in-time wrt α. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, s)$ and

$\gamma_F^-(\vec{x}, \alpha, s)$ of the successor state axiom for F are in context normal form. Let $\mathcal{J} = \{F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\}$ be the context set of α, and θ be a $\mathcal{J}$-model such that θ is the closure of $\langle F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\rangle$ on $\langle V_1, \ldots, V_n\rangle$. We define the set Γ_i^+ as the smallest set of constants e such that the following conditions hold:

1. $\vec{x} = \vec{d} \wedge \chi$ is a disjunct of $\gamma_{F_i}^+(\vec{x}, \alpha, s)$;
2. $\vec{d}$ is a vector of constants of the form $\langle \vec{c}_i, e\rangle$;
3. $\{\chi \wedge \theta\} \cup \mathcal{E}$ is consistent.

The set Γ_i^- is defined similarly based on the formula $\gamma_{F_i}^-(\vec{x}, \alpha, s)$ instead of $\gamma_{F_i}^+(\vec{x}, \alpha, s)$. The *progression* of θ wrt α is the closure of $\langle F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\rangle$ on the updated vector $\langle V_1', \ldots, V_n'\rangle$, where for all i, $1 \leq i \leq n$, V_i' is the set $(V_i - \Gamma_i^-) \cup \Gamma_i^+$. ■

A $\mathcal{J}$-model θ specifies an interpretation for all the ground fluent atoms that may be affected by α as well as the ground fluent atoms that the change is conditioned on. The progression of θ then only differs from θ in that some ground fluent atoms may have a different truth value. Essentially this amounts to updating sets of constants in $\langle V_1, \ldots, V_n\rangle$ so that a constant e is either removed or added to the closure of $F(\vec{c}_i, w, S_0)$ provided that the condition ϕ for the change is consistent with θ.

Example 4.3.34. Consider the basic action theory of the simple bomb domain of Example 4.3.16, and let α be the ground action *explode* and $\mathcal{J}$ the context set of α as in Example 4.3.30. Let $\vec{\tau}$ be the following vector of fluent atoms with a ground input that corresponds to the atoms in the context set $\mathcal{J}$:

$$\langle \mathit{Status}(\mathit{box}_1, w, S_0), \mathit{Status}(\mathit{box}_2, w, S_0), \mathit{Status}(\mathit{agent}, w, S_0), \mathit{Near}(\mathit{bomb}, w, S_0)\rangle,$$

and θ_1, θ_2 as in Example 4.3.32. The progression of the $\mathcal{J}$-model θ_1 is the closure θ_1^* of $\vec{\tau}$ on the vector

$$\langle \{\mathit{broken}\}, \{\mathit{closed}\}, \{\mathit{broken}\}, \{\mathit{agent}, \mathit{box}_1\}\rangle.$$

Similarly, the progression of the $\mathcal{J}$-model θ_2 is the closure θ_2^* of $\vec{\tau}$ on the vector

$$\langle\{closed\},\{broken\},\{broken\},\{agent, box_2\}\rangle.$$ ■

We now state the main result of this section that illustrates how the new database is constructed from $\mathcal{D}_0$.

Theorem 4.3.35. *Let $\mathcal{D}$ be a basic action theory that is range-restricted and has a finite $\mathcal{D}_0$, and α a ground action such that $\mathcal{D}$ is just-in-time wrt α. Let $\mathcal{J}$ be the context set of α, $\{\theta_1[S_0],\ldots,\theta_n[S_0]\}$ the set of all the $\mathcal{J}$-models, and $\{\phi_1[S_0],\ldots,\phi_m[S_0]\}$ the set of all axioms in $\mathcal{D}_0$ that do not mention any fluent atom in $\mathcal{J}$. Let $\mathcal{D}_\alpha$ be the following set:*

$$\Big\{\bigvee_{i=1}^{n}\theta_i^*[S_\alpha],\ \phi_1[S_\alpha],\ldots,\phi_m[S_\alpha]\Big\},$$

where $\theta_i^*[S_0]$ is the progression of $\theta_i[S_0]$ wrt α. Then, $\mathcal{D}_\alpha$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. ■

Proof. Observe that the set $\{\phi_1,\ldots,\phi_m\}$ of situation-suppressed axioms can be rewritten into a logically equivalent form such that each sentence in the set is unaffected wrt α, and that $\theta_i^*[S_\alpha]$ is a correct progression of $\theta_i[S_0]$. The formal argument follows the proof method for Theorem 4.2.30 for the strictly local-effect theories: the only difference is that here we rely on Lemma 4.3.13 to simplify the positive and negative effects formulas in the axioms of $\mathcal{D}_{ss}$ and we use closures instead of literals. □

A simple example of the application of Theorem 4.2.30 follows.

Example 4.3.36. Consider the basic action theory of the simple bomb domain of Example 4.3.16, and let α be the ground action *explode*. As shown in Example 4.3.30 the context set $\mathcal{J}$ of α is the set

$$\{Status(box_1,w,S_0), Status(box_2,w,S_0), Status(agent,w,S_0), Near(bomb,w,S_0)\}.$$

The set of all the $\mathcal{J}$-models is the set $\{\theta_1, \theta_2\}$, where θ_1, θ_2 are as in Example 4.3.32. The initial knowledge base $\mathcal{D}_0$ is the set $\{\chi_1 \vee \chi_2, \chi_3, \chi_4, \chi_5, \chi_6\}$ of possible closures axioms, where all the axioms except for χ_3 mention a fluent atom in $\mathcal{J}$. Let $\mathcal{D}_\alpha$ be the following set:

$$\big\{\theta_1^*[S_\alpha] \vee \theta_2^*[S_\alpha],\ \chi_3[S_\alpha]\big\},$$

where $\theta_1^*[S_0]$ is the progression of $\theta_1[S_0]$ and $\theta_2^*[S_0]$ the progression of $\theta_1[S_0]$ as in Example 4.3.34. By Theorem 4.3.35 it then follows that $\mathcal{D}_\alpha$ is a progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. That is, $\mathcal{D}_\alpha$ is the set that consists of the following two sentences:

$$\begin{aligned}
&\big(\forall w(\mathit{Status}(\mathit{box}_1, w, S_\alpha) \equiv w = \mathit{broken}) \wedge \forall w(\mathit{Status}(\mathit{box}_2, w, S_\alpha) \equiv w = \mathit{closed}) \wedge \\
&\ \forall w(\mathit{Status}(\mathit{agent}, w, S_\alpha) \equiv w = \mathit{broken}) \wedge \\
&\ \forall w(\mathit{Near}(\mathit{bomb}, w, S_\alpha) \equiv (w = \mathit{agent} \vee w = \mathit{box}_1))\big) \quad \vee \\
&\big(\forall w(\mathit{Status}(\mathit{box}_1, w, S_\alpha) \equiv w = \mathit{closed}) \wedge \forall w(\mathit{Status}(\mathit{box}_2, w, S_\alpha) \equiv w = \mathit{broken}) \wedge \\
&\ \forall w(\mathit{Status}(\mathit{agent}, w, S_\alpha) \equiv w = \mathit{broken}) \wedge \\
&\ \forall w(\mathit{Near}(\mathit{bomb}, w, S_\alpha) \equiv (w = \mathit{agent} \vee w = \mathit{box}_2))\big),
\end{aligned}$$

and

$$\forall w.\mathit{Near}(\mathit{agent}, w, S_0) \equiv w = \mathit{bomb}.$$

■

Theorem 4.3.35 shows a method for progressing such a $\mathcal{D}_0$ wrt a ground action α under the assumption that α is just-in-time wrt $\mathcal{D}_0$. This is a semantic assumption that captures the intuitive idea that the effects of α may depend only on information that is explicitly represented in $\mathcal{D}_0$ in terms of possible closures. For instance in the simple bomb domain of Example 4.3.36, it is not known whether the object box_1 is near the bomb, but it is known that there are exactly two possible closures for the fluent atom $\mathit{Near}(\mathit{bomb}, w, S_0)$. If there was no axiom in $\mathcal{D}_0$ that listed the possible closures of $\mathit{Near}(\mathit{bomb}, w, S_0)$ then we would not be able to use our progression method.

4.4 Concluding remarks

In this chapter we presented three cases where it is possible to obtain a first-order progression that is correct. In Section 4.1 we examined a case where a weak progression is always correct when we restrict our attention to a practical class of queries that allow a limited form of quantification over situations. In Sections 4.2 and 4.3 we examined two cases where a weak progression qualifies as a strong progression (hence is correct) when we restrict the expressiveness of the action theory. In these cases unrestricted queries are allowed but actions are limited to local or range-restricted effects.

As we also mentioned in Section 4.1, Lin and Reiter [1997] were the first to note that a weak progression may be correct with respect to certain queries even though it may not qualify as a strong progression. Lin and Reiter showed that as long as we restrict our attention to queries that are uniform in some (future) situation term, then a weak progression is always correct. A similar but weaker result is due to Shirazi and Amir [2005]. Shirazi and Amir showed that for the cases that a strong progression is first-order definable their variant of weak progression is correct for answering queries uniform in some situation term. The main result of Section 4.1, namely Theorem 4.1.12, goes beyond the previous results as it shows that a weak progression is always correct for a practical class that includes queries that are not uniform in any situation term.

Theorem 4.1.12 of Section 4.1 shares intuitions with a result that appears in [Savelli, 2006]. In the context of regression Savelli showed that whenever a basic action theory entails that there exists a situation satisfying a condition, then at least one such situation must be found within a *predetermined distance* from the initial situation. The proof of this result relies on two ideas that we also used for the proof of Theorem 4.1.12, namely the treatment of the existential quantification over situations in the proof of Lemma 4.1.11 and the use of the Compactness Theorem in the proof of Lemma 3.2.18. Our work separates the use of these two ideas so that Lemma 4.1.10 essentially specifies a proof method for showing similar results as Theorem 4.1.12 based on different assumptions for

the class of queries in question.

Lin and Reiter [1997] were also the first to identify cases where a strong progression is always first-order definable. As we showed in the end of Section 4.1 in Lemma 4.1.13, this is equivalent to the condition that a weak progression qualifies as a strong progression. Lin and Reiter suggested some strong syntactic restrictions on the structure of the action theories that allow for a first-order strong progression, including the *context-free* restriction for actions and the *relatively complete* assumption for the initial knowledge base. Along the same lines Shirazi and Amir [2005] introduced the *unit-case* actions and the *permuted* domains, and focused on the complexity of computing their variant of weak progression.

Liu and Levesque [2005] introduced a class of theories that are more expressive than the theories studied by Lin and Reiter and Shirazi and Amir. Liu and Levesque introduced the *local-effect* assumption for actions and the *proper* knowledge bases, and proposed a version of progression that is logically incomplete but remains first-order and tractable. The main results of Section 4.2, namely Theorems 4.2.16 and 4.2.30, extend this line of research as they show that a weak progression always qualifies as a strong progression when actions have local effects and the initial knowledge base is unrestricted, and also specify a method for computing the progressed knowledge base under a slightly stronger assumption.

Nonetheless, the local-effect assumption is quite restrictive and cannot represent scenarios where the effects of actions are specified by "indexical" information in the knowledge base rather than information in the arguments of the action. For example this is the case for an action with no arguments of the form *move-fw* that essentially results in the agent moving toward the direction she is facing. In Section 4.3 we examined a class of theories that go beyond the local-effect assumption and are able to represent this kind of actions. The main result of Section 4.3, namely Theorem 4.3.35, shows that when actions are range-restricted, the initial knowledge base is a database of possible closures, and a

just-in-time condition holds, then a weak progression qualifies as a strong progression.

The just-in-time assumption essentially captures the intuition that the knowledge base contains disjunctive information for all the "indexical" properties that are needed for specifying the effects of the action. This can be a limitation when a range-restricted theory is used *offline* where no other information may be available other than the knowledge base itself. Nonetheless, in settings where the action theory is used by an agent that is able to interact with the environment *online* and get information, we can overcome this limitation by using sensing actions that may provide the missing information.

Liu and Lakemeyer [2009] also worked toward extending the results of Section 4.2 about the local-effect theories. Liu and Lakemeyer first illustrate the strong connection between the progression of a basic action theory and the concept of *forgetting* [Lin and Reiter, 1994]. Using this observation they provide an alternative proof for Theorem 4.2.16, that is, the fact that a strong progression of a local-effect theory is always first-order definable, that also specifies a method for computing the progression without the strictly local-effect restriction of Theorem 4.2.30. Their method is based on the same technique that we used in our results about local-effect theories, namely that we can separate a formula into two parts: one that can be easily progressed by updating the truth value of ground fluent atoms, and one that remains unaffected by the action in question. In our progression method for the strictly local-effect theories we assume a convenient syntactic structure for the successor state axioms in $\mathcal{D}_{ss}$ and apply this technique to the initial knowledge base $\mathcal{D}_0$ while Liu and Lakemeyer apply this technique both to $\mathcal{D}_0$ and the axioms in $\mathcal{D}_{ss}$ to obtain the progressed knowledge base. The main difference between the two approaches is that in the case of Liu and Lakemeyer the progressed knowledge base needs to incorporate sentences from $\mathcal{D}_{ss}$ unlike our approach where the progressed knowledge base is obtained by updating the sentences in $\mathcal{D}_0$. The intuition behind our approach is that it is often easier for the progression method to preserve the syntactic structure of $\mathcal{D}_0$. For example, following this idea in Section 4.3 we presented a progres-

sion method for a type of theories where $\mathcal{D}_0$ is a database of possible closures, which is preserved after we perform a progression step.

Liu and Lakemeyer [2009] also investigated a class of theories that goes beyond the local-effect assumption. They observe that the effects of non-local-effect actions often do not depend on the fluents on which they have non-local effects. For example, moving a briefcase will move all the objects in it as well without affecting the fluent that represents which objects are in the briefcase. Based on this observation they introduce the *normal* actions that may have non-local effects as long all the fluents that appear in the positive and negative effects formulas are not affected by the action or the action has local-effects on those fluents. For this case Liu and Lakemeyer provide a method for computing a first-order strong progression when the theory has a *proper*$^+$ initial knowledge base. Compared to the range-restricted theories of Section 4.3 we note that both classes of theories have their limitations. For example, a normal action cannot be used represent the effects of a bomb exploding that may affect the status of the objects near the bomb as well as their location. The range-restricted theories can represent this scenario but on the other hand the definability of progression in this case relies on the just-in-time assumption, which as we noted earlier can be problematic in offline settings.

We should also note that our work in Section 4.3 is influenced a lot by ideas and syntactic restrictions that are common in databases and logic programming. The notion of input and output arguments is similar to that of *modes* in logic programming [Apt and Pellegrini, 1994]. The notions of the safe-range and range-restricted queries come from the database theory where this form of "safe" queries has been extensively studied [Abiteboul *et al.*, 1994]. Moreover, the notion of just-in-time formulas was introduced for a different setting in [De Giacomo *et al.*, 2001] and, in our case, is also related to the notion of an *active domain* in the database theory which is the subset of the domain that is explicit in a database instance [Abiteboul *et al.*, 1994].

Finally, a brief comment on a line of research about progression that is done outside

of the situation calculus in a similar logical formalism. Thielscher [1999] defined a dual representation for basic action theories based on the *state update axioms* that explicitly define the direct effects of each action, and studied a case of first-order progression in the *fluent calculus*. The main difference between the results in Sections 4.2 and 4.3 and this work is that, unlike our work where the sentences in the initial knowledge base are *replaced* by an *updated version*, there, the progression relies on adding a *logical description of the changes* between the initial state and the updated state. In order to implement the progression method of [Thielscher, 1999] one then needs to rely on some kind of inference procedure that is capable of producing the new state by the old state and the logical description. In particular in the implementation of FLUX [Thielscher, 2004], a language of logical constraints is used to express the change between states and a sound but incomplete constraint solver is used to compute the current state.

Chapter 5

A progression procedure for a practical case

In this chapter we focus on a special case of the range-restricted basic action theories that we examined in Section 4.3. We adopt a practical assumption and specify an algorithm for the core computational task of the progression method of Section 4.3. We prove the correctness and the complexity of the algorithm, discuss the overall performance of the progression method, and show its application in a simple example that is inspired from video games.

5.1 Progression of range-restricted theories by computing possible answers

In Section 4.3, we studied the range-restricted theories and specified a progression method that relies on simplifying the successor state axioms with respect to a ground action α using Lemmas 4.3.17 and 4.3.13. The first lemma specifies a simple substitution for some of the variables in the context formulas by ground arguments of α based on the uniqueness of names for actions. The second lemma rewrites each axiom based on the

set of the possible answers to each of the context formulas in the axiom. This is the part where the just-in-time assumption is needed in order to ensure that each of these sets is finite. After the axioms are simplified it is then straightforward to specify the context set $\mathcal{J}$ of the action as in Definition 4.3.29 and progress $\mathcal{D}_0$ as in Theorem 4.3.35 by progressing each of the $\mathcal{J}$-models as in Definition 4.3.33. We now look closer into this method and show that as far as implementing this method is concerned, we essentially need to provide a practical algorithm for computing the possible answers.

First, we show how the context set $\mathcal{J}$ can be specified directly from the set of possible answers to queries on the positive and negative effects formulas of the successor state axioms.

Definition 5.1.1. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a ground action term such that $\mathcal{D}$ is just-in-time wrt α. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ of the successor state axiom for F have been simplified based on the uniqueness of names for actions to the range-restricted form that Lemma 4.3.13 implies. We define the *context set* of α as the smallest set that includes all the fluent atoms with a ground input $F(\vec{c}, w, S_0)$ such that one of the following holds:

- for some closure χ and constant e, the pair $(\langle \vec{c}, e \rangle, \chi)$ is a possible answer to $\gamma_F^+(\langle \vec{x}, w \rangle, \alpha, S_0)$ or $\gamma_F^-(\langle \vec{x}, w \rangle, \alpha, S_0)$ wrt $\mathcal{D}_0$;
- for some closure χ, vector of constants $\vec{b}$, and constant e, the pair $(\langle \vec{b}, e \rangle, \chi)$ is a possible answer to $\gamma_F^+(\langle \vec{x}, y \rangle, \alpha, S_0)$ or $\gamma_F^-(\langle \vec{x}, y \rangle, \alpha, S_0)$ wrt $\mathcal{D}_0$, and $F(\vec{c}, w, S_0)$ is mentioned in χ or there is a fluent atom τ with a ground input that is mentioned in χ such that τ and $F(\vec{c}, w, S_0)$ are both mentioned in the same axiom in $\mathcal{D}_0$. ■

Note that we do not require that the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ are in context normal form but we use Lemma 4.3.17 that implies a simple substitution based on the uniqueness of names for actions. Note also that the two cases of this definition correspond

to those of Definition 4.3.29 that specify the atomic closures that may need updating after α is performed as well as the closures on which the change may be conditioned.

