A Mathematical Investigation of Reasoning About Actions

by

G. Neelakantan Kartha, B.E., M.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

May 1995

A Mathematical Investigation of Reasoning About Actions

Approved by
Dissertation Committee:

To my teachers

Acknowledgments

This dissertation has been significantly influenced by my doctoral advisor Vladimir Lifschitz. His rare ability to patiently listen to half-baked ideas and ask key questions has, time and again, helped me to better focus my research and move forward when stuck. He was a rich source of ideas and pointers to relevant literature, and was always accessible. He patiently read through several drafts of everything I wrote and always gave me suggestions that led to significant improvements in clarity and rigor. I am deeply grateful to him for all this.

I am thankful to the other members of my doctoral committee: Woodrow Bledsoe, Robert Boyer, Allen Emerson and Raymond Reiter. Each one of them has been very generous with his time and has taken a keen interest in my work.

I owe special thanks to Edsger Dijkstra and the Austin Tuesday Afternoon Club from whom I learned a lot about writing papers and presenting proofs.

During my stay at the University of Texas, the collegue from whom I learned the most via innumerable discussions and arguments is Norman Mccain. Thanks Norm! I have also benefited from discussions with Paolo Bouquet, Enrico Giunchiglia and Hudson Turner.

My office-mates Tim Collins, Rich Mallory, Jeff Rickel and Siddarth Subramanian were extraordinarily patient with my ever-growing desk and were always willing to share a joke. I have had discussions on almost all topics under the sun with Dinesh Das and Thomas Woo. My other friends in the computer science department including Guohui Feng, Bert Kay, Sharad Mehrotra, Kedar Namjoshi, Sowmya Ramachandran, Rajeev Rastogi, Robert Schrag and Joseph Sirosh have helped me in many different ways. I am thankful to all of them for sharing the joys and pains of graduate school.

My aunt and uncle in Pennsylvania, “Lachi” and Chittappan, were always just a phone call away during the course of my studies. They made their home my “home away from home”. More recently, my uncle “Kochaps” and his family in New York have played a similar role. I owe all of them a special debt of gratitude for all their support and love.

My father and mother have been a constant source of support and encouragement in my educational endeavours. I wish especially to thank my mother for always having faith in me and my father for his reminders that helped me finish up. Thanks to my sister, Renuka, for looking after my well-being from so far, and to my aunt, Radha Chitta, for enriching my life in many ways.

My wife, Rohini, has cheerfully put up with my long hours and prolonged periods of work. Her companionship and sense of humour has helped me to keep a more balanced perspective on life. For all that and more, I am thankful to her.

G. NEELAKANTAN KARTHA

The University of Texas at Austin
May 1995

A Mathematical Investigation of Reasoning About Actions

Publication No. ______________________

G. Neelakantan Kartha, Ph.D.
The University of Texas at Austin, 1995

Supervisor: Vladimir Lifschitz

Reasoning about actions is a central area of research in artificial intelligence, related to the study of commonsense and nonmonotonic reasoning, knowledge representation, planning and theorem proving. The methodology of research in this area has not been quite satisfactory; typically, a new proposal for reasoning about actions is illustrated by way of a few examples that have been known to be challenging to formalize, and then claims are made that the method works in general. This is not very satisfactory since minor modifications of the examples might prove (indeed, have proven, in many cases) to be difficult or impossible for the new proposal to handle correctly. Such an example-oriented methodology also makes it difficult to compare various formalisms and thus to synthesize new ones.

In this dissertation, we present a systematic study of reasoning about actions. The shortcomings of the example-oriented methodology are avoided as follows. First, declarative languages for describing actions are introduced and their semantics defined in such a way as to capture the underlying commonsense intuitions. Different methods of reasoning about actions are presented as translations from these languages and then the adequacy of the different formalizations is established by proving the soundness and completeness of the translations.

The new declarative languages we propose are capable of representing rich action domains, such as those where actions can have indirect effects. In addition, we introduce a new approach to reasoning about actions and show its adequacy in formalizing a large class of action domains. We show that, in conjunction with this approach, symbolic methods can often be employed for automating reasoning about actions. Finally, we point out some limitations of proposed methods of reasoning about actions and suggest ways to overcome these limitations.

Contents

Chapter 1

Introduction

1.1 Background

Reasoning about actions is a central topic in artificial intelligence. Its intimate connections to commonsense reasoning and planning were recognized quite early. It has also close connections with nonmonotonic reasoning; in fact, one of the main motivations behind the emergence of several nonmonotonic formalisms in 1980's was the *frame problem*, namely the problem of representing succinctly what does not change as a result of performing an action.

However, untill quite recently, the systematic study of reasoning about actions has mostly been ignored. Most of the research effort has focused on studying the mathematical properties of various nonmonotonic formalisms and in applying these formalisms to small, individual examples of reasoning about actions. While not denying that much insight was gained by the study of such examples, this work in applying the formalisms to reasoning about actions has not been entirely satisfactory with respect to the precision with which claims are made and defended. The pattern that seems to be followed by most of the work in the literature is as follows: an author informally describes an approach to reasoning about actions, illustrates it with a few examples, and then claims that the approach works in general. This methodology has the major defect that the exactness and confidence that the formal methods are supposed to provide are lacking to a large extent. Also, the important task of sharply delineating the realm of applicability of various methods is largely left unaddressed.

1.2 Research Approach and Significance

In this thesis, we study reasoning about actions in a systematic fashion. We carry out our study in a declarative framework; thus, we insist on formal languages with clear syntax and semantics. Our research approach is as follows: we first introduce declarative languages designed specifically for describing actions. The semantics of such languages will formalize what we mean by the "intuitively correct" effects of actions. Different approaches to reasoning about actions will be presented as translations from such languages. To investigate the adequacy of the approaches, we will then study the soundness and completeness of these translations.

This viewpoint has a number of advantages. First of all, once the soundness and completeness has been proved for a certain approach to reasoning about actions, we get the result that the approach works correctly for a class of action domains, namely, the class expressible in the language for representing actions. This is stronger than the results obtained by previous studies, where we only get that an approach works correctly for a few examples that have proved in the past to be troublesome to formalize. Note that another major difficulty that arises in using an example-oriented methodology is the following: even though such a methodology is able to guarantee that a proposed approach to reasoning about actions handles certain (possibly complex) examples, the approach might still fail on simpler ones. In such a case, it is not clear what has been accomplished by the proposed approach. The viewpoint presented in this thesis, by insisting that the proposed method work on a *class* of action domains, goes much further towards ensuring the general applicability of a method.

A second advantage of this viewpoint is that it forces us to examine different approaches to reasoning about actions in more depth than is usually done and to discover the strengths and limitations of each. In other words, trying to prove soundness and completeness is an excellent way of uncovering (heretofore unsuspected) shortcomings of an approach. Also, proving soundness and completeness from an underlying action language provides another dimension in which different approaches to reasoning about action may be compared.

A third advantage is that once such a soundness and completeness is established about a particular methodology, it may be possible to use procedural methods (which are usually faster than methods based on theorem proving) to actually compute the effects of reasoning about action within that methodology. Thus we can possibly combine the advantages of a declarative framework with computational speed of procedural representations.

The study of action languages themselves is quite interesting in its own right. The semantics of such languages can be thought of as a means for specifying unambiguously what counts as solutions to various problems (such as the frame problem) associated with reasoning about actions. This is important because in many cases, there are conflicting intuitions as to what counts as a correct solution. Clearly, only after specifying what counts as a correct solution can we embark on the task of verifying that a proposed approach gives the correct answers. The task of formalizing our intuitions about what counts as a correct solution turns out to be surprisingly challenging.

1.3 Overview of Contributions

This thesis studies reasoning about actions using the methodology described above. Several approaches to reasoning about action from the literature are viewed as translations from languages designed specifically for representing actions and the soundness and completeness of the translations is proved. New declarative languages for representing richer action domains (for instance, domains where actions might have indirect effects) are introduced and their properties studied. A new approach to reasoning about action is introduced and its adequacy in formalizing a large class of action domains is shown by viewing it as a translation from an action language and showing the soundness and completeness of the translation. Some limitations of proposed methods of reasoning about action are pointed out.

1.4 Organization

Chapter 2 explains what we mean by reasoning about actions and discusses problems that make this task difficult. It then briefly surveys the literature on nonmonotonic formalisms and how they are applied to reasoning about actions. This chapter also fixes the notation that will be used throughout the dissertation.

Chapter 3 presents translations based on several methodologies for reasoning about actions from an action language $\mathcal{A}$ and prove the soundness and completeness of the translations.

Chapter 4 introduces $\mathcal{AR}^-$, an action language that is more expressive than $\mathcal{A}$, and defines its semantics. It gives several examples of how various action domains can be expressed in this language and studies its properties.

Chapter 5 exhibits certain shortcomings that Baker's circumscriptive method has in dealing with nondeterministic actions and points out the reason for these shortcomings.

Chapter 6 gives a brief review of the formalism of nested abnormality theories and presents a sound and complete translation from $\mathcal{AR}^-$ into a nested abnormality theory based on Baker's ideas.

Chapter 7 introduces an extension $\mathcal{AR}$ of the language $\mathcal{AR}^-$ and studies its properties.

Chapter 8 presents a sound and complete translation from $\mathcal{AR}$ using a new approach to formalizing reasoning about actions. It also illustrates by way of examples the potential of the new approach in the automation of reasoning about actions.

Chapter 9 summarizes the main contributions of this thesis and suggests directions for future research.

The appendix contains certain mathematical results from the literature that we will use in proving various theorems.

Chapter 2

Motivation and Background

2.1 Introduction

This chapter introduces the topic of reasoning about actions, and surveys the research on this topic.

Section 2.2 explains what we mean by reasoning about actions and Section 2.3 presents a brief survey of the temporal formalisms used for this task. Section 2.4 examines various problems that arise in reasoning about actions. Sections 2.5 and 2.6 survey the literature on nonmonotonic formalisms and how they are applied in formalizing reasoning about actions. Section 2.7 fixes the mathematical notation used in this thesis.

2.2 Reasoning About Actions

An action is something that changes the state of the world. For instance, picking up a ball is an action. Throwing it is another one. By reasoning about actions, we mean reasoning about how the world changes as a result of executing such actions. For instance, we would like to conclude that after performing the action of picking up the ball, the ball is in the hand. It seems clear that any agent requires some ability to reason about actions, before it can be called truly intelligent.

There are various reasons why reasoning about actions in commonsense domains is difficult. One such reason is that it is often the case with commonsense domains that they are incompletely specified. As an example of this, note that only in very restricted domains is it possible to give an exhaustive list of *all* the preconditions that need to be satisfied before an action can be performed. Other difficulties stem from the necessity of dealing with nondeterministic actions, actions with indirect effects and from the computational need to specify the effects of actions in a concise manner.

We now turn to a brief description of temporal formalisms for studying reasoning about actions.

2.3 Temporal Formalisms for Reasoning About Actions

2.3.1 Situation Calculus

The temporal formalism used in our study of reasoning about actions is the *situation calculus* of McCarthy and Hayes [1969]. Intuitively, a situation can be thought of as a snapshot of the world at a point in time. The world changes from one situation to another when an action is performed. More formally, the result of performing an action a in a situation s is the new situation denoted by $Result(a, s)$. Situation dependent properties are called "fluents". Formally, a *fluent* is a function on the set of situations. A *propositional fluent* is a fluent whose codomain is the set $\{\mathit{True}, \mathit{False}\}$. Another way to view a propositional fluent is as a property which may hold in a situation. If f is a propositional fluent and s is a situation, we write $Holds(f, s)$ to mean that f is true in s.

Let us illustrate the use of situation calculus with an example. Consider the statement "If the battery is charged and there is gas in the tank, the car will start by turning the ignition-key". This can be formalized as

$$\begin{aligned} &Holds(\mathit{Charged}, s) \wedge Holds(\mathit{GasInTank}, s) \supset \\ &\qquad\qquad Holds(\mathit{EngineRunning}, Result(\mathit{TurnKey}, s)). \end{aligned} \tag{2.1}$$

Here *Charged*, *GasInTank* and *EngineRunning* are propositional fluents and *Turnkey* is an action.

The situation calculus is one of the oldest temporal formalisms to be used in artificial intelligence. Many criticisms have been raised against the situation calculus—for instance, that it cannot represent actions with time durations or those with delayed effects and that it is not suited for modeling scenarios involving concurrent actions. However, we base our study of reasoning about actions on the situation calculus for the following reasons:

- It is simple and well understood.
- Since most of the existing work on reasoning about action has been based on the situation calculus, and one of our goals is to study the relationship between existing formalizations, choosing situation calculus as our underlying temporal formalism facilitates comparisons.
- The difficulties that arise in reasoning about action occur even when a simple temporal formalism like the situation calculus is taken as the underlying temporal formalism. It is thus worthwhile to study these these difficulties in the context of situation calculus before moving to richer temporal formalisms.

- A closer study [Gelfond *et al.*, 1991b] reveals that many of the criticisms raised against the situation calculus are not justified—there the authors show how to formalize actions with time durations, actions with delayed effects, simple forms of concurrent actions etc. within the framework of the situation calculus. In addition, there has been a renewed interest in extending the situation calculus to make it more expressive. For instance, Reiter [1993] studies the use of induction in the situation calculus to prove properties of states. Pinto's thesis [Pinto, 1994] considers a number of ways of extending the situation calculus, including adding a time line to it. Miller and Shanahan [1994] consider how to represent narratives in the situation calculus. By basing our study on the situation calculus, it is reasonable to expect that our results can be extended to these more expressive variants.

2.3.2 Other Formalisms

Several other ontologies for representing actions have been proposed to address some of the criticisms raised against the situation calculus. Among the most prominent of these are the logics of McDermott [1982b] and Allen [1984] and the *event calculus* of Kowalski and Sergot [1986]. McDermott bases his logic on the notion of a partially ordered dense set of states, whereas Allen bases his logic on intervals. The event calculus of Kowalski and Sergot is similar to Allen's logic, but it is defined using the framework of logic programming. The event calculus has been criticized by Pinto and Reiter [1993] who point out that assigning a semantics to the event calculus is difficult. Kowalski and Sadri [1994] compare the situation calculus and the event calculus.

2.4 Problems in Reasoning About Actions

2.4.1 The Frame, Qualification and Ramification Problems

In this section, we briefly discuss three important problems that arise in reasoning about actions: the *frame problem* [McCarthy and Hayes, 1969], the *qualification problem* [McCarthy, 1977], and the *ramification problem* [Finger, 1986].

We will use the example introduced in Section 2.3.1 to illustrate the frame problem. Suppose that we are also given that the car was in the garage initially. We can formalize this fact in the situation calculus by the axiom

$$Holds(\mathit{InGarage}, S_0), \tag{2.2}$$

where S_0 represents the initial situation. Let us also assume the following facts about the initial situation:

$$Holds(\mathit{Charged}, S_0) \wedge Holds(\mathit{GasInTank}, S_0). \tag{2.3}$$

We would like to conclude that the car is still in the garage after turning the ignition key. That is, we would like to conclude

$$Holds(\mathit{InGarage}, Result(\mathit{TurnKey}, S_0)) \tag{2.4}$$

from the theory T consisting of the axioms (2.1), (2.2) and (2.3). But, it would not be possible to do that without another axiom, such as

$$Holds(\mathit{InGarage}, s) \supset Holds(\mathit{InGarage}, Result(\mathit{TurnKey}, s)),$$

which states that the location of the car does not change as a result of turning the ignition key. Such axioms, which state that a fluent does not change its value when an action is performed, are called *frame axioms.* Since the number of fluents affected by a particular action is relatively small compared to the total number of fluents, we see that a large number of such frame axioms are needed for describing any reasonable sized action domain. For instance, in the example above, if we had fluents for the colour of the car, the number of passengers etc., we would have to include frame axioms that state that these do not change when the action *Turnkey* is performed. Ideally, we would like to be able to state only the changes that occur as a result of performing an action, without having to mention what does not change.

The frame problem refers to the need for the large number of frame axioms in reasoning about the effects of actions. It is important to note that even though we have illustrated the frame problem within the framework of the situation calculus, the problem is more general and arises in every temporal logic.

The *qualification problem* arises because it is very difficult, if not impossible, to specify an exhaustive list of *all* the preconditions which need to be satisfied for performing an action. For instance, in (2.1) we considered only two preconditions which need to be satisfied before turning the key can make the car start. Some other preconditions are: no wires in the electrical system are disconnected, the key is in fact the ignition key, the key is not stuck ...., the list can be extended infinitely. Ordinarily, we want to jump to the conclusion that the action of turning the key will succeed provided that the two preconditions in (2.1) hold, and only think about other preconditions (called *qualifications*) if something unexpected happens. The problem of formalizing actions so as to achieve this desirable feature is called the qualification problem. Note that the qualification problem disappears if we assume that all the preconditions of an action are completely known. But this is rarely the case.

Another important problem in reasoning about actions is the *ramification problem.* This refers to the problem of formalizing actions that have indirect effects. As an example, let us assume that the exhaust fan is running whenever the car has started. We also assume that initially the fan is not running. We formalize these assumptions by introducing a propositional fluent *FanRunning* and introducing the following two axioms:

$$\neg Holds(\mathit{FanRunning}, S_0) \tag{2.5}$$

and

$$Holds(\mathit{EngineRunning}, s) \supset Holds(\mathit{FanRunning}, s). \tag{2.6}$$

Let T' be the theory consisting of (2.1), (2.2), (2.3), (2.5) and (2.6). We would like to be able to conclude

$$Holds(\mathit{FanRunning}, Result(\mathit{TurnKey}, S_0))$$

from T'. This seems like an easy task, yet some of the approaches suggested in the literature cannot handle such examples, without including an axiom that shows the effect of the action *TurnKey* on the fluent *FanRunning*.

In general, while specifying an action, it is infeasible to axiomatize the effect it has on each fluent directly. Instead, we would like to axiomatize only some of the effects and have the others follow from these effects and other axioms in the theory (that capture background knowledge). The ramification problem is the problem of formalizing actions in such a way that this goal is achieved.

2.4.2 Different Aspects of Reasoning About Actions

In this section, we will use examples to illustrate the different aspects of reasoning about action one might be interested in.

One of the simplest kind of questions one can ask in reasoning about actions is to compute the effect of performing a sequence of actions, starting from an initial state which is completely specified. This type of problems is called *temporal projection problems*. An example of this is the Yale shooting problem of Hanks and McDermott [1987].

Example 1. The following facts are known

1. Initially, a gun is loaded.
2. Initially, Fred is alive.
3. Firing a loaded gun kills Fred.

Conclusion: Fred is not alive after performing the sequence of actions *wait,shoot* in the initial situation.

Intuitively, the conclusion makes sense: since waiting can be assumed to have no effect (since nothing is mentioned in the problem statement about waiting), the gun is still loaded after performing the *wait* action, and Fred is not alive at the end of the sequence *wait,shoot*. The challenge here is to have a formalization for handling the frame problem which will give the intuitively expected result.

Another case of temporal projection is when the initial situation is not completely specified. The following example, based on a similar example due to Fangzhen Lin (reported in [Lifschitz, 1993c]), belongs to this class of problems.

Example 2. Same as Example 1, except that condition 1 is missing from the problem description. That is, it is not known whether or not the gun is loaded in the initial situation.

The challenge here is not to draw overspecific conclusions: for instance, it should not be possible to conclude that the gun is not loaded in the initial situation.

The following example, from Baker [1991], also belongs to the class of temporal projection problems.

Example 3. We augment Example 1 with the following additional facts:

1. If Fred is walking, he is alive
2. Intially, he is walking.

Conclusion: Fred is not walking after performing the sequence of actions *wait,shoot*.

Note that actions now have indirect effects— we want to obtain that Fred is not walking after the action shoot is performed as an indirect effect of the fact that shooting kills Fred. In particular, we do not want to add an axiom that specifies the effect of shooting on walking in order to obtain this conclusion.

The following two examples are not in the class of temporal projection problems. We may wish to use some facts about the current situation to derive conclusions about the past. The following example from [Baker, 1991] illustrates this.

Example 4. Condition 1 of Example 1 is replaced by

1. Fred is not alive after performing the sequence of actions *shoot,wait*.

Conclusion: The gun is loaded in the initial situation.

Note that there are methods that can solve temporal projection problems, but which give incorrect results with problems such as the above.

Another possibility is reasoning about alternate futures (counterfactual reasoning). This is illustrated by the following example from [Gelfond and Lifschitz, 1993].

Example 5. This is the same as Example 4, except that now we want to draw a different conclusion.

Conclusion: Fred is not alive after performing the sequence *wait,shoot*. (Note that the order of shoot and wait is reversed from Example 4.)

Such examples involving alternate futures are not commonly addressed in the literature. One can easily think of scenarios where what is known about the world is a mixture of information about future and past situations, and the task is to reason about the effect of actions based on this partial information. It is interesting to note that nontrivial problems in reasoning about action, such as the ones discussed above, arise even in extremely simple settings—for instance, when we assume that actions are deterministic and assume a discrete model of time. If these restrictions are relaxed, we run into more difficulties.

To see an example of the kind of problems which arise under nondeterministic actions, consider the following example due to Reiter.

Example 6. Suppose there is a table which is half white and half black. The action *Putdown* causes the block to touch only the white half, only the black half or both. This can be formalized as

$$Holds(\mathit{TouchW}, Result(\mathit{Putdown}, s)) \vee Holds(\mathit{TouchB}, Result(\mathit{Putdown}, s)).$$

Question : What do we know after the action *P*utdown is performed?

Intuitively, there are three possibilities. But many of the methods for reasoning about action based on minimization rule out the possibility in which both *TouchW* and *TouchB* are true. The challenge is to avoid such overspecific conclusions.

Researchers have proposed using nonmonotonic formalisms to solve some of the problems in reasoning about action. We now turn to a brief discussion of these formalisms.

2.5 Nonmonotonic Formalisms

A consequence relation $\vdash$ is is said to be *monotonic* if it follows from $T \vdash \gamma$ that $T' \vdash \gamma$ for a superset T' of T, where T and T' are theories and γ is a formula. A consequence relation $\vdash$ is is said to be *nonmonotonic* if it is not monotonic. The main nonmonotonic formalisms that have been proposed are circumscription [McCarthy, 1977; McCarthy, 1980] , default logic [Reiter, 1980], nonmonotonic logic [McDermott and Doyle, 1980] and autoepistemic logic [Moore, 1985]. Logic programming, which was proposed around 1974, can also be viewed as a nonmonotonic formalism, though the close connection of logic programming to other nonmonotonic formalisms became clear only in the latter half of the 1980's.

We now turn to a brief examination of each of these nonmonotonic formalisms. How these formalisms can be applied to reasoning about action will be discussed in Section 2.6.

2.5.1 Circumscription

In [McCarthy, 1980], McCarthy introduced the notion of predicate circumscription, which generalized domain circumscription introduced in his earlier paper [McCarthy, 1977]. Intuitively, circumscribing a predicate in a sentence means assuming that its extent is as small as possible. Model-theoretically, this means that we focus attention on those models where the extent of the circumscribed predicate is minimal. (The reader who is interested in a precise definition of circumscription and the minimal models is referred to the Appendix.)

In [Etherington *et al.*, 1985], Etherington, Mercer and Reiter pointed out the inability of predicate circumscription to derive new positive ground instances of any predicate. Formula circumscription, introduced in [McCarthy, 1986] does not suffer from this limitation. The main technical idea here was to vary other predicates and functions, in addition to the predicate being minimized. The SCAN algorithm [Gabbay and Ohlbach, 1992] and the SET-VAR method of Bledsoe and Feng [1993] can be used for computing circumscriptions.

A recent survey of circumscription can be found in [Lifschitz, 1993a].

2.5.2 Default Logic

Reiter's default logic [Reiter, 1980] augments a first order theory with a set of default rules. Informally, a default rule of the form $\alpha : \beta/\gamma$, where α, β and γ are formulae means the following: if α is true and it is consistent to assume β, then conclude γ is true. Reiter then defines the notion of *extensions*, which are (informally) meant to capture the theories sanctioned by these defaults. One of the omissions of Reiter's paper was a semantics for default logic. In [Etherington, 1987], Etherington attempts to remedy this omission. An alternative semantics, similar to Etherington's, was given by Guerreiro and Casanova [1990]. A close connection between default logic and autoepistemic logic was established in [Konolige, 1988]. A generalization of the notion of a default is that of a "disjunctive default", introduced in [Gelfond *et al.*, 1991a].

A recent survey of default logic can be found in [Poole, 1993].

2.5.3 Nonmonotonic Logic and Autoepistemic Logic

The nonmonotonic logic proposed by McDermott and Doyle in [1980] augmented first-order logic with a modal operator M, whose informal interpretation was "consistent". But the formulation in this paper proved to have have many technical problems: for instance Mp was not inconsistent with $\neg p$, where p is a proposition. In [McDermott, 1982a], McDermott proposed defining a non-monotonic logic based on a modal logic, instead of first order logic. But it was found that when the modal logic is S5, his nonmonotonic logic reduces to monotonic S5. Moore [1985] proposed autoepistemic logic as a way out of some of the difficulties of McDermott's logic and and in [Moore, 1984] gave a satisfactory semantics for it based on Kripke models. In [Shvarts, 1990], it was noticed that if we take the McDermott's nonmonotonic logic and base it on the modal logic Weak S5 instead of S5, we get precisely Moore's autoepistemic logic. A close connection between autoepistemic logic and default logic was established in [Konolige, 1988].

A recent survey of autoepistemic logic can be found in [Konolige, 1993].

2.5.4 Logic Programming

Nonmonotonicity in logic programs arises from the use of the negation as failure operator not. A close connection between default theories and logic programs has been observed [Bidoit and Froidevaux, 1988]. For instance, by identifying a logic programming rule

$$L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n,$$

(where L_i's are literals) with the default rule

$$\frac{L_1 \wedge \ldots \wedge L_m : \overline{L_{m+1}}, \ldots, \overline{L_n}}{L_0},$$

where $\overline{L}$ stands for the literal complementary to L, Gelfond and Lifschitz [1991] show that there is a one to one correspondence between the *answer sets* of a logic program and its extensions in the sense of default logic. In view of this connection, approaches to reasoning about action that employ logic programming can be classified under the approaches that use default logic.

2.6 Survey of the work on Reasoning About Action

In this section, we give a brief survey of different approaches to reasoning about action that have been proposed in the literature. This survey sets the stage for the research reported in this thesis. Work that was performed concurrently with the research reported here is discussed at the appropriate places in the thesis.

2.6.1 Approaches Based on Circumscription

Let us now examine how circumscription is applied to reasoning about action. To solve the frame problem, the key idea is to formalize the following default rule (known as the *commonsense law of inertia* (*LI*)): "Typically, the value of a fluent remains unchanged as a result of performing an action". The difficulty lies in formalizing the word *typically* in the above. Clearly, it is easy to formalize the statement that the value of a fluent remains unchanged as a result of performing an action: in situation calculus, we may write this as

$$Holds(f, s) \equiv Holds(f, Result(a, s)).$$

To formalize the commonsense law of inertia, following McCarthy's suggestion [McCarthy, 1986], a new predicate $Ab(f, a, s)$ is introduced. Intuitively, this means that the fluent f is abnormal with respect to action a in situation s. The *LI* is then formalized using axiom

$$\neg Ab(f, a, s) \supset Holds(f, s) \equiv Holds(f, Result(a, s)), \tag{2.7}$$

and $Ab(f, a, s)$ is circumscribed (with *Holds* allowed to vary).

Let us illustrate this idea on the example from Section 2.4.1. Let T_1 be the theory consisting of the axioms (2.1), (2.2), (2.3) and (2.7). Let T_2 be obtained from T_1 by circumscribing Ab while allowing *Holds* to vary (Formally, $T_2 = \mathbf{CIRC}(T_1; Ab; Holds)$). It is easy to see that (2.4) follows from T_2, as desired.

Though this idea (essentially due to McCarthy) is simple and attractive, it does not always yield the expected results, as illustrated by the the Yale Shooting Problem [Hanks and McDermott, 1987] (Example 1 from Section 2.4.2). The problem is that if we formalize this problem using the circumscriptive approach explained above, in addition to the intended model, there is another model where the extent of Ab is minimal. In the desired model, waiting preserves all the fluents and hence the action of shooting changes the fluent *Alive* from true to false. Using (2.7), we see that in this model, the fluent *Alive* is abnormal with respect to the action of shooting in the situation S_1 where $S_1 = Result(\mathit{Wait}, S_0)$. In the unintended model, the action of waiting unloads the gun and hence the fluent *Loaded* is abnormal with respect to the action of waiting in the situation S_0. However, shooting now has no effect, since the gun is unloaded and hence shooting preserves the values of all the fluents. That this model is minimal follows because this model can be thought of as being obtained from the desired model by "trading" the abnormality in the situation S_1 in the desired model for the abnormality in the situation S_0. Thus, the conclusion that Fred is dead after waiting and shooting does not follow from McCarthy's circumscriptive approach—the conclusions obtained are too weak in this case.

Many solutions were proposed in the literature to handle this problem. For instance, in *chronological minimization* proposed by Shoham [1987], models where abnormalities occur later in time are preferred over ones in which they occur earlier. Though the YSP is handled correctly by this method, it fails to handle Example 2 from Section 2.4.2—chronological minimization concludes in this case that the gun is not loaded in the initial situation and hence that Fred is alive after shooting (This was noted by Fangzhen Lin (reported in [Lifschitz, 1993c])). Note that in contrast to the previous example, the conclusions obtained by chronological minimization are too strong.

The method of chronological minimization and the logic of persistence proposed by Kautz[1986] both fail to handle Example 4 from Section 2.4.2 correctly—both conclude incorrectly that the gun was unloaded initially and that Fred died during the waiting [Baker, 1991].

Many methods proposed in the literature have trouble dealing with the indirect effects of actions. For instance, to get the intended conclusion in Example 3 from Section 2.4.2, the method of *causal minimization* [Lifschitz, 1987a] requires an axiom that specifies the effect of shooting on walking.

Baker [1991] proposed a circumscriptive approach that could handle many of the above problems correctly. Since we will have occasion later to discuss in detail the advantages and disadvantages of Baker's method, we will postpone discussion of this method.

Using Baker's ideas, Lifschitz [1991] proves a theorem which is applicable to a whole class of domains involving situations and actions. As noted in [Lifschitz, 1991], the theorem suffers from some limitations. For instance, the assumptions of the theorem make it impossible to have initial conditions in the axiom set. Also, the effect of each action is assumed to make fluents true, never false. Another drawback is that some of the assumptions of the theorem are non-syntactic in nature.

Lin and Shoham [1991] propose evaluating a nonmonotonic theory relative to an equivalent (and usually larger) monotonic theory. They introduce the notion of *epistemological completeness*, which is meant to capture the notion that a theory is a complete formalization of an action. Then they introduce the notion of a *causal theory* and prove a theorem which specifies four conditions under which a causal theory is epistemologically complete. The notion of causal theories they introduce is complicated and assumes that the effect of each action is always to make a fluent true, never false. Initial conditions cannot be formulated in their theory and nonsyntactic checks have to be made for applying the theorem referred to above. They also attempt to capture the causal theories using circumscription, but they use an unusual version of circumscription, which does not seem to be related to any other applications of circumscription to knowledge representation.

2.6.2 Approaches Based on Default Logic and Logic Programming

Hanks and McDermott [1987] showed that if one formalized the frame default principle by the combination of the commonsense law of inertia (2.7) and the normal default[1]

$$\frac{: \neg Ab(f,a,s)}{Ab(f,a,s)}$$

(that stands for minimizing Ab), an unsatisfactory formalization of the YSP results. In particular, there is an extension of the default theory in which the action of waiting unloads the gun.

Morris [1988] showed that formalizing the frame default with the non-normal default

$$\frac{: \neg Ab(f,a,s)}{Holds(f,s) \equiv Holds(f,Result(a,s))} \tag{2.8}$$

solves the Yale Shooting Problem.

Evans [1989] and Apt and Bezem [1990] represent properties of actions in logic programming languages with negation as failure. Gelfond and Lifschitz [1993] point out the

[1] A default of the form $\alpha : w/w$ is called a *normal default*.

advantages of representing actions in extended logic programs (logic programs that can represent both negation as failure and classical negation). Turner [1994] represents actions in extended disjunctive logic programs.

2.6.3 Approaches Based on Autoepistemic Logic

Gelfond [1989] shows how autoepistemic logic can be used to formalize reasoning about actions.

2.6.4 Other Approaches

One of the first programs to reason about action is described in Cordell Green's work [Green, 1969]. This program used the situation calculus and resolution to do planning. The STRIPS language of Fikes and Nilsson [1971] was another early attempt to formalize descriptions of actions. In STRIPS, actions were represented by three lists of formulae: the PRECONDITION, ADD and DELETE lists, which indicated when an action can be performed and how to change the state description as a result of performing an action. Though computationally efficient, the STRIPS system suffers from the disadvantage that seemingly minor modifications in standard examples can cause STRIPS to produce incorrect results [Siklóssy and Roach, 1975]. Also, STRIPS failed to handle the ramification problem since the indirect effects of actions had to be directly encoded in the ADD and DELETE lists. A formal account of the STRIPS language was given by Lifschitz [1987b]. The ADL formalism of Pednault [1989] is an attempt to integrate much of the expressive power of the situation calculus with the computational benefits of the STRIPS language.

Ginsberg and Smith [1988a] proposed a *possible worlds approach* to reasoning about actions. A *world* is identified with a set of formulae that satisfy certain restrictions (for instance, that they satisfy the domain constraints). The result of performing an action in a world leads to a world hat satisfies the effects of the action and that is "nearest" to the starting world. However, as pointed out by [Winslett, 1988], Ginsberg and Smith's approach suffers from a number of problems that stem from syntactic character of the definition of a world—for instance $\{p, q\}$ is a different world from $\{p \wedge q\}$, according to Ginsberg and Smith. Because of this, they get unintuitive conclusions when description of the starting world is incomplete. Winslett [1988] suggests modifying the Ginsberg and Smith approach to use possible models. However, even this approach has recently been criticized [Brewka and Hertzberg, 1993].

Sandewall's *filter preferential entailment* [Sandewall, 1989] modifies chronological minimization by changing the definition of preferred models. Also, of his preferred models, he considers only those which satisfy the *observation axioms*, i.e. those axioms specifying which fluents hold at specific times. Thus, the set of models he considers is obtained by 'filtering' the preferred models with observations. His approach is presented in the context of a richer temporal formalism in which continuous change can be represented. Sandewall shows that this formalism is able to handle many examples on which chronological minimization fails to produce intuitively correct results.

Recently, some (closely related) first order methods of reasoning about action [Reiter, 1991; Pednault, 1989; Schubert, 1990; Elkan, 1992; Davis, 1991] have appeared in the literature. Since we will have occasion to discuss these methods in detail later, we will postpone the discussion of these.

2.7 Notation

In formulae, constants are denoted by any sequence of characters starting with an uppercase letter (e.g. *InGarage*) and variables are denoted by lowercase letters (eg. $x, y, x, \ldots$). In a formulae, all free variables are assumed to be universally quantified from the outside. The binding power of logical operators is as follows (from highest to lowest): $\neg$, $\wedge$, $\vee$, $\supset$, $\equiv$.

The symbol $\subseteq$ denotes subset (not necessarily proper) and $\subset$ denotes proper subset. The expression 'iff' stands for 'if and only if'. The end of a proof will be denoted by $\Box$.

In some cases, proofs are presented in the calculational style of [Dijkstra and Scholten, 1990] and [Gries and Schneider, 1993].

Chapter 3

Translations from $\mathcal{A}$

3.1 Introduction

In this chapter, we consider three translations from $\mathcal{A}$, a simple declarative language for representing actions that was introduced in [Gelfond and Lifschitz, 1993] and prove the soundness and completeness of the translations.

We first review the syntax and semantics of $\mathcal{A}$ in Section 3.2. Sections 3.3, 3.4 and 3.5 present reviews of the approaches of Pednault [1989], Reiter [1991] and Baker [1991] and give sound and complete translations from $\mathcal{A}$ based on these approaches.

3.2 The Language $\mathcal{A}$

In this section, we describe the language $\mathcal{A}$ introduced in [Gelfond and Lifschitz, 1993]. The reader who is interested in more examples and motivation is referred to that paper.

Consider two disjoint nonempty sets of symbols, called *fluent names* and *action names*. A *fluent expression* is a fluent name possibly preceded by $\neg$. A *value proposition* specifies the value of a fluent after performing zero or more actions from the initial situation. It is of the form

$$F \textbf{ after } A_1; \ldots; A_m \tag{3.1}$$

where F is a fluent expression, and A_1,. . .,A_m ($m \geq 0$) are action names. If $m = 0$, we will write (3.1) as

$$\textbf{initially } F.$$

An *effect proposition* describes the effect of an action on a fluent and is of the form

$$A \textbf{ causes } F \textbf{ if } P_1, \ldots, P_n, \tag{3.2}$$

where A is an action name, and each of $F, P_1, \ldots, P_n$ ($n \geq 0$) is a fluent expression. Let us denote by $|F|$ the fluent name occurring in the fluent expression F. If $|F|$ occurs unnegated (negated) in (3.2), we call it a *positive (negative) effect proposition* and say that A *affects* $|F|$ *positively (negatively).* We also say that this effect proposition *describes the effect of* A *on* F, and that $P_1, \ldots, P_n$ are its *preconditions.* If $n = 0$, we will drop **if** and write simply

$$A \textbf{ causes } F.$$

A *proposition* is a value proposition or an effect proposition. A *domain description*, or simply *domain*, is a set of propositions (not necessarily finite).

Example 1. The Blocks World domain D_1, motivated by problem $D4$ from [Lifschitz, 1989], is defined as follows. The fluent names are *Ontable* and *Inhand*; the action names are *Putdown*, *Pickup*, *Lower* and *Wait.* The propositions of D_1 are:

initially $\neg$*Ontable*,
Ontable **after** *Wait;Putdown*,
Putdown **causes** *Ontable* **if** *Inhand*,
Putdown **causes** $\neg$*Inhand* **if** *Inhand*,
Pickup **causes** *Inhand* **if** *Ontable*,
Pickup **causes** $\neg$*Ontable* **if** *Ontable*,
Lower **causes** *Ontable* **if** *Inhand.*

The following definitions will be useful in describing the semantics of $\mathcal{A}$.

A *state* is a set of fluent names. Given a fluent name F and a state σ, we say that F *holds* in σ if $F \in \sigma$; $\neg F$ *holds* in σ if $F \notin \sigma$. A *transition function* is a mapping Φ of the set of pairs (A, σ), where A is an action name and σ is a state, into the set of states. A *structure* is a pair (σ_0, Φ), where σ_0 is a state (the *initial state* of the structure), and Φ is a transition function.

For any structure M and any action names $A_1, \ldots, A_m$, by $M^{A_1;\ldots;A_m}$ we denote the state

$$\Phi(A_m, \Phi(A_{m-1}, \ldots, \Phi(A_1, \sigma_0) \ldots)),$$

where Φ and σ_0 are the transition function and the initial state of M. We say that a value proposition (3.1) is *true* in a structure M if F holds in the state $M^{A_1;\ldots;A_m}$, and that it is *false* otherwise.

A structure (σ_0, Φ) is a *model* of a domain description D if every value proposition from D is true in (σ_0, Φ) and for every action name A, every fluent name F, and every state σ, the following two conditions are satisfied:

(i) if D includes an effect proposition describing the effect of A on F ($\neg F$) whose preconditions hold in σ, then F ($\neg F$) holds in the state $\Phi(A, \sigma)$.

(ii) otherwise, F holds in the state $\Phi(A, \sigma)$ iff F holds in the state σ.

Note that there can be at most one transition function Φ, satisfying conditions (i) and (ii). Hence different models of the same domain differ only by their initial states.

A value proposition is *entailed* by a domain description D if it is true in every model of D.

A domain description is *consistent* if it has a model, and *complete* if it has exactly one model.

Example 1 (continued). The domain description D_1 is complete. Its only model is defined by the equations

$$\sigma_0 = \{\mathit{Inhand}\},$$

$$\Phi(\mathit{Putdown}, \sigma) = \begin{cases} \sigma \cup \{\mathit{Ontable}\} \setminus \{\mathit{Inhand}\}, & \text{if } \mathit{Inhand} \in \sigma, \\ \sigma, & \text{otherwise,} \end{cases}$$

$$\Phi(\mathit{Pickup}, \sigma) = \begin{cases} \sigma \cup \{\mathit{Inhand}\} \setminus \{\mathit{Ontable}\}, & \text{if } \mathit{Ontable} \in \sigma, \\ \sigma, & \text{otherwise,} \end{cases}$$

$$\Phi(\mathit{Lower}, \sigma) = \begin{cases} \sigma \cup \{\mathit{Ontable}\}, & \text{if } \mathit{Inhand} \in \sigma, \\ \sigma, & \text{otherwise,} \end{cases}$$

$$\Phi(\mathit{Wait}, \sigma) = \sigma.$$

It is easy to see, for instance, that

initially *Inhand*,
Ontable **after** *Lower;Wait*,
¬*Inhand* **after** *Lower;Wait;Putdown*,

is entailed by D_1.

3.3 Translation into Pednault's Scheme

3.3.1 Review of Pednault's Scheme

In this section, we review Pednault's scheme [Pednault, 1989] for generating the frame axioms. We will conform for the most part to the notation introduced in [Reiter, 1991].

Pednault assumes that corresponding to each fixed fluent F and each fixed action A

there are two effect axioms, a positive effect axiom of the form[1]:

$$\pi_A(s) \wedge \varepsilon_F^+(s) \supset Holds(F, Result(A, s)), \tag{3.3}$$

and a negative effect axiom of the form

$$\pi_A(s) \wedge \varepsilon_F^-(s) \supset \neg Holds(F, Result(A, s)). \tag{3.4}$$

Here, $\pi_A(s)$ denotes the *action preconditions* of the action A. These are the prerequisites that must be satisfied for the execution of action A. They depend only on A. The formula $\varepsilon_F^+(s)$ $(\varepsilon_F^-(s))$ denotes the *positive (negative) fluent preconditions* under which action A, if performed, will lead to F becoming true (false) in the resulting situation $Result(A, s)$.

Pednault observes that he can obtain the frame axioms from the effect axioms systematically if a completeness assumption for the fluent preconditions is made. The assumption is that the positive (negative) fluent preconditions specify *all* the conditions under which an action if performed will lead to the truth (falsity) of the fluent in the resulting situation. Using this assumption, we can say that if $\pi_A(s)$ holds in situation s, then the only way that F can change from false in situation s to true in situation $Result(A, s)$ is if $\varepsilon_F^+(s)$ were true. This can be formalized as

$$\pi_A(s) \wedge \neg Holds(F, s) \wedge Holds(F, Result(A, s)) \supset \varepsilon_F^+(s).$$

Rewriting the above as

$$\pi_A(s) \wedge \neg Holds(F, s) \wedge \neg\varepsilon_F^+(s) \supset \neg Holds(F, Result(A, s)),$$

we see that this is a frame axiom. Similarly, we can generate the frame axiom

$$\pi_A(s) \wedge Holds(F, s) \wedge \neg\varepsilon_F^-(s) \supset Holds(F, Result(A, s))$$

from (3.4).

3.3.2 The Translation $P(D)$

Let D be a finite domain. The following notation will be used: If $A_1, \ldots, A_m$ are action names, $[A_1; \ldots; A_m]$ stands for the ground term

$$Result(A_m, Result(A_{m-1}, \ldots, Result(A_1, S_0), \ldots)).$$

If t is a term,

$$Holds(\neg F, t) \text{ stands for } \neg Holds(F, t).$$

[1] The form of the effect axioms in our review is more special than the ones actually used by Pednault: we deal with individual actions and fluents rather than families parameterized by x and y. We use such a simplification in our review of Reiter's proposal also.

The translation of a value proposition α of the form (3.1), denoted $P(\alpha)$, is

$$Holds(F, [A_1; \ldots; A_m]). \tag{3.5}$$

Since corresponding to an F, A pair (where F is a fluent name and A is an action), Pednault has just two effect axioms, effect propositions are not translated individually. Rather the set of all effect propositions describing the effect of A on F or $\neg F$, $S_{F,A}$, is partitioned into two subsets $S^+_{F,A}$ and $S^-_{F,A}$. $S^+_{F,A}$ ($S^-_{F,A}$) consists of all the effect propositions describing the effect of A on F ($\neg F$). The effect propositions in each of these two sets are then translated as a group. To express this formally, define for any effect proposition E of the form (3.2),

$$pre_E(s) = \bigwedge_{i=1}^{n} Holds(P_i, s).$$

Also define,

$$precond^+_{F,A}(s) = \bigvee_{E \in S^+_{F,A}} pre_E(s),$$

$$precond^-_{F,A}(s) = \bigvee_{E \in S^-_{F,A}} pre_E(s),$$

$$precond_{F,A}(s) = precond^+_{F,A}(s) \vee precond^-_{F,A}(s).$$

Now, we can translate the collection of effect propositions describing the effect of A on F or $\neg F$ as

$$precond^+_{F,A}(s) \supset Holds(F, Result(A, s)), \tag{3.6}$$

$$precond^-_{F,A}(s) \supset \neg Holds(F, Result(A, s)). \tag{3.7}$$

Comparing with the notation used in the previous section, we see that $\pi_A(s)$ can be identified with $true$, $\varepsilon^+_F(s)$ can be identified with $precond^+_{F,A}(s)$ and $\varepsilon^-_F(s)$ can be identified with $precond^-_{F,A}(s)$.

Let **F** and **A** be, respectively, the set of all the fluent names and action names mentioned in the domain description D. Then we will also add the following axioms:

1. The domain closure axiom (DCA) for fluents:

$$\bigvee_{F \in \mathbf{F}} f = F.$$

2. The domain closure axiom (DCA) for actions:

$$\bigvee_{A \in \mathbf{A}} a = A.$$

3. The unique name assumption (UNA) axiom for fluents:

$$\bigwedge_{\substack{F_1, F_2 \in \mathbf{F} \\ F_1, F_2 \text{ distinct}}} F_1 \neq F_2.$$

4. The unique name assumption (UNA) axiom for actions:

$$\bigwedge_{\substack{A_1, A_2 \in \mathbf{A} \\ A_1, A_2 \text{ distinct}}} A_1 \neq A_2.$$

By $P(D)$, we denote the collection of axioms which consist of these four axioms, the axioms of the form (3.5), (3.6) and (3.7) and the frame axioms obtained from the effect axioms by the process outlined in the previous section. That is, we add for each fluent-action pair F, A, the two frame axioms

$$\neg Holds(F, s) \wedge \neg precond^{+}_{F,A}(s) \supset \neg Holds(F, Result(A, s)), \tag{3.8}$$

$$Holds(F, s) \wedge \neg precond^{-}_{F,A}(s) \supset Holds(F, Result(A, s)). \tag{3.9}$$

Example 1 (continued). $P(D_1)$ consists of the following axioms:

1. $\neg Holds(\mathit{Ontable}, S_0)$.
2. $Holds(\mathit{Ontable}, Result(\mathit{Putdown}, Result(\mathit{Wait}, S_0)))$.
3. $Holds(\mathit{Inhand}, s) \supset Holds(\mathit{Ontable}, Result(\mathit{Putdown}, s))$.
4. $Holds(\mathit{Inhand}, s) \supset \neg Holds(\mathit{Inhand}, Result(\mathit{Putdown}, s))$.
5. $Holds(\mathit{Ontable}, s) \supset Holds(\mathit{Inhand}, Result(\mathit{Pickup}, s))$.
6. $Holds(\mathit{Ontable}, s) \supset \neg Holds(\mathit{Ontable}, Result(\mathit{Pickup}, s))$.
7. $Holds(\mathit{Inhand}, s) \supset Holds(\mathit{Ontable}, Result(\mathit{Lower}, s))$.
8. $f = \mathit{Ontable} \vee f = \mathit{Inhand}$.
9. $a = \mathit{Putdown} \vee a = \mathit{Pickup} \vee a = \mathit{Lower} \vee a = \mathit{Wait}$.
10. $\mathit{Ontable} \neq \mathit{Inhand}$.
11. $\mathit{Putdown} \neq \mathit{Pickup} \wedge \mathit{Putdown} \neq \mathit{Lower} \wedge \mathit{Putdown} \neq \mathit{Wait} \wedge$
    $\mathit{Pickup} \neq \mathit{Lower} \wedge \mathit{Pickup} \neq \mathit{Wait} \wedge \mathit{Lower} \neq \mathit{Wait}$.
12. $\neg Holds(\mathit{Inhand}, s) \supset \neg Holds(\mathit{Inhand}, Result(\mathit{Putdown}, s))$.

13. $\neg Holds(\textit{Inhand}, s) \wedge \neg Holds(\textit{Ontable}, s) \supset$
$\neg Holds(\textit{Inhand}, Result(\textit{Pickup}, s))$.

14. $Holds(\textit{Inhand}, s) \supset Holds(\textit{Inhand}, Result(\textit{Pickup}, s))$.

15. $\neg Holds(\textit{Inhand}, s) \supset \neg Holds(\textit{Inhand}, Result(\textit{Lower}, s))$.

16. $Holds(\textit{Inhand}, s) \supset Holds(\textit{Inhand}, Result(\textit{Lower}, s))$.

17. $\neg Holds(\textit{Inhand}, s) \supset \neg Holds(\textit{Inhand}, Result(\textit{Wait}, s))$.

18. $Holds(\textit{Inhand}, s) \supset Holds(\textit{Inhand}, Result(\textit{Wait}, s))$.

19. $\neg Holds(\textit{Ontable}, s) \wedge \neg Holds(\textit{Inhand}, s) \supset$
$\neg Holds(\textit{Ontable}, Result(\textit{Putdown}, s))$.

20. $Holds(\textit{Ontable}, s) \supset Holds(\textit{Ontable}, Result(\textit{Putdown}, s))$.

21. $\neg Holds(\textit{Ontable}, s) \supset \neg Holds(\textit{Ontable}, Result(\textit{Pickup}, s))$.

22. $\neg Holds(\textit{Ontable}, s) \wedge \neg Holds(\textit{Inhand}, s) \supset$
$\neg Holds(\textit{Ontable}, Result(\textit{Lower}, s))$.

23. $Holds(\textit{Ontable}, s) \supset Holds(\textit{Ontable}, Result(\textit{Lower}, s))$.

24. $\neg Holds(\textit{Ontable}, s) \supset \neg Holds(\textit{Ontable}, Result(\textit{Wait}, s))$.

25. $Holds(\textit{Ontable}, s) \supset Holds(\textit{Ontable}, Result(\textit{Wait}, s))$.

3.3.3 Soundness and Completeness

We can relate the notion of entailment in the sense of first order logic and entailment relation for the language $\mathcal{A}$ using the following theorem.

Theorem 3.1 *Let D be any finite, consistent domain, and let $P(D)$ be defined as before. For any value proposition α,*

$$P(D) \models P(\alpha) \iff D \textit{ entails } \alpha.$$

Theorem 3.1 expresses the soundness and completeness of the translation P. Note that it gives a very strong characterization of what can be proved from $P(D)$. For instance, using this theorem, we can conclude in the case of Example 1 that $P(D) \models Holds(\textit{Inhand}, S_0)$. Note that this corresponds to reasoning about the past. Similarly, we can conclude that $P(D) \models \neg Holds(\textit{Inhand}, [\textit{Lower}; \textit{Wait}; \textit{Putdown}])$. This corresponds to reasoning about the future, when the initial situation is incompletely specified. Domains in which each value proposition has the form **initially** F, are just a simple special case of this theorem. Such domains correspond to temporal projection problems, which have been the focus of much attention in the literature.

In the next two sections, we shall propose similar translations from D into other methods for reasoning about action. In each case, we will prove theorems of exactly the same form as Theorem 3.1, the only change being that instead of the translation P, we will have other translation schemes. Thus all the remarks of the previous paragraph apply to these translation schemes as well.