The set of the $\mathcal{J}$-models can be easily specified by going through all the combinations of possible closures wrt $\mathcal{D}_0$ for each atom in $\mathcal{J}$. The progression of each $\mathcal{J}$-model can then be specified using information directly from the sets of the possible answers to the effects formulas of the successor state axioms as in Definition 5.1.1.

Definition 5.1.2. Let $\mathcal{D}$ be a basic action theory that is range-restricted and α a ground action term such that $\mathcal{D}$ is just-in-time wrt α. Without loss of generality we assume that for all fluent symbols F in $\mathcal{L}$, the formulas $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ of the successor state axiom for F have been simplified to the range-restricted form that Lemma 4.3.13 implies. Let $\mathcal{J} = \{F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\}$ be the context set of α, and θ be a $\mathcal{J}$-model such that θ is the closure of the vector $\langle F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\rangle$ on $\langle V_1, \ldots, V_n\rangle$. We define the set Γ_i^+ as the smallest set of constants e such that the following conditions hold:

1. for some closure χ, the pair $(\langle \vec{c}_i, e\rangle, \chi)$ is a possible answer to $\gamma_{F_i}^+(\langle \vec{x}, w\rangle, \alpha, S_0)$ or $\gamma_{F_i}^-(\langle \vec{x}, w\rangle, \alpha, S_0)$ wrt $\mathcal{D}_0$;

2. $\{\chi \wedge \theta\} \cup \mathcal{E}$ is consistent.

The set Γ_i^- is defined similarly based on the formula $\gamma_{F_i}^-(\vec{x}, \alpha, s)$ instead of $\gamma_{F_i}^+(\vec{x}, \alpha, s)$. The *progression* of θ wrt α is the closure of $\langle F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\rangle$ on the updated vector $\langle V_1', \ldots, V_n'\rangle$, where for all i, $1 \leq i \leq n$, V_i' is the set $(V_i - \Gamma_i^-) \cup \Gamma_i^+$. ■

Note that the condition 1 in this definition corresponds to the conditions 1 and 2 of Definition 4.3.33, and the condition 2 is the same as the condition 3 of Definition 4.3.33.

Definitions 5.1.1 and 5.1.2 illustrate that we can specify the context set $\mathcal{J}$ and the progression for each of the $\mathcal{J}$-models by performing a series of simple tasks over the possible answers to the positive and negative effects formulas of the successor state axioms: in

order to specify the context set we only need to go through each of the possible answers to these formulas and populate the set $\mathcal{J}$, while for the progression of a $\mathcal{J}$-model we also need to verify that two closures are consistent, which according to Lemma 4.3.3 also reduces to a simple task of comparing lists. Moreover according to Theorem 4.3.35, in order to progress $\mathcal{D}_0$ we only need to separate the axioms that do not overlap with $\mathcal{J}$ and replace those that overlap by a larger axiom that consists of the progressed $\mathcal{J}$-models.

So, the challenging computational problem we need to solve in order to progress $\mathcal{D}_0$ is essentially that of specifying the set of possible answers to each of the positive and negative effects formulas. We now proceed to present an algorithm for computing the possible answers for a special case of the range-restricted formulas. Based on this restriction on formulas we will then define a practical class of range-restricted basic theories and discuss the progression of these theories.

5.2 An algorithm for range-restricted conjunctive$^+$ queries

In Section 5.1 we showed that the core procedure for progressing a range-restricted theory is that of computing the possible answers to a formula with respect to a database of possible closures. We now focus on a class of formulas that is similar to the so-called *conjunctive queries* of the database theory [Abiteboul *et al.*, 1994] and provide an algorithm for computing the possible answers.

Definition 5.2.1. Let σ be a situation term. A formula ϕ uniform in σ is a *conjunctive query* iff it has the following form:

$$\exists \vec{x}(\tau_1 \wedge \cdots \wedge \tau_n),$$

where each τ_i is fluent atom uniform in σ whose free variables are not necessarily in $\vec{x}$. ■

The conjunctive queries are capable of expressing a wide range of practical queries over a database of possible closures. In particular they offer an intuitive way of building range-restricted formulas based on the idea of using the output of a fluent with a ground input as an input to another fluent. A simple example follows.

Example 5.2.2. The following formula of Example 4.3.14 is a conjunctive query:

$$Near(bomb, y, S_0) \wedge Status(y, z, S_0).$$

Similarly the following formula is also a conjunctive query:

$$\exists z(Near(bomb, y, S_0) \wedge Status(y, z, S_0)).$$

Note that both queries are range-restricted formulas according to Definition 4.3.12. ■

We now present a function that computes the set of possible answers to a range-restricted conjunctive query. Algorithm 1 describes the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ that takes a range-restricted conjunctive query γ and a finite database of possible values $\mathcal{D}_0$ as its input, and returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$ if γ is just-in-time wrt $\mathcal{D}_0$ and *failure* otherwise.

Algorithm 1 $\mathsf{PANS}(\gamma, \mathcal{D}_0)$

1. $\Gamma := \{F(\vec{c}, t, S_0) \mid F(\vec{c}, t, S_0)$ is an atom of γ that is mentioned in $\mathcal{D}_0\}$
2. $\Delta := \{\tau \mid \tau$ is an atom of γ such that $\tau \notin \Gamma\}$
3. $X := \mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$
4. **if** $X =$ *failure* **then**
5. **return** *failure*
6. **else**
7. $X' :=$ the projection of X to the free variables of γ
8. **return** X'
9. **end if**

Algorithm 2 PANS-REC$(\Gamma, \Delta, \mathcal{D}_0)$

1. **if** $\Gamma \cup \Delta = \emptyset$ **then**
2. **return** $\{\}$ // all atoms have been processed successfully
3. **end if**
4. **if** $\Gamma = \emptyset$ **then**
5. **return** *failure* // no appropriate atom to continue
6. **else**
7. $X := \{\}$ // initialize the set of possible answers
8. $\tau :=$ the first atom $F(\vec{c}, t, S_0)$ in Γ
9. **for all** χ, χ is a possible atomic closure of $F(\vec{c}, w, S_0)$ wrt $\mathcal{D}_0$, **do**
10. **if** t is a constant **then**
11. **if** $t \in V$, where χ is the closure of $F(\vec{c}, w, S_0)$ on V **then**
12. $Y :=$ PANS-REC$(\Gamma - \{\tau\}, \Delta, \mathcal{D}_0)$ // recursively solve the smaller problem
13. **if** $Y =$ *failure* **then**
14. **return** *failure*
15. **else**
16. $Y' := \{(\omega, \chi \wedge \chi') \mid (\omega, \chi') \in Y, \chi \wedge \chi' \text{ consistent}\}$ // merge the answers
17. $X := X \cup Y'$ // update the set of possible answers
18. **end if**
19. **end if**
20. **else**
21. **for all** constants $e \in V$, where χ is the closure of $F(\vec{c}, w, S_0)$ on V **do**
22. $\Gamma' := \{\pi|_e^t \mid \pi \in \Gamma, \pi \neq \tau\}, \quad \Delta' := \{\pi|_e^t \mid \pi \in \Delta\}$ // t is a variable
23. $\Delta'_1 := \{\pi \mid \pi \in \Delta', \pi \text{ has a ground input and is mentioned in } \mathcal{D}_0\}$
24. $\Delta'_2 := \Delta' - \Delta'_1$
25. $Y :=$ PANS-REC$(\Gamma' \cup \Delta'_1, \Delta'_2, \mathcal{D}_0)$ // recursively solve the smaller problem
26. **if** $Y =$ *failure* **then**
27. **return** *failure*
28. **else**
29. $Y' := \{(t = e \wedge \omega, \chi \wedge \chi') \mid (\omega, \chi') \in Y, \chi \wedge \chi' \text{ and } t = e \wedge \omega \text{ consistent}\}$
30. $X := X \cup Y'$ // merge and update the set of possible answers
31. **end if**
32. **end for**
33. **end if**
34. **end for**
35. **return** X // return the set of possible answers
36. **end if**

The function PANS$(\gamma, \mathcal{D}_0)$ essentially performs the initialization for the recursive function PANS-REC$(\Gamma, \Delta, \mathcal{D}_0)$ that is described in Algorithm 2. PANS$(\gamma, \mathcal{D}_0)$ separates the atoms in γ in two sets: i) the set Γ that consists of all the atoms τ such that it is guaranteed that there are finitely many possible answers to τ wrt $\mathcal{D}_0$, and ii) the set Δ that

consists of the rest of the atoms. The function $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ works by specifying the possible answers to an atom in Γ, updating the sets Γ, Δ based on the computed answers, recursively solving the simpler problem that is obtained, and finally merging the answers.[1]

In some more detail, the function $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ first checks if there are no more atoms to process (line 1), in which case it returns the empty set that is our convention for the possible answers to the empty conjunctive query. In case there are more atoms to process but there is no atom in Γ the function returns *failure* (line 5), otherwise selects the first atom τ in Γ (line 8). In the case that an atom τ is selected the function forms a loop that ranges over all the possible closures χ of τ wrt $\mathcal{D}_0$ (line 9). Inside this loop the function checks whether τ has a ground output or not (line 10). In the first case, if the output is implied by χ (line 11), then the possible answers of the rest of the atoms in Γ, Δ are computed recursively (line 12), and the closure χ is merged into those answers provided that they are compatible (line 16). The case where τ has a variable as its output is similar.

As we mentioned earlier, the intuition is that the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ returns *failure* if γ is not just-in-time wrt $\mathcal{D}_0$, otherwise it returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$. We now proceed to state this formally.

5.2.1 Correctness

The following theorem states the correctness of the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$:

Theorem 5.2.3. *Let γ be a range-restricted conjunctive query uniform in S_0 and $\mathcal{D}_0$ a finite database of possible closures. The function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ always terminates and if γ is just-in-time wrt $\mathcal{D}_0$, then $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$, otherwise it returns failure.* ■

[1] The notation $\pi|_e^t$ is used to denote that result of substituting the term t by the constant e in π.

Proof. It suffices to show that for all $n \geq 0$, the function $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ always terminates and if γ is just-in-time wrt $\mathcal{D}_0$, then it returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$, otherwise it returns *failure*, where Γ, Δ are sets of atoms of the appropriate form, n is the size of $\Gamma \cup \Delta$, and γ is the conjunction of the atoms in $\Gamma \cup \Delta$. We prove this by induction on n. The base case follows from our convention for the empty conjunctive query and the line 2 of Algorithm 2. For the induction step we assume that the hypothesis holds for all $n < k$ and we prove for k. Let $\Gamma \cup \Delta$ be nonempty and consider two cases as follows.

Case 1: Let Γ be empty. Then the function $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ returns *failure*. We will show that γ is not just-in-time wrt $\mathcal{D}_0$, therefore the induction hypothesis holds for k. Let τ be any atom in Δ. Then τ is either an atom without a ground input or an atom of F with a ground input $\vec{c}$ such that $F(\vec{c}, w, S_0)$ is not mentioned in $\mathcal{D}_0$. In both cases since $\mathcal{D}_0$ is finite and the constants in $\mathcal{L}$ are infinite, it follows that there is a possible answer (ω, χ) to τ wrt $\mathcal{D}_0$ such that χ does not consist of possible closures wrt $\mathcal{D}_0$. By Definition 4.3.8 of possible answers it follows that in all the cases we can always find a possible answer (ω', χ') to γ such that χ is a conjunct of χ'. By Definition 4.3.11 then it follows that γ is not just-in-time wrt $\mathcal{D}_0$.

Case 2: Let Γ be nonempty. By the fact that the induction hypothesis holds for $k-1$ it follows that the recursive calls of the lines 12 and 25 return a correct value. If any of the recursive calls return *failure* then the function also returns *failure*. By a similar argument as in the Case 1 we can show that in this case γ is not just-in-time wrt $\mathcal{D}_0$ and so the induction hypothesis holds for k. Assuming that no recursive call returns *failure*, by Definition 4.3.8 of possible answers it follows that the merging of the possible answers is also correct and the function returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$. □

Now we turn our attention to the complexity of the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$.

5.2.2 Complexity

First note that for the case that γ is a range-restricted conjunctive query that is also just-in-time wrt $\mathcal{D}_0$ the size of $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is $O(N^n)$, where n is the size of γ and N is the size of $\mathcal{D}_0$. This is because the assumptions for γ and Lemma 4.3.13 ensure that in the worst case for each atom τ in γ, the set of possible answers to τ wrt $\mathcal{D}_0$ corresponds to the information in the entire $\mathcal{D}_0$. As a result this is a lower bound for the running time of the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ that computes the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$. The following lemma shows that in the worst case the running time of $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ is close to this lower bound:

Lemma 5.2.4. *Let γ be a range-restricted conjunctive query and $\mathcal{D}_0$ a finite database of possible closures. Let n be the number of atoms in γ, k the number of distinct fluent atoms with a ground input in $\mathcal{D}_0$, m the maximum number of disjuncts in a possible closures axiom in $\mathcal{D}_0$, and l the maximum number of constants that appear in an atomic closure in an axiom in $\mathcal{D}_0$. Then the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ runs in time $O((k \cdot l + n^2) \cdot (m \cdot l)^n)$.* ■

Proof. The time complexity of the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ is essentially the same as the complexity of the function $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ which is characterized by the following recurrence relation:

$$T(n) = \begin{cases} c_1, & \text{if } n = 0; \\ c_1 + m \cdot l \cdot (c_2 \cdot n + T(n-1) + c_3 \cdot (m \cdot l)^{n-1} \cdot (k \cdot l + n)), & \text{if } n \geq 1, \end{cases}$$

where c_1, c_2, c_3 are constants that represent the steps that $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ performs that do not depend on the parameters n, m, k, l. This is obtained as follows:

1. m is due to the line 9 of Algorithm 2 that forms a loop over all the possible atomic closures χ of the selected atom τ;

2. l is due to the line 21 that forms a loop over all the constants e in the closure χ;

3. n is due to the lines 22–24 that perform the update of the sets Γ, Δ which can be performed in linear time wrt n assuming that the sets are sorted;

4. $T(n-1)$ is due to the line 25 that performs a recursive call for a problem of size $n-1$;

5. $(m \cdot l)^{n-1}$ is due to the line 29 that essentially forms a loop over all the possible answers to a formula of size $n-1$, and $(k \cdot l + n)$ reflects that the atomic closure χ is tested for consistency wrt all the atomic closures of χ' of each possible answer.

The recurrence relation can be simplified as follows:

$$T(n) = \begin{cases} 1, & \text{if } n = 0; \\ m \cdot l \cdot T(n-1) + (m \cdot l)^n \cdot (k \cdot l + n)), & \text{if } n \geq 1, \end{cases}$$

It follows that for $n \geq 1$,

$$T(n) = (m \cdot l)^n + (m \cdot l)^n (k \cdot l + n + (n-1) + \cdots + 1).$$

It follows that the function $\mathsf{PANS\text{-}REC}(\Gamma, \Delta, \mathcal{D}_0)$ runs in time $O((k \cdot l + n^2) \cdot (m \cdot l)^n)$. □

Assuming that $N = m \cdot k \cdot l$ is an upper bound for the size of $\mathcal{D}_0$ Lemma 5.2.4 implies that the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ runs in time $O((n^2 + N) \cdot N^n)$.

We now proceed to discuss how the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ can be extended to handle more expressive queries without affecting the upper bounds for the running time and the size of the return value.

5.2.3 Extending the algorithm to handle equality and negated atoms

The conjunctive queries are quite expressive and extensively used in databases. Nonetheless for the purposes of reasoning about action they have some important limitations. Recall that our intention is to use the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ in order to compute the set of possible answers to the positive and negative effects formulas γ of the successor state axioms of a range-restricted theory $\mathcal{D}$. Therefore, in order for the function to be applicable it is necessary that the positive and negative effects formulas are range-restricted conjunctive queries.

For instance consider the theory $\mathcal{D}$ of the simple bomb domain of Example 4.3.16. Even though the only context formula of $\gamma^+_{Status}(x_1, x_2, a, s)$ and $\gamma^-_{Status}(x_1, x_2, a, s)$ is in both cases a conjunction of literals, a negated atom is also included and as a result it does not qualify as conjunctive query. Note that it is important that both equality atoms and negated atoms can be used in the context formulas so that a wide range of conditions under which a fluent atom may change truth value can be represented. We introduce the following extension to the conjunctive queries that accounts for this limitation.

Definition 5.2.5. Let σ be a situation term. A formula ϕ uniform in σ is a *conjunctive*$^+$ *query* iff it has the following form:

$$\exists \vec{x}(\phi_1 \wedge \cdots \wedge \phi_n),$$

where each ϕ_i is one of the following:

- a possibly negated fluent atom uniform in σ whose free variables are not necessarily in $\vec{x}$;
- an atom of the form $y = c$, where y is a variable not necessarily in $\vec{x}$ and c is a constant;
- an atom of the form $c = d$ or $c \neq d$, where c, d are constants. ■

Note that when γ is a range-restricted conjunctive$^+$ query that is also just-in-time wrt $\mathcal{D}_0$ the size of $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is again $O(N^n)$, where n is the size of γ and N is the size of $\mathcal{D}_0$. Moreover the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ can be easily extended to handle range-restricted conjunctive$^+$ queries as follows. First, Algorithms 1 and 2 need to be extended so that they also include in the set Γ all the equality literals of γ as well as all the ground negated fluent atoms of the form $\neg F(\vec{c}, d, S_0)$ such that $F(\vec{c}, w, S_0)$ is mentioned in $\mathcal{D}_0$. These literals can be then processed in a very similar way as the fluent atoms that Algorithm 2 can already handle.