3.4 Translation into Reiter's Scheme

3.4.1 Review of Reiter's Scheme

Like Pednault, Reiter [1991] has two effect axioms for each fluent F. But in these effect axioms, the actions are universally quantified (following the proposal by Schubert [1990]). So he calls them *general* effect axioms. They are of the form

$$Poss(a, s) \wedge \gamma_F^+(a, s) \supset Holds(F, Result(a, s)), \tag{3.10}$$

$$Poss(a, s) \wedge \gamma_F^-(a, s) \supset \neg Holds(F, Result(a, s)). \tag{3.11}$$

The predicate $Poss(a, s)$ intuitively means that a is possible in situation s. $Poss$ is defined by action precondition axioms of the form

$$\pi_A(s) \supset Poss(A, s)$$

for each action A, where $\pi_A(s)$ stands for the action A's preconditions.

Reiter also makes a completeness assumption, according to which (3.10) and (3.11) account for *all* the conditions under which action a can affect the value of F. As before, this completeness assumption leads to what Schubert and Reiter calls *explanation closure axioms*. They are equivalent to frame axioms of the following form:

$$Poss(a, s) \wedge \neg Holds(F, s) \wedge \neg\gamma_F^+(a, s) \supset \neg Holds(F, Result(a, s)),$$

$$Poss(a, s) \wedge Holds(F, s) \wedge \neg\gamma_F^-(a, s) \supset Holds(F, Result(a, s)).$$

Comparing these frame axioms with the ones obtained by Pednault, we see that the reification of actions allows Reiter to express the frame axioms in a more compact way.

3.4.2 The Translation $R(D)$

The translation of the domain D into Reiter's scheme, $R(D)$, is similar to the translation $P(D)$ into Pednault's scheme: the translation of value propositions is exactly the same and the UNA and DCA for fluents and actions are included in the translation, as before. The only differences are in the translation of the effect propositions and in the form of the frame axioms. These differences arise due to the fact that here, actions in the general effect statements are universally quantified.

With a fixed fluent name F, we associate two sets: the set A_F^+ of action names which affect it positively and the set A_F^- of those which affect it negatively. Note that it is possible for a particular action name to be in both these sets or to be in neither of them.

Now we can give the translation of effect propositions into Reiter's scheme as follows:

$$\bigvee_{A \in A_F^+} (a = A \wedge precond_{F,A}^+(s)) \supset Holds(F, Result(a,s)), \tag{3.12}$$

$$\bigvee_{A \in A_F^-} (a = A \wedge precond_{F,A}^-(s)) \supset \neg Holds(F, Result(a,s)). \tag{3.13}$$

In comparison to the notation used in Section 3.4.1, $Poss(a,s)$ corresponds to $true$, $\gamma_F^+(a,s)$ to

$$\bigvee_{A \in A_F^+} (a = A \wedge precond_{F,A}^+(s)),$$

and $\gamma_F^-(a,s)$ to

$$\bigvee_{A \in A_F^-} (a = A \wedge precond_{F,A}^-(s)).$$

For each fluent F, we can generate the following two frame axioms as described in the last section:

$$\neg Holds(F,s) \wedge \neg \bigvee_{A \in A_F^+} (a = A \wedge precond_{F,A}^+(s)) \supset \neg Holds(F, Result(a,s)), \tag{3.14}$$

$$Holds(F,s) \wedge \neg \bigvee_{A \in A_F^-} (a = A \wedge precond_{F,A}^-(s)) \supset Holds(F, Result(a,s)). \tag{3.15}$$

$R(D)$ consists of the translation of the value propositions and effect propositions of the domain, that is, all sentences of the form (3.5), (3.12) and (3.13), the UNA and DCA for fluents and actions, and all the frame axioms of the form (3.14) and (3.15).

Example 1 (continued). $R(D_1)$ contains the axioms 1-2 and 8-11 of $P(D_1)$. The effect axioms 3-7 of $P(D_1)$ are replaced here by the general effect axioms (i)-(iv), and the frame axioms 12-25 are replaced by the general frame axioms (v)-(viii).

(i) $(a = \textit{Lower} \wedge Holds(\textit{Inhand}, s)) \vee (a = \textit{Putdown} \wedge Holds(\textit{Inhand}, s)) \supset$
$Holds(\textit{Ontable}, Result(a, s))$.

(ii) $(a = \textit{Pickup} \wedge Holds(\textit{Ontable}, s)) \supset \neg Holds(\textit{Ontable}, Result(a, s))$.

(iii) $(a = \textit{Pickup} \wedge Holds(\textit{Ontable}, s)) \supset Holds(\textit{Inhand}, Result(a, s))$.

(iv) $(a = \textit{Putdown} \wedge Holds(\textit{Inhand}, s)) \supset \neg Holds(\textit{Inhand}, Result(a, s))$.

(v) $\neg Holds(\textit{Inhand}, s) \wedge \neg(a = \textit{Pickup} \wedge Holds(\textit{Ontable}, s)) \supset$
$\neg Holds(\textit{Inhand}, Result(a, s))$.

(vi) $Holds(\textit{Inhand}, s) \wedge \neg(a = \textit{Putdown} \wedge Holds(\textit{Inhand}, s)) \supset$
$Holds(\textit{Inhand}, Result(a, s))$.

(vii) $\neg Holds(\textit{Ontable}, s) \wedge \neg[(a = \textit{Lower} \wedge Holds(\textit{Inhand}, s)) \vee$
$(a = \textit{Putdown} \wedge Holds(\textit{Inhand}, s))] \supset \neg Holds(\textit{Ontable}, Result(a, s))$.

(viii) $Holds(\textit{Ontable}, s) \wedge \neg(a = \textit{Pickup} \wedge Holds(\textit{Ontable}, s)) \supset$
$Holds(\textit{Ontable}, Result(a, s))$.

We can now prove a theorem about $R(D)$ exactly similar to Theorem 3.1.

Theorem 3.2 *Let D be any finite, consistent domain and let $R(D)$ be defined as before. For any value proposition α,*

$$R(D) \models R(\alpha) \iff D \textit{ entails } \alpha.$$

3.5 Translation into Baker's Approach

3.5.1 Review of Baker's Approach

Baker's approach to reasoning about action [Baker, 1991] makes it possible to write axioms for the direct effects of actions and the domain constraints in the situation calculus and obtain the effect of appropriate frame axioms by circumscription. We will first illustrate Baker's approach with an example and point out its salient features.

Suppose we wish to formalize the actions of a commuter who has to take a bus to work. We assume that the commuter has to buy a ticket to board the bus. We also assume that initially, he does not have a ticket and that he is not on the bus.

To describe the given scenario in the situation calculus, we introduce a situation constant S_0 to stand for the initial situation, two actions *BuyTicket* and *GetOnBoard*, and two fluents *HasTicket* and *OnBoard*. Then the following axioms capture what holds in the initial situation and what the effects of actions are:

$$\begin{aligned}
&\neg Holds(\mathit{HasTicket}, S_0),\\
&\neg Holds(\mathit{OnBoard}, S_0),\\
&\neg Holds(\mathit{HasTicket}, s) \supset Holds(\mathit{HasTicket}, Result(\mathit{BuyTicket}, s)),\\
&Holds(\mathit{HasTicket}, s) \wedge \neg Holds(\mathit{OnBoard}, s) \supset\\
&\qquad\qquad\qquad Holds(\mathit{OnBoard}, Result(\mathit{GetOnBoard}, s)).
\end{aligned} \tag{3.16}$$

We also include the unique names axioms (UNA) and the domain closure axioms (DCA) for fluents and actions. For our example, these are

$$\begin{aligned}
&\mathit{HasTicket} \neq \mathit{OnBoard},\\
&f = \mathit{HasTicket} \vee f = \mathit{OnBoard},\\
&\mathit{BuyTicket} \neq \mathit{GetOnBoard},\\
&a = \mathit{BuyTicket} \vee a = \mathit{GetOnBoard}.
\end{aligned} \tag{3.17}$$

As is common with circumscriptive approaches to reasoning about action, we include a "commonsense law of inertia":

$$\neg Ab(f, a, s) \supset [Holds(f, s) \equiv Holds(f, Result(a, s))]. \tag{3.18}$$

This axiom states that fluents normally persist when actions are performed. Here, $Ab(f, a, s)$ is an "abnormality" predicate which stands for the fluent f being abnormal with respect to action a in situation s. We want a fluent to be abnormal with respect to an action in a situation only when it is forced to change by the effect axioms for that action. To achieve this, Ab is circumscribed.

The axioms we have described up to this point are common to most circumscriptive approaches to reasoning about action. The major change suggested by Baker is to employ a new circumscription policy: to minimize Ab while varying $Result$ and S_0 (as opposed to varying $Holds$, as is done in [Hanks and McDermott, 1987]). For this to work correctly, Baker includes an existence of situations axiom, which says that corresponding to each set of fluents, there is a situation in which exactly these fluents hold. Thus for our example, we include

$$\begin{aligned}
&\exists s[Holds(\mathit{HasTicket}, s) \wedge Holds(\mathit{OnBoard}, s)] \wedge\\
&\exists s[Holds(\mathit{HasTicket}, s) \wedge \neg Holds(\mathit{OnBoard}, s)] \wedge\\
&\exists s[\neg Holds(\mathit{HasTicket}, s) \wedge Holds(\mathit{OnBoard}, s)] \wedge\\
&\exists s[\neg Holds(\mathit{HasTicket}, s) \wedge \neg Holds(\mathit{OnBoard}, s)].
\end{aligned} \tag{3.19}$$

Recall that we denote the circumscription of the predicate P in the formula A, with the predicates and functions in the tuple Z varied by $\mathbf{CIRC}(A; P; Z)$. Thus, for our example above, we consider $\mathbf{CIRC}(W; Ab; Result, S_0)$ when using Baker's approach, where W is the conjunction of the universal closures of the axioms (3.16)–(3.19).

It can be shown, for instance, that

$$\textbf{CIRC}(W; Ab; Result, S_0) \models \\ Holds(\textit{OnBoard}, Result(\textit{GetOnBoard}, Result(\textit{BuyTicket}, S_0))).$$

If we denote by W' the set of axioms obtained from W by replacing the axiom $\neg Holds(\textit{HasTicket}, S_0)$ of W by $Holds(\textit{OnBoard}, Result(\textit{GetOnBoard}, S_0))$, we can show that

$$\textbf{CIRC}(W'; Ab; Result, S_0) \models Holds(\textit{HasTicket}, S_0).$$

This example illustrates that Baker's approach can easily deal with reasoning about the past.

An advantage of the method that Baker points out is that it can deal with indirect effects of actions (*ramifications*) in a simple fashion. For instance, assume that we would like to include a domain constraint stating that if the commuter is on board the bus, he is at a well-lighted place. The axiomatization is easily modified in the following fashion: we first introduce a new fluent name *WellLighted* which stands for the commuter being at a well-lighted place. We then add the axiom

$$Holds(\textit{OnBoard}, s) \supset Holds(\textit{WellLighted}, s). \tag{3.20}$$

We also modify the UNA and DCA for fluents in the obvious fashion and replace the old existence of situations axiom by

$$\begin{array}{l}
\exists s[Holds(\textit{HasTicket}, s) \wedge Holds(\textit{OnBoard}, s) \wedge Holds(\textit{WellLighted}, s)] \wedge \\
\exists s[Holds(\textit{HasTicket}, s) \wedge \neg Holds(\textit{OnBoard}, s) \wedge Holds(\textit{WellLighted}, s)] \wedge \\
\exists s[Holds(\textit{HasTicket}, s) \wedge \neg Holds(\textit{OnBoard}, s) \wedge \neg Holds(\textit{WellLighted}, s)] \wedge \\
\exists s[\neg Holds(\textit{HasTicket}, s) \wedge Holds(\textit{OnBoard}, s) \wedge Holds(\textit{WellLighted}, s)] \wedge \\
\exists s[\neg Holds(\textit{HasTicket}, s) \wedge \neg Holds(\textit{OnBoard}, s) \wedge Holds(\textit{WellLighted}, s)] \wedge \\
\exists s[\neg Holds(\textit{HasTicket}, s) \wedge \neg Holds(\textit{OnBoard}, s) \wedge \neg Holds(\textit{WellLighted}, s)].
\end{array} \tag{3.21}$$

Note that in the new existence of situations axiom, the two possible conjunctive terms that contradict the domain constraint (3.20) are not included.

3.5.2 The Translation $B(D)$

Let us first define an auxiliary translation $T_B(D)$. $T_B(D)$ is similar to $P(D)$ in that the translations of the value propositions are exactly the same as in $P(D)$, and in that $T_B(D)$ includes the UNA and DCA for actions and the DCA for fluents. Each effect proposition is translated individually in $T_B(D)$. The translation of an effect proposition (3.2) is

$$Holds(P_1, s) \wedge \ldots \wedge Holds(P_n, s) \supset Holds(F, Result(A, s)). \tag{3.22}$$

In addition, $T_B(D)$ includes the existence of situations axiom for the domain D. This can be formulated as follows: Let σ range over states, that is, subsets of the set of all fluents. Define

$$St_\sigma(s) = \bigwedge_{F\in\sigma} Holds(F,s) \wedge \bigwedge_{F\notin\sigma} \neg Holds(F,s).$$

Now, the existence of situations axiom can be written as

$$\bigwedge_{\sigma} \exists s\ St_\sigma(s). \tag{3.23}$$

Finally, $T_B(D)$ includes the commonsense law of inertia (3.18). (Note that we do not need to add the UNA for fluents, since it follows from (3.23).)

Example 1 (continued). $T_B(D_1)$ consists of axioms 1-9 and 11 of $P(D_1)$, the law of inertia (3.18) and the following existence of situations axiom:

$$\begin{array}{c}
\exists s[Holds(\mathit{Ontable},s) \wedge Holds(\mathit{Inhand},s)]\wedge \\
\exists s[Holds(\mathit{Ontable},s) \wedge \neg Holds(\mathit{Inhand},s)]\wedge \\
\exists s[\neg Holds(\mathit{Ontable},s) \wedge Holds(\mathit{Inhand},s)]\wedge \\
\exists s[\neg Holds(\mathit{Ontable},s) \wedge \neg Holds(\mathit{Inhand},s)].
\end{array}$$

Let B' be the conjunction of the universal closures of the axioms of $T_B(D)$. Let us denote the formula **CIRC**$(B'; Ab; Result, S_0)$ by $B(D)$. The following theorem about $B(D)$ can now be proved:

Theorem 3.3 *Let D be any finite, consistent domain and let $B(D)$ be defined as before. For any value proposition α,*

$$B(D) \models B(\alpha) \iff D \textit{ entails } \alpha.$$

Note that this theorem is of the same form as Theorems 3.1 and 3.2. The proof of this theorem is based on the following lemma about the result of circumscribing Ab.

Main Lemma *Let D be a finite, consistent domain. Then $B(D)$ is equivalent to*

$$B' \wedge \forall f\ a\ s\, [Ab(f,a,s) \equiv \bigvee_{F\in\mathbf{F}, A\in\mathbf{A}} (f = F \wedge a = A \wedge \mathit{affected}_{F,A}(s))] \tag{3.24}$$

where

$$\mathit{affected}_{F,A}(s) = (precond^{+}_{F,A}(s) \wedge \neg Holds(F,s)) \vee (precond^{-}_{F,A}(s) \wedge Holds(F,s)). \tag{3.25}$$

Note that this lemma characterizes Ab completely.

3.6 Related Work on $\mathcal{A}$

The language $\mathcal{A}$ is introduced in [Gelfond and Lifschitz, 1993] and a new method for representing actions in logic programming is specified as a translation from a subset of $\mathcal{A}$ into a logic programming language, and the translation is proved to be sound relative to the semantics of $\mathcal{A}$. The three translations discussed in this chapter were first reported in [Kartha, 1993]. Translations from $\mathcal{A}$ into abductive logic programming in can be found in [Dung, 1993] and [Denecker and De Schreye, 1993], and into equational logic programming in [Thielscher, 1994b].

If a theorem prover, or a query evaluation procedure, is available for the target language of such a translation, then it becomes possible to use the translation for the automation of reasoning about action. For example, the logic programming interpreter XOLDT [Chen and Warren, 1992] is put to such use in [Lifschitz *et al.*, 1993], the abductive procedure SLDNFA in [Denecker and De Schreye, 1993], and the theorem prover of [Boyer and Moore, 1988] in [Subramanian, 1993].

In [Thielscher, 1994a], the language $\mathcal{A}$ is related to the ideas of [Sandewall, 1992].

In many ways, the expressive possibilities of $\mathcal{A}$ are limited. For instance, in $\mathcal{A}$, all the fluents are assumed to be independent. Thus we cannot have, for instance, two fluent *Alive* and *Dead*, which are negations of each other. Thus, $\mathcal{A}$ cannot express domain constraints and hence the ramification problem cannot be tackled within this framework. Also, all actions are assumed to have determinisitic effects. Such expressive limitations of $\mathcal{A}$ have led researchers to enhance it in various ways. Dung [1993] proposed a "relational" version of $\mathcal{A}$, in which fluents and actions may have arguments. In [Denecker and De Schreye, 1993] and [Thielscher, 1994b], dialects of $\mathcal{A}$ are outlined in which one can describe actions with nondeterministic effects. Baral and Gelfond [1993] extend $\mathcal{A}$ by an operator for the concurrent execution of a set of actions. The dialect of $\mathcal{A}$ defined in [Lifschitz, 1993b] provides symbols for time intervals.

3.7 Discussion

The first two translations we provide use first order methods for reasoning about action, whereas the third one uses the nonmonotonic formalism of circumscription. The soundness and completeness of the three translations allows us to compare these three different methods in a mathematically precise fashion: we get the result that they are equivalent in the sense that they all faithfully capture the semantics of $\mathcal{A}$. Thus, the work reported here introduces an important dimension along which different ideas of reasoning about action can be compared.

The work reported in this chapter is applicable as long as the action domain of interest can be formalized in $\mathcal{A}$. In spite of its extremely simple syntax, $\mathcal{A}$ provides a framework for discussing some interesting cases of reasoning about action, including temporal projection with incomplete information about the initial situation, reasoning from the future to the past ("explanation"), and some forms of hypothetical reasoning ("if a different sequence of actions were performed. . ."). Many problems which have been discussed extensively in the literature, such as the Yale shooting problem, are special cases of the formalization presented here. Instead of focusing on individual examples, our work yields three *classes* of successful formalizations.

Finally, the technically most difficult result—the soundness and completeness of the formalization based on Baker's approach—shows how to apply the methods of [Baker, 1991] in a general setting.

When we try to apply the method of this chapter to other formalizations of reasoning about action, we may expect two kinds of outcomes. One is that we see that there is a simple translation from $\mathcal{A}$ that is sound and complete. In this case, we are satisfied that the method works on at least on the class of problems formalizable $\mathcal{A}$. In the other case, we might see that there is no such translation, in which case we might try to modify the idea so that we again obtain soundness and completeness.

As an example of the first kind of outcome, we can easily prove a theorem of the same form as Theorem 3.2 for the first order scheme (similar to that of Pednault and Reiter) proposed by Elkan [1992]. An example of the second kind of outcome, the translation from $\mathcal{A}$ into the language of logic programs presented in [Gelfond and Lifschitz, 1993] is incomplete. Turner [1994] exhibits a sound and complete translation from $\mathcal{A}$ into the language of disjunctive logic programs.

As noted before, the expressive possibilities of the language $\mathcal{A}$ are limited. Hence we turn to studying extensions of this language and translations from them in the next few chapters.

3.8 Proofs

3.8.1 Proof of Theorem 3.1

Theorem 3.1 will be proved using Lemma 3.1 and Lemma 3.2.

Lemma 3.1 *For any model M of a finite domain D, there is a model M_P of $P(D)$ such that, for any value proposition α,*

$$\alpha \textit{ is true in } M \iff M_P \models P(\alpha). \tag{3.26}$$

Proof. In the statement of this lemma, the word "model" is used first in the sense defined in Section 3.2 and the other time in the sense of classical logic.

To describe the model M_P, we first need to define three different universes: the universe of fluents $| M_P |_f$, the universe of actions $| M_P |_a$, and the universe of situations $| M_P |_s$. We define

$$\begin{array}{rcl} | M_P |_f & = & \mathbf{F}, \\ | M_P |_a & = & \mathbf{A}, \\ | M_P |_s & = & \{\sigma \mid \sigma \subseteq \mathbf{F}\}. \end{array}$$

Interpret the fluent and action constants as themselves. It remains for us to define the interpretations for the situation constant S_0, the predicate $Holds$ and the function $Result$. Let us denote the interpretation of an object in the model M_P by superscripting the object with M_P. Thus, $S_0^{M_P}$ denotes the interpretation of the situation constant S_0.

Let $M = (\sigma_0, \Phi)$. Define

$$\begin{array}{lcl} S_0^{M_P} & = & \sigma_0, \\ \langle F, \sigma \rangle \in Holds^{M_P} & \iff & F \in \sigma, \\ Result^{M_P}(A, \sigma) & = & \Phi(A, \sigma). \end{array}$$

This completes the definition of M_P.

For convenience in presenting the proofs, let us extend the language of $P(D)$ by adding each element of the universe of situations of M_P as a new object constant. Thus we may define the interpretation of $Holds^{M_P}$ equivalently as follows:

$$M_P \models Holds(F, \sigma) \quad \iff \quad F \in \sigma.$$

Let us check that the axioms of $P(D)$ are true in M_P. For all axioms other than (3.5)-(3.9) this is obvious. Let us first check that M_P is a model of (3.5).

First of all, it is easy to verify by induction on n that

$$[A_1; \ldots; A_n]^{M_P} = M^{A_1;\ldots;A_n}. \tag{3.27}$$

It then follows that M_P satisfies (3.26). Since M is a model of the domain D, all value propositions in D are *true* in M. It then follows from (3.26) that M_P is a model of $P(\alpha)$ for any value proposition α in D.

Let us check that M_P satisfies axiom (3.6) for any fluent name F and action name A. Assume $M_P \models precond^+_{F,A}(\sigma)$. We want to show that $M_P \models Holds(F, Result(A, \sigma))$ or equivalently $F \in \Phi(A, \sigma)$. Since $M_P \models precond^+_{F,A}(\sigma)$, all the preconditions of one of the effect propositions in $S^+_{F,A}$ is satisfied by σ. By clause (i) from the definition of the model of a domain, we conclude that F holds in $\Phi(A, \sigma)$. That M_P satisfies axiom (3.7) is similar.

Finally, let us check that M_P satisfies the frame axiom of the form (3.8) for any fluent name F and action name A. Let $M_P \models \neg Holds(F, \sigma) \wedge \neg precond^+_{F,A}(\sigma)$. This means that F does not hold in σ and none of the effect propositions in $S^+_{F,A}$ have all of their preconditions hold in σ. By clause (ii) from the definition of the model of a domain, it follows that F does not hold in $\Phi(A, \sigma)$ so that $M_P \models \neg Holds(F, Result(A, \sigma))$. That M_P satisfies axiom (3.9) is similar. $\Box$

Lemma 3.2 *Let D be a finite, consistent domain. For any model M_P of $P(D)$, there is a model M of D such that for any value proposition α,*

$$\alpha \textit{ is true in } M \iff M_P \models P(\alpha). \tag{3.28}$$

Proof. Since $P(D)$ includes the UNA and DCA for fluents and actions, we can assume without loss of generality that

$$\begin{aligned} \mid M_P \mid_f &= \mathbf{F}, \\ \mid M_P \mid_a &= \mathbf{A}. \end{aligned}$$

For any ground situation term t, let

$$t^* = \{F \mid M_P \models Holds(F, t)\}.$$

We want to specify a model $M = (\sigma_0, \Phi)$ of D. Since D is consistent, Φ can be uniquely selected in such a way that the conditions (i) and (ii) from the definition of a model (Section 3.2) are satisfied.

Define

$$\sigma_0 = S_0^*.$$

Now, we need to check that every value proposition in D is true in $M = (\sigma_0, \Phi)$, and that M satisfies (3.28). Both assertions will follow if we prove that

$$[A_1; \ldots; A_n]^* = M^{A_1;\ldots;A_n}. \tag{3.29}$$

We now prove (3.29) by induction on n. The base case $S_0^* = \sigma_0$ is true by the the definition of σ_0. Assume inductively that

$$[A_1; \ldots; A_{n-1}]^* = M^{A_1;\ldots;A_{n-1}}.$$

To prove the inductive step, let us focus attention on an arbitrary, but fixed fluent F. We will prove (3.29) by showing that

$$F \in [A_1; \ldots ; A_n]^* \Longrightarrow F \in M^{A_1;\ldots;A_n}, \tag{3.30}$$

and

$$F \notin [A_1; \ldots ; A_n]^* \Longrightarrow F \notin M^{A_1;\ldots;A_n}. \tag{3.31}$$

Let us now prove (3.30). The proof of (3.31) is similar.

Assume $F \in [A_1; \ldots ; A_n]^*$. Equivalently,

$$M_P \models Holds(F, [A_1; \ldots ; A_n]). \tag{3.32}$$

Let $S_{n-1} = [A_1; \ldots ; A_{n-1}]$. Note that $M_P \not\models precond^-_{F,A_n}(S_{n-1})$, because otherwise, we would have

$$M_P \models \neg Holds(F, Result(A_n, S_{n-1})),$$

since M_P is a model of the translations of the effect axioms describing the effect of A_n on F. This contradicts (3.32).

We now consider the following two cases:

Case 1: $M_P \models precond^+_{F,A_n}(S_{n-1})$

Using the inductive hypothesis and the case assumption, we can conclude that $M^{A_1;\ldots;A_{n-1}}$ satisfies all the preconditions of at least one of the effect propositions of S^+_{F,A_n}. Since Φ satisfies clause (i) in the definition of a model (Section 3.2), we get that $F \in M^{A_1;\ldots;A_n}$.

Case 2: $M_P \not\models precond^+_{F,A_n}(S_{n-1})$

Using the contrapositive of the frame axiom (3.8), the case assumption and (3.32), we get that $M_P \models Holds(F, S_{n-1})$. By the inductive hypothesis, $F \in M^{A_1;\ldots;A_{n-1}}$. By the case assumption and the inductive hypothesis, there are no effect propositions describing the effect of A_n on F, whose preconditions hold in $M^{A_1;\ldots;A_{n-1}}$. Since Φ satisfies clause (ii) in the definition of a model (Section 3.2), we get that F holds in $M^{A_1;\ldots;A_n}$.

This completes the induction. □

Proof of Theorem 3.1.

Left to right: Let M be any model of D. By Lemma 3.1 there is a model M_P of $P(D)$ such that for any value proposition α,

$$\alpha \text{ is true in } M \iff M_P \models P(\alpha).$$

Assume $P(D) \models P(\alpha)$. Then $M_P \models P(\alpha)$. Hence α is true in M.

Right to left: Assume $P(D) \not\models P(\alpha)$. Then, there exists a model M_P of $P(D)$ such that $M_P \models \neg P(\alpha)$. By Lemma 3.2, there is a model M of D such that $\neg\alpha$ is true in M. Thus D does not entail α. □

3.8.2 Proof of Theorem 3.2

Theorem 3.2 is an immediate consequence of Theorem 3.1 and the following lemma:

Lemma 3.3 *Let D be a finite domain. Then the theories $P(D)$ and $R(D)$ are equivalent.*

Proof. It is sufficient to show that, in the presence of UNA and DCA for fluents and actions, the effect axioms and frame axioms of $P(D)$ are equivalent to the effect axioms and frame axioms of $R(D)$. The equivalence of the effect axioms of $P(D)$ and $R(D)$ is trivial to prove. The equivalence of the frame axioms of $P(D)$ and $R(D)$ is also easy to see: for instance, let us check that the frame axioms of $P(D)$ follow from those of $R(D)$. In the presence of the DCA for actions, we can rewrite (3.14) and (3.15) as

$$\begin{array}{l}\bigwedge_{A' \in \mathbf{A}}[\neg Holds(F,s) \wedge \neg \bigvee_{A \in A_F^+}(A' = A \wedge precond^+_{F,A}(s)) \supset \\ \qquad \neg Holds(F, Result(A', s))],\end{array} \tag{3.33}$$

and

$$\begin{array}{l}\bigwedge_{A' \in \mathbf{A}}[\neg Holds(F,s) \wedge \neg \bigvee_{A \in A_F^-}(A' = A \wedge precond^-_{F,A}(s)) \supset \\ \qquad \neg Holds(F, Result(A', s))].\end{array} \tag{3.34}$$

Let us focus attention on (3.33)((3.34) is similar). For $A' \in A_F^+$, (3.33) reduces (using the UNA for actions) to

$$\neg Holds(F,s) \wedge \neg precond^+_{F,A}(s) \supset \neg Holds(F, Result(A,s)),$$

which (3.8), a negative frame axiom of $P(D)$.

Similarly, for $A' \notin A_F^+$, (3.33) reduces to

$$\neg Holds(F,s) \supset \neg Holds(F, Result(A', s)). \tag{3.35}$$

Note that for $A' \notin A_F^+$, $precond^+_{F,A'}(s) = false$ and hence (3.8) reduces to (3.35). Similarly it can be shown that frame axioms of $R(D)$ follow from those of $P(D)$. □

3.8.3 The Proof of the Main Lemma

In this proof, unless otherwise specified, F will range over $\mathbf{F}$, A will range over $\mathbf{A}$ and σ will range over subsets of $\mathbf{F}$. Recall that B' denotes the conjunction of the universal closures of the axioms of $T_B(D)$ and $B(D)$ denotes $\mathbf{CIRC}(B'; Ab; Result, S_0)$. Since we need to substitute variables for the constants Ab, $Result$ and S_0, we will sometime write B' as $B'(Ab, Result, S_0)$.

Define

$$\mathit{affected}^+_{F,A}(s) = precond^+_{F,A}(s) \wedge \neg Holds(F,s), \tag{3.36}$$

$$\mathit{affected}^-_{F,A}(s) = precond^-_{F,A}(s) \wedge Holds(F,s). \tag{3.37}$$

Thus, $\mathit{affected}_{F,A}(s)$ can be written using these abbreviations as

$$\mathit{affected}_{F,A}(s) = \mathit{affected}^{+}_{F,A}(s) \vee \mathit{affected}^{-}_{F,A}(s). \tag{3.38}$$

Let Q be defined as

$$Q = \lambda f\ a\ s \bigvee_{F,A} (f = F \wedge a = A \wedge \mathit{affected}_{F,A}(s)). \tag{3.39}$$

We will use Proposition 5 from the Appendix (with Γ chosen as $\emptyset$) to prove the Main Lemma. It thus suffices to prove that the following two propositions are universally valid:

$$B'(Ab, Result, S_0) \quad \supset \quad Q \leq Ab, \tag{3.40}$$

$$B'(Ab, Result, S_0) \quad \supset \quad \exists g, s^* B'(Q, g, s^*). \tag{3.41}$$

The universal validity of (3.40) can be equivalently expressed as follows.

Lemma 3.4

$$B' \text{ entails } \forall f, a, s[\bigvee_{F,A} (f = F \wedge a = A \wedge \mathit{affected}_{F,A}(s)) \supset Ab(f, a, s)].$$

Proof. Assume B'. Using the DCA for fluents and actions, we can rewrite the formula to be proved as

$$\forall s \bigwedge_{F',A'} [\bigvee_{F,A} (F' = F \wedge A' = A \wedge \mathit{affected}_{F,A}(s)) \supset Ab(F', A', s)].$$

Using the UNA for actions, and the UNA for fluents (which is provable from B', using the existence of situations axiom), this can be further simplified to

$$\bigwedge_{F,A} \forall s\ \ [\mathit{affected}_{F,A}(s) \supset Ab(F, A, s)].$$

Let us focus attention on one of the conjuncts to be proved. Using (3.38), our proof obligation can be written as

$$\forall s[\mathit{affected}^{+}_{F,A}(s) \supset Ab(F, A, s)], \tag{3.42}$$

and

$$\forall s[\mathit{affected}^{-}_{F,A}(s) \supset Ab(F, A, s)]. \tag{3.43}$$

We will prove (3.42) (proof of (3.43) is similar). If $\mathit{affected}^{+}_{F,A}$ is *false*, this is trivial. Otherwise, using (3.36) and the definition of $precond^{+}_{F,A}$ from Section 3.3.2, we note that our proof obligation again splits up into proving, for each effect proposition E in the set $S^{+}_{F,A}$,

$$\forall s[pre_E(s) \wedge \neg Holds(F,s) \supset Ab(F,A,s)]. \tag{3.44}$$

Using the axiom of B' that is the translation of the effect proposition E, we get

$$pre_E(s) \wedge \neg Holds(F,s) \supset Holds(F, Result(A,s)).$$

From this, and the contrapositive of (3.18), we get $Ab(F,A,s)$, as was required. □

Since D is consistent, Φ can be uniquely selected in such a way that the conditions (i) and (ii) from the definition of a model (Section 3.2) are satisfied. Define

$$G(a,s,s') = \bigvee_{A,\sigma} [a = A \wedge St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s')].$$

Lemma 3.5 *The existence of situations axiom (3.23) and the DCA and UNA for actions entail*

$$\exists g\ \forall a\ s\ G(a,s,g(a,s)).$$

Proof. By the axiom of choice, it suffices to prove

$$\forall a\ s\ \exists s'\ G(a,s,s').$$

The DCA for actions enable our proof obligation to be written as

$$\forall s \bigwedge_A \exists s'\ G(A,s,s').$$

Using the UNA for actions, this can again be rewritten as

$$\forall s \bigwedge_A \exists s'\ [\bigvee_\sigma (St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s'))].$$

Thus, we have to prove for an arbitrary A

$$\exists s'\ [\bigvee_\sigma (St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s'))]. \tag{3.45}$$

We will first show, for any σ:

$$St_\sigma(s) \supset \exists s'(St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s')). \tag{3.46}$$

Consider an arbitrary σ. Since D has a model, $\Phi(A,\sigma)$ is well defined. By the existence of situations axiom, we have $\exists s' St_{\Phi(A,\sigma)}(s')$. (3.46) now follows easily.

From (3.46), it easily follows that

$$St_\sigma(s) \supset \bigvee_\sigma \exists s'(St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s')).$$

Since σ was arbitrary, we can conclude from the above that

$$\bigwedge_{\sigma}[St_{\sigma}(s) \supset \bigvee_{\sigma} \exists s'(St_{\sigma}(s) \wedge St_{\Phi(A,\sigma)}(s'))].$$

That is,

$$\bigvee_{\sigma} St_{\sigma}(s) \supset \bigvee_{\sigma} \exists s'(St_{\sigma}(s) \wedge St_{\Phi(A,\sigma)}(s')).$$

From the above and the universally valid formula $\bigvee_{\sigma} St_{\sigma}(s)$, we get the desired conclusion (3.45). □

The universal validity of (3.41) can be equivalently expressed as follows.

Lemma 3.6 *Let B' and Q be defined as before.*

$$B'(Ab, Result, S_0) \text{ entails } \exists g\ s^* B'(Q, g, s^*).$$

Proof. Assume $B'(Ab, Result, S_0)$. By Lemma 3.5, there exists a function g such that $\forall a\ s\ G(a, s, g(a, s))$ holds. This will serve as the g in Lemma 3.6.

Let $M = (\sigma_0, \Phi)$ be a model of the domain D. By the existence of situations axiom, $\exists s\ St_{\sigma_0}(s)$. Choose one such s to be the s^* in Lemma 3.6.

We must check that with this definition of g and s^*, the formula

$$Holds(F, g(A_m, g(A_{m-1}, \ldots g(A_1, s^*, \ldots)) \tag{3.47}$$

is entailed for each value proposition (3.1) in D, the sentence

$$\forall s(pre_E(s) \supset (Holds(F, g(A, s))) \tag{3.48}$$

is entailed for each effect proposition (3.2) in D, and that the sentence

$$\forall f, a, s[\neg Q(f, a, s) \supset (Holds(f, s) \equiv Holds(f, g(a, s)))] \tag{3.49}$$

is entailed.

To prove (3.47), it suffices to prove that B' entails the formula

$$St_{M^{A_1;\ldots;A_n}} g(A_n, \ldots, g(A_1, s^*) \ldots), \tag{3.50}$$

and to observe that since M is a model of D, F holds in $M^{A_1;\ldots;A_n}$ for any value proposition (3.1) from D, and that (3.47) is one of the conjuncts of (3.50).

The proof of (3.50) is by induction on n. The base case that B' entails $St_{\sigma_0}(s^*)$ follows from the definition of s^*. Let us denote by s^*_n the term $g(A_n, \ldots, g(A_1, s^*) \ldots)$. Thus, the inductive hypothesis is that B' entails

$$St_{M^{A_1;\ldots;A_{n-1}}}(s^*_{n-1}).$$

Since $g(A_n, s^*_{n-1}) = s^*_n$, we get from Lemma 3.5 that B' entails $G(A_n, s^*_{n-1}, s^*_n)$. That is, B' entails

$$\bigvee_\sigma [St_\sigma(s^*_{n-1}) \wedge St_{\Phi(A_n,\sigma)}(s^*_n)]. \tag{3.51}$$

From the fact that for any $\sigma \neq \sigma'$, $St_\sigma(s) \supset \neg St_{\sigma'}(s)$ is universally valid, and the inductive hypothesis, we get that B' entails $\neg St_\sigma(s^*_{n-1})$ for any $\sigma \neq M^{A_1;\ldots;A_{n-1}}$. From this and the second conjunct of (3.51), we get that B' entails $St_{M^{A_1;\ldots;A_n}}(s^*_n)$, as desired.

To prove (3.48), we proceed as follows: Let $g(A, s) = s'$. Then by Lemma 3.5, we have, $G(A, s, s')$, and consequently

$$\bigvee_\sigma (St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s')). \tag{3.52}$$

We will first show, for any σ,

$$pre_E(s) \supset [St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s') \supset Holds(F, s')]. \tag{3.53}$$

If all the preconditions of E hold in σ, F holds in $\Phi(A, \sigma)$ and we are done since

$$St_{\Phi(A,\sigma)}(s') \supset Holds(F, s').$$

Otherwise,

$$pre_E(s) \supset \neg St_\sigma(s) \tag{3.54}$$

is a tautology and the conclusion holds trivially.

From (3.53), we can conclude that

$$pre_E(s) \supset \bigwedge_\sigma [St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s') \supset Holds(F, s')].$$

That is,

$$pre_E(s) \supset [\bigvee_\sigma (St_\sigma(s) \wedge St_{\Phi(A,\sigma)}(s')) \supset Holds(F, s')]. \tag{3.55}$$

(3.52) and (3.55) yield the desired conclusion.

To prove (3.49), we first note using the definition of Q from (3.39) and the DCA for for fluents and actions, we can rewrite (3.49) as

$$\forall s \bigwedge_{F,A} [\bigwedge_{F',A'} (F \neq F' \vee A \neq A' \vee \neg \mathit{affected}_{F',A'}(s)) \supset \\ (Holds(F, s) \equiv Holds(F, g(A, s)))].$$

Using the UNA for actions, and the UNA for fluents (which is provable from B', using the existence of situations axiom), for any fixed F and A, all the terms in the antecedent reduce to $true$, except when $F = F'$ and $A = A'$. Thus our proof obligation can be simplified to

$$\forall s \bigwedge_{F,A} [\neg \mathit{affected}_{F,A}(s) \supset (Holds(F, s) \equiv Holds(F, g(A, s)))]. \tag{3.56}$$

Let us focus attention on one of the conjuncts to be proved. Using (3.25), we note that

$$\begin{aligned}\neg \mathit{affected}_{F,A}(s) \quad = \quad & \neg precond^{+}_{F,A}(s) \wedge \neg precond^{-}_{F,A}(s) \vee \\ & \neg precond^{+}_{F,A}(s) \wedge \neg Holds(F, s) \vee \\ & \neg precond^{-}_{F,A}(s) \wedge Holds(F, s).\end{aligned} \tag{3.57}$$

Recalling that

$$precond_{F,A}(s) = precond^{+}_{F,A}(s) \vee precond^{-}_{F,A}(s)$$

we see that (3.56) (for a fixed F and A) can be simplified as the conjunction of the following three formulae.

$$\neg precond_{F,A}(s) \supset [Holds(F, s) \equiv Holds(F, g(A, s))], \tag{3.58}$$

$$\neg precond^{+}_{F,A}(s) \wedge \neg Holds(F, s) \supset [Holds(F, s) \equiv Holds(F, g(A, s))], \tag{3.59}$$

and

$$\neg precond^{-}_{F,A}(s) \wedge Holds(F, s) \supset [Holds(F, s) \equiv Holds(F, g(A, s))]. \tag{3.60}$$

To prove (3.58), we proceed as in the proof of (3.48) above. Let $g(A, s) = s'$. As before, it suffices to show for any σ

$$\neg precond_{F,A}(s) \supset [(St_{\sigma}(s) \wedge St_{\Phi(A,\sigma)}(s')) \supset (Holds(F, s) \equiv Holds(F, s'))].$$

If there are not any effect proposition in $S_{F,A}$ such that all its preconditions hold in σ, then $F \in \sigma \;\Leftrightarrow\; F \in \Phi(A, \sigma)$, and we are done. Otherwise,

$$\neg precond_{F,A}(s) \supset \neg St_{\sigma}(s),$$

and the conclusion follows trivially.

Let us now prove (3.59) (The proof of (3.60) is similar). Let $g(A, s) = s'$. As before, it suffices to show for any σ

$$\begin{aligned}&\neg precond^{+}_{F,A}(s) \wedge \neg Holds(F, s) \supset \\ &\quad [(St_{\sigma}(s) \wedge St_{\Phi(A,\sigma)}(s')) \supset (Holds(F, s) \equiv Holds(F, s'))].\end{aligned}$$

If no effect proposition of $S^{+}_{F,A}$ has all its preconditions hold in σ and F does not hold in σ, then $F \in \sigma \;\Leftrightarrow\; F \in \Phi(A, \sigma)$, and we are done. Otherwise,

$$\neg precond_{F,A}(s) \wedge \neg Holds(F, s) \supset \neg St_{\sigma}(s),$$

and the conclusion follows trivially. □

The Main Lemma follows from Lemma 3.4 and Lemma 3.6.

3.8.4 Proof of Theorem 3.3

The following lemma shows that the models of circumscription are also models of $P(D)$.

Lemma 3.7 *Let D be a finite, consistent domain and let $B(D)$ be defined as before. Any model of $B(D)$ is a model of $P(D)$.*

Proof. We will show that the axioms of $P(D)$ follow from $B(D)$: The DCA for fluents and actions and the UNA for actions in $P(D)$ and $B(D)$ are identical. The UNA for fluents in $B(D)$ follows from the existence of situation axiom. The equivalence of the effect axioms in $P(D)$ and $B(D)$ is trivial to prove. Hence we need just show how to obtain the frame axioms of $P(D)$ from $B(D)$.

Substituting the definition of Ab from (3.24) in the commonsense law of inertia (3.18), we get for any F and A (using the expression for $\neg affected_{F,A}(s)$ from (3.57)),

$$\neg precond^{+}_{F,A}(s) \wedge \neg precond^{-}_{F,A}(s) \supset [Holds(F,s) \equiv Holds(F, Result(A,s))],$$

$$\neg precond^{+}_{F,A}(s) \wedge \neg Holds(F,s) \supset [Holds(F,s) \equiv Holds(F, Result(A,s))], \quad (3.61)$$

and

$$\neg precond^{-}_{F,A}(s) \wedge Holds(F,s) \supset [Holds(F,s) \equiv Holds(F, Result(A,s))]. \quad (3.62)$$

The frame axioms of $P(D)$, (3.8) and (3.9), clearly follow from (3.61) and (3.62) respectively. □

The following lemma expresses the completeness of the translation into the circumscriptive scheme.

Lemma 3.8 *Let D be a finite, consistent domain and $B(D)$ be as before. For any model M_B of $B(D)$, there is a model M of D such that for any value proposition α,*

$$\alpha \text{ is true in } M \iff M_B \models B(\alpha). \quad (3.63)$$

Proof. Direct consequence of Lemma 3.7 and Lemma 3.2. □

The following lemma expresses the soundness of the translation into the circumscriptive scheme.

Lemma 3.9 *For any model M of a finite domain D, there is a model M_B of $B(D)$ such that for any value proposition α,*

$$\alpha \text{ is true in } M \iff M_B \models B(\alpha). \quad (3.64)$$

Proof. The model M_B is constructed in exactly the same way as the model M_P in the proof of Lemma 3.1, except that here, we also need to define the interpretation of the predicate Ab. This is done as follows:

$$\langle F, A, \sigma\rangle \in Ab^{M_B} \Longleftrightarrow \neg[F \in \sigma \;\Leftrightarrow\; F \in \Phi(A, \sigma)]. \tag{3.65}$$

We must now check that M_B is a model of (3.24). That M_B satisfies $B(E)$ for the translation $B(E)$ of any effect proposition E is quite similar to the corresponding proof in Lemma 3.1 and is omitted. The proof showing that M_B satisfies $B(\alpha)$ for any value proposition α in D is identical to that in Lemma 3.1. Hence, we just need to check that M_B is a model of the commonsense law of inertia (3.18), the existence of situations axiom (3.23) and that M_B satisfies

$$\forall f\ a\ s[Ab(f, a, s) \equiv \bigvee_{F,A} (f = F \wedge a = A \wedge \mathit{affected}_{F,A}(s))]. \tag{3.66}$$

We first check that M_B satisfies (3.18). Using the DCA for fluents and actions, it suffices to check that M_B satisfies the following for every fluent F and action A:

$$\forall s[\neg Ab(F, A, s) \supset (Holds(F, s) \equiv Holds(F, Result(A, s)))].$$

That is, we must check for every σ that

$$[F \in \sigma \;\Leftrightarrow\; F \in \Phi(A, \sigma)] \Rightarrow [F \in \sigma \;\Leftrightarrow\; F \in \Phi(A, \sigma)].$$

But this is trivial.

That M_B satisfies the existence of situations axiom (3.23) is easy to check. We now check that M_B satisfies (3.66). Using the UNA and DCA for fluents and actions, it suffices to check that M_B satisfies the following for every fluent F and action A:

$$\forall s[Ab(F, A, s) \equiv \mathit{affected}_{F,A}(s)].$$

Left to right: Let $M_B \models Ab(F, A, \sigma)$. Then by (3.65),

$$\neg[F \in \sigma \;\Leftrightarrow\; F \in \Phi(A, \sigma)]. \tag{3.67}$$

F changes between σ and $\Phi(A, \sigma)$. Assume $F \in \sigma$ (the proof is similar when $F \notin \sigma$). Then, $F \notin \Phi(A, \sigma)$. Hence, there is an effect proposition E in $S^-_{F,A}$ all of whose preconditions hold in σ. That is, $M_B \models \mathit{affected}^-_{F,A}(\sigma)$ and hence $M_B \models \mathit{affected}_{F,A}(\sigma)$ (by (3.38)).

Right to left: Let $M_B \models \mathit{affected}_{F,A}(\sigma)$. By (3.25), we get that

$$M_B \models (precond^+_{F,A}(\sigma) \wedge \neg Holds(F, \sigma)) \vee (precond^-_{F,A}(\sigma) \wedge Holds(F, \sigma)).$$

Assume

$$M_B \models precond^+_{F,A}(\sigma) \wedge \neg Holds(F, \sigma).$$

(The proof is similar in the other case). That is, all the preconditions of an effect proposition E in $S^+_{F,A}$ hold in σ and $F \notin \sigma$. By clause(i) in the definition of a model (Section 3.2) and the first fact above, $F \in \Phi(A, \sigma)$. That is, (3.67) holds. Thus, $M_B \models Ab(F, A, \sigma)$. □

Theorem 3.3 follows from Lemma 3.8 and Lemma 3.9. □

It is instructive to see why we need to vary S_0 in addition to $Result$, while circumscribing Ab in B'. Consider a domain description D_1 that consists of a single value proposition, F **after** A. This has just one model, where $\sigma_0 = \{F\}$ and $\Phi(A, \sigma) = \sigma$ for any subset σ of $\{F\}$. Let B'_1 be the conjunction of the universal closures of $T_B(D_1)$ and let $B^*(D_1)$ be **CIRC**$(B'_1; Ab; Result)$. We would like to conclude that $Ab(f, a, s) \equiv$ *False*, as a result of the circumscription. However, this is not the case, as shown by the following result.

$$B^*(D_1) \not\models \forall fas[Ab(f, a, s) \equiv \mathit{False}].$$

To show this, we will exhibit a model of M of $B^*(D_1)$ where the extent of Ab is nonempty. The model M is defined as follows.

$$\begin{array}{lcl}
\mid M \mid_f & = & \{F\}, \\
\mid M \mid_a & = & \{A\}, \\
\mid M \mid_s & = & \{0, 1\}, \\
S_0^M & = & 0, \\
Result^M(A, i) & = & 1, \quad i \in \{0, 1\}, \\
Holds^M & = & \{\langle F, 1\rangle\}, \\
Ab^M & = & \{\langle F, A, 0\rangle\}.
\end{array}$$

It is easy to check that M is a model of B'_1. We will now show that M is a model of $B^*(D_1)$. I If it is not a model of $B^*(D_1)$, assume that M' is a model of B'_1 where the extent of Ab is smaller (and hence empty). Since M and M' interpret S_0 and $Holds$ identically, and since $M'$is a model of $Holds(F, [A])$, we get $Result^{M'}(A, 0) = 1$. By the commonsense law of inertia, $\langle F, A, 0\rangle$ is contained within the extent of Ab in M', which contradicts the assumption that the extent of Ab in M' is empty. Thus we get that M is a model of $B^*(D_1)$.

Chapter 4

$\mathcal{AR}^-$: A Language for Ramifications

4.1 Introduction

In this chapter, we introduce $\mathcal{AR}^-$, a language that is more expressive in many ways than the language $\mathcal{A}$ described in the previous chapter. For instance, domain constraints can be represented in $\mathcal{AR}^-$. Let us consider an example to illustrate this important expressive capability of $\mathcal{AR}^-$.

The "Yale Shooting" domain [Hanks and McDermott, 1987] can be encoded in $\mathcal{A}$ as follows:

$$\begin{array}{l} \textbf{initially } \neg \mathit{Loaded}, \\ \textbf{initially } \mathit{Alive}, \\ \mathit{Load} \textbf{ causes } \mathit{Loaded}, \\ \mathit{Shoot} \textbf{ causes } \neg \mathit{Alive} \textbf{ if } \mathit{Loaded}, \\ \mathit{Shoot} \textbf{ causes } \neg \mathit{Loaded}. \end{array} \tag{4.1}$$

Recall that in $\mathcal{A}$, any combination of fluent values represents a possible state of the system. However, this is not so in $\mathcal{AR}^-$. We will be able, for instance, to extend domain description (4.1) by the constraint

$$\textbf{always } \mathit{Walking} \supset \mathit{Alive}, \tag{4.2}$$

where *Walking* is a new fluent name. This constraint tells us that an assignment of truth values to fluents represents a state only if it makes the conditional $\mathit{Walking} \supset \mathit{Alive}$ true.

The domain description in $\mathcal{AR}^-$ consisting of propositions (4.1) and proposition (4.2) entails, for instance, the value proposition

$$\neg \mathit{Walking} \textbf{ after } \mathit{Load}; \mathit{Wait}; \mathit{Shoot}.$$

Note that the fluent *Walking* does not occur in the effect propositions. Thus any change in the value of *Walking* is an *indirect effect*, or ramification, of the actions that have been performed. The possibility of ramifications is the main feature of the dialect of $\mathcal{A}$ introduced here. We call this language $\mathcal{AR}^-$; $\mathcal{A}$ stands for "actions," $\mathcal{R}$ for "ramifications," and the superscript $-$ is used to distinguish this language from the more expressive language $\mathcal{AR}$ that will be defined in Chapter 7.

Besides the ability to describe indirect effects, the new language has other expressive possibilities that $\mathcal{A}$ lacks. It allows us to specify not only "fluent preconditions"—the conditions that need to be satisfied in order for an action to affect a specific fluent—but also "action preconditions," which, if violated, make the execution of an action impossible[1]. In $\mathcal{AR}^-$, we can specify a "coordinate frame" in the space of situations, as suggested in [Lifschitz, 1990]; the commonsense law of inertia applies only to the fluents that belong to the frame.

The syntax of the new action language is described in Section 4.2. Examples illustrating the use of the language are provided in Section 4.3, and its semantics is defined in Section 4.4. In Section 4.5, we study the properties of $\mathcal{AR}^-$. In Section 4.6, the language $\mathcal{A}$ from [Gelfond and Lifschitz, 1993] is embedded into $\mathcal{AR}^-$.