As far as the equality literals are concerned we need to do something similar to the treatment of the ground fluent atoms as in the lines 10–19 of Algorithm 2: as long as the literal is consistent we only need to simplify the atoms in Γ and Δ in case the literal has the form $y = c$, recursively compute the possible answers for the simpler problem, and merge the answers in the straightforward way. Similarly, when it comes to a negated ground fluent literal of the form $\neg F(\vec{c}, d, S_0)$ we need to do something similar to the treatment of a non-ground fluent atom of the form $F(\vec{c}, t, S_0)$ as in the lines 21–33 of Algorithm 2: the only difference now is that we need to iterate over the constants e in the possible closure χ of $F(\vec{c}, w, S_0)$ such that $e \neq d$. Observe that this is similar to the way logic programming techniques handle *negation as failure* [Apt and Pellegrini, 1994].

We omit the formal specification of the extended algorithm but note that it is straightforward to extend Algorithms 1 and 2 as we described and obtain the extended version of the function $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ that we denote as $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$. The intuition for $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$ is that it takes as input a range-restricted conjunctive$^+$ query γ and a finite database of possible closures $\mathcal{D}_0$, and if γ is just-in-time wrt $\mathcal{D}_0$, then it returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$, otherwise returns *failure*. The correctness of $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$ follows by a similar argument as the one for the proof of Theorem 5.2.3, and it is also straightforward to show that $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$ has the same time complexity as $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ following the same reasoning as in the proof of Lemma 5.2.4. This is stated formally in the following two results:

Corollary 5.2.6. *Let γ be a range-restricted conjunctive$^+$ query and $\mathcal{D}_0$ a finite database of possible closures. The function $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$ always terminates and if γ is just-in-time wrt $\mathcal{D}_0$, then $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ returns the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$, otherwise returns failure.* ■

Corollary 5.2.7. *Let γ be a range-restricted conjunctive$^+$ query and $\mathcal{D}_0$ a finite database of possible closures. The time complexity of $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$ is $O((n^2 + N) \cdot N^n)$, where n is the size of γ and N is the size of $\mathcal{D}_0$.* ■

We now proceed to introduce a class of basic action theories that can be progressed using $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$.

5.3 Progression of range-restricted conjunctive$^+$ theories

In this section we focus on a special case of the range-restricted basic action theories that we examined in Section 4.3. Following the intuitions in Sections 5.1 and 5.2 we adopt the assumption that the context formulas of the successor state axioms are conjunctive$^+$ queries and discuss the completeness and complexity of a progression method that is based on the function $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$.

Definition 5.3.1. Let the successor state axiom for the fluent symbol F have the following form:

$$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee (F(\vec{x}, s) \wedge \neg\gamma_F^-(\vec{x}, a, s)).$$

The successor state axiom for F is *range-restricted* and *conjunctive$^+$* iff both $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^-(\vec{x}, a, s)$ are disjunctions of formulas of the following form:

$$\exists\vec{z}(a = A(\vec{y}) \wedge \phi(\vec{x}, \vec{z}, s)),$$

where A is an action function, $\vec{y}$ may contain some variables from $\vec{x}$, $\vec{z}$ corresponds to the

remaining variables of $\vec{y}$, and the context formula ϕ is a conjunctive$^+$ query uniform in s, does not mention any other free variable other than $\vec{x}, \vec{z}, s$, and is such that $\phi(\vec{x}, \vec{z}, S_0)$ is safe-range wrt the variables in $\vec{x}$. A basic action theory $\mathcal{D}$ is *range-restricted* and *conjunctive*$^+$ iff all the successor state axioms in $\mathcal{D}_{ss}$ are range-restricted and conjunctive$^+$ in the previous sense, $\mathcal{D}_0$ is a database of possible closures, and the theory also includes the set $\mathcal{E}$ of the unique-names axioms for objects. ■

Algorithm 3 that appears in the next page describes the function $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ that progresses a range-restricted and conjunctive$^+$ theory $\mathcal{D}$ wrt a ground action α. $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ essentially follows the steps of the progression method for range-restricted action theories of Section 4.3. The correctness and complexity of the algorithm follows.

5.3.1 Correctness

The correctness of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ is stated formally in the next theorem.

Theorem 5.3.2. *Let $\mathcal{D}$ be a basic action theory that is range-restricted and conjunctive$^+$ and has a finite $\mathcal{D}_0$, and α a ground action term. Then, the function $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ always terminates and if $\mathcal{D}$ is just-in-time wrt α, then $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ returns a set that is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$, otherwise returns failure.* ■

Proof. The theorem follows from Theorem 4.3.35 that specifies a method for progressing a range-restricted theory, Corollary 5.2.6 that shows the correctness of the function $\mathsf{PANS}^+(\gamma, \mathcal{D}_0)$ for a range-restricted conjunctive$^+$ formula γ, and Definitions 5.1.1 and 5.1.2 that show how to specify a the context set $\mathcal{J}$ and the progression of a $\mathcal{J}$-model by inspecting the set of possible answers to the effects formulas of the successor state axioms in $\mathcal{D}_{ss}$. □

Algorithm 3 PROGRESS$(\alpha, \mathcal{D})$

1. Y:= {} // initialize the context set $\mathcal{J}$
2. **for all** fluent symbols F in $\mathcal{L}$ **do**
3. simplify $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^+(\vec{x}, \alpha, S_0)$ using $\mathcal{D}_{una}$
4. X_F^+ := PANS$^+(\gamma_F^+(\vec{x}, \alpha, S_0), \mathcal{D}_0)$ // compute the possible answers to the
5. X_F^- := PANS$^+(\gamma_F^-(\vec{x}, \alpha, S_0), \mathcal{D}_0)$ // positive and negative effects formulas of F
6. **if** $X_F^+ =$ *failure* or $X_F^- =$ *failure* **then**
7. **return** *failure* // $\mathcal{D}$ is not just-in-time wrt α
8. **else**
9. **for all** pairs (ω, χ) in $X_F^+ \cup X_F^-$ **do**
10. **if** (ω, χ) specifies the atoms $\tau_1, \ldots, \tau_n$ according to Definition 5.1.1 **then**
11. **if** $\tau_1, \ldots, \tau_n$ are mentioned in $\mathcal{D}_0$ **then**
12. $Y := Y \cup \{\tau_1, \ldots, \tau_n\}$ // update the context set $\mathcal{J}$
13. **else**
14. **return** *failure* // $\mathcal{D}$ is not just-in-time wrt α
15. **end if**
16. **end if**
17. **end for**
18. **end if**
19. **end for**
20. let $\vec{\tau}$ be a vector of the elements in Y // compute the set of the $\mathcal{J}$-models
21. Z:= $\{\theta \mid \theta$ is a closure of $\vec{\tau}$ that consists of possible atomic closures wrt $\mathcal{D}_0\}$
22. **for all** $\mathcal{J}$-models θ in Z **do**
23. **for all** atoms $F(\vec{c}, w, S_0)$ in Y **do**
24. Γ^+:= {}, Γ^-:= {} // initialize the progression of the atomic closure of $F(\vec{c}, w, S_0)$
25. **for all** pairs (ω, χ) in X_F^+ and X_F^- **do**
26. **if** (ω, χ) specifies a constant e^+ or e^- according to Definition 5.1.2 **then**
27. $\Gamma^+ := \Gamma^+ \cup \{e^+\}$ or $\Gamma^- := \Gamma^- \cup \{e^-\}$
28. **end if**
29. **end for**
30. replace the atomic closure of $F(\vec{c}, w, S_0)$ on V in θ with the closure of $F(\vec{c}, w, S_0)$ on $(V - \Gamma^-) \cup \Gamma^+$ // progress each atomic closure in the $\mathcal{J}$-model
31. **end for**
32. **end for**
33. $U := \mathcal{D}_0$ // initialize the progressed database U as the original $\mathcal{D}_0$
34. **for all** atoms τ in Y **do**
35. **for all** axioms ϕ in U **do**
36. **if** ϕ mentions τ **then**
37. $U := U - \{\phi\}$ // the part of $\mathcal{D}_0$ that needs updating is removed from U
38. **end if**
39. **end for**
40. **end for**
41. $U := U \cup \{\bigvee_{\psi_i \in Z} \psi_i\}$ // the axiom of the progressed $\mathcal{J}$-models is added to U
42. **return** the set $\{\phi[S_\alpha] \mid \phi[S_0] \in U\}$

Now we turn our attention to the running time of the algorithm and the size of the progressed database $\mathcal{D}_\alpha$ that is the return value of the algorithm.

5.3.2 Complexity

As far as the running time of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ the following holds:

Lemma 5.3.3. *Let $\mathcal{D}$ be a basic action theory that is range-restricted and conjunctive$^+$ and has a finite $\mathcal{D}_0$, and α a ground action term. Let n be the size of $\mathcal{D}_{ss}$, k the number of distinct fluent atoms with a ground input in $\mathcal{D}_0$, m the maximum number of disjuncts in a possible closures axiom in $\mathcal{D}_0$, and l the maximum number of constants that appear in an atomic closure in an axiom in $\mathcal{D}_0$. Then the function $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ runs in time $O(m^k \cdot k \cdot (m \cdot l)^n \cdot (k \cdot l + n))$.* ■

Proof. This follows from the fact that the running time of the function is dominated by the three nested loops in the lines 22–32 of Algorithm 3. The three first factors in $O(m^k \cdot k \cdot (m \cdot l)^n \cdot (k \cdot l + n))$ correspond to the iterations of each of the three loops and the last factor corresponds to the processing of the line 26:

1. m^k corresponds to the maximum size of the set Z of $\mathcal{J}$-models;
2. k corresponds to the maximum size of the context set $\mathcal{J}$;
3. $(m \cdot l)^n$ corresponds to the maximum size of the set of the possible answers to all positive and negative effects formulas in $\mathcal{D}_{ss}$;
4. $(k \cdot l + n)$ corresponds to the maximum size of a pair (ω, χ) that needs to be inspected. □

According to the previous result the running time of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ is in the worst case exponential with respect to the parameters k and n that are bound by the size of $\mathcal{D}_0$ and the size of $\mathcal{D}_{ss}$, respectively. Similarly, in the worst case the size of $\mathcal{D}_\alpha$ is exponential

with respect to the size of $\mathcal{D}_0$. This is because in the worst case the context set of α includes all the fluent atoms with a ground input that are mentioned in $\mathcal{D}_0$, and $\mathcal{D}_\alpha$ is one large possible closures axiom that lists all the combinations of possible atomic closures for all the fluent atoms in the context set. Essentially this corresponds to "unwinding" a formula that is in conjunctive normal form into a logically equivalent one in disjunctive normal form.

Finally, as far as performing a series of progression steps is concerned, the size complexity of the resulting database is the same as for one step. This is because the exponential blow-up in the size of the progressed database depends on the total number of fluent atoms with a ground input that are mentioned in $\mathcal{D}_0$, which is fixed for progression steps that follow the just-in-time requirement. In other words, when $\mathcal{D}$ is just-in-time wrt α, the action α may only affect the output of fluent atoms that are already mentioned in $\mathcal{D}_0$. As a result $\mathcal{D}_\alpha$ mentions exactly the same set of fluent atoms with a ground input as $\mathcal{D}_0$. The way these atoms are grouped in the resulting database after a series of progression steps is performed may lead to an exponential blow-up, but it can get no worse than the case we described for one step of progression where $\mathcal{D}_\alpha$ is one large possible closures axiom that mentions all the atoms of $\mathcal{D}_0$.

In the next section we discuss the performance of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ in practice where it is often the case that the application scenarios are not comparable to the worst case scenario that we considered in our analysis, and the average complexity of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ is more appealing.

5.4 Discussion

The results about the performance of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ seem disheartening but nonetheless we expect both the size of $\mathcal{D}_\alpha$ and the running time of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ to be manageable in practice as the scenarios we have in mind are quite far from being sim-

ilar to the worst case scenario. We therefore argue for the existence of constant upper bounds to the parameters k, m, n as they are used in the analysis of the complexity of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ in Lemma 5.3.3 as follows.

First, we expect that the effects formulas that need to be evaluated in order to compute the context set of α are small in number and typically much smaller in size than the size of $\mathcal{D}_{ss}$, which is the upper bound in the worst case. This is because in a large theory that represents the effects of many actions we expect that for the vast amount of the effects formulas in $\mathcal{D}_{ss}$, α does not unify with the action term in the formula and the formula only mentions a small number of atoms in the context.

Similarly, we expect that the number of fluent atoms with a ground input that may be either affected by α or necessary to decide when the effects of α apply is small and not comparable to the total number of atoms in $\mathcal{D}_0$, which is the upper bound in the worst case. The context set of α may still be large in size though because these fluent atoms may appear in $\mathcal{D}_0$ in some large axioms that also mention other atoms. Nonetheless, we expect that at any given time there is a constant upper bound on the number of fluent atoms with a ground input that are both *unknown* and *mutually constrained*, and thus need to appear in the same possible closures axiom. Moreover, we also expect that there is a constant upper bound on the number of disjuncts in a possible closures axiom. In other words we expect that the amount of incomplete information that is represented in $\mathcal{D}_0$ comes in (many) small packages of manageable size.

With these assumptions it follows that the parameters k, n, m are bounded by some small constants K, N, M. The only parameter then that may indeed grow arbitrarily is l that corresponds to the complete information that each closure represents. In this case the running time of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ and the size of $\mathcal{D}_\alpha$ are both $O(l^C)$ for some constant C that depends on the actual value of the constants K, N, M.

The assumptions we described are relatively strong and do not apply to all applications we have in mind but we believe that they are safe for many "organic" scenarios

where the theory $\mathcal{D}$ is used to represent the effects of a domain that is similar to the physical world. Furthermore, as far as the assumptions on the incomplete information are concerned, we believe that they can be enforced with an appropriate account of sensing.

We conclude with a detailed example of the application of $\mathsf{PROGRESS}(\alpha, \mathcal{D})$ that is inspired from video games.

5.5 A simple example from video games

We examine an example of progression in the case that the theory represents the game-world of a simple video game. For purposes of illustrating how some of the typical elements of a game-world are handled in the range-restricted and conjunctive$^+$ theories, we assume that the only character in the game is a non-player character (NPC) that is equipped with a basic action theory.

Moreover, in order to show that our representation of the game-world is practical also with respect to the interaction of a reasoning module equipped with a basic action theory and the game-engine, we will use the same ontology that is common in game-engines for implementing the properties and the functionality of the various elements of the game-world. To that end, we adopt the following assumptions for representing the game-world:

- every location that the character may occupy in space is represented as a *node*;
- the nodes are interconnected forming a directed graph that specifies how the character may move in space;
- every item in the game-world that the character may interact with is located at some node.

Note that following these assumptions in order to represent that there is a door that connects the nodes n_1 and n_2 in the domain we need to represent the door as an object

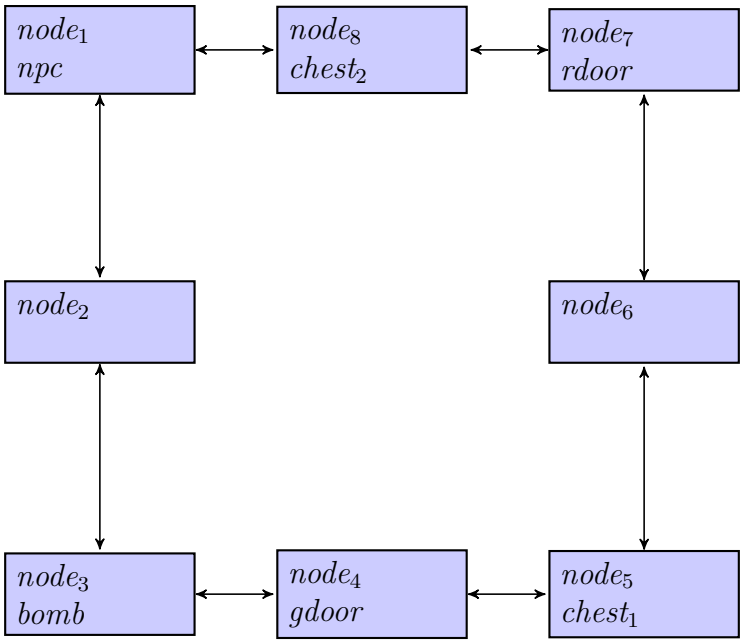

Figure 5.1: The initial configuration of the game-world.

that is located in another node n_3, such that n_1 and n_3 are adjacent, n_2 and n_3 are adjacent, and the NPC may occupy n_3 iff the door that is located in n_3 is open.

We now proceed to give the details of the example.

Example 5.5.1 (The simple video game domain). Consider a simple video game where the game-world is a big castle with many floors. Each level of the game is a floor of the castle with rooms that are interconnected and a door that leads to the next level. We look into a particular level where the red door leads to the next level and various other objects are available including a green door, two closed chests, a red key, and a bomb. The red door is locked and may only be unlocked using the red key that is stored in one of the chests.

Let $\mathcal{L}$ be the situation calculus language that consists of the standard logical symbols and the symbols $Poss, do, S_0$, the fluents $Adj(x_1, x_2, s)$, $At(x_1, x_2, s)$, $Status(x_1, x_2, s)$, $Inventory(x, s)$, $Inside(x_1, x_2, s)$, $InRange(x_1, x_2, x)$, the action functions $walk(y_1, y_2)$, $trigger(y)$, $pickup(y)$, $drop(y)$, $use(y_1, y_2)$, and the object constants $node_1, \ldots, node_8$, $rdoor$, $gdoor$, npc, $chest_1$, $chest_2$, $bomb$, $rkey$, $broken$, $locked$, ok.

The fluent $Adj(x_1, x_2, s)$ is used to represent the way the nodes of the game-world are connected. The fluent $At(x_1, x_2, s)$ represents that the object x_2 is located at the node x_1 in s, $Status(x_1, x_2, s)$ that the object x_1 has the status x_2 in s, $Inventory(x, s)$ that the object x is in the inventory of the NPC in s, $Inside(x_1, x_2, s)$ that the object x_2 is inside the object x_1 in s, and $InRange(x_1, x_2, s)$ represents the generic property that the object x_2 is the range of the object x_1 in s. The action $walk(y_1, y_2)$ represents that the NPC walks from the node y_1 to the node y_2, $trigger(y)$ that the bomb y is forced to explode affecting all the objects in range setting their status to *broken*, and $use(y_1, y_2)$ is a generic action that is used to represent that the object y_1 is used to affect the status of the object y_2. The constant *npc* represents the NPC, *rdoor* and *gdoor* represent the red and green doors in the domain, and *rkey* represents the red key that unlocks the red door. The rest of the symbols have the intuitive meaning.