4.2 Syntax of $\mathcal{AR}^-$

4.2.1 Formulae and Propositions

To be precise, $\mathcal{AR}^-$, like $\mathcal{A}$, is not a single language, but rather a family of languages. A particular language in this group is characterized by

- a nonempty set of symbols, that are called *fluent names*, or *fluents*,
- a subset of fluent names, that is called the *frame*,
- a nonempty set of symbols, that are called *action names*, or *actions*.

A *formula* is a propositional combination of fluents.

There are three types of *propositions* in $\mathcal{AR}^-$—value propositions, effect propositions and constraints.

A *value proposition* is an expression of the form

$$C \textbf{ after } \overline{A}, \tag{4.3}$$

where C is a formula, and $\overline{A}$ is a string of actions. Informally, (4.3) asserts that C holds after the sequence of actions $\overline{A}$ is performed in the initial situation. For instance,

$$\textit{Heads} \textbf{ after } \textit{Toss}$$

is a value proposition, where *Heads* is a fluent and *Toss* is an action.

[1]This terminology belongs to Reiter [1991].

An *effect proposition* is an expression of the form

$$A \textbf{ causes } C \textbf{ if } P, \tag{4.4}$$

where A is an action, and C and P are formulae. Intuitively, (4.4) asserts that A, if executed in a situation in which the precondition P is true, makes C true. For instance,

GetOnBoard **causes** *OnBus* **if** *HasTicket*

is an effect proposition, where *GetOnBoard* is an action, and *OnBus* and *HasTicket* are fluents.

Finally, a *constraint* is a proposition of the form

$$\textbf{always } C, \tag{4.5}$$

where C is a formula. Intuitively, (4.5) asserts that C holds in all possible situations. We will sometimes identify a constraint (4.5) with the formula C.

A *domain description*, or *domain*, is a set of propositions.

4.2.2 Notational Conventions

In formulae, we will omit some parentheses, as customary in classical logic. If $\overline{A}$ in a value proposition (4.3) is empty, we will write this proposition as

$$\textbf{initially } C.$$

Otherwise, the members of $\overline{A}$ will be separated by semicolons. An effect proposition (4.4) will be written as

$$A \textbf{ causes } C$$

if P is *True*, and as

$$\textbf{impossible } A \textbf{ if } P$$

if C is *False*.

4.3 Examples

In this section we show how some systems involving action and change can be described in $\mathcal{AR}^-$. Our comments on the "models" of these domain descriptions will turn into precise statements after the semantics of $\mathcal{AR}^-$ is described in the next section.

Note that any domain description in $\mathcal{A}$ can be viewed as a domain description in $\mathcal{AR}^-$ (with no constraints). In Section 4.6, this will be discussed in more detail.

The following examples illustrate some of the expressive possibilities of $\mathcal{AR}^-$ that $\mathcal{A}$ lacks.

Example 1: Action Preconditions. The door in a hotel room can be opened by inserting the entry card. In the initial situation, the door is closed. We use two propositional fluents *HasCard* and *DoorOpen*, both belonging to the frame. There is one action *InsertCard*. The propositions are

$$\begin{array}{l} \textbf{initially } \neg \mathit{DoorOpen}, \\ \mathit{InsertCard} \textbf{ causes } \mathit{DoorOpen}, \\ \textbf{impossible } \mathit{InsertCard} \textbf{ if } \neg \mathit{HasCard}. \end{array}$$

This domain has two models; in one model, *HasCard* is false in the initial situation, and in the other model it is initially true. In the first model, the action of inserting the card cannot be executed.

Example 2: Nonframe Fluents. There are two switches and a light; the light is on only when the switches are both on or both off [Lifschitz, 1990]. This system can be described using three propositional fluents *Switch1, Switch2* and *Light*; only the first two belong to the frame. There are two action names, *Toggle1* and *Toggle2*. The propositions are:

$$\begin{array}{l} \textbf{always } \mathit{Light} \equiv (\mathit{Switch1} \equiv \mathit{Switch2}), \\ \mathit{Toggle1} \textbf{ causes } \neg \mathit{Switch1} \textbf{ if } \mathit{Switch1}, \\ \mathit{Toggle1} \textbf{ causes } \mathit{Switch1} \textbf{ if } \neg \mathit{Switch1}, \\ \mathit{Toggle2} \textbf{ causes } \neg \mathit{Switch2} \textbf{ if } \mathit{Switch2}, \\ \mathit{Toggle2} \textbf{ causes } \mathit{Switch2} \textbf{ if } \neg \mathit{Switch2}. \end{array}$$

This domain has 4 models that differ by the initial states of the switches.

It is essential in this example that the fluent *Light* is not included in the frame. Otherwise, the effects of the actions would be indeterminate. For instance, the effect of *Toggle1* would be to change either the values of *Switch1* and *Light* or, surprisingly, the values of *Switch1* and *Switch2*.

Example 3: Ramifications. As discussed in the introduction, this can be illustrated by the domain description consisting of propositions (4.1) and proposition (4.2). All fluents are included in the frame. There are two models that differ by the initial value of *Walking*.

Example 4: Nondeterminism. Each of two light bulbs is controlled by a switch. The action of turning on the light is nondeterministic, because it is not specified which of the switches is used. In the following description, *Switch1* and *Switch2* belong to the frame, and *Light* does not:

$$\begin{array}{l} \textbf{always } \mathit{Light} \equiv (\mathit{Switch1} \vee \mathit{Switch2}), \\ \mathit{TurnOn} \textbf{ causes } \mathit{Light}, \\ \mathit{TurnOff} \textbf{ causes } \neg \mathit{Light}. \end{array}$$

Note that the effect propositions do not include disjunctions; the nondeterminism in this domain description is due entirely to the presence of a constraint.

4.4 Semantics of $\mathcal{AR}^-$

4.4.1 States and Transition Functions

In this section, we consider truth-valued functions on the set of fluents called *valuations*. Such a function can be extended to arbitrary formulae according to the truth tables of propositional logic.

We say that a valuation σ is a *state* of a domain D if it maps every constraint in D to T ("true").

For instance, the domain consisting of propositions (4.1) and (4.2) has 6 states; they are the truth-valued functions on

$$\{Loaded, Alive, Walking\}$$

that make the formula $Walking \supset Alive$ true.

The semantics of $\mathcal{AR}^-$ shows how the effect propositions of D define a nondeterministic transition system with this set of states, whose input symbols are actions. We will describe this transition system by a function *Res* that maps an action and a state to a set of states. The elements of $Res(A, \sigma)$ are, intuitively, the states that differ from σ as dictated by the effect propositions for A, but, at the same time, differ from σ as little as possible. In determining the "difference" between σ and the elements of $Res(A, \sigma)$, we will consider only the frame fluents.

As a preliminary step, define $Res_0(A, \sigma)$ to be the set of states σ' such that, for each effect proposition A **causes** C **if** P in D, $\sigma'(C) = \mathsf{T}$ whenever $\sigma(P) = \mathsf{T}$. The set $Res(A, \sigma)$ will be defined as the subset of $Res_0(A, \sigma)$ whose elements are "close" to σ.

In order to make this precise, the following notation is needed. The *distance* $\rho(\sigma_1, \sigma_2)$ between states σ_1 and σ_2 is the set of fluents on which the states differ:

$$\rho(\sigma_1, \sigma_2) = \{F \mid \sigma_1(F) \neq \sigma_2(F)\}.$$

Let **Fr** denote the frame.

Now the transition function *Res* corresponding to D is defined as follows: $Res(A, \sigma)$ is the set of states $\sigma' \in Res_0(A, \sigma)$ for which $\rho(\sigma, \sigma') \cap \mathbf{Fr}$ is minimal relative to set inclusion—in other words, for which there is no $\sigma'' \in Res_0(A, \sigma)$ such that $\rho(\sigma, \sigma'') \cap \mathbf{Fr}$ is a proper subset of $\rho(\sigma, \sigma') \cap \mathbf{Fr}$.

Consider, for instance, the domain of Example 3. We will represent a state σ by the set of fluents F such that $\sigma(F) = \mathsf{T}$. In this notation,

$$\begin{array}{r}
\mathit{Res}_0(\mathit{Load}, \{\mathit{Alive}\}) = \{\{\mathit{Loaded}\}, \{\mathit{Loaded}, \mathit{Alive}\}\}, \\
\mathit{Res}(\mathit{Load}, \{\mathit{Alive}\}) = \{\{\mathit{Loaded}, \mathit{Alive}\}\};
\end{array}$$

4.4.2 Models and Entailment

A *structure* is a partial function from strings of actions to valuations whose domain is nonempty and prefix-closed. If a structure Ψ is defined on a string $\overline{A}$, we say that $\overline{A}$ is *executable* in Ψ. Thus, note that in any structure, the empty string of actions is always executable. As we will see shortly, $\Psi(\overline{A})$ in a model of D represents the state that results from the execution of the members of $\overline{A}$ sequentially from the initial state.

A value proposition (4.3) is *true* in a structure Ψ if $\overline{A}$ is executable in Ψ and $\Psi(\overline{A})(C) = \mathsf{T}$. A constraint **always** C is *true* in a structure Ψ if for any string $\overline{A}$ executable in Ψ, $\Psi(\overline{A})(C) = \mathsf{T}$.

A structure Ψ is a *model* of a domain D if every value proposition and every constraint in D is true in Ψ, and, for every string of actions $\overline{A}$ executable in Ψ and every action A,

- if $\overline{A}A$ is executable in Ψ, then
$$\Psi(\overline{A}A) \in Res(A, \Psi(\overline{A})),$$
- otherwise, $Res(A, \Psi(\overline{A})) = \emptyset$.

Consider, for instance, the domain of Example 1. Its states can be represented by subsets of $\{\mathit{HasCard}, \mathit{DoorOpen}\}$. The only action in this domain is *InsertCard*, and its transition function is defined by the equations

$$\mathit{Res}(\mathit{InsertCard}, \sigma) = \begin{cases} \{\sigma \cup \{\mathit{DoorOpen}\}\}, & \\ \qquad \text{if } \mathit{HasCard} \in \sigma, \\ \emptyset, \qquad \text{otherwise.} \end{cases}$$

It is easy to see that this domain has two models. In one model, Ψ_1, the only executable string of actions is the empty string ϵ, and $\Psi_1(\epsilon) = \emptyset$. In the other model, Ψ_2, every string of actions is executable, and

$$\Psi_2(\overline{A}) = \begin{cases} \{\mathit{HasCard}\}, & \text{if } \overline{A} = \epsilon, \\ \{\mathit{HasCard}, \mathit{DoorOpen}\}, & \text{otherwise.} \end{cases}$$

We say that a domain description is *consistent* if it has a model. Two domain descriptions are *equivalent* if they have the same models.

A value proposition is *entailed* by a domain description D if it is true in every model of D.

4.5 Some Theorems about Domain Descriptions

In this section, we first show how two ideas familiar from classical logic—the "replacement property" and the notion of an explicit definition—can be applied to the language $\mathcal{AR}^-$. We then show how a domain description can be replaced by an equivalent domain description that does not include any constraints. Finally, we state a "restricted monotonicity" property of domain descriptions in $\mathcal{AR}^-$.

4.5.1 Replacement

Let T be a set of constraints. We say that formulae G and H are *equivalent* with respect to T if the formula $G \equiv H$ is a propositional consequence of T. If, for instance, T includes the constraint from Example 4, then *Light* is equivalent to *Switch1* $\vee$ *Switch2* with respect to T.

Theorem 4.1 *Let D be a domain description, let T be a subset of the constraints in D, and let G and H be formulae equivalent with respect to T. If a domain description D' is obtained from D by replacing G by H in some or all occurrences in any value propositions, effect propositions, and constraints that do not belong to T, then D and D' are equivalent.*

This theorem shows, for instance, that the domain of Example 4 is equivalent to

$$\begin{array}{l} \textbf{always } \mathit{Light} \equiv (\mathit{Switch1} \vee \mathit{Switch2}), \\ \mathit{TurnOn} \textbf{ causes } \mathit{Switch1} \vee \mathit{Switch2}, \\ \mathit{TurnOff} \textbf{ causes } \neg\mathit{Switch1} \wedge \neg\mathit{Switch2} \end{array} \tag{4.6}$$

and to

$$\begin{array}{l} \textbf{always } \mathit{Light} \equiv (\mathit{Switch2} \vee \mathit{Switch1}), \\ \mathit{TurnOn} \textbf{ causes } \mathit{Light}, \\ \mathit{TurnOff} \textbf{ causes } \neg\mathit{Light} \end{array}$$

(take $T = \emptyset$).

4.5.2 Explicit Definitions

A constraint in a domain description D is an *explicit definition* of a fluent F if it has the form

$$\textbf{always } F \equiv R, \tag{4.7}$$

and F

- does not belong to the frame,
- does not occur in R,

- does not occur in any other propositions in D.

For instance, the constraint in (4.6) is an explicit definition of *Light*.

Theorem 4.2 *Let D be a domain description that includes an explicit definition of a fluent F, and let D' be the domain description obtained from D by deleting this definition and deleting F from the language. A value proposition that does not contain F is entailed by D if and only if it is entailed by D'.*

Thus in $\mathcal{AR}^-$, as in classical logic, definitional extensions are conservative.

4.5.3 Eliminating Constraints

The following theorem shows how a domain that includes constraints may be replaced by an equivalent domain without any constraints.

Theorem 4.3 *Let D be a domain description and let S be a set of constraints in D. Let D' be the domain description obtained from D by deleting all constraints in S and adding the following propositions:*

- *For each action A and for each constraint* **always** *C in S, the effect proposition A* **causes** *C is added and*
- *For each constraint* **always** *C in S, the value proposition* **initially** *C is added.*

Then D and D' are equivalent.

4.5.4 Restricted Monotonicity

When an effect proposition or a constraint is added to a domain description, the set of value propositions entailed by the description often changes in a nonmonotonic fashion: Some of the propositions entailed by the original description are not entailed by the extended one. This cannot happen, however, when a value proposition is added, as shown by the following.

Theorem 4.4 *Let D be a domain description and let D' be the domain description obtained from D by adding a value proposition. Then for any value proposition α, if D entails α, D' entails α.*

The proof of this is easy: it is clear from the definition of a model given in Section 4.4 that adding a value proposition to a domain description can only make the set of its models smaller, and thus make the set of value propositions entailed by it larger. This is what one would intuitively expect, because an additional value proposition in a domain description expresses merely an additional assumption about the "trajectory" under consideration.

The observation made above is a "restricted monotonicity property" in the sense of [Lifschitz, 1993c]. Propositions of all four types play the role of postulates, and value propositions play the role of sentences, assertions and parameters. A similar fact about the language $\mathcal{A}$ is stated in [Lifschitz, 1993c] as Proposition 3.

4.6 Relation to $\mathcal{A}$

We claimed in Section 4.3 that the language $\mathcal{A}$ from [Gelfond and Lifschitz, 1993] can be viewed as a subset of $\mathcal{AR}^-$. Now this assertion can be given a precise meaning.

First of all, note that any domain description in $\mathcal{A}$ can be viewed as a domain description in $\mathcal{AR}^-$ in which all fluents belong to the frame; we only need to agree to identify an effect proposition

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

in the sense of $\mathcal{A}$ with

$$A \textbf{ causes } F \textbf{ if } P_1 \wedge \ldots \wedge P_n.$$

For a consistent domain description in the language $\mathcal{A}$, there is a simple one-one correspondence between its models as defined here and its models in the sense of Section 4.4:

Theorem 4.5 *Let D be a consistent domain description in the language $\mathcal{A}$. If M is a model of D in the sense of $\mathcal{A}$, then the function*

$$\overline{A} \mapsto M^{\overline{A}} \tag{4.8}$$

is a model of D in the sense of $\mathcal{AR}^-$. Moreover, every model of D in the sense of $\mathcal{AR}^-$ has form (4.8) for exactly one model M of D.

Corollary. *Let D be a consistent domain description in the language $\mathcal{A}$. A value proposition is entailed by D in the sense of $\mathcal{A}$ if and only if it is entailed by D in the sense of $\mathcal{AR}^-$.*

Without assuming the consistency of D, the assertions of Theorem 3 and of the corollary would be incorrect. For instance, the domain description

$$A \textbf{ causes } F,$$
$$A \textbf{ causes } \neg F,$$

although inconsistent in $\mathcal{A}$, has models in the sense of $\mathcal{AR}^-$ (with ϵ as the the only executable string of actions).

4.7 Discussion

In this chapter, we have defined the syntax and semantics of the language $\mathcal{AR}^-$ and studied its properties. The language $\mathcal{AR}^-$ is a subset of the language $\mathcal{AR}_0$ introduced in [Kartha and Lifschitz, 1994]—the latter has one additional kind of propositions called *release propositions*. Syntactically, they are of the form

$$A \textbf{ releases } F \textbf{ if } P, \tag{4.9}$$

where A is an action, F a fluent which belongs to the frame, and P a formula. Intuitively, (4.9) says that F is exempt from obeying the commonsense law of inertia when the action A is executed in a situation when the precondition P is true.

These propositions make it possible to express simple kinds of nondeterminism in $\mathcal{AR}_0$. However, as observed by Fangzhen Lin (personal communication, October 26, 1993), the behavior of nondeterministic actions in $\mathcal{AR}_0$ may be unintuitive when indirect effects are involved. The language $\mathcal{AR}$ we introduce in Chapter 7 allows a more satisfactory formalization of nondeterministic actions.

4.8 Proofs

4.8.1 Proof of Theorem 4.1

In the proof of this theorem, we will have occasion to talk about the sets $Res_0(A, \sigma)$, $Res(A, \sigma)$ etc. in the domains D and D'. To avoid confusion, let us agree to denote these sets in D' by $Res_0'(A, \sigma)$, $Res'(A, \sigma)$ etc.

We need to show that every model of D is a model of D' and vice versa. It suffices to show that

1. The states of D and D' are identical.

2. For every action A and state σ, $Res(A, \sigma) = Res'(A, \sigma)$.

3. Any value proposition of D is true in a model of D' and vice versa.

We will first show that the states of D and D' are identical. Let σ be a valuation such that $\sigma(G) = \sigma(H)$. Hence for any formula P, if P' is the formula obtained by replacing all or some occurrences of G in P by H,

$$\sigma(P) = \sigma(P'). \tag{4.10}$$

Let σ be a state of D. Then $\sigma(T) = \mathsf{T}$, and hence $\sigma(G \equiv H) = \mathsf{T}$. That is, $\sigma(G) = \sigma(H)$. Hence, by (4.10), we see that for each constraint C in $D \setminus T$, $\sigma(C) = \sigma(C')$

where C' is obtained by replacing all or some occurrences of G in C by H. Thus, σ is a state of D'. Similarly, we can see that every state of D' is a state of D.

Let us now show that $Res(A,\sigma) = Res'(A,\sigma)$. To show this, we first need to check that $Res_0(A,\sigma) = Res'_0(A,\sigma)$. Let P' (C') be obtained by replacing all or some occurrences of G in P (C) by H.

$Res_0(A,\sigma)$
$= \{$ Definition $\}$
$\{\sigma' \mid \sigma'$ is a state of D such that for any effect proposition
A **causes** C **if** P in D, $\sigma(P) = \mathsf{T} \Rightarrow \sigma'(C) = \mathsf{T}\}$
$= \{$By (4.10) $\}$
$\{\sigma' \mid \sigma'$ is a state of D' such that for any effect proposition
A **causes** C' **if** P' in D', $\sigma(P') = \mathsf{T} \Rightarrow \sigma'(C') = \mathsf{T}\}$
$= \{$ Definition $\}$
$Res'_0(A,\sigma)$

Similarly, we get that for any action name A and state σ, $Res(A,\sigma) = Res'(A,\sigma)$.

Let us check that any value proposition of D is true in a model of D'. Consider any value proposition C **after** $\overline{A}$ in D. Let the corresponding value proposition in D' obtained from it by replacing some or all occurrences of G by H be C' **after** $\overline{A}$. Let Ψ' be a model of D'. Then $\Psi'(\overline{A})(C') = \mathsf{T}$. Using (4.10), we see that

$$\Psi'(\overline{A})(C) = \Psi'(\overline{A})(C'),$$

and hence $\Psi'(\overline{A})(C) = \mathsf{T}$. Similarly, it is easy to show that any value proposition of D' is true in a model of D. □

4.8.2 Proof of Theorem 4.2

Let σ be any state of D and let $g(\sigma)$ be the restriction of σ to the fluents in D'. Note that g is a function from the states of D to the states of D' and that for any formula C of D',

$$\sigma(C) = g(\sigma)(C). \tag{4.11}$$

Lemma 4.1 *The function g as defined above is a bijection between the states of D and the states of D'.*

Proof. Let us first check that g is an injection. Assume that $g(\sigma_1) = g(\sigma_2)$. Hence $g(\sigma_1)(R) = g(\sigma_2)(R)$, where R is the formula in the explicit definition of the fluent F in D. Since R is also a formula of D', the previous equality in conjunction with (4.11) implies that $\sigma_1(R) = \sigma_2(R)$. Hence $\sigma_1(F) = \sigma_2(F)$, and consequently, $\sigma_1 = \sigma_2$.

Now we need to show that g is a surjection and to this end, we let σ' be any state of D'. Let $\sigma'(R) = v$, where $v \in \{\mathsf{T}, \mathsf{F}\}$. Define $\sigma = \sigma' \cup \{\langle F, v\rangle\}$. It is clear that $g(\sigma) = \sigma'$. It remains to check that σ is a state of D. Clearly, σ satisfies any constraint **always** C' in D', because $\sigma(C') = \sigma'(C')$. Also, by construction, σ satisfies the explicit definition **always** $F \equiv R$. Thus, σ satisfies all the constraints of D. □

The above lemma allows us to write any state of D' as $g(\sigma)$, for some state σ of D. We will make extensive use of this fact in the proofs of Lemma 4.2 and Lemma 4.3.

The g-image of a set of states is defined in the usual manner: if Γ is a set of states of D,

$$g(\Gamma) = \{g(\sigma) \mid \sigma \in \Gamma\}.$$

As before, the sets $Res_0(A, \sigma')$ etc. of D' etc will be denoted by $Res'_0(A, \sigma')$ etc.

Lemma 4.2 *For any action name A and state σ,*

$$Res'_0(A, g(\sigma)) = g(Res_0(A, \sigma)).$$

Proof.

$$\begin{aligned}
&\quad Res'_0(A, g(\sigma)) \\
&= \{\text{ Definition }\} \\
&\quad \{g(\sigma') \mid \ \sigma' \text{ is a state of } D \text{ such that for any effect proposition} \\
&\quad\ \ A \textbf{ causes } G \textbf{ if } P \text{ in } D,\ \ g(\sigma)(P) = \mathsf{T} \Rightarrow g(\sigma')(G) = \mathsf{T}\} \\
&= \{\text{By (4.11) }\} \\
&\quad \{g(\sigma') \mid \sigma' \text{ is a state of } D \text{ such that for any effect proposition} \\
&\quad\ \ A \textbf{ causes } G \textbf{ if } P \text{ in } D,\ \ \sigma(P) = \mathsf{T} \Rightarrow \sigma'(G) = \mathsf{T}\} \\
&= \{\text{Definition of } g \text{ on a set }\} \\
&\quad g(Res_0(A, \sigma))
\end{aligned}$$

□

Lemma 4.3 *For any action name A and state σ,*

$$Res'(A, g(\sigma)) = g(Res(A, \sigma))$$

Proof. For any action name A, state σ, and state σ_1 belonging to $Res_0(A, \sigma)$ (in D), we will first show that

$$\rho(\sigma, \sigma_1) \cap \mathbf{Fr} = \rho(g(\sigma), g(\sigma_1)) \cap \mathbf{Fr}. \tag{4.12}$$

(Note that the frame in D and D' are identical).

If σ and σ_1 assign the same value to F, then $\rho(\sigma, \sigma_1) = \rho(g(\sigma), g(\sigma_1))$, and (4.12) follows. If not,

$$\rho(\sigma, \sigma_1) = \rho(g(\sigma), g(\sigma_1))) \cup \{F\},$$

and (4.12) again follows, since $F \notin \mathbf{Fr}$.

$$
\begin{aligned}
& Res'(A, g(\sigma)) \\
= & \{\text{Definition}\} \\
& \quad \{g(\sigma') \mid g(\sigma') \in Res'_0(A, g(\sigma)) \text{ and there does not exist a } g(\sigma'') \text{ in} \\
& \quad\; Res'_0(A, g(\sigma)) \text{ so that } \rho(g(\sigma), g(\sigma'')) \cap \mathbf{Fr} \subset \rho(g(\sigma), g(\sigma')) \cap \mathbf{Fr}\} \\
= & \{\text{Lemma 4.2}\} \\
& \quad \{g(\sigma') \mid \sigma' \in Res_0(A, \sigma) \text{ and there does not exist a } \sigma'' \text{ in } Res_0(A, \sigma) \\
& \quad\; \text{so that } \rho(g(\sigma), g(\sigma'')) \cap \mathbf{Fr} \subset \rho(g(\sigma), g(\sigma')) \cap \mathbf{Fr}\} \\
= & \{\ (4.12)\ \} \\
& \quad \{g(\sigma') \mid \sigma' \in Res_0(A, \sigma) \text{ and there does not exist a } \sigma'' \text{ in } Res_0(A, \sigma) \\
& \quad\; \text{so that } \rho(\sigma, \sigma'') \cap \mathbf{Fr} \subset \rho(\sigma, \sigma') \cap \mathbf{Fr}\} \\
= & \{\text{ Definition of } g \text{ on a set }\} \\
& \quad g(Res(A, \sigma))
\end{aligned}
$$

□

Consider a function f that maps a model Ψ of D into a structure Ψ' defined as follows:

1. For each tuple of action names $\overline{A}$, $\Psi'(\overline{A})$ is defined iff $\Psi(\overline{A})$ is defined.
2. $\Psi'(\overline{A}) = g(\Psi(\overline{A}))$.

Lemma 4.4 *For any model Ψ of D, $f(\Psi)$ is is a model of D'.*

Proof. Let $\Psi' = f(\Psi)$. For any value proposition C **after** $\overline{A}$ of D, $\overline{A}$ is executable in Ψ and $\Psi(\overline{A})(C) = \mathsf{T}$, since Ψ is a model of D. Hence $\overline{A}$ is also executable in Ψ'. Since the value propositions in D and D' are identical, we can use (4.11) to conclude that

$$\Psi(\overline{A})(C) = \Psi'(\overline{A})(C). \tag{4.13}$$

Thus, we see that the value propositions of D' are true in Ψ'. Similarly, it is easy to see that for any constraint **always** C in D',

$$\Psi(\overline{A})(C) = \Psi'(\overline{A})(C)$$

and hence the constraints of D' are true in Ψ'.

Assume that $\overline{A}$ is executable in Ψ'. Then $\overline{A}$ is executable in Ψ.

$$
\begin{array}{ll}
& \overline{A}; A \text{ is executable in } \Psi' \\
\Leftrightarrow & \{\text{Definition of } \Psi' \} \\
& \overline{A}; A \text{ is executable in } \Psi \\
\Rightarrow & \{ \Psi \text{ is a model}\} \\
& \Psi(\overline{A}; A) \in Res(A, \Psi(\overline{A})) \\
\Leftrightarrow & \{\text{Definition of } g \text{ on a set} \} \\
& g(\Psi(\overline{A}; A)) \in g(Res(A, \Psi(\overline{A}))) \\
\Leftrightarrow & \{\text{Definition of } \Psi' \} \\
& \Psi'(\overline{A}; A) \in g(Res(A, \Psi(\overline{A}))) \\
\Leftrightarrow & \{\text{Lemma 4.3} \} \\
& \Psi'(\overline{A}; A) \in Res'(A, g(\Psi(\overline{A}))) \\
\Leftrightarrow & \{\text{Definition of } \Psi' \} \\
& \Psi'(\overline{A}; A) \in Res'(A, \Psi'(\overline{A}))
\end{array}
$$

Similarly,

$$
\begin{array}{ll}
& \overline{A}; A \text{ is not executable in } \Psi' \\
\Leftrightarrow & \{ \text{Definition of } \Psi' \} \\
& \overline{A}; A \text{ is not executable in } \Psi \\
\Rightarrow & \{ \Psi \text{ is a model} \} \\
& Res(A, \Psi(\overline{A})) = \emptyset \\
\Leftrightarrow & \{ \text{Definition of } g \text{ on a set} \} \\
& g(Res(A, \Psi(\overline{A}))) = \emptyset \\
\Leftrightarrow & \{ \text{Lemma 4.3} \} \\
& Res'(A, g(\Psi(\overline{A}))) = \emptyset \\
\Leftrightarrow & \{\text{Definition of } \Psi'\} \\
& Res'(A, \Psi'(\overline{A})) = \emptyset
\end{array}
$$

Thus Ψ' is a model of D'. □

Lemma 4.5 *The function f from the models of D to the models of D' is a bijection.*

Proof. Clearly, f is injective. Let Ψ' be any model of D'. Let us define a structure Ψ as follows:

1. For any tuple of actions $\overline{A}$, $\Psi(\overline{A})$ is defined iff $\Psi'(\overline{A})$ is defined.

2. If $\Psi'(\overline{A})(R) = v$ where $v \in \{\mathsf{T}, \mathsf{F}\}$,

$$\Psi(\overline{A}) = \Psi'(\overline{A}) \cup \{\langle F, v \rangle\}$$

Note that if $\Psi(\overline{A})$ is defined, $g(\Psi(\overline{A})) = \Psi'(\overline{A})$. Using this, we can prove, in a manner quite analogous to the proof of Lemma 4.4, that Ψ is a model of D and $f(\Psi) = \Psi'$. $\square$

Proof of Theorem 4.2

Let C **after** $\overline{A}$ be any value proposition that does not contain F.

C **after** $\overline{A}$ is entailed by D

$\Leftrightarrow$ {Definition of entailment }

For each model Ψ of D, $\overline{A}$ is executable in Ψ and $\Psi(\overline{A}) = C$

$\Leftrightarrow$ { Lemma 4.5, Eqn (4.13) }

For each model Ψ' of D', $\overline{A}$ is executable in Ψ' and $\Psi'(\overline{A}) = C$

$\Leftrightarrow$ {Definition of entailment }

C **after** $\overline{A}$ is entailed by D'.

$\square$

4.8.3 Proof of Theorem 4.3

First of all, note that the set of states of D is a subset of the set of states of D'. We will first show that for any action name A and state σ of D,

$$Res_0(A, \sigma) = Res'_0(A, \sigma). \tag{4.14}$$

$\sigma_1 \in Res_0(A, \sigma)$

$\Leftrightarrow$ { Definition }

σ_1 is a state of D and for any effect proposition

A **causes** G **if** P in D, $\sigma(P) = \mathsf{T} \Rightarrow \sigma_1(G) = \mathsf{T}$

$\Leftrightarrow$ { Definition of state }

σ_1 is a valuation of D and $\sigma_1(C) = \mathsf{T}$ for any constraint

always C in D and for any effect proposition

A **causes** G **if** P in D, $\sigma(P) = \mathsf{T} \Rightarrow \sigma_1(G) = \mathsf{T}$

$\Leftrightarrow$ { Definition of the set of effect propositions of D' }

σ_1 is a state of D' for any effect proposition

A **causes** G **if** P in D', $\sigma(P) = \mathsf{T} \Rightarrow \sigma_1(G) = \mathsf{T}$

$\Leftrightarrow$ { Definition }

$\sigma_1 \in Res'_0(A, \sigma)$

Using (4.14), it is easy to show that for any action name A and state σ of D, $Res(A, \sigma) = Res'(A, \sigma)$. It is then easy to prove (using induction) that for any model Ψ' of D' and for any sequence of actions $\overline{A}$ executable in Ψ', $\Psi'(\overline{A})$ is a state of D. The equivalence of D and D' follows easily from these facts. $\square$

4.8.4 Proof of Theorem 4.5

Recall that a domain D in $\mathcal{A}$ can be considered to be a domain D' in $\mathcal{AR}^-$ if we consider each effect proposition

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

of D as a syntactic variant of the effect proposition

$$A \textbf{ causes } F \textbf{ if } P_1 \wedge \ldots \wedge P_n$$

of D' and assume that the frame is the set of all fluent names.

We will identify a state σ of the domain description D in $\mathcal{A}$ with a truth valued function on the set of fluents in the standard way: $\sigma(F) = \mathsf{T}$ iff $F \in \sigma$. We will also identify any sequence of actions $A_1; \ldots; A_n$ in $\mathcal{A}$ with the string of actions $A_1 \ldots A_n$ in $\mathcal{AR}^-$.

Lemma 4.6 *Let D be a domain in the language $\mathcal{A}$ and let Φ be the transition function of any model of D. For any state σ and action name A, $Res(A, \sigma) = \{\Phi(A, \sigma)\}$.*

Proof. Fix σ and A. Let us first check that $\Phi(A, \sigma) \in Res_0(A, \sigma)$. That is, we need to check that for each effect proposition of the form A **causes** F **if** $P_1 \wedge \ldots \wedge P_n$ in D, if $\sigma(P_1 \wedge \ldots P_n) = \mathsf{T}$, then $\Phi(A, \sigma)(F) = \mathsf{T}$. But this is precisely condition (i) in the definition of a model of $\mathcal{A}$.

To show that $Res(A, \sigma) = \{\Phi(A, \sigma)\}$, it suffices to show that for any state σ and any σ' in $Res_0(A, \sigma)$ different from $\Phi(A, \sigma)$,

$$\rho(\sigma, \Phi(A, \sigma)) \cap \mathbf{Fr} \subset \rho(\sigma, \sigma') \cap \mathbf{Fr}.$$

Since **Fr** is the set of all fluents, it suffices to show that $\rho(\sigma, \Phi(A, \sigma)) \subset \rho(\sigma, \sigma')$.

We will first check that $\rho(\sigma, \Phi(A, \sigma))$ is a subset of $\rho(\sigma, \sigma')$. Let $F \in \rho(\sigma, \Phi(A, \sigma))$. This means that $\Phi(A, \sigma)(F) \neq \sigma(F)$. Assume that $\sigma(F) = \mathsf{T}$. (The case when $\sigma(F) = \mathsf{F}$ is similar). By condition (ii) in the definition of a model of a domain in $\mathcal{A}$, we see that there is an effect proposition in D of the form A **causes** $\neg F$ **if** P, where $\sigma(P) = \mathsf{T}$. By the definition of $Res_0(A, \sigma)$, $\sigma'(F) = \mathsf{F}$. Hence $F \in \rho(\sigma, \sigma')$.

It remains to check that $\rho(\sigma, \Phi(A, \sigma)) \neq \rho(\sigma, \sigma')$. This follows from the easily checked fact that for any states σ, σ' and σ'',

$$\rho(\sigma, \sigma') = \rho(\sigma, \sigma'') \text{ iff } \sigma' = \sigma''.$$

$\square$

Lemma 4.7 *For any model M of a domain D in the sense of $\mathcal{A}$, the structure*

$$\overline{A} \mapsto M^{\overline{A}} \tag{4.15}$$

is a model of D in the sense of $\mathcal{AR}^-$.

Proof. Let Ψ be the structure defined by (4.15). Since M is a model of D, $M^{\overline{A}}$ is defined for each $\overline{A}$ and hence every string $\overline{A}$ is executable in Ψ. Also, it is easy to see that any value proposition is true in M iff it is true in Ψ and hence all the value propositions of the domain are true in Ψ. To verify that Ψ is a model of the domain, it remains to check that $\Psi(\overline{A}; A) \in Res(A, \Psi(\overline{A}))$ for every tuple of action names $\overline{A}$ and every action name A. That is, we need to check that $M^{\overline{A};A} \in Res(A, M^{\overline{A}})$. But this follows from Lemma 4.6. □

Lemma 4.8 *Let D be a consistent domain in the language $\mathcal{A}$. For any model Ψ of D in the sense of $\mathcal{AR}$, there is a unique model M of D in the sense of $\mathcal{A}$ such that Ψ is the function defined as in (4.15).*

Proof. Using Lemma 4.6, it is easy to see that every string $\overline{A}$ is executable in Ψ. Since D is consistent, it has a model (in the sense of $\mathcal{A}$). To define the desired model $M = (\sigma_0, \Phi)$, it suffices to specify σ_0, as Φ is completely determined by the effect propositions of D. We define $\sigma_0 = \Psi(\epsilon)$. Using Lemma 4.6, it is easy to check (using induction) that for any tuple of action names $\overline{A}$, $\Psi(\overline{A}) = M^{\overline{A}}$. Hence, any value proposition is true in M iff it is true in Ψ. In particular, all the value propositions of D are true in M. □

Theorem 4.5 follows from Lemmas 4.7 and 4.8. □

Chapter 5

Two Counterexamples

5.1 Introduction

In this chapter, we describe two limitations of the method of [Baker, 1991] that were uncovered while investigating its applicability to nondeterministic actions: in the presence of such actions, neither does Baker's method allow us to draw some intuitively plausible conclusions nor does it generate the appropriate existence of situations axiom. Note that nondeterministic actions (like the action of tossing a coin) arise naturally in many settings. We suggest that the reason for these shortcomings of Baker's method is the unexpected interference of the axioms encoding "observations" with the minimization.

In the next section, we give an example to illustrate the unexpected difficulties that Baker's method has in dealing with nondeterministic actions. Section 5.3 reviews Baker's proposal to obtain the existence of situations axiom via circumscription and Section 5.4 shows that this proposal does not always work.

5.2 Two Buses

In this section, we will show that in the presence of nondeterministic actions, Baker's method sometimes gives results that are intuitively implausible.

We consider a simple modification of the example from Section 3.5.1. Let us assume that there are two buses that take the commuter to his workplace and that they are of two different colors: one red and the other yellow. To get to work, he takes the first bus that comes by. He can board either bus only if he has a ticket. We consider a particular scenario where he does not have a ticket initially, and after purchasing a ticket and boarding, he is on the red bus.

This example may be formalized in the situation calculus as follows. As before, we have two actions, *BuyTicket* (B) and *GetOnBoard* (G). The fluents are *HasTicket* (H), *OnRedBus* (R), and *OnYellowBus* (Y). The axioms are divided into three groups. The first set of axioms captures the constraints that hold between various fluents in the domain and are called constraint axioms. They are

C.1 $\neg(Holds(R, s) \wedge Holds(Y, s)),$
C.2 $Holds(R, s) \supset Holds(H, s),$
C.3 $Holds(Y, s) \supset Holds(H, s).$

The second group of axioms captures what the various effects of actions are. These "effect axioms" are

E.1 $\neg Holds(H, s) \supset Holds(H, Result(B, s)),$
E.2 $Holds(H, s) \wedge \neg Holds(R, s) \wedge \neg Holds(Y, s) \supset$
$Holds(R, Result(G, s)) \ \vee \ Holds(Y, Result(G, s)).$

The third group of axioms captures information about the initial situation and situations reached by performing actions from the initial situation. These are called "observation axioms". They are

O.1 $\neg Holds(H, S_0),$
O.2 $Holds(R, Result(G, Result(B, S_0))).$

In addition, we add the UNA and DCA for fluents and actions:

$$\begin{aligned} &H \neq R \wedge H \neq Y \wedge R \neq Y, \\ &f = H \vee f = R \vee f = Y, \\ &B \neq G, \\ &a = B \vee a = G. \end{aligned} \tag{5.1}$$

Finally, we add the following existence of situations axiom:

$$\begin{aligned} &\exists s[\neg Holds(H, s) \wedge \neg Holds(R, s) \wedge \neg Holds(Y, s)] \wedge \\ &\exists s[Holds(H, s) \wedge \neg Holds(R, s) \wedge \neg Holds(Y, s)] \wedge \\ &\exists s[Holds(H, s) \wedge Holds(R, s) \wedge \neg Holds(Y, s)] \wedge \\ &\exists s[Holds(H, s) \wedge \neg Holds(R, s) \wedge Holds(Y, s)]. \end{aligned} \tag{5.2}$$

Let us denote the conjunction of the universal closures of axioms **C, E** and **O**, axioms (5.1) and (5.2), and the commonsense law of inertia (3.18) by D_1.

We expect that after executing the action B in the initial state, we will be in a situation where only the fluent H holds, because R and Y were false initially and these fluents persist as a result of performing the action B. However, we will exhibit a model M of **CIRC**$(D_1; Ab; Result, S_0)$ such that the fluents H and R are true after performing the action B. Thus, in this model, buying the ticket magically transports the commuter on board the red bus!

Before presenting the unexpected model M, the reason behind the existence of such an unexpected model can be explained informally as follows. Note that in the absence of the observation axioms, when the action B is executed in a situation where none of the fluents hold, the resulting situation that is most preferred is the one where only the fluent H holds. However, circumscription has to take into account the entire set of axioms (including the observation axioms) while minimizing Ab. In the presence of the observation axioms, there is no reason to prefer the situation in which only H holds over the situation in which both H and R hold. Thus, the observation axioms play an unwanted role, by interfering with the minimization.

Let us now turn to a precise description of the unexpected model. Consider the following structure M. The domains of the variables are defined by the equations

$$\begin{aligned} \mid M \mid_f &= \{H, R, Y\}, \\ \mid M \mid_a &= \{B, G\}, \\ \mid M \mid_s &= \{0, 1, 2, 3\}. \end{aligned}$$

The fluent and action constants are interpreted as themselves and the situation constant S_0 is interpreted as 0. The interpretations of the predicates *Holds* and *Ab* are as follows:

$$\begin{aligned} \mathit{Holds}^M &= \{\langle H, 1\rangle, \langle H, 2\rangle, \langle H, 3\rangle, \langle R, 2\rangle, \langle Y, 3\rangle\}, \\ \mathit{Ab}^M &= \{\langle H, B, 0\rangle, \langle R, B, 0\rangle, \langle Y, G, 1\rangle\}. \end{aligned}$$

The function $Result$ is defined as follows:

$$\begin{aligned} Result^M(B, 0) &= 2, \\ Result^M(G, 1) &= 3. \end{aligned}$$

Otherwise, $Result^M(A, i) = i$. (Here A and i range over $\mid M \mid_a$ and $\mid M \mid_s$ respectively).

Figure 5.1 is a pictorial representation of M. Here, the circles denote the situations, the symbols inside a circle represent the fluents that hold in that situation, and the arrows represent the $Result$ function.

It is easy to check that M is a model of D_1. We can show the following.

Theorem 5.1 *M (as defined above) is a model of* **CIRC**$(D_1; Ab; Result, S_0)$.

We will now show that in the presence of nondeterministic actions, Baker's approach fails to generate the proper existence of situations axiom. To do that, let us now examine Baker's proposal to obtain the existence of situations axiom via circumscription.

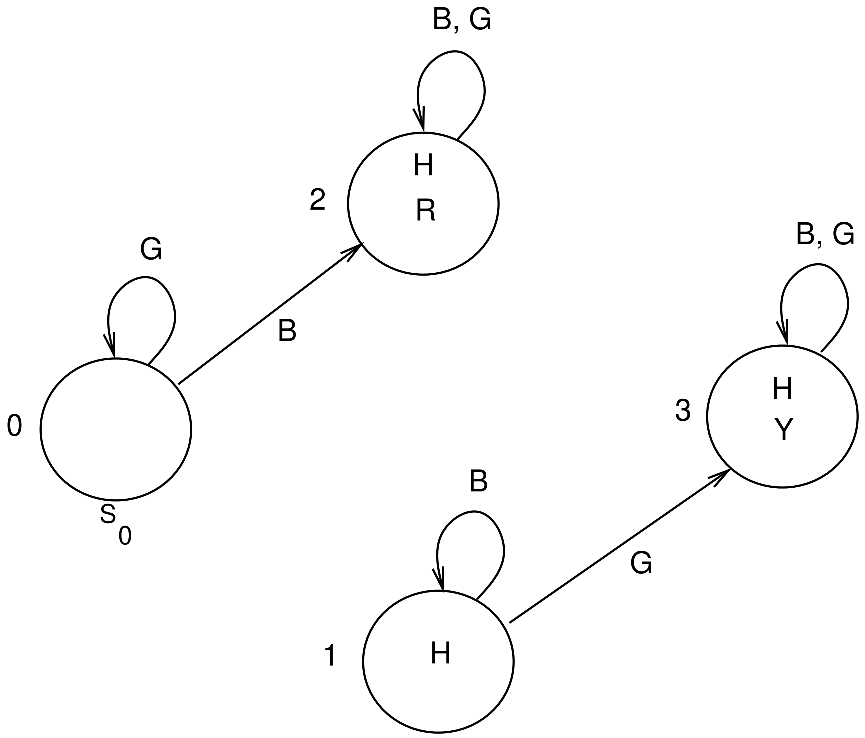

Figure 5.1: An Unexpected Model in the Bus Example.

5.3 General Existence of Situations Axiom

Recall that when we enhanced the example in Section 3.5.1 by adding the constraint (3.20), we had to change the existence of situations axiom (3.19) to (3.21) to reflect that change. Making all such changes into the existence of situations axiom by hand is clearly not satisfactory, since updating the axiom would become a significant task as the constraints change.

To deal with this issue, Baker proposed to derive the existence of situations axiom, using circumscription, from domain constraints and from a general default version of the existence of situations principle. The default version expresses that normally, corresponding to every subset of the fluents, there exists a situation in which precisely these fluents hold. To help express quantification over subsets of fluents, we introduce unary predicate variables over fluents. We will have occasion to use only one such variable and we denote it by h. To obtain situations corresponding to different unary predicates on the fluents, a function *Sit* from unary predicates to situations is introduced. Since the existence of situations principle only holds by default (note that domain constraints restrict the applicability of this default), an abnormality predicate $Absit$ on fluent predicates is introduced.

The default version of the existence of situations principle can now be expressed by writing an axiom saying that normal fluent predicates satisfy the existence of situations principle. Formally,

$$\neg Absit(h) \supset [Holds(f, Sit(h)) \equiv h(f)]. \tag{5.3}$$

The idea is that by circumscribing $Absit$, we would get that h is in the extent of $Absit$ iff the set of all fluents f such that $h(f)$ holds violates some constraint. In other words, for each subset T of the set of fluents consistent with the set of constraints, there is a situation where precisely those fluents in T hold. In order to ensure that this circumscription works properly, Baker includes the following unique names axiom for Sit:

$$Sit(h_1) = Sit(h_2) \supset h_1 = h_2. \tag{5.4}$$

For the example with the single domain constraint (3.20) given in Section 3.5.1, let us see how to apply this idea. Let us denote by V the conjunction of the universal closures of the axioms (3.16), (3.18), (3.20), (5.3), (5.4) and the UNA and DCA for fluents[1] and actions. Then Baker suggests considering the theory **CIRC**$(V; Absit; Ab, Holds, Result, S_0)$. Note that this theory turns out to be a conservative extension of the theory obtained by adding the formula (3.21).

5.4 Coin Tossing

This section presents an example which shows that, in the presence of nondeterministic actions, circumscription of $Absit$ does not always generate the existence of situations axiom that we intuitively expect.

Consider the action A of tossing a coin. Let the fluents be H and T (corresponding to heads and tails). The action of tossing the coin can be formalized by the following effect axiom:

$$Holds(H, Result(A, s)) \vee Holds(T, Result(A, s)). \tag{5.5}$$

We also have the constraint that we cannot have both the fluents H and T holding at the same time. This is expressed as the following axiom:

$$\neg(Holds(H, s) \wedge Holds(T, s)). \tag{5.6}$$

The following set of axioms give the outcome in a series of tosses starting from the initial situation. As before, we call these the observation axioms. Let us first introduce some

[1]As noted before, the UNA and DCA of fluents have to be modified to take into account the new fluent $WellLighted$.

notation. Let $[A^m]$ denote the ground term obtained by applying the $Result$ function (with A as the first argument) m times to S_0.

Using this notation, the observation axioms are:

$$\begin{array}{ll} (a) & Holds(H, S_0) \wedge \neg Holds(T, S_0) \\ (b) & Holds(H, [A]) \wedge \neg Holds(T, [A]), \\ (c) & \neg Holds(H, [A^2]) \wedge Holds(T, [A^2]) \\ (d) & Holds(H, [A^3]) \wedge \neg Holds(T, [A^3]) \\ (e) & Holds(H, [A^4]) \wedge \neg Holds(T, [A^4]) \\ (f) & Holds(H, [A^5]) \wedge \neg Holds(T, [A^5]) \end{array} \tag{5.7}$$

Finally, we add the UNA and DCA for fluent and actions:

$$\begin{array}{l} H \neq T, \\ f = H \vee f = T, \\ a = A. \end{array} \tag{5.8}$$

Let D_2 denote the conjunction of universal closures of axioms (5.5)–(5.8), the commonsense law of inertia (3.18), the axiom for $Absit$ (5.3) and the UNA for $Absit$ (5.4). Following Baker, we consider

$$\textbf{CIRC}(D_2; Absit; Ab, Holds, Result, S_0). \tag{5.9}$$

For this domain, we would expect that the above circumscription of $Absit$ would guarantee the existence of (at least) three situations, corresponding to the three possible assignments of truth values to the set $\{H, T\}$ satisfying the domain constraint (5.6). That is, we expect that (5.9) entails the following existence of situations axiom:

$$\begin{array}{l} \exists s[\neg Holds(H, s) \wedge \neg Holds(T, s)] \wedge \\ \exists s[\neg Holds(H, s) \wedge Holds(T, s)] \wedge \\ \exists s[Holds(H, s) \wedge \neg Holds(T, s)]. \end{array} \tag{5.10}$$

We will now show that this is not the case. The intuition behind this failure to obtain the existence of situations corresponding to *all* fluent combinations (consistent with the constraints) can be explained as follows. In a domain with nondeterministic actions, we may need more than one situation corresponding to each fluent combination—for instance, to satisfy the observation axioms. This leads to the possibility that there are some fluent combinations that do not have any corresponding situation.

We prove the following theorem:

Theorem 5.2 *There exists a model of* **CIRC**$(D_2; Absit; Ab, Holds, Result, S_0)$ *where the sentence* $\exists s[\neg Holds(H, s) \wedge \neg Holds(T, s)]$ *is false.*

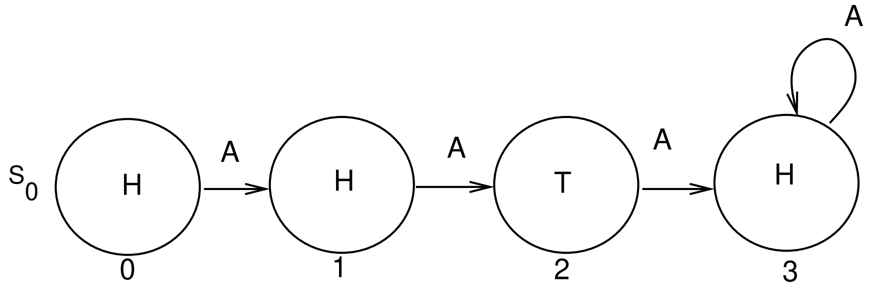

Figure 5.2: An Unexpected Model in the Coin Tossing Example.

Figure 5.2 gives a pictorial representation of a model of D_2 (A precise description is given in the proof of Lemma 5.1.) This model is, in fact, a model of **CIRC**$(D_2; Absit; Ab, Holds, Result, S_0)$. However, verifying this is cumbersome and hence we give a different proof.

5.5 Discussion

Note that in both of the counterexamples, there was at least one action whose effects were nondeterministic—it was the action G in the two-bus example and the action A in the coin tossing example. Also, both counterexamples involve reasoning backward in time. The counterexamples thus show that when we stray outside the realm of deterministic and temporal projection domains, Baker's method may not allow us to justify some intuitively plausible results. The question now arises as to whether it is possible to modify the method to handle wider classes of domains (for instance, those that include the examples) so that we again get results that are in tune with our intuition.

Note that in both examples, the observation axioms play an unwanted role, by interfering with the minimization. Intuitively, as more observation axioms are added to a theory, the class of models of the theory should get smaller monotonically. That is, observation axioms should serve only to rule out some models, never to include new ones. This property seems quite desirable in view of the similar use made of observations by humans in many everyday situations. Both examples do not have this *restricted monotonicity* property [Lifschitz, 1993c]. This suggests that a way out is to apply circumscription not to the whole theory of action, but to the theory of action minus the observation axioms, and then to include the observation axioms. Such a method of excluding the observations from the minimization was proposed by Sandewall [1989] in the context of chronological minimization [1988] and by Crawford and Etherington [Crawford and Etherington, 1992] in the context of avoiding interference between minimizations solving the frame and the qualification problems.