Let $\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_0 \cup \mathcal{E} \cup \mathcal{D}_{fnd}$ be the basic action theory of the *simple video game domain* that represents the effects of the available actions and the initial configuration of the game-world as appears in Figure 5.1. Each of the parts of $\mathcal{D}$ is as follows.

1. $\mathcal{D}_{ap}$ consists of the following five sentences:[2]

$$\begin{aligned} Poss(walk(y_1, y_2, s)) \equiv\ & At(y_1, npc, s) \wedge Adj(y_1, y_2, s) \wedge \neg At(y, rdoor, s)) \vee \\ & At(y_1, npc, s) \wedge Adj(y_1, y_2, s) \wedge At(y_2, rdoor, s) \wedge \neg Status(rdoor, locked)) \vee \\ & At(y_1, npc, s) \wedge Adj(y_1, y_2, s) \wedge At(y_2, gdoor, s) \wedge \neg Status(rdoor, locked)), \end{aligned}$$

$$\begin{aligned} Poss(trigger(y, s)) \equiv & \exists z_1 \exists z_2 (At(z_1, npc, s) \wedge At(z_2, y, s) \wedge Adj(z_1, z_2, s) \wedge \\ & \neg Status(bomb, broken)), \end{aligned}$$

[2]Note that the right hand side of the axioms in $\mathcal{D}_{ap}$ is not always a range-restricted formula. This implies that the part of the module that is responsible for the projection problem, i.e., answering queries about the current and future situations, needs to be able to handle more than range-restricted queries.

$$Poss(pickup(y,s)) \equiv \exists z(At(z,npc,s) \wedge At(z,y,s) \wedge \neg Inventory(y,s)) \vee$$
$$\exists z_1 \exists z_2(At(z_1,npc,s) \wedge At(z_1,z_2,s) \wedge Inside(z_2,y,s) \wedge Status(z_2,open,s) \wedge$$
$$\neg Inventory(y,s)),$$

$$Poss(drop(y,s)) \equiv Inventory(y,s),$$

$$Poss(use(y_1,y_2,s)) \equiv \exists z(Inventory(y_1,s) \wedge At(z,npc,s) \wedge At(z,y_2,s)) \vee$$
$$\exists z_1 \exists z_2(Inventory(y_1,s) \wedge At(z_1,npc,s) \wedge At(z_2,y_2,s) \wedge Adj(z_1,z_2,s)).$$

2. $\mathcal{D}_{ss}$ consists of the following six sentences:

$$Adj(x_1,x_2,do(a,s)) \equiv Adj(x_1,x_2,s)$$

$$At(x_1,x_2,do(a,s)) \equiv \gamma^+_{At}(x_1,x_2,a,s) \vee$$
$$(At(x_1,x_2,s) \wedge \neg\gamma^-_{At}(x_1,x_2,a,s)),$$

$$Status(x_1,x_2,do(a,s)) \equiv \gamma^+_{Status}(x_1,x_2,a,s) \vee$$
$$(Status(x_1,x_2,s) \wedge \neg\gamma^-_{Status}(x_1,x_2,a,s)),$$

$$Inventory(x,do(a,s)) \equiv \gamma^+_{Inventory}(x,a,s) \vee$$
$$(Inventory(x,s) \wedge \neg\gamma^-_{Inventory}(x,a,s)),$$

$$Inside(x_1,x_2,do(a,s)) \equiv \gamma^+_{Inside}(x_1,x_2,a,s) \vee$$
$$(Inside(x_1,x_2,s) \wedge \neg\gamma^-_{Inside}(x_1,x_2,a,s)),$$

$$InRange(x_1,x_2,do(a,s)) \equiv \gamma^+_{InRange}(x_1,x_2,a,s) \vee$$
$$(InRange(x_1,x_2,s) \wedge \neg\gamma^-_{InRange}(x_1,x_2,a,s)),$$

where the effects formulas of the successor state axioms are as follows:

- $\gamma^{+}_{At}(x_1, x_2, a, s)$ is the following formula:

$$\exists z(a = walk(z, x_1) \wedge x_2 = npc) \vee$$
$$\exists z(a = walk(z, x_1) \wedge Inventory(x_2, s)) \vee$$
$$\exists z_1 \exists z_2(a = walk(z_1, x_1) \wedge Inventory(z_2, s) \wedge Inside(z_2, x_2, s)),$$

- $\gamma^{-}_{At}(x_1, x_2, a, s)$ is the following formula:

$$\exists z(a = walk(x_1, z) \wedge x_2 = npc) \vee$$
$$\exists z(a = walk(x_1, z) \wedge Inventory(x_2, s)) \vee$$
$$\exists z_1 \exists z_2(a = walk(x_1, z_1) \wedge Inventory(z_2, s) \wedge Inside(x_2, z, s)),$$

- $\gamma^{+}_{Status}(x_1, x_2, a, s)$ is the following formula:

$$a = trigger(bomb) \wedge InRange(bomb, x_1, s) \wedge x_2 = broken \vee$$
$$\exists z(a = trigger(bomb) \wedge InRange(bomb, z, s) \wedge Inside(z, x_1, s) \wedge x_2 = broken) \vee$$
$$a = use(rkey, rdoor) \wedge x_1 = rdoor \wedge x_2 = unlocked,$$

- $\gamma^{-}_{Status}(x_1, x_2, a, s)$ is the following formula:

$$a = trigger(bomb) \wedge InRange(bomb, x_1, s) \wedge Status(x_1, x_2, s) \wedge x_2 \neq broken \vee$$
$$\exists z(a = trigger(bomb) \wedge InRange(bomb, z, s) \wedge Inside(z, x_1, s) \wedge$$
$$Status(x_1, x_2, s) \wedge x_2 \neq broken),$$

- $\gamma^{+}_{Inventory}(x, a, s)$ is the following formula:

$$a = pickup(x),$$

- $\gamma^-_{Inventory}(x, a, s)$ is the following formula:

$$a = drop(x),$$

- $\gamma^+_{Inside}(x_1, x_2, a, s)$ is the empty disjunction,

- $\gamma^-_{Inside}(x_1, x_2, a, s)$ is the following formula:

$$a = pickup(x_1) \wedge Inside(x_1, x_2, s),$$

- $\gamma^+_{InRange}(x_1, x_2, a, s)$ is the following formula:

$$\exists z_1 \exists z_2 (a = walk(z_1, z_2) \wedge Inventory(bomb, s) \wedge x_1 = bomb \wedge At(z_2, x_2, s) \vee$$
$$\exists z_1 \exists z_2 \exists z_3 (a = walk(z_1, z_2) \wedge Inventory(bomb, s) \wedge x_1 = bomb \wedge$$
$$At(z_2, z_3, s) \wedge Inside(z_3, x_2, s)),$$

- $\gamma^-_{InRange}(x_1, x_2, a, s)$ is the following formula:

$$\exists z_1 \exists z_2 (a = walk(z_1, z_2) \wedge Inventory(bomb, s) \wedge x_1 = bomb \wedge$$
$$InRange(x_1, x_2, s) \wedge x_2 \neq bomb,$$

3. $\mathcal{D}_{una}$ is the set of unique-names axioms for the action functions in $\mathcal{L}$.

4. $\mathcal{D}_0$ is the following database of possible closures:

$$\{\chi_1, \ldots, \chi_{25},\ \chi_{26} \vee \chi_{27}\},$$

where $\chi_1, \ldots, \chi_{27}$ are the following closures:

- $\chi_1, \ldots, \chi_8$ are closures of a fluent atom of Adj with a ground input as follows:

$$\begin{aligned}
\chi_1 : &\quad \forall w.Adj(node_1, w, S_0) \equiv (w = node_2 \vee w = node_8),\\
\chi_2 : &\quad \forall w.Adj(node_2, w, S_0) \equiv (w = node_1 \vee w = node_3),\\
\chi_3 : &\quad \forall w.Adj(node_3, w, S_0) \equiv (w = node_2 \vee w = node_4),\\
\chi_4 : &\quad \forall w.Adj(node_4, w, S_0) \equiv (w = node_3 \vee w = node_5),\\
\chi_5 : &\quad \forall w.Adj(node_5, w, S_0) \equiv (w = node_4 \vee w = node_6),\\
\chi_6 : &\quad \forall w.Adj(node_6, w, S_0) \equiv (w = node_5 \vee w = node_7),\\
\chi_7 : &\quad \forall w.Adj(node_7, w, S_0) \equiv (w = node_6 \vee w = node_8),\\
\chi_8 : &\quad \forall w.Adj(node_8, w, S_0) \equiv (w = node_7 \vee w = node_2),
\end{aligned}$$

- $\chi_9, \ldots, \chi_{16}$ are closures of a fluent atom of At with a ground input as follows:

$$\begin{aligned}
\chi_9 : &\quad \forall w.At(node_1, w, S_0) \equiv w = npc,\\
\chi_{10} : &\quad \forall w.At(node_2, w, S_0) \equiv false,\\
\chi_{11} : &\quad \forall w.At(node_3, w, S_0) \equiv w = bomb,\\
\chi_{12} : &\quad \forall w.At(node_4, w, S_0) \equiv w = gdoor,\\
\chi_{13} : &\quad \forall w.At(node_5, w, S_0) \equiv w = chest_1,\\
\chi_{14} : &\quad \forall w.At(node_6, w, S_0) \equiv false,\\
\chi_{15} : &\quad \forall w.At(node_7, w, S_0) \equiv w = rdoor,\\
\chi_{16} : &\quad \forall w.At(node_7, w, S_0) \equiv w = chest_2,
\end{aligned}$$

- $\chi_{17}, \ldots, \chi_{23}$ are closures of a fluent atom of $Status$ with a ground input as

follows:

$$\chi_{17}: \quad \forall w.Status(npc, w, S_0) \equiv w = ok,$$

$$\chi_{18}: \quad \forall w.Status(gdoor, w, S_0) \equiv w = ok,$$

$$\chi_{19}: \quad \forall w.Status(rdoor, w, S_0) \equiv (w = ok \vee w = locked),$$

$$\chi_{20}: \quad \forall w.Status(chest_1, w, S_0) \equiv (w = ok \vee w = locked),$$

$$\chi_{21}: \quad \forall w.Status(chest_2, w, S_0) \equiv (w = ok \vee w = locked),$$

$$\chi_{22}: \quad \forall w.Status(bomb, w, S_0) \equiv w = ok,$$

$$\chi_{23}: \quad \forall w.Status(rkey, w, S_0) \equiv w = ok,$$

- χ_{24} is a closure of the fluent atom $Inventory(w, S_0)$ as follows:

$$\chi_{24}: \quad \forall w.Inventory(w, S_0) \equiv false,$$

- χ_{25} is a closure of the fluent atom $InRange(bomb, w, S_0)$ as follows:

$$\chi_{25}: \quad \forall w.InRange(bomb, w, S_0) \equiv w = bomb,$$

- χ_{26} and χ_{27} are non-atomic closures of the vector

$$\langle Inside(chest_1, w, S_0), Inside(chest_2, w, S_0)\rangle$$

as follows:

$$\chi_{26}: \quad \forall w(Inside(chest_1, w, S_0) \equiv w = rkey) \wedge \forall w(Inside(chest_2, w, S_0) \equiv false),$$

$$\chi_{27}: \quad \forall w(Inside(chest_1, w, S_0) \equiv false) \wedge \forall w(Inside(chest_2, w, S_0) \equiv w = rkey).$$

5. $\mathcal{E}$ is the set of unique-names axioms for the constants $node_1, \ldots, node_8$, $rdoor$, $gdoor$,

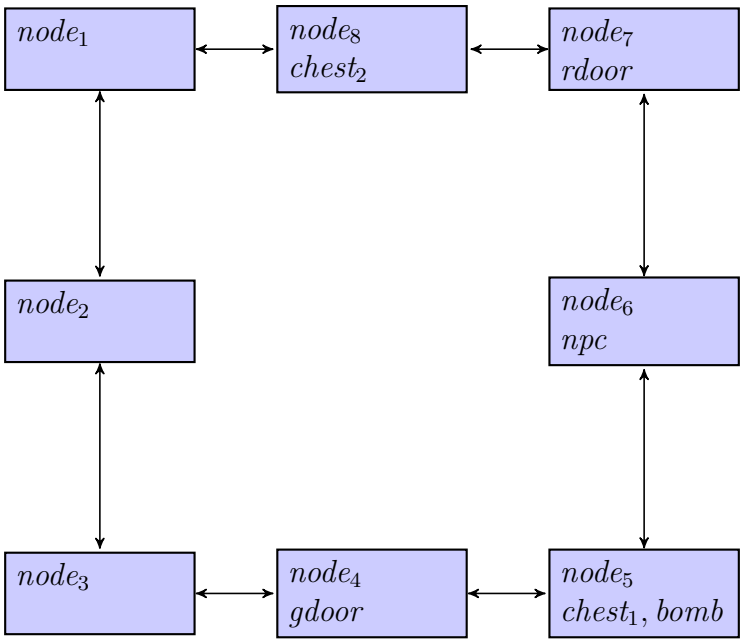

Figure 5.2: The configuration of the game-world after the action sequence λ is performed.

npc, $chest_1$, $chest_2$, *bomb*, *rkey*, *broken*, *locked*, and *ok*.

6. $\mathcal{D}_{fnd}$ is as in Definition 3.2.1.

Observe that all the successor state axioms in $\mathcal{D}_{ss}$ are range-restricted and conjunctive$^+$. Note also that the successor state axiom for $Inventory(x, s)$ is the only one that is local-effect. The rest of the axioms in $\mathcal{D}_{ss}$ are clearly not local-effect as the action terms that are mentioned in the effects formulas do not include always all the arguments of the fluents. Furthermore, $\mathcal{D}$ includes the unique-names axioms for constants and $\mathcal{D}_0$ is database of possible closures. Therefore, $\mathcal{D}$ is range-restricted and conjunctive$^+$.

Following the progression method of Theorem 4.3.35 for range-restricted theories it is easy to show that the just-in-time assumption holds when we do seven progression steps that correspond to the following sequence λ of actions:

$$walk(node_1, node_2), walk(node_2, node_3), pickup(bomb), walk(node_3, node_4),$$
$$walk(node_4, node_5), drop(bomb), walk(node_5, node_6),$$

and the progression of $\mathcal{D}_0$ wrt the action sequence is the following set $\mathcal{D}_0'$:[3]

$$(\mathcal{D}_0 - \{\chi_9, \chi_{11}, \chi_{13}, \chi_{14}, \chi_{25}\}) \cup \{\chi_9', \chi_{11}', \chi_{13}', \chi_{14}', \chi_{25}'\},$$

where $\chi_9', \chi_{11}', \chi_{13}', \chi_{14}', \chi_{25}'$ are as follows:

$$\begin{aligned}
\chi_9' :&\quad \forall w.At(node_1, w, S_0) \equiv false,\\
\chi_{11}' :&\quad \forall w.At(node_3, w, S_0) \equiv false,\\
\chi_{13}' :&\quad \forall w.At(node_5, w, S_0) \equiv (w = chest_1 \vee w = bomb),\\
\chi_{14}' :&\quad \forall w.At(node_6, w, S_0) \equiv w = npc\\
\chi_{25}' :&\quad \forall w.InRange(bomb, w, S_0) \equiv (w = bomb \vee w = chest_1).
\end{aligned}$$

The resulting configuration of the game-world appears in Figure 5.2.

It is easy to verify that we obtain the same set $\mathcal{D}_0'$ if we use the function PROGRESS for progressing range-restricted and conjunctive$^+$ theories. Now we proceed to examine in detail the application of $\mathsf{PROGRESS}(\mathcal{D}_0', \alpha)$, where α is the action term $trigger(bomb)$.

Lines 1–19 of Algorithm 3 We instantiate the effects formulas of the axioms in $\mathcal{D}_{ss}$ using the ground action α and S_0, simplify them using the unique-names axioms for actions, compute the possible answers to the simplified formulas, and store them to the variables of the form X^+, X^- as follows:

- The effect formulas $\gamma_{At}^+(x_1, x_2, \alpha, S_0)$ and $\gamma_{At}^-(x_1, x_2, \alpha, S_0)$ are simplified to *false*, therefore $\mathsf{pans}(\gamma_{At}^+(x_1, x_2, \alpha, S_0), \mathcal{D}_0) = \mathsf{pans}(\gamma_{At}^-(x_1, x_2, \alpha, S_0), \mathcal{D}_0) = \{\}$ and

$$X_{At}^+ = X_{At}^- = \{\}.$$

[3]Here we take the progressed version $\mathcal{D}_\alpha$ of $\mathcal{D}_0$ at every step to be uniform in S_0 instead of S_α. As we mentioned in the end of Section3.2.2, this is the intended use of progression in practice, and $\mathcal{D}_\alpha$ is typically assumed to be uniform in S_α only for simplifying the theoretical analysis.