Let us now see how we can apply this idea to the examples. For the two-bus example, we can show that

$$\textbf{CIRC}(D_1 \setminus \{\textbf{O.1, O.2}\}; Ab; Result, S_0) \cup \{\textbf{O.1, O.2}\} \models Holds(H, Result(B, S_0)) \wedge \\ \neg Holds(R, Result(B, S_0)) \wedge \neg Holds(Y, Result(B, S_0)).$$

Similarly, for the coin tossing example, we can show that

$$\textbf{CIRC}(D_2 \setminus (5.7); Absit; Ab, Holds, Result, S_0) \cup (5.7) \models (5.10).$$

Thus, it seems that the flexibility for excluding certain parts of a theory from the scope of the circumscription operator is quite useful in applications of circumscription to reasoning about actions. The framework of *nested abnormality theories* (NATs)[Lifschitz, 1995], by limiting the applicability of various circumscriptions to subsets of the theory under consideration, allows this kind of flexibility. In the next chapter, we present a quick review of NATs and then present a sound and complete translation from $\mathcal{AR}^-$ into NATs.

The two counterexamples were first reported in [Kartha, 1994].

5.6 Proofs

5.6.1 Proof of Theorem 5.1

We need to show that M as defined in Section 5.2 is a model of $\textbf{CIRC}(D_1; Ab; Result, S_0)$. Assume that M is not a minimal model with respect to $\leq^{Ab;Result,S_0}$. Then there is a model M' identical to M except (possibly) in the interpretations of $Result$ and S_0 and where the extent of Ab is strictly smaller than its extent in M. We will show that assuming that such an M' exists leads to a contradiction.

Since M' is also a model of the observation axiom **O.1** and the constraint axioms **C.2** and **C.3**, we get $S_0^{M'} = 0$ $(= S_0^M)$. Also, for each action A and situation i such that $Result^M(A, i) = i$, $Result^{M'}(A, i) = i$, for otherwise, by the commonsense law of inertia, $Ab^{M'}$ will contain some elements not in Ab^M. Hence, M and M' can differ in the interpretation of $Result$ only in how $Result(B, 0)$ and $Result(G, 1)$ are defined in each.

There are three possible values $Result^{M'}(B, 0)$ can take, while still satisfying the first effect axiom **E.1**, namely $1, 2$ or 3. Similarly, $Result^{M'}(G, 1)$ could possibly be defined as 2 or 3, while satisfying **E.2**. But if $Result^{M'}(B, 0) = 3$, then the observation axiom **O.2** will not be satisfied. If $Result^{M'}(G, 1) = 2$, then $\langle R, G, 1\rangle \in Ab^{M'}$ and hence $Ab^{M'}$ is not a subset of Ab^{M}. Thus there are only two cases to be considered. The first case, where $Result^{M'}(B, 0) = 2$ and $Result^{M'}(G, 1) = 3$ implies by the commonsense law of inertia that Ab^{M} is a subset of $Ab^{M'}$. The second case, where $Result^{M'}(B, 0) = 1$ and $Result^{M'}(G, 1) = 3$ does not satisfy the observation axiom **O.2**. □

5.6.2 Proof of Theorem 5.2

Lemma 5.1 *There exists a finite model of D_2 where the cardinality of the universe of situations is four.*

Proof. Define a structure M as follows:

$$\begin{array}{lcl} \mid M \mid_f &=& \{H, T\}, \\ \mid M \mid_a &=& \{A\}, \\ \mid M \mid_s &=& \{0, 1, 2, 3\}. \end{array}$$

The fluent, action and value constants are interpreted as themselves. The situation constant S_0 is interpreted as 0. Let us agree to identify a unary predicate on the fluents by the subset of the fluents for which the predicate is true. Thus, let $\sigma_0, \sigma_1, \sigma_2, \sigma_3$ denote respectively the following four predicates $\emptyset$, $\{H\}$, $\{T\}$, $\{H, T\}$. Now, we can give the interpretations of the predicates of D_2 as follows:

$$\begin{array}{lcl} \mathit{Holds}^M &=& \{\langle H, 0\rangle, \langle H, 1\rangle, \langle T, 2\rangle, \langle H, 3\rangle\}, \\ \mathit{Ab}^M &=& \{\langle H, A, 1\rangle, \langle H, A, 2\rangle, \langle T, A, 1\rangle, \langle T, A, 2\rangle\}, \\ \mathit{Absit}^M &=& \{\sigma_0, \sigma_3\}. \end{array}$$

We can give the interpretations of the functions of D_2 as follows:

$$\begin{array}{l} Result^M(A, 0) = 1, \\ Result^M(A, 1) = 2, \\ Result^M(A, 2) = 3, \\ Result^M(A, 3) = 3, \\ Sit^M(\sigma_i) = i \ \ (0 \le i \le 3). \end{array}$$

It is easy to check that M is a model of D_2. □

Figure 5.2 gives a pictorial representation of this model.

Lemma 5.2 . *For any model M' of D_2, there are at least four distinct situations in the universe of situations such that in each such situation t, either*
$\langle H, t\rangle \in Holds^{M'} \wedge \langle T, t\rangle \notin Holds^{M'}$ or $\langle H, t\rangle \notin Holds^{M'} \wedge \langle T, t\rangle \in Holds^{M'}$.

Proof. By the UNA and DCA for fluents and actions, we can assume that the universes of fluents and actions in M' are identical to those in M (where M is defined as in the proof of Lemma 5.1). Let s_0 be the element of the universe of situations of M' such that $s_0 = S_0^{M'}$. Since M' is a model of the observation axiom (5.7a), we see that
$\langle H, s_0\rangle \in Holds^{M'} \wedge \langle T, s_0\rangle \notin Holds^{M'}$.

Now, $Result^{M'}(A, s_0) \neq s_0$, since otherwise M' will not be a model for the observation axiom (5.7c). Denote $Result^{M'}(A, s_0)$ by s_1. Since M' is a model of (5.7b), $\langle H, s_1\rangle \in Holds^{M'} \wedge \langle T, s_1\rangle \notin Holds^{M'}$.

Again, since M' is a model for (5.7c), $Result^{M'}(A, s_1) \neq s_0$ and $Result^{M'}(A, s_1) \neq s_1$. Denoting $Result^{M'}(A, s_1)$ by s_2, we get that $\langle H, s_1\rangle \notin Holds^{M'} \wedge \langle T, s_1\rangle \in Holds^{M'}$.

Similarly, using the observation axioms (5.7d)–(5.7f), we conclude that $Result^{M'}(A, s_2)$ is not equal to s_0, s_1 or s_2. Denoting $Result^{M'}(A, s_2)$ by s_3, we get that $\langle H, s_3\rangle \in Holds^{M'} \wedge \langle T, s_3\rangle \notin Holds^{M'}$. □

Proof of Theorem 5.2. By Lemma 5.1, there exists a finite model of D_2 where the cardinality of the universe of situations is four. Since the existence of a finite model of D_2 implies the existence of a finite model of the circumscription with identical universes for the three sorts, we see that there is a model M^* of **CIRC**$(D_2; Absit; Ab, Holds, Result, S_0)$ where also the cardinality of the universe of situations is four. Since M^* is a model of D_2, we can use Lemma 5.2 to conclude that there are at least four distinct situations in the universe of situations of M^* such that in each such situation t, either $\langle H, t\rangle \in Holds^{M^*} \wedge \langle T, t\rangle \notin Holds^{M^*}$ or $\langle H, t\rangle \notin Holds^{M^*} \wedge \langle T, t\rangle \in Holds^{M^*}$. Since the universe of situations of M^* is of cardinality four, we see that $M^* \not\models \exists s[\neg Holds(H, s) \wedge \neg Holds(T, s)]$. □

Chapter 6

A Sound and Complete Translation from $\mathcal{AR}^-$

6.1 Introduction

In this chapter, we present a sound and complete translation from $\mathcal{AR}^-$. As noted in the previous chapter, we need to be able to exclude certain parts of the theory from the scope of the circumscription operator. We will use the formalism of nested abnormality theories [Lifschitz, 1995] (NATs) for this purpose.

In Section 6.2, we briefly discuss the formalism of nested abnormality theories and in in Section 6.3, we present a sound and complete translation from $\mathcal{AR}^-$ to nested abnormality theories.

6.2 Nested Abnormality Theories

The formalism of nested abnormality theories is introduced in [Lifschitz, 1995]. Its use is demonstrated there by recasting several familiar applications of circumscription in its framework—inheritance hierarchies, the domain closure assumption, and the causal minimization approach to the frame problem.

The difference between this formalism and earlier uses of circumscription for formalizing knowledge can be characterized as follows. A "circumscriptive theory" is usually defined by a list of axioms $A_1, \ldots, A_n$ that may contain the abnormality predicate *Ab*, and by a list of predicate and/or function constants[1] $C_1, \ldots, C_m$ that are "described" by the axioms and thus are allowed to vary in the process of circumscribing *Ab* [McCarthy, 1986].

[1] Object constants are viewed as function constants of arity 0.

Such theories may be represented in the following syntactic form:

$$C_1, \ldots, C_m \;:\; A_1, \ldots, A_n. \tag{6.1}$$

The expression (6.1) can be translated into the language of classical logic by forming the circumscription of the abnormality predicate *Ab* relative to the conjunction of the axioms $A_1 \wedge \ldots \wedge A_n$ with $C_1, \ldots, C_m$ allowed to vary; we denote this circumscription by

$$\textbf{CIRC}[A_1 \wedge \ldots \wedge A_n; Ab; C_1, \ldots, C_m]. \tag{6.2}$$

Unfortunately, this is not general enough for the purpose of representing defaults, and "prioritized circumscription" is proposed in [McCarthy, 1986] as a more general representation tool. In nested abnormality theories, we generalize (6.1) in a different way: each A_i in (6.1) is allowed to be a "block" of form (6.1), so that axioms become "nested." It is also convenient to replace the predicate constant *Ab* in (6.2) by an existentially quantified predicate variable. This is motivated by the fact that the abnormality predicate usually plays an auxiliary role in a formalization; what we are actually interested in are the logical consequences of (6.2) that do not include *Ab*. For the formal definition of the syntax and semantics of nested abnormality theories, the reader is referred to the Appendix.

6.3 Translating from $\mathcal{AR}^-$ into NATs

In this section, we show how to transform a domain description in $\mathcal{AR}^-$ into a nested abnormality theory. We use some ideas from [Baker, 1991] in the translation. Note however that there are several differences from the formalization in [Baker, 1991]—for instance, the statement of the existence of situations principle is simplified here. We restrict attention to *finite* domain descriptions, that is, to domain descriptions with finitely many fluents, actions and propositions.

6.3.1 Language

We will transform a finite domain D in the sense of the language $\mathcal{AR}^-$ into a nested abnormality theory whose language L has variables of three sorts: for fluents, actions and situations. These variables will be denoted respectively by $f, f_1, f_2, \ldots$; $a, a_1, a_2, \ldots$; $s, s_1, s_2, \ldots$. In addition, L has predicate variables $\lambda, \lambda_1, \lambda_2, \ldots$ that range over properties of fluents.

The language has the following object constants:

- the fluents of D,
- the actions of D,

- the situation constants S_0 and $\perp$.

It also includes the binary function constant $Result$, which takes an action and a situation as arguments, and produces a situation.

The constant $\perp$ is needed because we want to talk about actions that may not be executable, such as *InsertCard* in Example 1 from Chapter 4. Intuitively, $\perp$ represents the value "undefined", so that the assertion that a is not executable in situation s will be represented by

$$Result(a, s) = \perp.$$

Finally, L has two predicate constants:

- the unary predicate $FrameFluent$, whose argument is a fluent,
- the binary predicate $Holds$, whose two arguments are a fluent and a situation.

Note that "formulae" as defined in Section 4.2 are not among the formulae of L. To avoid confusion, we will refer to formulae in the sense of Section 4.2 as "domain formulae." For any domain formula C, by $T(C, s)$ we will denote the formula of L obtained from C as the result of replacing each fluent name F by $Holds(F, s)$. For instance, $T(\neg \mathit{Alive} \wedge \mathit{Loaded}, s)$ stands for

$$\neg Holds(\mathit{Alive}, s) \wedge Holds(\mathit{Loaded}, s).$$

For any string of actions $A_1 \ldots A_m$, by $[A_1 \ldots A_m]$ we will denote the ground term

$$Result(A_m, Result(A_{m-1}, \ldots, Result(A_1, S_0), \ldots)).$$

6.3.2 Law of Inertia and Existence of Situations

Our translation β from $\mathcal{AR}^-$ into nested abnormality theories uses two defaults that play an important role in applications of default reasoning to the frame problem.

According to the *commonsense law of inertia*, the value of a frame fluent normally remains unchanged after performing an action. Formulae describing how actions change the world represent exceptions to this default. The commonsense law of inertia is expressed by the formula

$$FrameFluent(f) \wedge \neg \mathit{Ab}(f, a, s) \supset [Holds(f, Result(a, s)) \equiv Holds(f, s)].$$

This formula will be denoted by *LI*.

According to the *existence of situations principle*, for any assignment of values to fluents, there normally exists a situation in which every fluent takes the assigned value. Recognizing the role of this principle was a major contribution of [Baker, 1991]. Constraints represent exceptions to this default. For instance, the constraint

$$Holds(\mathit{Walking}, s) \supset Holds(\mathit{Alive}, s) \tag{6.3}$$

shows that there is no situation in which *Walking* is true and *Alive* is false. The existence of situations principle is expressed by the formula

$$\neg \mathit{Ab}(\lambda) \supset \exists s[s \neq \bot \wedge \forall f(Holds(f, s) \equiv \lambda(f))].$$

This formula will be denoted by *ES*.

As defined below, the nested abnormality theory βD representing a finite domain D consists of several formulae, including *LI* and *ES*, arranged into a system of blocks. We should not worry about the fact that *Ab* has three arguments in *LI* and only one argument in *ES*: these formulae will appear in two different blocks.

6.3.3 Translating Propositions

We will define how to construct, for each proposition P in D, the corresponding formula βP. These formulae will be included in the translation βD of D.

If P is a value proposition C **after** $\overline{A}$, then βP is

$$[\overline{A}] \neq \bot \wedge\ T(C, [\overline{A}]).$$

The first term expresses that the sequence of actions $\overline{A}$ is executable. For instance, the proposition

Heads **after** *Toss*

is translated as

$$Result(\mathit{Toss}, S_0) \neq \bot \wedge\ Holds(\mathit{Heads}, Result(\mathit{Toss}, S_0)).$$

If P is an effect proposition A **causes** C **if** P, then βP is

$$T(P, s) \wedge Result(A, s) \neq \bot \ \supset\ T(C, Result(A, s)).$$

For instance, the proposition

Shoot **causes** $\neg$*Alive* **if** *Loaded*

is translated as

$$Holds(\mathit{Loaded}, s) \wedge Result(Shoot, s) \neq \bot \ \supset \\ \neg Holds(\mathit{Alive}, Result(\mathit{Shoot}, s)).$$

Finally, if P is a constraint **always** C, then βP is $T(C, s)$. For instance, **always** *Walking* $\supset$ *Alive* is translated as (6.3).

6.3.4 Definition of βD

Now we are ready to define the representation βD of a finite domain D as a nested abnormality theory.

By **F** we will denote the set of fluents in the language of D, by **Fr** the set of frame fluents, by **A** the set of actions. Furthermore, let D_v, D_e and D_c be the parts of D consisting of its value propositions, effect propositions and constraints, so that

$$D = D_v \cup D_e \cup D_c.$$

The axioms of βD are as follows.
Group 1. Unique names axioms:

$$F_1 \neq F_2$$

for all pairs of distinct fluents $F_1, F_2 \in \mathbf{F}$,

$$A_1 \neq A_2$$

for all pairs of distinct actions $A_1, A_2 \in \mathbf{A}$, and

$$S_0 \neq \bot.$$

Group 2. Domain closure axioms:

$$\bigvee_{F \in \mathbf{F}} f = F,$$

$$\bigvee_{A \in \mathbf{A}} a = A.$$

Group 3. The universe of situations is assumed to be sufficiently large:

$$\exists s_1, \ldots, s_N \bigwedge_{1 \leq i < j \leq N} s_i \neq s_j,$$

where $N = 2^{|\mathbf{F}|} + 1$.
Group 4. Translations of the value propositions:

$$\beta P \qquad (P \in D_v).$$

Group 5. Characterization of $FrameFluent$:

$$\begin{array}{l} \{FrameFluent\ : \\ \quad FrameFluent(f) \supset Ab(f), \\ \quad FrameFluent(F) \qquad (F \in \mathbf{Fr}) \\ \ \}. \end{array}$$

By enclosing the formulae $FrameFluent(F)$ for all F in **Fr** in a block along with

$$FrameFluent(f) \supset Ab(f),$$

we express the "closed world assumption" for FrameFluent: a fluent satisfies this predicate *only* if it represented by a constant from **Fr**.

Group 6. Characterization of $Holds$ and $Result$:

$$\begin{array}{l}
\{Result\ : \\
\quad \mathit{LI}, \\
\quad \{Result\ : \\
\quad\quad Result(a,s) = \bot \supset Ab(a,s), \\
\quad\quad Result(a,\bot) = \bot, \\
\quad\quad \beta P \qquad (P \in D_e), \\
\quad\quad \{Holds\ : \\
\quad\quad\quad \mathit{ES}, \\
\quad\quad\quad \beta P \qquad (P \in D_c) \\
\quad\quad \} \\
\quad \} \\
\}.
\end{array}$$

The innermost block represents the basic assumption about the universe of situations: for any assignment of truth values to fluents, there exists a situation in which every fluent takes the assigned value, unless such a situation is declared impossible by one of the constraints in D. In the intermediate block, the formula

$$Result(a,s) = \bot \ \supset Ab(a,s)$$

expresses that, normally, an action is executable. Thus the intermediate block tells us that an action can be executed unless this is prohibited by the propositions in D. The outermost block says that the values of fluents do not change unless this is required by the propositions in D.

The nesting of blocks reflects our intention to decide first what kinds of situations exist, then which actions can be executed, and then what the effects of actions are.

6.3.5 Soundness and Completeness

Consider a finite domain D. Let Ψ be a model of D, and let M be a model of βD. We say that M is *similar* to Ψ if, for every value proposition P, P is true in Ψ if and only if M satisfies βP.

Theorem 6.1 *Let D be a finite domain. For any model Ψ of D, there exists a model M of βD similar to Ψ. For any model M of βD, there exists a model Ψ of D such that M is similar to Ψ.*

The following corollary expresses the soundness and completeness of the translation.

Corollary. *For any finite domain D and any value proposition P, βD entails βP if and only if D entails P.*

6.4 Discussion

We showed in Chapter 5 that Baker's method has difficulties dealing with nondeterministic actions. Are there classes of problems for which Baker's method (without modification) works correctly? We will now define two important classes of domains for which this question can be answered in the affirmative.

A domain D is a *temporal projection domain* if every value proposition in D is of the form **initially** C. For instance, the domains of Examples 1–4 from Section 4.3 are temporal projection domains. Obviously, the domains that contain no value propositions are temporal projection domains.

A domain D is *deterministic* if, for every action A and every state σ, $Res(A, \sigma)$ is a singleton. For instance, the domains of Examples 2 and 3 from Section 4.3 are deterministic.

For these two classes of domains, it can be shown that there is a sound and complete translation into Baker's method. The proof, though tedious, is similar to the proof of Theorem 6.1 and is omitted.

The treatment of value propositions in the semantics of $\mathcal{AR}^-$ is analogous to the "filtering" method advocated by Sandewall [1989], [1992]. Indeed, the value propositions included in D do not affect the construction of the corresponding transition function *Res*; they are taken into account only when we define which of the "trajectories" described by this transition function are counted as the models of D. Similarly, in the nested abnormality theory βD, the translations of the value propositions form a separate group of axioms that is not placed in the range of any circumscriptions.

The use of nested abnormality theories, instead of conjunctions of circumscriptions as in [Baker, 1991] and [Lifschitz, 1991], leads to a simpler and more natural presentation of Baker's method. Moreover, as shown by the examples from Chapter 5, the use of conjunctions of circumscriptions does not seem to lead to satisfactory results when actions can be nondeterministic.

6.5 Proofs

The following lemma about domains in $\mathcal{AR}^-$ will be useful later.

Lemma 6.1 *For a domain in $\mathcal{AR}^-$ with a finite number of fluent names, for any action A and state σ,*

$$Res_0(A, \sigma) = \emptyset \;\Leftrightarrow\; Res(A, \sigma) = \emptyset.$$

Proof. The left to right direction follows easily since $Res(A, \sigma)$ is a subset of $Res_0(A, \sigma)$. We will prove the contrapositive of the right to left direction. Assume that $Res_0(A, \sigma) \neq \emptyset$. Since the domain has only a finite number of fluent names, for any σ' in $Res_0(A, \sigma)$, the cardinality of $\rho(\sigma, \sigma') \cap \mathbf{Fr}$ is finite. Choose a state σ' in $Res_0(A, \sigma)$ for which the cardinality of $\rho(\sigma, \sigma') \cap \mathbf{Fr}$ is minimal. Clearly, σ' is an element of $Res(A, \sigma)$ and thus $Res(A, \sigma) \neq \emptyset$. □

6.5.1 Proof of Theorem 6.1

Recall that for a block $\{C_1, \ldots, C_m : A_1, \ldots, A_n\}$, $F(\mathit{Ab})$ denotes the formula

$$F(\mathit{Ab}) = \mathbf{CIRC}[\varphi A_1 \wedge \ldots \wedge \varphi A_n; \mathit{Ab}; C_1, \ldots, C_m].$$

When the block is referred to by a name, say B, we will denote this formula by $F_B(\mathit{Ab})$.

For any domain D, let $\mathit{IsState}^D(\lambda)$ be the formula obtained as follows:

1. For each constraint **always** C of the domain D, form the formula obtained from C by replacing in it each fluent F by $\lambda(F)$.
2. Take the conjunction of all such formulae obtained in step 1.

Informally, $\mathit{IsState}^D(\lambda)$ says that λ obeys the domain constraints. When the domain under consideration is clear from the context, we will drop the superscript D. Let *Block 1* denote the innermost block of Group 6 set of axioms of βD.

Lemma 6.2 *Assume that the set of constraints is consistent. Then, the UNA and DCA for fluents and the Group 3 axiom entail that $F_{Block\ 1}(\mathit{Ab})$ is equivalent to the conjunction of the axioms of Block 1 and the formula*

$$\forall \lambda \; (\mathit{Ab}(\lambda) \equiv \neg IsState(\lambda)).$$

Proof. Let B denote the conjunction of the UNA and DCA for fluents and the Group 3 axiom. Let $A(Ab, Holds)$ denote the conjunction of the axioms in *Block 1*. We apply Proposition 5 from the Appendix with E chosen as the expression $\neg IsState(\lambda)$.

Let us first show that B entails

$$A(Ab, Holds) \supset \forall\lambda(\neg IsState(\lambda) \supset Ab(\lambda)).$$

That is, we need to show that B and $A(Ab, Holds)$ entail

$$\forall\lambda(\neg Ab(\lambda) \supset IsState(\lambda)).$$

Now, assume B, $A(Ab, Holds)$ and $\neg Ab(\lambda)$. By the axiom ES of $A(Ab, Holds)$, we see that

$$\exists s[s \neq \perp \wedge \forall f(Holds(f, s) \equiv \lambda(f)]. \tag{6.4}$$

Consider any constraint **always** C of the domain D. Note that the formula $T(C, s)$ is a part of the formulae $\beta(P)(P \in D_c)$ of $A(Ab, Holds)$. From $T(C, s)$ and (6.4), the formula obtained from C by replacing in it each fluent F by $\lambda(F)$ follows. Since $IsState(\lambda)$ is the conjunction of all such formulae, this concludes the proof.

Let us now show that B entails

$$A(Ab, Holds) \supset \exists H \quad A(E, H). \tag{6.5}$$

Our first task is to define H. First of all, observe that the DCA for fluents and the Group 3 axiom (that guarantees that the universe of situations is sufficiently large) entail the following formula:

$$\exists g[\forall\lambda\exists s(s = g(\lambda) \wedge s \neq \perp) \wedge \forall\lambda_1\lambda_2(g(\lambda_1) = g(\lambda_2) \supset \lambda_1 = \lambda_2)],$$

where g is a variable for a function that maps properties of fluents to situations. Choose such a function g. Since we assume that the set of constraints is consistent, there is a λ^* such that $IsState(\lambda^*)$ holds. Define H as

$$\begin{array}{rcl} H(f, s) & = & [\exists\lambda(IsState(\lambda) \wedge s = g(\lambda) \wedge \lambda(f))] \vee \\ & & [\neg\exists\lambda(IsState(\lambda) \wedge s = g(\lambda)) \wedge \lambda^*(f)]. \end{array}$$

We are now ready to check (6.5). Note that $A(E, H)$ is the conjunction of formula

$$IsState(\lambda) \supset \exists s[s \neq \perp \wedge \forall f(H(f, s) \equiv \lambda(f))]. \tag{6.6}$$

and the formulae $T'(C, s)$ for each constraint **always** C, where $T'(C, s)$ denotes the formula obtained by replacing each fluent name F in C by $H(F, s)$.

Let us first check that B and $A(Ab, Holds)$ entail (6.6). Assume $IsState(\lambda_1)$ for some λ_1 and define s_1 as $g(\lambda_1)$. We now need to check that

$$s_1 \neq \perp \wedge \forall f(H(f, s_1) \equiv \lambda_1(f)) \tag{6.7}$$

That $s_1 \neq \perp$ follows from the definition of g. By the DCA for fluents, checking that the second conjunct in (6.7) holds reduces to checking for each $F \in \mathbf{F}$ that $H(F, s_1) \equiv \lambda_1(F)$.

Fix an arbitrary $F \in \mathbf{F}$. The formula $H(F, s_1)$ is

$$[\exists\lambda(IsState(\lambda) \wedge s_1 = g(\lambda) \wedge \lambda(F))] \vee$$
$$[\neg\exists\lambda(IsState(\lambda) \wedge s_1 = g(\lambda)) \wedge \lambda^*(F)]\,.$$

Note that the second disjunct of the above formula can be dropped (choose λ_1 as the desired λ). Also, the first disjunct reduces to

$$IsState(\lambda_1) \wedge s_1 = g(\lambda_1) \wedge \lambda_1(F), \tag{6.8}$$

because $s_1 = g(\lambda_1)$ and $s_1 = g(\lambda_2)$ yields $\lambda_1 = \lambda_2$ by the definition of g. The formula (6.8) reduces to $\lambda_1(F)$ in the presence of the first two conjuncts. This shows that B and $A(Ab, Holds)$ entail (6.6).