- The effect formula $\gamma^+_{Status}(x_1, x_2, \alpha, S_0)$ is simplified to the following formula:

$$\begin{aligned}&InRange(bomb, x_1, S_0) \wedge x_2 = broken \vee\\ &\exists z(InRange(bomb, z, S_0) \wedge Inside(z, x_1, S_0) \wedge x_2 = broken).\end{aligned}$$

  The set $\mathsf{pans}(\gamma^+_{Status}(x_1, x_2, \alpha, S_0), \mathcal{D}_0)$ is computed and stored in the variable X^+_{Status} as follows:

$$\begin{aligned}X^+_{Status} = \{&(\langle x_1 = bomb, x_2 = broken\rangle, \chi'_{25}),\\ &(\langle x_1 = chest_1, x_2 = broken\rangle, \chi'_{25}),\\ &(\langle x_1 = rkey, x_2 = broken\rangle, \chi'_{25} \wedge \chi_{26})\}.\end{aligned}$$

- The effect formula $\gamma^-_{Status}(x_1, x_2, \alpha, S_0)$ is simplified to the following formula:

$$\begin{aligned}&InRange(bomb, x_1, S_0) \wedge Status(x_1, x_2, S_0) \wedge x_2 \neq broken \vee\\ &\exists z(InRange(bomb, z, S_0) \wedge Inside(z, x_1, S_0) \wedge\ \ Status(x_1, x_2, S_0) \wedge x_2 \neq broken).\end{aligned}$$

  The set $\mathsf{pans}(\gamma^-_{Status}(x_1, x_2, \alpha, S_0), \mathcal{D}_0)$ is computed and stored in the variable X^-_{Status} as follows:

$$\begin{aligned}X^-_{Status} = \{&(\langle x_1 = bomb, x_2 = ok\rangle, \chi'_{25}),\\ &(\langle x_1 = chest_1, x_2 = ok\rangle, \chi'_{25}),\\ &(\langle x_1 = rkey, x_2 = ok\rangle, \chi'_{25} \wedge \chi_{26})\}.\end{aligned}$$

- The effect formulas $\gamma^+_{Inventory}(x, \alpha, S_0)$ and $\gamma^-_{Inventory}(x, \alpha, S_0)$ are simplified to *false*, therefore $\mathsf{pans}(\gamma^+_{Inventory}(x, \alpha, S_0), \mathcal{D}_0) = \mathsf{pans}(\gamma^-_{Inventory}(x, \alpha, S_0), \mathcal{D}_0) = \{\}$ and

$$X^+_{Inventory} = X^-_{Inventory} = \{\}.$$

- The effect formulas $\gamma^+_{Inside}(x_1, x_2, \alpha, S_0)$ and $\gamma^-_{Inside}(x_1, x_2, \alpha, S_0)$ are simplified to *false*, therefore $\mathsf{pans}(\gamma^+_{Inside}(x_1, x_2, \alpha, S_0), \mathcal{D}_0) = \mathsf{pans}(\gamma^-_{Inside}(x_1, x_2, \alpha, S_0), \mathcal{D}_0) = \{\}$ and
$$X^+_{Inside} = X^-_{Inside} = \{\}.$$

- The effect formulas $\gamma^+_{InRange}(x_1, x_2, \alpha, S_0)$ and $\gamma^-_{InRange}(x_1, x_2, \alpha, S_0)$ are simplified to *false*, therefore $\mathsf{pans}(\gamma^+_{InRange}(x_1, x_2, \alpha, S_0), \mathcal{D}_0) = \mathsf{pans}(\gamma^-_{InRange}(x_1, x_2, \alpha, S_0), \mathcal{D}_0) = \{\}$ and
$$X^+_{InRange} = X^-_{InRange} = \{\}.$$

At each step we also update the variable Y using the variables X^+, X^- and Definition 5.1.1 so that at the end of the loop the variable Y holds the context set $\mathcal{J}$ of α. The variable Y is then the following set:

$$\{Status(bomb, w, S_0), Status(chest_1, w, S_0), Status(rkey, w, S_0),$$
$$InRange(bomb, w, S_0), Inside(chest_1, w, S_0), Inside(chest_2, w, S_0)\}.$$

Lines 20–21 of Algorithm 3 We compute the set of the $\mathcal{J}$-models and store it as the variable Z. The variable Z is then the following set:

$$\{\theta_1, \theta_2\},$$

where θ_1 is the following closure:

$$\chi_{20} \wedge \chi_{22} \wedge \chi_{23} \wedge \chi'_{25} \wedge \chi_{26},$$

and θ_2 is the following closure:

$$\chi_{20} \wedge \chi_{22} \wedge \chi_{23} \wedge \chi'_{25} \wedge \chi_{27}.$$

Lines 22–32 of Algorithm 3 We progress the set of the $\mathcal{J}$-models. The set Z is then updated to the following set:

$$\{\theta_1', \theta_2'\},$$

where θ_1' is the following closure:

$$\chi_{20}' \wedge \chi_{22}' \wedge \chi_{23}' \wedge \chi_{25}' \wedge \chi_{26},$$

θ_2' is the following closure:

$$\chi_{20}' \wedge \chi_{22}' \wedge \chi_{23} \wedge \chi_{25}' \wedge \chi_{27},$$

and $\chi_{20}', \chi_{22}', \chi_{23}'$ are as follows:

$$\begin{aligned}
\chi_{20}' : &\quad \forall w.\mathit{Status}(\mathit{bomb}, w, S_0) \equiv w = \mathit{broken},\\
\chi_{22}' : &\quad \forall w.\mathit{Status}(\mathit{chest}_1, w, S_0) \equiv w = \mathit{broken},\\
\chi_{23}' : &\quad \forall w.\mathit{Status}(\mathit{rkey}, w, S_0) \equiv w = \mathit{broken}.
\end{aligned}$$

Lines 33–42 of Algorithm 3 We update the database of possible closures $\mathcal{D}_0'$ to the set $\mathcal{D}_0''$ which is the following set:

$$(\mathcal{D}_0' - \{\chi_{20}, \chi_{22}, \chi_{23}, \chi_{25}',\ \chi_{26} \vee \chi_{27}\}) \cup \{\theta_1' \vee \theta_2'\}.$$

Essentially, the update we did in $\mathcal{D}_0'$ is that the status of the objects *bomb* and chest_1 is changed from *ok* to *broken*, and the status of the object *rkey* is changed similarly only for the case that the object *rkey* is inside the object chest_1.

For clarity we also give a more detailed list of the axioms in $\mathcal{D}_0''$ by showing the

closures that each axioms consists of. The set $\mathcal{D}_0''$ is the following set of axioms:

$$
\begin{aligned}
&\{\chi_1, \ldots, \chi_8,\\
&\chi_9' \ : \quad \forall w.At(node_1, w, S_0) \equiv false,\\
&\chi_{10} : \quad \forall w.At(node_2, w, S_0) \equiv false,\\
&\chi_{11}' : \quad \forall w.At(node_3, w, S_0) \equiv false,\\
&\chi_{12} : \quad \forall w.At(node_4, w, S_0) \equiv w = gdoor,\\
&\chi_{13}' : \quad \forall w.At(node_5, w, S_0) \equiv (w = chest_1 \vee w = bomb),\\
&\chi_{14}' : \quad \forall w.At(node_6, w, S_0) \equiv w = npc\\
&\chi_{15} : \quad \forall w.At(node_7, w, S_0) \equiv w = rdoor,\\
&\chi_{16} : \quad \forall w.At(node_7, w, S_0) \equiv w = chest_2,\\
&\chi_{17} : \quad \forall w.Status(npc, w, S_0) \equiv w = ok,\\
&\chi_{18} : \quad \forall w.Status(gdoor, w, S_0) \equiv w = ok,\\
&\chi_{19} : \quad \forall w.Status(rdoor, w, S_0) \equiv (w = ok \vee w = locked),\\
&\chi_{21} : \quad \forall w.Status(chest_2, w, S_0) \equiv (w = ok \vee w = locked),\\
&\chi_{24} : \quad \forall w.Inventory(w, S_0) \equiv w = false,\\
&\theta_1' \vee \theta_2'\},
\end{aligned}
$$

where θ_1' is the conjunction of the following closures:

$$
\begin{aligned}
&\{\chi_{20}' : \quad \forall w.Status(bomb, w, S_0) \equiv w = broken,\\
&\chi_{22}' : \quad \forall w.Status(chest_1, w, S_0) \equiv w = broken,\\
&\chi_{23}' : \quad \forall w.Status(rkey, w, S_0) \equiv w = broken,\\
&\chi_{25}' : \quad \forall w.InRange(bomb, w, S_0) \equiv (w = bomb \vee w = chest_1),\\
&\chi_{26} : \quad \forall w(Inside(chest_1, w, S_0) \equiv w = rkey) \wedge \forall w(Inside(chest_2, w, S_0) \equiv false)\}.
\end{aligned}
$$

and θ_2' is the conjunction of the following closures:

$$\begin{aligned}
\{\chi'_{20}: &\quad \forall w.Status(bomb, w, S_0) \equiv w = broken,\\
\chi'_{22}: &\quad \forall w.Status(chest_1, w, S_0) \equiv w = broken,\\
\chi_{23}: &\quad \forall w.Status(rkey, w, S_0) \equiv w = ok,\\
\chi'_{25}: &\quad \forall w.InRange(bomb, w, S_0) \equiv (w = bomb \vee w = chest_1),\\
\chi_{27}: &\quad \forall w(Inside(chest_1, w, S_0) \equiv false) \wedge \forall w(Inside(chest_2, w, S_0) \equiv w = rkey)\}.
\end{aligned}$$

■

Chapter 6

Conclusions

In this thesis we examined a reasoning module that is intended to be used in the design and implementation of agents that have cognitive abilities such as memory, perception, action, problem solving, etc, and are long-lived in the sense that they are expected to function autonomously for long periods of time. The module we presented provides the ability to reason about action and change using the language of the situation calculus and some variants of the basic action theories. The main focus of this thesis was on the functionality of the module that corresponds to the problem of progressing an action theory. The technical contributions of this thesis are as follows:

1. We investigated a conjecture by Lin and Reiter [1997] that a implied that a practical first-order definition of progression is not correct in the general case, and showed that Lin and Reiter were indeed correct in their intuitions by providing a proof for the conjecture. In this way we resolved the open question about the first-order definability of progression and justified the need for a second-order definition.

2. We investigated restrictions on the basic action theories under which a first-order progression is always correct and proved three major results as follows:

   (a) We showed that a first-order progression is always correct when we restrict our

attention to queries that may only quantify over situations in a limited way. This extends the result by Lin and Reiter [1997] that a first-order progression is always correct when we restrict our attention to queries that refer to a specific situation only.

(b) We revisited the local-effect assumption of Liu and Levesque [2005] that requires that the effects of every action are fixed by the arguments of the action term, and proved that under this restriction a first-order progression is always correct. Moreover we presented a method for computing the progression provided that a slightly stronger assumption holds.

(c) We investigated a way that the local-effect assumption can be relaxed so that a first-order progression is always correct for the more practical case that the effects of the actions may be specified also using information from the initial knowledge base. We showed that when the initial knowledge base is a *database of possible closures* and the effects of the actions are *range-restricted* then a first-order progression is always correct provided that a *just-in-time* assumption holds.

3. We examined a special case of the range-restricted theories and specified an algorithm for the core computational task that our progression method relies on. We proved the correctness and the complexity of the algorithm, and discussed the overall performance of the progression method.

We conclude with a brief list of topics for future research.

1. In this thesis we focused on the problem of progression and provided a clean formalization for the functionality of the reasoning module in the situation calculus. We also investigated practical methods for the problem of progression but we did not examine similar methods for computing the projection problem. One possible

line of future research is to investigate whether our results about progression imply similar results about projection. For instance, our result in Section 4.1 about progression shares intuitions with the result of Savelli [2006] about regression. We believe that there may be more cases like this.

2. As far as the problem of progression is concerned, one line of future research is to investigate more expressive variants of the basic action theories that are able to represent more features that are common in practical domains and also allow for a practical progression method. Three possible research directions follow.

   (a) Sensing actions: The theories we studied are able to represent incomplete information about the state of the world in the initial situation. Nonetheless we did not investigate the case that an action may provide *new* information that resolves some of the incompleteness through a sensing result. This is a necessary feature for many practical scenarios and several approaches for representing such actions have been studied in the literature, e.g., [Levesque, 1996], [Scherl and Levesque, 2003]. One line of future work is to investigate how the existing techniques apply in the classes of theories we studied with respect to progression. In particular for the range-restricted theories, it is crucial that the sensing results provide not only information about a finite number of ground fluent atoms but also what amounts to a new possible closures axiom for some atoms that may be unrestricted before the action is performed.

   (b) Nondeterministic actions: Similar to the point (a), we did not investigate the case that an action may result in *losing* information, that is, lead to a situation where there is incomplete information about the truth value of a fluent that is not derived from the incomplete information about the previous situation. This is also important for many practical scenarios where the agent may lose

track of certain properties of the domain. This functionality is similar to the notion of *forgetting* [Lin and Reiter, 1994] which was recently shown to be similar in spirit to the notion of progression [Liu and Lakemeyer, 2009]. Another line of future research is then to investigate how these intuitions apply in the theories we studied so that we can provide a practical account of nondeterministic actions with respect to progression.

(c) Ramifications: The basic action theories typically represent both the direct and indirect effects of actions in the successor state axioms. The problem of ramification refers to a more general setting where a chain of indirect effects may occur as a result of axioms that represent the general laws of the domain. This is important for practical scenarios where a"domino" effect of bounded size may occur. A possible line of future research is to investigate the notions of safe-range and just-in-time formulas as a means to formalize a variant of the basic action theories that allows for a range-restricted chain of effects.

3. As far as the practical task of implementing the reasoning module is concerned, we believe that it is important to investigate the details of a generic architecture design for agents that makes it easy for reasoning modules to interact and cooperate in a real software system.

   For instance consider the case of a video game that is implemented as a complex software system, and a non-player character that is implemented in this system but is also equipped with a reasoning module like the one we study. The game-engine typically provides the necessary functionality for representing some properties of the world such as the connectivity of the places in the game-world, and solving simple problems such as path-finding problems. We think it is important that this kind of functionality that the game-engine provides be treated as another reasoning module that may interact with other modules such as the one we study. For instance, our

module might use information from the path-finding functionality of the game-engine in order progress. It then becomes necessary to define a generic language for specifying the properties of the domain and the appropriate interfaces so that each reasoning module may provide useful services based on a common language.

Appendix A

A note on the various definitions of progression in the literature

Lin and Reiter were first to formally characterize when a set $\mathcal{D}_\alpha$ qualifies as a progression of $\mathcal{D}_0$ [Lin and Reiter, 1997, Definition 4.1]. In this seminal paper Lin and Reiter gave a model-theoretic definition for $\mathcal{D}_\alpha$ and proved that their definition is always correct. In their definition they required that the models of $\mathcal{D}_\alpha$ have a specific relation with the models of $\mathcal{D}$ as far as the situation S_α is concerned.

An important detail about the definition of progression by Lin and Reiter is that it is based on a slightly different version of the basic action theories than the ones we consider nowadays. In particular, the successor state axioms in [Lin and Reiter, 1997] had the following form:

$$Poss(a, s) \supset F(\vec{x}, do(a, s)) \equiv \Phi(\vec{x}, a, s),$$

that is, it should be the case that the action α is *executable* in S_0 in order for S_α to be affected in any way. Therefore, the model-theoretic definition of progression they provided also accounted for the occurrence of *Poss* in the successor state axioms. *Poss* is not actually mentioned in the definition but since the definition requires that the models of $\mathcal{D}_\alpha$ have a specific relation with the models of $\mathcal{D}$ as far as the situation S_α is concerned,

then *Poss* is also considered. For example, if α is not possible in S_0 then the models of $\mathcal{D}$ are such that S_0 and S_α are identical. It follows that the models of $\mathcal{D}_\alpha$ need to reflect this as well.

Another subtle detail is that the model-theoretic definition by Lin and Reiter requires that $\mathcal{D}_\alpha$ entails $\mathcal{D}_{una}$. This requirement and the effect of *Poss* is illustrated in their result about the second-order definability of progression [Lin and Reiter, 1997, Theorem 2]: they were able to prove that when $\mathcal{D}_0$ is a finite set, then the set $\mathcal{D}_{una}$ along with a particular second-order sentence that mentions *Poss* always qualifies as a progression of $\mathcal{D}_0$. The definition of progression of [Lin and Reiter, 1997] also implies that any two progressions of $\mathcal{D}_0$ are logically equivalent. As a result it follows that $\mathcal{D}_\alpha$ necessarily includes $\mathcal{D}_{una}$ or a logically equivalent representation of $\mathcal{D}_{una}$.

In this thesis we chose to define the progression of $\mathcal{D}_0$ using the second-order sentence that was introduced by Lin and Reiter. The reason is that it is often easier to work with a second-order sentence than a model-theoretic relation between theories. In order to use their result though, we had to remove the dependency on *Poss* as in the definition of the basic action theories that we use the predicate *Poss* is not mentioned in the successor state axioms. This was in fact easy as we only had to remove the left hand side of the implication of the second-order sentence of [Lin and Reiter, 1997]. Intuitively this implies that the situations progress as if $Poss(a, s)$ is always true.

Moreover, we find that it is not aesthetically nice to include $\mathcal{D}_{una}$ in $\mathcal{D}_\alpha$ knowing that $\mathcal{D}_{una}$ is already included in $\mathcal{D}$. In other words, since $\mathcal{D}_\alpha$ includes a logically equivalent representation of $\mathcal{D}_{una}$, the updated basic action theory $(\mathcal{D}-\mathcal{D}_0)\cup\mathcal{D}_\alpha$ essentially includes $\mathcal{D}_{una}$ twice. In order to prevent this we also removed the dependency on $\mathcal{D}_{una}$ so that the progression of $\mathcal{D}_0$ corresponds to a more intuitive set that does not include information that is implied by $\mathcal{D}_{una}$ unless it is necessary for specifying the situation S_α.[1]

[1]Note that as we explained in Example 3.2.10 some information from $\mathcal{D}_{una}$ is indeed necessary but $\mathcal{D}_\alpha$ does not have to include by default all the information in $\mathcal{D}_{una}$ that is usually represented as an infinite set of sentences.

So, the definition of a strong progression is essentially the original definition of progression by Lin and Reiter [1997] applied to the newer form of basic action theories that do not use *Poss* in the successor state axioms. In fact, the correctness of strong progression follows from the results in [Lin and Reiter, 1997] in the sense that we can follow the same reasoning as in the proofs of Lin and Reiter with only slight changes that correspond to the two points we mentioned about *Poss* and $\mathcal{D}_{una}$. Finally, the only restriction in the definition of strong progression is that $\mathcal{D}_0$ needs to be a finite set, but since we are interested in practical cases where $\mathcal{D}$ is a finite theory this definition is sufficient.

There is one more definition in the literature that is also very related to the original definition by Lin and Reiter, namely the definition of progression that appears in [Reiter, 2001, Definition 9.1.1]. This is also model-theoretic, more compact than the original definition by Lin and Reiter, and also accounts for the aesthetic issue we noted about the set $\mathcal{D}_{una}$. This definition of $\mathcal{D}_\alpha$ requires that the models of $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha$ instead of the models of $\mathcal{D}_\alpha$ have a specific relation with the models of $\mathcal{D}$ as far as the situations in the future of S_α are concerned. Nonetheless the definition in [Reiter, 2001] suffers from the following problem: it allows two sets that are not logically equivalent to both qualify as a progression of $\mathcal{D}_0$. Unfortunately we can no longer then assume that $\mathcal{D}_{una}$ is unique up to logical equivalence which in some sense means that $\mathcal{D}_\alpha$ is not well-defined.

For instance consider the case where $\mathcal{D}_0$ is the empty set and consider the action A that makes $F(s)$ true and does not affect any other fluent. The successor state axiom for F is then the following:

$$F(do(a, s)) \equiv (a = A \vee F(s)).$$

According to the definition in [Reiter, 2001] it is not too difficult to show that the set

$$\{F(do(A, S_0)\}$$

is a progression of $\mathcal{D}_0$, which is indeed the intended progression of $\mathcal{D}_0$. Unfortunately,

there is one more set that qualifies as a progression of $\mathcal{D}_0$, one that is not intended to qualify as a progression. Let $\mathcal{D}_\alpha$ be the empty set. Then it is trivial to show that $\mathcal{D}_\alpha$ is a progression of $\mathcal{D}_0$ according to the definition in [Reiter, 2001] simply because $\mathcal{D}$ and $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha$ are identical as both $\mathcal{D}_0$ and $\mathcal{D}_\alpha$ are empty. So, in this simple example the empty set and the set $\{F(do(A, S_0)\}$ both qualify as a progression of $\mathcal{D}_0$ with respect to the definition in [Reiter, 2001] but they are clearly not logically equivalent. Note that this cannot arise in the case of strong progression or the original definition by Lin and Reiter.

Appendix B

Long proofs

B.1 Proof of Lemma 3.3.9

Lemma 3.3.9 Let $\mathcal{D}$ be the basic action theory of the infinite doors domain of Definition 3.3.3 and $\mathcal{F}_\alpha$ the following set of first-order sentences:

$$\{\forall x(x = 0 \equiv F(x, S_A)),\ (3.5),\ (3.6),\ (3.7)\}.$$

Then, $\mathcal{F}_\alpha$ is a weak progression of $\mathcal{D}_0$ wrt the ground action A and $\mathcal{D}$.

Proof. By Definition 3.3.1 we need to show that for all first-order sentences ϕ uniform in $\S_A$, $\mathcal{F}_\alpha \cup \mathcal{D}_{una} \models \phi$ iff $\mathcal{D} \models \phi$.

For the ($\Leftarrow$) direction we proceed as follows. The theory $\mathcal{D}$ entails the successor state axiom for F therefore also entails the sentence we get by replacing s by S_0 and a by A:

$$\mathcal{D} \models \forall x\ F(x, do(A, S_0)) \equiv A = A \wedge x = 0 \vee$$
$$A = B \wedge \neg F(x, S_0) \wedge \exists y(x = n(y) \wedge F(y, S_0)),$$

By the axioms for equality and the sentence (3.4), i.e., $A \neq B$, it follows that

$$\mathcal{D} \models \forall x\ x = 0 \equiv F(x, S_A).$$

It is clear that $\mathcal{D} \models \{(3.5), (3.6), (3.7)\}$ as these are axioms of $\mathcal{D}$. Therefore

$$\mathcal{D} \models \{\forall x(x = 0 \equiv F(x, S_A)),\ (3.5), (3.6), (3.7)\}.$$

It follows that $\mathcal{D} \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}$. Now let ϕ be a first-order sentence uniform in S_A and assume that $\mathcal{F}_\alpha \cup \mathcal{D}_{una} \models \phi$. Then since $\mathcal{D} \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}$ and $\mathcal{F}_\alpha \cup \mathcal{D}_{una} \models \phi$ it follows that $\mathcal{D} \models \phi$.

For the ($\Rightarrow$) direction let Σ_1 be the set $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$, Σ_2 the set $\mathcal{D}$, and Γ the set of first-order sentences uniform in S_A. Then by Lemma 3.2.17 it suffices to show that for every model M of $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$, there is a model M' of $\mathcal{D}$ such that M and M' satisfy the same set of sentences in Γ. Let M be an arbitrary model of $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$. Based on Definition 3.3.4 of named and unnamed objects we distinguish two cases for: i) M has at least one element in the object domain that is unnamed, and ii) all the elements of the object domain of M are named. For both cases we will show that there is a model M' of $\mathcal{D}$ such that M and M' satisfy the same set of sentences in Γ.

Case i): $M \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}$ and M has at least one element in the object domain that is unnamed. We construct the model M' as follows.

1. M' has the same domains for sort object and action as M.

2. M' has a domain of strings Sit for the situation sort that is defined as follows. Sit includes the string "S_0" and it is the smallest set that is closed under the string operation that produces the string "$do(a, s)$", where a can be either the string "A" or the string "B" and s is any string in Sit.

3. M' interprets $0, n, A, B$ exactly as M.

4. M' interprets S_0, do such that every situation term σ in $\mathcal{L}$ is interpreted to the corresponding string "σ" in Sit.

5. $M', \mu_q^x \models F(x, S_0)$, for all q in the object domain of M' such that q is named.

6. $M', \mu_q^x \not\models F(x, S_0)$, for all q in the object domain of M' such that q is unnamed.

7. For all q in the object domain of M' (or equivalently the object domain of M), $M', \mu_q^x \models F(x, S_A)$ iff $M, \mu_q^x \models F(x, S_A)$.

8. For all the elements of Sit other than "S_0" and "$do(A, S_0)$" let the interpretation of $F(x, s)$ be the one that is defined by the successor state axiom for F and the interpretation of $F(x, S_0)$, $F(x, S_A)$.

9. $M' \models \forall a \forall s Poss(a, s)$.

By the construction of M' and in particular by the points 1,3,7 and by induction on the construction of the first-order formulas uniform in S_A it follows that for all sentences ϕ in Γ, $M \models \phi$ iff $M' \models \phi$. We now proceed to show also that M' is a model of $\mathcal{D}$, that is, M' satisfies the sentences (1) to (7) as they appear in Definition 3.3.3, as well as the foundational axioms $\mathcal{D}_{fnd}$ for basic action theories.

- Sentence (3.1): $Poss(a, s) \equiv true$.

  By point 9 it follows that $M' \models$ (3.1).

- Sentence (3.3):

$$F(x, do(a, s)) \equiv a = A \wedge x = 0 \vee \\ a = B \wedge \neg F(x, s) \wedge \exists y(x = n(y) \wedge F(y, s)).$$

Let $\Phi(x, a, s)$ be the formula on the right hand side of the logical symbol $\equiv$ in the sentence (3.3). By the point 8 it follows that for all variable assignments μ except for μ' which interprets $do(a, s)$ as "$do(A, S_0)$", $M', \mu \models \forall x.F(x, do(a, s)) \equiv \Phi(x, a, s)$. By the point 7 and since M is a model of $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$ it follows that $M' \models \forall x.x = 0 \equiv F(x, do(A, S_0))$ which implies that $M', \mu' \models \forall x.F(x, do(a, s)) \equiv \Phi(x, a, s)$. Therefore, $M' \models (3.3)$.

- Sentences (3.4), (3.5), (3.6), (3.7): $A \neq B$, $\forall a\ a = A \vee a = B$, $\forall x\ x \neq 0 \equiv \exists y\ n(y) = x$, $\forall x \forall y\ n(x) = n(y) \supset x = y$.

  $M \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}$ therefore $M \models \{(3.4), (3.5), (3.6), (3.7)\}$. By the points 1,3 it follows that $M' \models \{(3.4), (3.5), (3.6), (3.7)\}$.

- Sentence (3.8): $F(0, S_0) \wedge \forall x\,(F(x, S_0) \supset F(n(x), S_0))$.

  By the point 5 it follows that $M' \models (3.8)$.

- Sentence (3.9): $\exists x\ \neg F(x, S_0)$.

  By the point 6 it follows that $M' \models (3.9)$.

- $\mathcal{D}_{fnd}$.

  By the point 8 it follows that $M' \models \mathcal{D}_{fnd}$.

Thus, $M' \models \mathcal{D}$.

Case ii): $M \models \mathcal{F}_\alpha \cup \mathcal{D}_{una}$ and all the elements of the object domain of M are named. For this case we will use the Lowenheim-Skolem theorem of first-order logic to show that there is structure M_c that satisfies the same set of sentences in Γ as M but also has unnamed objects. We can then construct M' in the same way as for the *Case i)* so that M' is a model of $\mathcal{D}$ and satisfies the same set of sentences in Γ as M_c (and thus satisfies the same set of sentences in Γ as M). Even though the intuition for the construction

of M_c is relatively simple, the actual construction involves several tedious intermediate steps. In particular, in order to use the (upward) Lowenheim-Skolem theorem we'll need to transform the three-sorted model M into a normal one-sorted model M_1 in a way that the satisfaction of formulas is preserved, then apply the theorem to get an elementarily equivalent model M_2 with unnamed objects, and then transform M_2 to a model M_c of the three-sorted situation calculus language. The formal details follow.

Let $\mathcal{L}^*$ be the (one-sorted) first-order language that consists of the constant symbols $0, A, B$, the unary function symbol n, the unary predicate symbols P, Q, and the countably infinite number of variables $a, a_1, a_2, \ldots$ and $x, x_1, x_2, \ldots$. The predicate symbol P will be used to encode the truth of the fluent atoms of the form $F(t, S_A)$ and the predicate Q will be used to specify a subset of the domain that will be used as the action domain in $\mathcal{L}^*$. The variables $x, x_1, \ldots$ will be used to quantify over the whole domain while the variables $a, a_1, \ldots$ will be used to quantify over the set of objects that lie in the extension of Q. In particular let $\mathcal{L}^*_{S_A}$ be the smallest set of formulas in $\mathcal{L}^*$ such that the following conditions hold:

- for every predicate atom $P(t)$ such that the term t is of the form $n^k(0)$ or t is one of the variables $x, x_1, x_2, \ldots$, $P(t)$ is in $\mathcal{L}^*_{S_A}$;
- for every equality atom $t_1 = t_2$ such that t_1, t_2 are terms of the form $n^k(0)$ or one of the variables $x, x_1, x_2, \ldots$, $t_1 = t_2$ is in $\mathcal{L}^*_{S_A}$;
- for every equality atom $t_1 = t_2$ such that t_1, t_2 are one of the constants A, B or one of the variables $a, a_1, a_2, \ldots$, $t_1 = t_2$ is in $\mathcal{L}^*_{S_A}$;
- if ϕ, ψ are in $\mathcal{L}^*_{S_A}$ then $\neg\phi$, $\phi \wedge \psi$, $\forall x \phi$, and $\forall a\ Q(a) \supset \phi$ are in $\mathcal{L}^*_{S_A}$.

For every formula ϕ in Γ let ϕ^* be ϕ with each fluent atom of the form $F(t, S_A)$ replaced by the atom $P(t)$ and each action quantifier of the form $\forall a \psi$ replaced by $\forall a Q(a) \supset \phi$. It is not difficult to show that if ϕ is in Γ then ϕ^* is in $\mathcal{L}^*_{S_A}$ and similarly that we can do the inverse transformation in order to obtain a formula in Γ from a formula in $\mathcal{L}^*_{S_A}$.

We now construct a structure M_1 of the language $\mathcal{L}^*$ such that the following hold.

1. M_1 has the object domain of M as its domain and interprets $n, 0$ in the same way as M.

2. M_1 interprets A as the denotation of 0, B to the denotation of $n(0)$, and the extension of Q is such that $Q(a)$ is true only for the denotations of 0 and $n(0)$.

3. For all q in the domain of M_1 (or equivalently the object domain of M), $M_1, \mu_q^x \models P(x)$ iff $M, \mu_q^x \models F(x, S_A)$.

Let g_1 be a function from the action domain of M to the set $\{A^{M_1}, B^{M_1}\}$ such that g_1 maps the denotation of A in M to the denotation of A in M_1 and similarly for B. Since M models the sentences (3.4) and (3.5) it follows that the domain of actions for M has exactly two elements and so g_1 is a bijection. Let h_1 be the extension of g_1 that also maps each of the elements of the object domain of M to the identical element in the domain of M_1. The mapping h_1 is a bijection with the obvious definition for the inverse mapping. By induction then on the construction of the first-order formulas uniform in S_A it follows that for all first-order formulas ϕ uniform in S_A, $M, \mu \models \phi$ iff $M_1, h_1(\mu) \models \phi^*$. Therefore it follows that for all sentences ϕ in Γ,

$$M \models \phi \text{ iff } M_1 \models \phi^*. \tag{B.1}$$

By the (upward) Lowenheim-Skolem theorem of first-order logic it follows that there is a structure M_2 of the language $\mathcal{L}^*$ such that M_1 and M_2 are elementarily equivalent (i.e. they satisfy the same set of sentences in $\mathcal{L}^*$) but M_1 and M_2 are not isomorphic; in particular the domain of M_2 has a greater cardinality than the domain of M_1. So there is a model M_2 of the language $\mathcal{L}^*$ such that the following conditions hold.

1. The cardinality of the domain of M_2 is greater than the cardinality of the domain of M_1.

2. For all $\phi^* \in \mathcal{L}^*$,

$$M_1 \models \phi^* \text{ iff } M_2 \models \phi^*. \tag{B.2}$$

By the assumption we did for the *Case ii)* all the objects in the domain of M are named by some term in $\mathcal{L}$ of the form $n^k(0)$ and so by the point 1 in the construction of M_1 and the fact that $\mathcal{L}^*$ includes the symbols $n, 0$ it follows that all the objects in the domain of M_1 are also named by some term in $\mathcal{L}^*$. Since the set of all terms of $\mathcal{L}^*$ is countable this implies that the domain of M_1 is also countable. Now since the cardinality of the domain of M_2 is greater than the cardinality of domain of M_1 this implies that the domain of M_2 is uncountable. Since the set of all terms of $\mathcal{L}^*$ is countable it then follows that there is at least one unnamed object in M_2.

Based on the $\mathcal{L}^*$-structure M_2 we will now construct an $\mathcal{L}$-structure M_c that satisfies the same set of sentences in Γ as M but also has at least one unnamed object. We construct M_c as follows.

1. M_c has the domain of M_2 as the object domain and interprets $n, 0$ exactly as M_2.

2. M_c has the domain $\{A^{M_2}, B^{M_2}\}$ as the action domain and interprets A, B exactly as M_2.

3. M_c has the set of strings $\{\text{“S”}\}$ as the domain for sort situation and interprets all situation terms as the string “S”.

4. For all q in the object domain of M_c (or equivalently the domain of M_2),
   $M_c, \mu_q^x \models F(x, S_A)$ iff $M_2, \mu_q^x \models P(x)$.

The set of ground terms of sort object in $\mathcal{L}$ is a subset of the set of ground terms of $\mathcal{L}^*$. By the fact that there is at least one unnamed object in the domain of M_2 and point 1 in the construction of M_c it follows that there is at least one element of the object domain of M_c that is unnamed. So in order to complete the proof we only need to show

that M_c satisfies the same set of sentences in $\mathcal{L}^*_{S_A}$ as M. Let g_2 be a function from the action domain of M_c to the set $\{A^{M_2}, B^{M_2}\}$ such that g_2 maps the denotation of A in M_c to the denotation of A in M_2 and the same for B. Similarly to the definition of h_1 let h_2 be the extension of g_2 that also maps each of the elements of the object domain of M_c to the identical element in the domain of M_2. The mapping h_2 is a bijection with the obvious definition for the inverse mapping. By induction then on the construction of first-order formulas uniform in S_A it follows that for all first-order formulas uniform in S_A, $M_c, \mu \models \phi$ iff $M_2, h_2(\mu) \models \phi^*$. Therefore it follows that for every sentence ϕ in Γ,

$$M_c \models \phi \text{ iff } M_2 \models \phi^*. \tag{B.3}$$

By (B.1), (B.2), and (B.3) it then follows that for all sentences ϕ in Γ, $M \models \phi$ iff $M_c \models \phi$ which reduces the *Case ii)* to the the *Case i)*. □

B.2 Proof of Theorem 4.2.16

Theorem 4.2.16 Let $\mathcal{D}$ be a basic action theory that is local-effect and has a finite $\mathcal{D}_0$, α a ground action term, and $\mathcal{F}_\alpha$ a weak progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$. Then $\mathcal{F}_\alpha$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$.

Proof. We need to show that $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$ is logically equivalent to $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una}$. By the definition of $\mathcal{F}_\alpha$ and Theorem 3.2.11 it follows that $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una} \models \mathcal{F}_\alpha$.

For the other direction we need to show that for every model M of $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$ there is a configuration for the truth value of all the fluents that qualifies as a *predecessor* of S_α according to the axioms in $\mathcal{D}$. As far as the unaffected fluents are concerned, we just need to specify their truth value in this configuration to be the same as in the situation S_α in M. For the fluents that are affected by α we find a model M' of $\mathcal{D}$ such that M and M' satisfy the same set of first-order sentences uniform in S_α, and use it as a guideline.

Let $F_1, \ldots, F_n$ be the relational fluent symbols of $\mathcal{L}$, and $\mathcal{D}_{ss}$ a set of successor state axioms of the form $F_i(\vec{x}, do(a, s)) \equiv \Phi_i(\vec{x}, a, s)$. To simplify the analysis we assume that the initial knowledge base $\mathcal{D}_0$ is empty, but we will lift this restriction at the end. In this case $Pro(\mathcal{D}, \alpha)$ is the following second-order sentence:

$$\exists \vec{Q}. \bigwedge_{i=1}^{n} \forall \vec{x}. F_i(\vec{x}, S_\alpha) \equiv (\Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F} : \vec{Q} \rangle),$$

where $Q_1, \ldots, Q_n$ are predicate variables of the appropriate arity, and $\Phi_i(\vec{x}, \alpha, S_0)$ has the form:

$$\gamma^+_{F_i}(\vec{x}, \alpha, S_0) \vee (F(\vec{x}, S_0) \wedge \neg\gamma^-_{F_i}(\vec{x}, \alpha, S_0)).$$

Let M be an arbitrary model of $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$. We will show that $\mathcal{F}_\alpha \cup \mathcal{D}_{una} \models Pro(\mathcal{D}, \alpha)$ by specifying a relation R_i as an interpretation for each predicate variable Q_i. For each F_i we do as follows. Let $\vec{o}$ be a vector of elements in the object domain of M of the same arity as $\vec{x}$, μ a variable assignment, and $\{c_1, \ldots, c_m\}$ the argument set of F_i wrt α. We take the following two cases:

- $M, \mu^{\vec{x}}_{\vec{o}} \models \bigwedge_{j=1}^{m} \vec{x} \neq \vec{c}_j$. We define that $\vec{o}$ is in the relation R_i iff $M, \mu^{\vec{x}}_{\vec{o}} \models F_i(\vec{x}, S_\alpha)$. The intuition is that the objects $\vec{o}$ correspond to fluent atoms that remain unchanged in the transition from S_0 to S_α.

- $M, \mu^{\vec{x}}_{\vec{o}} \models \vec{x} = \vec{c}_j$ for some j, $1 \leq j \leq n$. For this case we will use a model of $\mathcal{D}$ to guide us for specifying whether $\vec{o}$ should be in R_i or not.

  Let Σ_1 be $\mathcal{F}_\alpha \cup \mathcal{D}_{una}$, Σ_2 be $\mathcal{D} - \mathcal{D}_{fnd}$, and Γ be the set of the first-order sentences uniform in S_α. By Lemma 3.2.18 it follows that there is a model M'' of $\mathcal{D} - \mathcal{D}_{fnd}$ such that for every first-order sentence ϕ uniform in S_α, $M \models \phi$ iff $M'' \models \phi$. By Lemma 3.2.12 it follows that there is a model M' of $\mathcal{D}$ such that for every first-order sentence ϕ uniform in S_α,

  $$M \models \phi \text{ iff } M' \models \phi. \tag{B.4}$$

We define that $\vec{o}$ is in R_i iff $M' \models F(\vec{c}_j, S_0)$. The intuition is that the objects $\vec{o}$ correspond to the fluent atom $F(\vec{c}_j, S_0)$ that may have changed in the transition from S_0 to S_α, and we define R_i to include $\vec{o}$ iff $F(\vec{c}_j, S_0)$ is satisfied in S_0 in M'.

We now proceed to show that the relations R_i we have specified qualify as a predecessor of S_α. That is, for every i, $1 \leq i \leq n$, we will show that

$$M, \mu \models \forall \vec{x}\big(F_i(\vec{x}, S_\alpha) \equiv \Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F}{:}\vec{Q}\rangle\big),$$

where μ interprets each predicate variable Q_i as the relation R_i. Let $\vec{o}$ be a vector of elements in the object domain of M of the same arity as $\vec{x}$, and $\{c_1, \ldots, c_m\}$ the argument set of F_i wrt α. We will take the same cases again as follows.

- $M, \mu^{\vec{x}}_{\vec{o}} \models \bigwedge_{j=1}^{m} \vec{x} \neq \vec{c}_j$. By the definition of the argument set of F it follows that $\Phi(\vec{x}, \alpha, S_0)$ reduces to $F(\vec{x}, S_0)$, and so $\Phi(\vec{x}, \alpha, S_0)\langle \vec{F}, \vec{Q}\rangle$ reduces to $Q(\vec{x})$. Therefore it suffices to show that $M, \mu^{\vec{x}}_{\vec{o}} \models F_i(\vec{x}, S_\alpha)) \equiv Q_i(\vec{x})$, which follows from the way we defined R_i.

- $M, \mu^{\vec{x}}_{\vec{o}} \models \vec{x} = \vec{c}_j$ for some j, $1 \leq j \leq n$. We need to show that

$$M, \mu^{\vec{x}}_{\vec{o}} \models F_i(\vec{x}, S_\alpha) \equiv \Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F}{:}\vec{Q}\rangle,$$

  or equivalently,

$$M, \mu \models F_i(\vec{c}_j, S_\alpha) \equiv \Phi_i(\vec{c}_j, \alpha, S_0)\langle \vec{F}{:}\vec{Q}\rangle.$$

  For the ($\Rightarrow$) direction let $M \models F_i(\vec{c}_j, S_\alpha)$.

  By (B.4) it follows that $M' \models F_i(\vec{c}_j, S_\alpha)$. Since M' is a model of $\mathcal{D}$, it follows that $M' \models \mathcal{D}_{ss}$ and so

$$M' \models \Phi_i(\vec{c}_j, \alpha, S_0).$$

  Let $\mathcal{G}$ be the characteristic set of α and $\{\theta_1, \ldots, \theta_l\}$ be the set of all the $\mathcal{G}$-models.

By the definition of a $\mathcal{G}$-model it follows that there is exactly one k, $1 \leq k \leq l$, such that

$$M' \models \theta_k[S_0].$$

Without loss of generality assume that $\Phi_i(\vec{c}_j, \alpha, S_0)$ is in NNF^+ and let ϕ be the situation-suppressed version of the formula $\Phi_i(\vec{c}_j, \alpha, S_0)$. By Lemma 4.2.15 it then follows that

$$M' \models \mathcal{V}(\theta_k, \phi)[S_0].$$

By Lemma 4.2.13 it follows that $\mathcal{V}(\theta_k, \phi)$ is unaffected wrt α and by Lemma 4.2.10 it follows that

$$M' \models \mathcal{V}(\theta_k, \phi)[S_\alpha],$$

By (B.4) it follows that

$$M \models \mathcal{V}(\theta_k, \phi)[S_\alpha].$$

Since $\mathcal{V}(\theta_k, \phi)$ is unaffected wrt α and the way we defined the relations $R_1, \ldots, R_n$, by using an argument similar to that of Lemma 4.2.10 it follows that

$$M, \mu \models (\mathcal{V}(\theta_k, \phi)[S_0])\langle \vec{F} : \vec{Q} \rangle. \tag{B.5}$$

Recall that $M' \models \theta_k[S_0]$, and θ_k is a conjunction of literals of the form $F_i(\vec{c}_j)$ or $\neg F_i(\vec{c}_j)$, where $\vec{c}_j$ is in the argument set of F_i wrt α. By the way we defined the relations $R_1, \ldots, R_n$ it follows then that

$$M, \mu \models (\theta_k[S_0])\langle \vec{F} : \vec{Q} \rangle \tag{B.6}$$

By (B.5) and (B.6) using an argument similar to that of Lemma 4.2.15 it follows that

$$M, \mu \models \phi[S_0]\langle \vec{F} : \vec{Q} \rangle,$$

i.e.,

$$M, \mu \models \Phi_i(\vec{c}_j, \alpha, S_0)\langle \vec{F} : \vec{Q} \rangle.$$

For the ($\Leftarrow$) direction let $M \not\models F_i(\vec{c}_j, S_\alpha)$ and follow the same reasoning.

Finally, to lift the assumption that $\mathcal{D}_0$ is empty we do the same as we did to show that $M, \mu \models \Phi_i(\vec{c}_j, \alpha, S_0)\langle \vec{F} : \vec{Q} \rangle$, but in this case we let $\phi[S_0]$ be the conjunction of all sentences in $\mathcal{D}_0$. □

B.3 Proof of Theorem 4.2.30

Theorem 4.2.30 Let $\mathcal{D}$ be a basic action theory that is strictly local-effect and has a finite $\mathcal{D}_0$, and ϕ a situation-suppressed sentence such that $\phi[S_0]$ is the conjunction of all the sentences in $\mathcal{D}_0$. Let α be a ground action, $\mathcal{J}$ the context set of α, $\{\theta_1, \ldots, \theta_m\}$ the set of all the $\mathcal{J}$-models, and ϕ^* the following situation-suppressed sentence:

$$\bigvee_{j=1}^{m} \Big(\theta_j^* \wedge \mathcal{V}(\theta_j, \phi) \Big),$$

where θ_j^* is the progression of θ_j wrt α. Then, $\phi^*[S_\alpha]$ is a strong progression of $\mathcal{D}_0$ wrt α and $\mathcal{D}$.

Proof. We need to show that $\{\phi^*[S_\alpha]\} \cup \mathcal{D}_{una}$ is logically equivalent to $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una}$, where $Pro(\mathcal{D}, \alpha)$ is the second-order sentence of Definition 3.2.8.

For the ($\Leftarrow$) direction we show that $\{Pro(\mathcal{D}, \alpha)\} \cup \mathcal{D}_{una} \models \phi^*[S_\alpha]$. Since $\phi^*[S_\alpha]$ is uniform in S_α by the definition of a strong progression and Theorem 3.2.11 it follows that it suffices to show that $\mathcal{D} \models \phi^*[S_\alpha]$. Let M be an arbitrary model of $\mathcal{D}$. By the definition of a $\mathcal{J}$-model and the fact that $\{\theta_1, \ldots, \theta_m\}$ is the set of all the $\mathcal{J}$-models it follows that there is exactly one k, $1 \leq k \leq m$, such that $M \models \theta_k[S_0]$. Since M is a model of $\mathcal{D}$ it satisfies $\phi[S_0]$, therefore by Lemma 4.2.27 it follows that $M \models \mathcal{V}(\theta_k, \phi)[S_0]$.

By Lemmas 4.2.10 and 4.2.13 it follows that $M \models \mathcal{V}(\theta_k, \phi)[S_\alpha]$. Since M also satisfies $\mathcal{D}_{ss}$ and by the way we defined the progression of a $\mathcal{J}$-model it follows that $M \models \theta^*[S_\alpha]$. Therefore, $M \models (\theta^* \wedge \mathcal{V}(\theta_k, \phi))[S_\alpha]$ and so $M \models \phi^*[S_\alpha]$. Since M was arbitrary it follows that $\mathcal{D} \models \phi^*[S_\alpha]$.

For the ($\Rightarrow$) direction we will show that $\phi^*[S_\alpha] \models \mathit{Pro}(\mathcal{D}, \alpha)$. Let M be an arbitrary model of $\phi^*[S_\alpha]$. Similar to what we did for Theorem 4.2.16, we need to show that there is a configuration for the truth value of all the fluents that qualifies as a predecessor of S_α in M according to the axioms in $\mathcal{D}$. Let $F_1, \ldots, F_n$ be all the relational fluent symbols of $\mathcal{L}$, and recall that $\mathit{Pro}(\mathcal{D}, \alpha)$ is the following second-order sentence:

$$\exists \vec{Q}.\ (\phi[S_0])\langle \vec{F} : \vec{Q} \rangle \ \wedge \ \bigwedge_{i=1}^{n} \forall \vec{x}. F_i(\vec{x}, S_\alpha) \equiv (\Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F} : \vec{Q} \rangle).$$

We will show that $\phi^*[S_\alpha] \models \mathit{Pro}(\mathcal{D}, \alpha)$ by specifying a relation R_i as an interpretation for each predicate variable Q_i. For each F_i we do as follows. Let $\vec{o}$ be a vector of elements in the object domain of M of the same arity as $\vec{x}$, μ a variable assignment, and $\{c_1, \ldots, c_m\}$ the argument set of F_i wrt α. We take the following two cases:

- $M, \mu^{\vec{x}}_{\vec{o}} \models \bigwedge_{j=1}^{m} \vec{x} \neq \vec{c}_j$. We define that $\vec{o}$ is in the relation R_i iff $M, \mu^{\vec{x}}_{\vec{o}} \models F_i(\vec{x}, S_\alpha)$. The intuition is that the objects $\vec{o}$ correspond to fluent atoms that remain unchanged in the transition from S_0 to S_α.

- $M, \mu^{\vec{x}}_{\vec{o}} \models \vec{x} = \vec{c}_j$ for some j, $1 \leq j \leq n$. By the way we defined ϕ^* it follows that there exists a k, $1 \leq k \leq m$, such that

  $$M \models \theta^*_k[S_\alpha].$$

  We define that $\vec{o}$ is in the relation R_i iff the positive literal $F_i(\vec{c}_j)$ is in θ_k. The intuition is that the objects $\vec{o}$ correspond to fluent atoms that may have changed in the transition from S_0 to S_α and we use the predecessor of θ^*_k to specify their

truth value.

We now proceed to show that the relations R_i we have specified qualify as a predecessor of S_α. First note that from the way we defined the relations R_i it follows that

$$M \models (\theta_k[S_0])\langle \vec{F}{:}\vec{Q}\rangle. \tag{B.7}$$

First, we show that for every i, $1 \leq i \leq n$,

$$M, \mu \models \forall \vec{x}\big(F_i(\vec{x}, S_\alpha) \equiv \Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F}{:}\vec{Q}\rangle\big),$$

where μ interprets each predicate variable Q_i as the relation R_i. Let $\vec{o}$ be a vector of elements in the object domain of M of the same arity as $\vec{x}$, and $\{c_1, \ldots, c_m\}$ the argument set of F_i wrt α. We will take the same cases again as follows.

- $M, \mu^{\vec{x}}_{\vec{o}} \models \bigwedge_{j=1}^{m} \vec{x} \neq \vec{c}_j$. By the definition of the argument set of F it follows that $\Phi(\vec{x}, \alpha, S_0)$ reduces to $F(\vec{x}, S_0)$, and so $\Phi(\vec{x}, \alpha, S_0)\langle \vec{F}, \vec{Q}\rangle$ reduces to $Q(\vec{x})$. Therefore it suffices to show that $M, \mu^{\vec{x}}_{\vec{o}} \models F_i(\vec{x}, S_\alpha)) \equiv Q_i(\vec{x})$, which follows from the way we defined R_i.

- $M, \mu^{\vec{x}}_{\vec{o}} \models \vec{x} = \vec{c}_j$ for some j, $1 \leq j \leq n$. We need to show that

  $$M, \mu^{\vec{x}}_{\vec{o}} \models F_i(\vec{x}, S_\alpha) \equiv \Phi_i(\vec{x}, \alpha, S_0)\langle \vec{F}{:}\vec{Q}\rangle,$$

  or equivalently,

  $$M, \mu \models F_i(\vec{c}_j, S_\alpha) \equiv \Phi_i(\vec{c}_j, \alpha, S_0)\langle \vec{F}{:}\vec{Q}\rangle.$$

  This follows from the fact that $F_i(\vec{c}_j)$ is in the context set $\mathcal{J}$, the way we defined the relations R_i, and the fact that θ^*_k is a progression of θ_k.

Second, we show that

$$M, \mu \models (\phi[S_0])\langle \vec{F}{:}\vec{Q}\rangle.$$

By Lemma 4.2.27 it follows that

$$M \models \phi[S_0] \equiv \bigvee_{j=1}^{m} \Big(\theta_j \wedge \mathcal{V}(\theta_j, \phi)\Big)[S_0].$$

Therefore, by (B.7) and using an argument similar to that of Lemma 4.2.15 it follows that it suffices to show that

$$M, \mu \models (\mathcal{V}(\theta_k, \phi)[S_0])\langle \vec{F} : \vec{Q} \rangle.$$

This follows from the fact that $\mathcal{V}(\theta_k, \phi)$ is unaffected wrt α and the way we defined the relations $R_1, \ldots, R_n$, by using an argument similar to that of Lemma 4.2.10. □

B.4 Proof of Lemma 4.3.13

Lemma 4.3.13 Let $\mathcal{D}_0$ be a database of possible values and $\gamma(\vec{x})$ a first-order formula that is range-restricted, just-in-time wrt $\mathcal{D}_0$, and does not mention any free variable other than those in $\vec{x}$. Then, $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is a finite set $\{(\vec{c}_1, \chi_1), \ldots, (\vec{c}_n, \chi_n)\}$ such that every χ_i is a conjunction of possible atomic closures wrt $\mathcal{D}_0$ and every c_i consists of constants in $\mathcal{D}_0$ and $\gamma(\vec{x})$, and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}.\gamma(\vec{x}) \equiv \bigvee_{i=1}^{n} (\vec{x} = \vec{c}_i \wedge \chi_i).$$

Proof. It suffices to prove a stronger lemma about the safe-range formulas as follows. Let $\gamma(\vec{x}, \vec{y})$ be a first-order formula that is just-in-time wrt $\mathcal{D}_0$, safe-range wrt the variables in $\vec{x}$, and does not mention any free variable other than $\vec{x}, \vec{y}$. Then for every constant vector $\vec{d}$ that has the same size as $\vec{y}$, $\mathsf{pans}(\gamma(\vec{x}, \vec{d}), \mathcal{D}_0)$ is a finite set $\{(\vec{e}_1, \chi_1), \ldots, (\vec{e}_n, \chi_n)\}$ such that every χ_i is a conjunction of possible atomic closures wrt $\mathcal{D}_0$ and every e_i consists of

constants in $\mathcal{D}_0$ and $\gamma(\vec{x}, \vec{d})$, and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}.\gamma(\vec{x}, \vec{d}) \equiv \bigvee_{i=1}^{n} (\vec{x} = \vec{e}_i \wedge \chi_i).$$

We prove this lemma by induction on the construction of the formulas γ. Since γ is safe-range wrt the variables in $\vec{x}$ we only need to consider the cases of Definition 4.3.12.

Base case. We only show the case that $\gamma(x, y)$ is $F(\vec{c}_1, y, \vec{c}_2, x)$, where x, y are distinct variables and $\vec{c}_1, \vec{c}_2$ are vectors of constants of sort object, and the other cases are similar. Let d be an arbitrary constant of the language. Then $\gamma(x, d)$ is the formula $F(\vec{c}_1, d, \vec{c}_2, x)$. We first show that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$. Let e be a constant of $\mathcal{L}$. We take two cases as follows.

- Case i): $F(\vec{c}_1, d, \vec{c}_2, e)$ is consistent with $\mathcal{D}_0 \cup \mathcal{E}$. Then, by the fact that $\gamma(x, y)$ is just-in-time wrt $\mathcal{D}_0$ it follows that there is a closure χ such that $\{\chi\} \cup \mathcal{E} \models \gamma(e, d)$, where χ is a conjunction of closures each of which is a possible closure wrt $\mathcal{D}_0$. It follows that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$.

- Case ii): $F(\vec{c}_1, d, \vec{c}_2, e)$ is not consistent with $\mathcal{D}_0 \cup \mathcal{E}$. It follows that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$, such that $F(\vec{c}_1, d, \vec{c}_2, e)$ is not true in any of the possible closures wrt ϕ.

It follows that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$. Without loss of generality we assume that ϕ is a possible closures axiom for $F(\vec{c}_1, d, \vec{c}_2, w)$. We will show how to rewrite ϕ in the form that the lemma requires. The axiom ϕ has the form

$$\bigvee_{i=1}^{n} \chi_i,$$

where each χ_i is an atomic closure of $F(\vec{c}_1, d, \vec{c}_2, w)$ on some set of constants $\{e_1, \ldots, e_m\}$,

i.e, a sentence of the form

$$\forall w.F(\vec{c}_1, d, \vec{c}_2, w) \equiv w = e_1 \vee \ldots \vee w = e_m.$$

For each χ_i of this form let χ_i' be the formula

$$\bigvee_{j=1}^{m}(x = e_j \wedge \chi_i),$$

and let ϕ' be

$$\forall x.F(\vec{c}_1, d, \vec{c}_2, x) \equiv \bigvee_{i=1}^{n} \chi_i'.$$

Note that ϕ' has the form that the lemma requires. So, in order to prove the lemma it suffices to show that $\mathcal{D}_0 \cup \mathcal{E} \models \phi'$. Let M be an arbitrary model of $\mathcal{D}_0 \cup \mathcal{E}$. Since ϕ is a sentence in $\mathcal{D}_0$ it follows that $M \models \phi$. By the definition of a possible closures axiom and Lemma 4.3.3 it follows that there is exactly k, $1 \leq k \leq n$, such that $M \models \chi_k$. Observe that if we simplify χ_k to *true* and all the other χ_i to *false* in ϕ' we obtain the sentence χ_k. Therefore, $M \models \phi'$ and since M was an arbitrary model of $\mathcal{D}_0 \cup \mathcal{E}$, it follows that $\mathcal{D}_0 \cup \mathcal{E} \models \phi'$. Also, by the definition of a possible answer and the structure of ϕ' it follows that the set $\mathsf{pans}(\gamma(x, d), \mathcal{D}_0)$ is the set that the lemma requires. Finally, since d was arbitrary it follows that this holds for every d, thus the lemma holds for the case of $F(\vec{c}_1, d, \vec{c}_2, w)$.

Inductive step. We only show the case when γ is a disjunction of formulas and the other cases are similar. We assume the lemma holds for $\phi_1(\vec{x}_1, \vec{z}, \vec{y}_1)$ that is safe-range wrt the variables in $\vec{x}_1$ and $\vec{z}$, and does not mention any free variable other than $\vec{x}_1, \vec{z}, \vec{y}_1$. Similarly we assume that the lemma holds for $\phi_2(\vec{x}_2, \vec{z}, \vec{y}_2)$ that is safe-range wrt the variables in $\vec{x}_2$ and $\vec{z}$, and does not mention any free variable other than $\vec{x}_2, \vec{z}, \vec{y}_2$, where

$\vec{x}_1$ and $\vec{x}_2$ share no variables. Let $\gamma(\vec{x}_1, \vec{y}_1, \vec{x}_2, \vec{y}_2, \vec{z})$ be the formula

$$\phi_1(\vec{x}_1, \vec{z}, \vec{y}_1) \vee \phi_2(\vec{x}_2, \vec{z}, \vec{y}_2).$$

We will show that the lemma holds for γ. By the definition of the safe-range formulas it follows that γ is safe-range for the variables in $\vec{z}$. Let $\vec{d_1}$ be a vector of constants of the same size as $\vec{y}_1$, and $\vec{d_2}$ be a vector of constants of the same size as $\vec{y}_2$. By the induction hypothesis it follows that $\mathsf{pans}(\phi_1(\vec{x}_1, \vec{z}, \vec{d_1}), \mathcal{D}_0)$ is a finite set of the form

$$\{(\langle \vec{c}_{11}, \vec{e}_{11} \rangle, \chi_{11}), \ldots, (\langle \vec{c}_{1n}, \vec{e}_{1n} \rangle, \chi_{1n})\}$$

and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}_1 . \forall \vec{z} . \phi_1(\vec{x}_1, \vec{z}, \vec{d_1}) \equiv \bigvee_{i=1}^{n} (\vec{x}_1 = \vec{c}_{1i} \wedge \vec{z} = \vec{e}_{1i} \wedge \chi_{1i}).$$

Similarly, $\mathsf{pans}(\phi_2(\vec{x}_2, \vec{z}, \vec{d_2}), \mathcal{D}_0)$ is a finite set of the form

$$\{(\langle \vec{c}_{21}, \vec{e}_{21} \rangle, \chi_{21}), \ldots, (\langle \vec{c}_{2m}, \vec{e}_{2m} \rangle, \chi_{2m})\},$$

and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}_2 . \forall \vec{z} . \phi_2(\vec{x}_2, \vec{z}, \vec{d_2}) \equiv \bigvee_{i=1}^{m} (\vec{x}_2 = \vec{c}_{2i} \wedge \vec{z} = \vec{e}_{2i} \wedge \chi_{2i}).$$

Let $\vec{b_1}$ be a vector of constants of the same size as $\vec{x}_1$ and $\vec{b_2}$ a vector of constants of the same size as $\vec{x}_2$. It follows that

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{z} . \phi_1(\vec{b_1}, \vec{z}, \vec{d_2}) \vee \phi_2(\vec{b_2}, \vec{z}, \vec{d_2}) \equiv \bigvee_{i=1}^{n} (\vec{b_1} = \vec{c}_{1i} \wedge \vec{z} = \vec{e}_{1i} \wedge \chi_{1i}) \vee \bigvee_{i=1}^{m} (\vec{b_2} = \vec{c}_{2i} \wedge \vec{z} = \vec{e}_{2i} \wedge \chi_{2i}).$$

By the uniqueness of names for the constants of sort object it follows that each of the atoms of the form $\vec{b}_1 = \vec{c}_{1i}$ and $\vec{b}_2 = \vec{c}_{2i}$ can be simplified to either *true* or *false*. Therefore the previous sentence can be simplified to the form that the lemma requires and the lemma follows. □

List of definitions

List of lemmas

List of theorems

List of Symbols

$\mathcal{C}_F$ The argument set of F, page 77

χ A closure of $\vec{\tau}$, page 101

$\mathcal{D}_0$ The initial knowledge base, page 38

$\mathcal{D}_\alpha$ A strong progression of $\mathcal{D}_0$, page 45

$\mathcal{D}_{ap}$ The set of action precondition axioms, page 37

$\mathcal{D}_{fnd}$ The set of domain independent foundational axioms, page 38

$\mathcal{D}_{ss}$ The set of successor state axioms, page 37

$\mathcal{D}_{una}$ The set of unique-names axioms for actions, page 38

$\mathcal{E}$ The set of unique-names axioms for objects, page 100

$\mathcal{F}_\alpha$ A weak progression of $\mathcal{D}_0$, page 53

$\mathcal{G}$ The characteristic set of α, page 77

γ_F^+ The positive effects formula of F, page 72

γ_F^- The negative effects formula of F, page 72

$\mathcal{J}$ The context set of α, page 91

$\mathcal{L}$ The language of the situation calculus, page 30

$\mathcal{L}_\sigma$ The set of sentences uniform in σ, page 32

pans The possible answers to a query, page 106

Poss The predicate that expresses the executability of actions, page 31

Pro The second-order sentence in the definition of strong progression, page 44

$\mathcal{Q}_\sigma$ A class of restricted sentences about the future of σ, page 66

$\mathcal{R}$ The regression operator, page 50

$\mathcal{R}_\sigma$ The generalized regression operator wrt σ, page 65

$\mathcal{T}$ The operator $\mathcal{T}$ for formulas, page 93

τ A fluent atom with ground input, page 101

$\mathcal{V}$ The operator $\mathcal{V}$ for situation-suppressed formulas., page 81

do The function for building action sequences, page 30

$\sqsubset$ The ordering of situations, page 31

$\sqsubseteq$ A macro based on $\sqsubset$, page 32

S_0 The initial situation, page 30

S_α A macro for the situation term $do(\alpha, S_0)$, page 34

Index

Bibliography

[Abiteboul *et al.*, 1994] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases: The Logical Level*. Addison Wesley, 1994.

[Apt and Pellegrini, 1994] K. Apt and A. Pellegrini. On the occur-check free Prolog program. *ACM Toplas*, 16(3):687–726, 1994.

[Baral and Gelfond, 2005] C. Baral and M. Gelfond. Logic programming and reasoning about actions. In Michael Fisher, Dov Gabbay, and Lluis Vila, editors, *Handbook of Temporal reasoning in AI*. Elsevier Publications, 2005.

[Bordini *et al.*, 2005] Rafael H. Bordini, Mehdi Dastani, Jürgen Dix, and Amal El Fallah-Seghrouchni, editors. *Multi-Agent Programming: Languages, Platforms and Applications*, volume 15 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*. Springer, 2005.

[De Giacomo and Levesque, 1999a] G. De Giacomo and H. J. Levesque. An incremental interpreter for high-level programs with sensing. *Logical Foundations for Cognitive Agents: Contributions in Honor of Ray Reiter*, pages 86–102, 1999.

[De Giacomo and Levesque, 1999b] Giuseppe De Giacomo and Hector J. Levesque. Projection using regression and sensors. In *IJCAI*, pages 160–165, 1999.

[De Giacomo *et al.*, 1997] Giuseppe De Giacomo, Yves Lesperance, and Hector J. Levesque. Reasoning about concurrent execution prioritized interrupts, and exoge-

nous actions in the situation calculus. In *Proceedings of the Fifteenth International JointConference on AI (IJCAI'97)*, pages 1221–1226, Nagoya, August 1997.

[De Giacomo *et al.*, 2001] G. De Giacomo, H. J. Levesque, and S. Sardina. Incremental execution of guarded theories. *Computational Logic*, 2(4):495–525, 2001.

[Demolombe and Pozos Para, 2000] R. Demolombe and P. Pozos Para. A simple and tractable extension of situation calculus to epistemic logic. In *Proc. ISMIS-00*, number 1932 in LNAI, pages 515–524, Charlotte, NC, USA, 2000. Springer.

[Enderton, 1972] Herbert B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, Inc., Orlando, Florida, 1972.

[Fikes and Nilsson, 1971] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[Funge, 1999] J. Funge. Representing knowledge within the situation calculus using intervalvalued epistemic fluents, 1999.

[Gabaldon, 2002] A. Gabaldon. Non-markovian control in the situation calculus. In *Proceedings of the Eighteenth national conference on Artificial intelligence (AAAI-02)*, pages 519–524, Menlo Park, CA, USA, 2002. AAAI Press.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3/4):365–386, 1991.

[Gelfond and Lifschitz, 1993] Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 2-4:301–323, 1993.

[Ghallab *et al.*, 2004] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann Publishers, 2004.

[Hindriks *et al.*, 1999] Koen V. Hindriks, Frank S. De Boer, and Wiebe and van der Hoek. Agent programming in 3APL. *Autonomous Agents and Multi-Agent Systems*, 2(4):357–401, 1999.

[Hintikka, 1962] J. Hintikka. Knowledge and belief. *Knowledge and Belief*, 1962.

[Hölldobler and Schneeberger, 1990] Steffen Hölldobler and Josef Schneeberger. A new deductive approach to planning. *New Generation Comput.*, 8(3), 1990.

[Kelly and Pearce, 2007] Ryan F. Kelly and Adrian R. Pearce. Property persistence in the situation calculus. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 1948–1953, 2007.

[Kowalski and Sergot, 1986] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95, 1986.

[Lesperance *et al.*, 1994] Yves Lesperance, Hector J. Levesque, Fangzhen Lin, Daniel Marcu, Raymond Reiter, and Richard B. Scherl. A logical approach to high level robot programming – a progress report. In B. Kuipers, editor, *Working notes of the 1994 AAAI fall symposium on Control of the Physical World by Intelligent Systems*, New Orleans, LA, November 1994.

[Levesque and Lakemeyer, 2007] Hector Levesque and Gerhard Lakemeyer. Chapter 23: Cognitive robotics. In *Handbook of Knowledge Representation (Foundations of Artificial Intelligence)*, 2007.

[Levesque *et al.*, 1997] H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, and R. B. Scherl. Golog: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1-3):59–83, 1997.

[Levesque, 1996] Hector Levesque. What is planning in the presence of sensing? In *The Proceedings of the Thirteenth National Conference on Artificial Intelligence, AAAI-96*,

pages 1139–1146, Portland, Oregon, August 1996. American Association for Artificial Intelligence.

[Levesque, 2005] H. J. Levesque. Planning with loops. In *Proc. IJCAI-05*, Edinburgh, Scotland, August 2005.

[Lin and Reiter, 1994] Fangzhen Lin and Raymond Reiter. Forget it! In Russell Greiner and Devika Subramanian, editors, *Working Notes, AAAI Fall Symposium on Relevance*, pages 154–159, Menlo Park, California, 1994. American Association for Artificial Intelligence.

[Lin and Reiter, 1997] F. Lin and R. Reiter. How to progress a database. *Artificial Intelligence*, 92(1-2):131–167, 1997.

[Lin, 1996] Fangzhen Lin. Embracing causality in specifying the indeterminate effects of actions. In *AAAI/IAAI, Vol. 1*, pages 670–676, 1996.

[Liu and Lakemeyer, 2009] Y. Liu and G. Lakemeyer. On first-order definability and computability of progression for local-effect actions and beyond. In *Proc. IJCAI-09*, 2009.

[Liu and Levesque, 2005] Y. Liu and H. J. Levesque. Tractable reasoning with incomplete first-order knowledge in dynamic systems with context-dependent actions. In *Proc. IJCAI-05*, Edinburgh, Scotland, August 2005.

[McCarthy and Hayes, 1969] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[McCarthy, 1980] John McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[Mendelson, 1997] Elliott Mendelson. *Introduction to Mathematical Logic*. Chapman & Hall/CRC, fourth edition, 1997.

[Pednault, 1987] E. P. D. Pednault. *Toward a mathematical theory of plan synthesis*. PhD thesis, Stanford University, Stanford, CA, USA, 1987.

[Peppas *et al.*, 1995] P. Peppas, N. Foo, and M. Williams. On the expressibility of propositions. *Logique et Analyse*, 139–149:251–272, 1995.

[Petrick and Bacchus, 2002] R. Petrick and F. Bacchus. A knowledge-based approach to planning with incomplete information and sensing, 2002.

[Petrick and Bacchus, 2004] Ronald P. A. Petrick and Fahiem Bacchus. Extending the knowledge-based approach to planning with incomplete information and sensing. In Didier Dubois, Christopher Welty, and Mary-Anne Williams, editors, *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR-2004)*, pages 613–622, Menlo Park, CA, June 2004. AAAI Press.

[Pirri and Reiter, 1999] Fiora Pirri and Ray Reiter. Some contributions to the metatheory of the situation calculus. *Journal of the ACM*, 46(3):261–325, 1999.

[Rao, 1996] Anand S. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In Rudy van Hoe, editor, *Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, The Netherlands, 1996.

[Reiter, 1980] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[Reiter, 1991] Ray Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 359–380. Academic Press, San Diego, CA, 1991.

[Reiter, 1993] R. Reiter. Proving properties of states in the situation calculus. *Artificial Intelligence*, 64(2):337–351, 1993.

[Reiter, 2001] R. Reiter. *Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.

[Sandewall, 1995] Erik Sandewall. *Features and fluents: the representation of knowledge about dynamical systems*. Oxford University Press, 1995.

[Sardina and Vassos, 2005] S. Sardina and S. Vassos. The Wumpus World in IndiGolog: A Preliminary Report. In *Proc. NRAC-05*, pages 90–95, Edinburgh, Scottland, 2005.

[Sardina, 2005] Sebastian Sardina. *Deliberation in Agent Programming Languages*. PhD thesis, University of Toronto, 2005.

[Savelli, 2006] F. Savelli. Existential assertions and quantum levels on the tree of the situation calculus. *Artificial Intelligence*, 170(6):643–652, May 2006.

[Scherl and Levesque, 1993] Richard B. Scherl and Hector J. Levesque. The frame problem and knowledge-producing actions. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 689–697, Washington, D.C., USA, 1993. AAAI Press/MIT Press.

[Scherl and Levesque, 2003] R. Scherl and H. J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144(1–2):1–39, 2003.

[Schoppers, 1987] M. J. Schoppers. Universal plans for reactive robots in unpredictable environments. In John McDermott, editor, *Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87)*, pages 1039–1046, Milan, Italy, 1987. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

[Shanahan, 1999] M. Shanahan. The event calculus explained. *Lecture Notes in Computer Science*, 1600:409–430, 1999.

[Shirazi and Amir, 2005] Afsaneh Shirazi and Eyal Amir. First-order logical filtering. In *Proc. of IJCAI-05*, pages 589–595, 2005.

[Shoham, 1993] Yoav Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[Son and Baral, 2001] T. C. Son and C. Baral. Formalizing sensing actions a transition function based approach. *Artificial Intelligence*, 125(1-2):19–91, 2001.

[Thielscher, 1999] M. Thielscher. From situation calculus to fluent calculus: State update axioms as a solution to the inferential frame problem. *Artificial Intelligence*, 111(1-2):277–299, July 1999.

[Thielscher, 2000] Michael Thielscher. Representing the knowledge of a robot. In Anthony G. Cohn, Fausto Giunchiglia, and Bart Selman, editors, *KR2000: Principles of Knowledge Representation and Reasoning*, pages 109–120, San Francisco, 2000. Morgan Kaufmann.

[Thielscher, 2001] Michael Thielscher. The qualification problem: A solution to the problem of anomalous models. *Artificial Intelligence*, 131(1-2):1–37, 2001.

[Thielscher, 2004] M. Thielscher. FLUX: A logic programming method for reasoning agents. *Theory and Practice of Logic Programing*, 5(4-5):533–565, 2004.

[Thomas, 1995] Rebecca S. Thomas. The PLACA agent programming language. In *ECAI-94: Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents*, pages 355–370, New York, NY, USA, 1995. Springer-Verlag New York, Inc.

[van Ditmarsch *et al.*, 2007a] Hans van Ditmarsch, Andreas Herzig, and Tiago de Lima. Optimal regression for reasoning about knowledge and actions. In Giacomo Bonanno, James Delgrande, Jérôme Lang, and Hans Rott, editors, *Formal Models of Belief Change in Rational Agents*, number 07351 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[van Ditmarsch *et al.*, 2007b] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*. Springer Press, 2007.

[Vassos and Levesque, 2007] Stavros Vassos and Hector Levesque. Progression of situation calculus action theories with incomplete information. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2024–2029, Hyderabad, India, January 2007.

[Vassos and Levesque, 2008] Stavros Vassos and Hector Levesque. On the progression of situation calculus basic action theories: Resolving a 10-year-old conjecture. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-08)*, pages 1004–1009, Chicago, Illinois, USA, July 13–17 2008.

[Vassos *et al.*, 2008] Stavros Vassos, Lakemeyer Gerhard, and Hector Levesque. First-order strong progression for local-effect basic action theories. In *Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning (KR-08)*, pages 662–272, Sydney, Australia, September 16–19 2008.

[Vassos *et al.*, 2009] Stavros Vassos, Sebastian Sardina, and Hector Levesque. Progressing basic action theories with non-local effect actions. In *Proceedings of the Ninth International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense-09)*, Toronto, Canada, June 2009.