We now need to check that B and $A(Ab, Holds)$ entails $T'(C, s)$ for each constraint **always** C, where $T'(C, s)$ denotes the formula obtained by replacing each fluent name F in C by $H(F, s)$. Note that from the definition of H, it follows that B entails the formula

$$\forall f\ s\exists\lambda[IsState(\lambda) \wedge H(f, s) \equiv \lambda(f)]$$

Using this fact and the definition of $IsState(\lambda)$, the desired conclusion follows easily. □

Corollary. *Assume that the constraints are consistent. The UNA and DCA for fluents and the Group 3 axiom entail that $\varphi(Block\ 1)$ is equivalent to the conjunction of the translation of the constraints and the formula*

$$IsState(\lambda) \supset \exists s[s \neq \perp \wedge \forall f(Holds(f, s) \equiv \lambda(f))]. \tag{6.9}$$

For any state σ of the domain D, let $T(\sigma, s)$ be defined as

$$T(\sigma, s) = \bigwedge_{\sigma(F)=\mathsf{T}} Holds(F, s) \wedge \bigwedge_{\sigma(F)=\mathsf{F}} \neg Holds(F, s).$$

Let

$$\Sigma = \{\langle A, \sigma\rangle \mid Res_0(A, \sigma) = \emptyset\}.$$

Define G as the following predicate expression:

$$G = \lambda a\ s[s = \perp \vee \bigvee_{\langle A,\sigma\rangle\in\Sigma} (a = A \wedge T(\sigma, s))].$$

Let the intermediate subblock of the block in Group 6 be denoted by *Block 2*. Recall that this block includes *Block 1* as a subblock, the translation of the effect propositions, and the following formulae:

$$Result(a, s) = \perp\ \supset Ab(a, s), \tag{6.10}$$

$$Result(a, \perp) = \perp. \tag{6.11}$$

Lemma 6.3 *Assume that the constraints are consistent. The UNA and DCA for fluents and actions and the Group 3 axiom entail that $F_{Block\ 2}(Ab)$ is equivalent to the conjunction of (6.10), (6.11), the translation of effect propositions, the translation of constraints, (6.9) and the formula $Ab = G$.*

Proof. We will apply Proposition 5 from the Appendix with E chosen as G. Let B denote the conjunction of the UNA and DCA for fluents and actions and the Group 3 axiom. Let $A(Ab, Result)$ denote the the conjunction of (6.10), (6.11), the translation of effect propositions, the translation of constraints and (6.9).

We will first show that B entails

$$A(Ab, Result) \supset G \leq Ab.$$

That is, we need to show that B and $A(Ab, Result)$ entail

$$\forall a\ s \left[s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} (a = A \wedge T(\sigma, s)) \supset Ab(a, s) \right].$$

We will show that B and $A(Ab, Result)$ entail

$$\forall a\ s \left[s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} (a = A \wedge T(\sigma, s)) \supset Result(a, s) = \bot \right]$$

from which the desired conclusion follows, by (6.10).

The formula $\forall a, s(s = \bot \supset Result(a, s) = \bot)$ follows from the formula (6.11) of $A(Ab, Result)$. Hence, assuming B and $A(Ab, Result)$, we only need to check that

$$T(\sigma, s) \supset Result(A, s) = \bot,$$

where $\langle A, \sigma\rangle \in \Sigma$. By the definition of Σ, $Res_0(A, \sigma) = \emptyset$. That is, the set of domain formulae consisting of all constraints and all formula C such that there is an effect proposition A **causes** C **if** P where $\sigma(P) = \mathsf{T}$ is inconsistent. Since $A(Ab, Result)$ includes the translations of the effect propositions and constraints, this means that B and $A(Ab, Result)$ entail the following formula:

$$T(\sigma, s) \wedge Result(A, s) \neq \bot \supset \mathit{false}.$$

This formula is equivalent to

$$T(\sigma, s) \supset Result(A, s) = \bot.$$

We will now show that B entails

$$A(Ab, Result) \supset \exists R\ \ A(G, R). \tag{6.12}$$

Our first task is to define R.

Define

$$\Sigma' = \{\langle A, \sigma\rangle \mid Res_0(A, \sigma) \neq \emptyset\}.$$

For each $\langle A, \sigma\rangle$ in Σ', let $r(A, \sigma)$ be a fixed element of $Res_0(A, \sigma)$. Define $F(a, s, s')$ as follows:

$$\begin{array}{rcl} F(a,s,s') & = & (s = \bot \wedge s' = \bot)\vee \\ & & \bigvee_{\langle A,\sigma\rangle\in\Sigma}(a = A \wedge s \neq \bot \wedge T(\sigma, s) \wedge s' = \bot)\vee \\ & & \bigvee_{\langle A,\sigma\rangle\in\Sigma'}(a = A \wedge s \neq \bot \wedge T(\sigma, s) \wedge T(r(A,\sigma), s') \wedge s' \neq \bot). \end{array}$$

We will first show that B and $A(Ab, Result)$ entail

$$\exists g \forall as F(a, s, g(a, s)). \tag{6.13}$$

By the axiom of choice, it suffices to show that

$$\forall as \exists s' F(a, s, s').$$

The DCA for actions enables the proof obligation to be written as

$$\forall s \bigwedge_A \exists s' F(A, s, s').$$

Fix a particular A. Using UNA for actions, we see that we need to prove

$$\begin{array}{l} \exists s' \left[(s = \bot \wedge s' = \bot)\vee \right. \\ \quad \bigvee_{\langle A,\sigma\rangle\in\Sigma}(s \neq \bot \wedge T(\sigma, s) \wedge s' = \bot)\vee \\ \quad \left. \bigvee_{\langle A,\sigma\rangle\in\Sigma'}(s \neq \bot \wedge T(\sigma, s) \wedge T(r(A,\sigma), s') \wedge s' \neq \bot)\right]. \end{array} \tag{6.14}$$

We will first show that

$$\begin{array}{rcl} \bigvee_\sigma T(\sigma, s) & \supset & \exists s' \left[(s = \bot \wedge s' = \bot)\vee \right. \\ & & \quad \bigvee_{\langle A,\sigma\rangle\in\Sigma}(s \neq \bot \wedge T(\sigma, s) \wedge s' = \bot) \\ & & \quad \left. \bigvee_{\langle A,\sigma\rangle\in\Sigma'}(s \neq \bot \wedge T(\sigma, s) \wedge T(r(A,\sigma), s') \wedge s' \neq \bot)\right] \end{array} \tag{6.15}$$

Choose any arbitrary state σ. The proof is by cases: If $s = \bot$ define $s' = \bot$. Is $s \neq \bot$, there are two cases depending on whether $Res_0(A, \sigma) = \emptyset$ or not. In the former case, define $s' = \bot$. In the latter case, note that the state $r(A, \sigma)$ in $Res_0(A, \sigma)$ satisfies the predicate $IsState$. Hence, by (6.9)

$$\exists s'[s' \neq \bot \wedge \forall f(Holds(f, s') \equiv r(A, \sigma)(f))].$$

Choose this s' as the desired situation. This concludes the proof of (6.15).

Then, the desired conclusion (6.14) follows from (6.15) and the observation that the translation of constraints (which is a part of $A(Ab, Result)$) entail the formula $\bigvee_\sigma T(\sigma, s)$.

We now are ready to prove (6.12). Define $R(a, s)$ as the function g in (6.13). Then, we need to show that B and $A(Ab, Result)$ entail the following three formulae:

$$R(a, s) = \bot \supset \left[s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} (a = A \wedge T(\sigma, s)) \right], \tag{6.16}$$

$$R(a, \bot) = \bot, \tag{6.17}$$

$$T(P, s) \wedge R(A, s) \neq \bot \supset T(C, R(A, s)), \tag{6.18}$$

for each effect proposition of the form A **causes** C **if** P.

We will first show that B and $A(Ab, Result)$ entail (6.16). Using the DCA and UNA for actions, it suffices to prove for an arbitrary $A \in \mathbf{A}$ that

$$R(A, s) = \bot \supset [s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} T(\sigma, s))].$$

Assume $R(A, s) = \bot$. Hence $F(A, s, \bot)$ holds. Using the definition of F and the UNA for actions, this simplifies to

$$(s = \bot \wedge \bot = \bot) \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} (s \neq \bot \wedge T(\sigma, s) \wedge \bot = \bot),$$

which is equivalent to the desired conclusion.

Proving that B and $A(Ab, Result)$ entail (6.17) is quite similar: assume that for an arbitrary A, $R(A, \bot) = s'$ for some s'. Then $F(A, \bot, s')$ holds, which simplifies to $s' = \bot$.

Finally, we need to check that B and $A(Ab, Result)$ entail (6.18). Let $R(A, s) = s'$. Then $F(A, s, s')$ holds.

We will first show that

$$T(P, s) \wedge s' \neq \bot \wedge F(A, s, s') \supset T(C, s'). \tag{6.19}$$

Note that the desired conclusion,

$$T(P, s) \wedge s' \neq \bot \supset T(C, s')$$

is a direct consequence of (6.19) and $F(A, s, s')$.

The formula (6.19) simplifies, using the definition of $F(A, s, s')$ to

$$\left[\bigvee_{\langle A,\sigma\rangle \in \Sigma'} (s \neq \bot \wedge T(\sigma, s) \wedge T(r(A, \sigma), s')) \right] \wedge T(P, s) \wedge s' \neq \bot \supset T(C, s'). \tag{6.20}$$

Consider any σ such that $\langle A, \sigma \rangle \in \Sigma'$. If $\sigma(P) = \mathsf{T}$, we get that $r(A, \sigma)(C) = \mathsf{T}$. (Recall that $r(A, \sigma) \in Res_0(A, \sigma)$). Hence

$$T(r(A, \sigma), s') \supset T(C, s'),$$

and (6.20) follows. If $\sigma(P) = \mathsf{F}$,

$$T(\sigma, s) \supset \neg T(P, s)$$

and then (6.20) follows trivially. □

Corollary. *Assume that the constraints are consistent. The UNA and DCA for fluents and actions and the Group 3 axiom entail that $\varphi(Block\ 2)$ is equivalent to the conjunction of (6.11), the translation of effect propositions, the translation of constraints, (6.9) and the formula*

$$(Result(a, s) = \bot) \equiv [s = \bot \vee \bigvee_{\langle A, \sigma \rangle \in \Sigma} (a = A \wedge T(\sigma, s))]. \tag{6.21}$$

Let *Block 3* denote the outermost block in Group 6. Note that *Block 3* includes *Block 1* and *Block 2* as sub-blocks. Let *Block 4* denote the block (in Group 5) that characterizes $FrameFluent$.

Lemma 6.4 *The formula $F_{Block\ 4}(Ab)$ is equivalent to the conjunction of the axioms of the block and the formula*

$$\forall f (Ab(f) \equiv \bigvee_{F \in \mathbf{Fr}} f = F).$$

Proof. Let $A(Ab)$ denote the conjunction of the axioms in *Block 3*.

$F_{Block\ 4}(Ab)$
$\equiv$ { Definition }
CIRC$(A(Ab); Ab; FrameFluent)$
$\equiv$ { Proposition 3 from Appendix }
$A(Ab) \wedge$ **CIRC**$(\exists FrameFluent\ A(Ab); Ab)$
$\equiv$ { Simplification—see below }
$A(Ab) \wedge$ **CIRC**$(\forall f (\bigvee_{F \in \mathbf{Fr}} f = F \supset Ab(f)); Ab)$
$\equiv$ { Proposition 2 from Appendix }
$A(Ab) \wedge [\forall f (Ab(f) \equiv \bigvee_{F \in \mathbf{Fr}} f = F)]$

To see that $\exists FrameFluent\ A(Ab)$ is equivalent to

$$\forall f (\bigvee_{F \in \mathbf{Fr}} f = F \supset Ab(f)), \tag{6.22}$$

note that $\exists FrameFluent\ A(Ab)$ can be equivalently expressed as

$$\begin{array}{c}\exists FrameFluent[\forall f((\bigvee_{F\in \mathbf{Fr}} f = F \supset FrameFluent(f))\wedge \\ (FrameFluent(f) \supset Ab(f)))].\end{array}$$

By second order logic, the above formula is equivalent to (6.22).

Corollary. *$\varphi(Block\ 4)$ is equivalent to the formula*

$$\forall f(FrameFluent(f) \equiv \bigvee_{F\in \mathbf{Fr}} f = F).$$

Lemma 6.5 *Let D be a finite domain. For any model Ψ of D, there exists a model M of $\beta(D)$ similar to Ψ.*

Proof. In defining the model M, our first task is to describe the universe of fluents $\mid M\mid_f$, the universe of actions $\mid M\mid_a$, and the universe of situations $\mid M\mid_s$.

Defining the first two is straightforward:

$$\begin{array}{rcl}\mid M\mid_f &=& \mathbf{F},\\ \mid M\mid_a &=& \mathbf{A}.\end{array}$$

Let W be the set of all tuples of actions over which Ψ is defined. Let L be the set of all functions from **F** to $\{\mathsf{T},\mathsf{F}\}$. The universe of situations $\mid M\mid_s$ contains distinct symbols corresponding to each element of W and L and a separate symbol to denote $\perp$. Formally,

$$\mid M\mid_s \;=\; \{\xi_{\overline{A}} \mid \overline{A}\in W\}\cup\{\xi_\lambda \mid \lambda\in L\}\cup\{\xi_\perp\}.$$

With each element ξ of $\mid M\mid_s$, we associate a state of the domain D. The state associated with $\xi_{\overline{A}}$ is $\Psi(\overline{A})$. The state associated with ξ_λ is λ if λ is a state of the domain D and is $\Psi(\epsilon)$ otherwise. The state associated with $\xi_\perp$ is $\Psi(\epsilon)$. These states will be useful in defining the interpretation of certain functions (e.g. $Result$) and predicates (e.g. $Holds$).

We will interpret the fluent and action constants as themselves. The interpretation of $FrameFluent$, S_0 and $\perp$ are straightforward:

$$\begin{array}{lcl}F\in FrameFluent^M &\Leftrightarrow& F\in \mathbf{Fr},\\ S_0^M &=& \xi_\epsilon,\\ \perp^M &=& \xi_\perp.\end{array}$$

To define the interpretation of $Holds$, consider any element ξ in the universe of situations and let the state associated with ξ be σ. Then

$$\langle F,\xi\rangle\in Holds^M \quad\Leftrightarrow\quad \sigma(F)=\mathsf{T}.$$

The definition of $Result^M$ is by cases. For any action A, $Result^M(A, \xi_\perp)$ is defined to be $\xi_\perp$. Otherwise, let the state associated with ξ be σ. If $Res_0(A, \sigma) = \emptyset$, then $Result^M(A, \xi)$ is defined to be $\xi_\perp$. Otherwise, if $\xi = \xi_{\overline{A}}$, $Result^M(A, \xi)$ is defined to be $\xi_{\overline{A};A}$. If $\xi = \xi_\lambda$, $Result^M(A, \xi)$ is defined to be $\xi_{\sigma'}$, where σ' is some element of $Res(A, \sigma)$.

This completes the definition of M. We now need to check that M is a model of the axioms in Groups 1–4, $\varphi(\textit{Block 4})$ and $\varphi(\textit{Block 3})$.

For simplifying the presentation of the proofs to follow, we will extend the language of βD by adding each element of the universe of situations of M as a new object constant. These new objects constants will be called *names*. Thus, we may write the definition of $Holds^M$ above equivalently as

$$M \models Holds(F, \xi) \ \Leftrightarrow \ \sigma(F) = \mathsf{T},$$

where σ is the state associated with ξ. In fact, it is easy to show, by induction, that for any domain formula C and any name ξ,

$$M \models T(C, \xi) \ \Leftrightarrow \ \sigma(C) = \mathsf{T}. \tag{6.23}$$

It is easy to check that M is a model of the axioms in Groups 1–3. That it is a model of $\varphi(\textit{Block 4})$ follows easily from the corollary to Lemma 6.4.

Let us check that M is a model of the axioms in Group 4, namely, that it is a model of the translations of the value propositions. First of all, we will prove by induction that

$$\Psi(\overline{A}) \text{ is defined} \Rightarrow [\overline{A}]^M = \xi_{\overline{A}}. \tag{6.24}$$

The base case

$$\Psi(\epsilon) \text{ is defined} \Rightarrow S_0^M = \xi_\epsilon$$

follows trivially from the definition of S_0^M. Let the inductive hypothesis be (6.24). To show the induction step, assume $\Psi(\overline{A}; A)$ is defined. This implies that (i) $\Psi(\overline{A})$ is defined and (ii) $Res(A, \Psi(\overline{A})) \neq \emptyset$. From (i) and the inductive hypothesis, we get $[\overline{A}]^M = \xi_{\overline{A}}$. From (ii), it is clear that $Res_0(A, \Psi(\overline{A})) \neq \emptyset$. Since $\Psi(\overline{A})$ is the state associated with $\xi_{\overline{A}}$, the previous fact and the definition of $Result^M$ gives us that $Result^M(A, \xi_{\overline{A}}) = \xi_{\overline{A};A}$. That is, $Result^M(A, [\overline{A}]^M) = \xi_{\overline{A};A}$ or equivalently, $[\overline{A}; A]^M = \xi_{\overline{A};A}$, as desired.

Let us now show that

$$\Psi(\overline{A}) \text{ is defined} \ \Leftrightarrow \ M \models [\overline{A}] \neq \perp. \tag{6.25}$$

This proof is also by induction. The base case follows easily since both sides of the equivalence (6.25) are true when $\overline{A} = \epsilon$. Let (6.25) be the inductive hypothesis. We will now prove that

$$\Psi(\overline{A}; A) \text{ is defined} \ \Leftrightarrow \ M \models [\overline{A}; A] \neq \perp.$$

holds by proving each implication separately.

$$
\begin{array}{ll}
 & \Psi(\overline{A}; A) \text{ is defined} \\
\Leftrightarrow & \{ \text{Definition} \} \\
 & \Psi(\overline{A}) \text{ is defined and } Res(A, \Psi(\overline{A})) \neq \emptyset \\
\Leftrightarrow & \{ \text{Lemma 6.1} \} \\
 & \Psi(\overline{A}) \text{ is defined and } Res_0(A, \Psi(\overline{A})) \neq \emptyset \\
\Rightarrow & \{ \text{Eqn (6.24)} \} \\
 & [\overline{A}]^M = \xi_{\overline{A}} \text{ and } Res_0(A, \Psi(\overline{A})) \neq \emptyset \\
\Rightarrow & \{ \text{The state associated with } \xi_{\overline{A}} \text{ is } \Psi(\overline{A}), \text{Defn. of } Result^M \} \\
 & M \models Result(A, [\overline{A}]) \neq \bot \\
\Leftrightarrow & \{ \text{Definition} \} \\
 & M \models [\overline{A}; A] \neq \bot.
\end{array}
$$

$$
\begin{array}{ll}
 & M \models [\overline{A}; A] \neq \bot \\
\Rightarrow & \{ M \text{ is a model of } Result(a, \bot) = \bot \} \\
 & M \models [\overline{A}] \neq \bot \text{ and } M \models [\overline{A}; A] \neq \bot \\
\Leftrightarrow & \{ \text{Inductive Hypothesis} \} \\
 & \Psi(\overline{A}) \text{ is defined and } M \models [\overline{A}; A] \neq \bot \\
\Rightarrow & \{ \text{Eqn (6.24)} \} \\
 & [\overline{A}]^M = \xi_{\overline{A}} \text{ and } M \models [\overline{A}; A] \neq \bot \\
\Rightarrow & \{ \text{The state associated with } \xi_{\overline{A}} \text{ is } \Psi(\overline{A}), \text{Defn. of } Result^M \} \\
 & Res_0(A, \Psi(\overline{A})) \neq \emptyset \\
\Leftrightarrow & \{ \text{Lemma 6.1} \} \\
 & Res(A, \Psi(\overline{A})) \neq \emptyset \\
\Rightarrow & \{ \Psi(\overline{A}) \text{ is defined} \} \\
 & \Psi(\overline{A}; A) \text{ is defined}
\end{array}
$$

We will now show that for every value proposition P, P is true in Ψ iff M satisfies βP. (That M is a model of the translations of the value propositions of the domain D is a direct consequence of this result). Again, we will prove both implications separately.

$$
\begin{array}{ll}
 & C \textbf{ after } \overline{A} \text{ is true in } \Psi \\
\Leftrightarrow & \{ \text{Definition} \} \\
 & \Psi(\overline{A}) \text{ is defined and } \Psi(\overline{A})(C) = \mathsf{T} \\
\Rightarrow & \{ \text{Eqn (6.24), Eqn (6.25)} \} \\
 & \xi_{\overline{A}} = [\overline{A}]^M \text{ and } M \models [\overline{A}] \neq \bot \text{ and } \Psi(\overline{A})(C) = \mathsf{T} \\
\Leftrightarrow & \{ \text{The state corresponding to } \xi_{\overline{A}} \text{ is } \Psi(\overline{A}), \text{Eqn (6.23)} \} \\
 & M \models \xi_{\overline{A}} = [\overline{A}] \wedge [\overline{A}] \neq \bot \wedge T(C, \xi_{\overline{A}}) \\
\Rightarrow & \{ \text{Equality} \} \\
 & M \models T(C, [\overline{A}]) \wedge [\overline{A}] \neq \bot.
\end{array}
$$

$M \models T(C, [\overline{A}]) \wedge [\overline{A}] \neq \perp$

$\Leftrightarrow$ { Eqn (6.25) }

$M \models T(C, [\overline{A}])$ and $\Psi(\overline{A})$ is defined

$\Rightarrow$ { Eqn (6.24) }

$M \models T(C, [\overline{A}])$ and $\Psi(\overline{A})$ is defined and $\xi_{\overline{A}} = [\overline{A}]^M$

$\Rightarrow$ { Equality }

$M \models T(C, \xi_{\overline{A}})$ and $\Psi(\overline{A})$ is defined

$\Leftrightarrow$ { Eqn (6.23), the state associated with $\xi_{\overline{A}}$ is $\Psi(\overline{A})$ }

$\Psi(\overline{A})(C) = \mathsf{T}$ and $\Psi(\overline{A})$ is defined

$\Leftrightarrow$ { Definition }

C **after** $\overline{A}$ is true in Ψ

We now turn to verifying that M is a model of $\varphi(\textit{Block 3})$. To do that, we need to check that M combined with an appropriate interpretation of Ab satisfies $F_{Block\ 3}(Ab)$. That is, we have to show that M (combined with an appropriate interpretation of Ab) satisfies **CIRC**$(Z; Ab; Result)$, where Z is the conjunction of the commonsense law of inertia and $\varphi(\textit{Block 2})$.

Let Ab be interpreted as follows: Let ξ be an element in the universe of situations and let $Result^M(A, \xi) = \xi'$. Let the state associated with ξ be σ and that associated with ξ' be σ'. Then

$$\langle F, A, \xi\rangle \in \mathit{Ab}^M \;\Leftrightarrow\; F \in \mathbf{Fr} \wedge F \in \rho(\sigma, \sigma'). \tag{6.26}$$

Let us first check that M is a model of $\varphi(\textit{Block 2})$. By the corollary to Lemma 6.3, we need to check that M is a model of (6.9), (6.11), (6.21), the translation of constraints and the translation of effect propositions. That M is model of (6.9), (6.11) and (6.21) follows easily from the definition of M.

Let us now check that for each constraint of the form **always** C and each name ξ, $M \models T(C, \xi)$. Let σ be the state associated with ξ. Since σ is a state of the domain, $\sigma(C) = \mathsf{T}$ and the result follows by (6.23).

Let us now check that for each effect proposition A **causes** C **if** P, M is a model of its translation. Let ξ be any element in the universe of situations and let $Result^M(A, \xi)$ be ξ'. Also, let the states associated with ξ and ξ' be σ and σ' respectively. Assume that $M \models \xi' \neq \perp$. We need to show that $M \models T(P, \xi) \supset T(C, \xi')$.

Since $\xi' \neq \perp$, using the definition of $Result^M$, we get that

$$\sigma' \in Res(A, \sigma). \tag{6.27}$$

$$
\begin{aligned}
& M \models T(P,\xi) \supset T(C,\xi') \\
\Leftrightarrow\ & \{\text{Definition}\} \\
& M \models T(P,\xi) \Rightarrow M \models T(C,\xi') \\
\Leftrightarrow\ & \{\text{ Eqn.(6.23)}\} \\
& \sigma(P) = \mathsf{T} \Rightarrow \sigma'(C) = \mathsf{T} \\
\Leftrightarrow\ & \{\text{ Eqn (6.27) }\} \\
& \textit{True}
\end{aligned}
$$

Let us now check that M is a model of the law of inertia LI. As before, let ξ be any element in the universe of situations and let $Result^M(A,\xi)$ be ξ'. Let the states associated with ξ and ξ' be σ and σ' respectively. Fix an arbitrary fluent F and action A. We first check that M is a model of LI.

$$
\begin{aligned}
& M \models FrameFluent(F) \wedge \neg Ab(F,A,\xi) \supset \\
& \qquad\qquad\qquad (Holds(F,\xi) \equiv Holds(F, Result(A,\xi))) \\
\Leftrightarrow\ & \{\text{Definition}\} \\
& M \models FrameFluent(F) \wedge \neg Ab(F,A,\xi) \Rightarrow \\
& \qquad\qquad\qquad M \models (Holds(F,\xi) \equiv Holds(F,\xi')) \\
\Leftrightarrow\ & \{\text{ Defn. of } FrameFluent^M,\ Ab^M,\ Holds^M\ \} \\
& [F \in \mathbf{Fr} \wedge F \notin \rho(\sigma,\sigma')] \Rightarrow \sigma(F) = \sigma'(F) \\
\Leftrightarrow\ & \{\text{Definition of } \rho\ \} \\
& [F \in \mathbf{Fr} \wedge F \notin \rho(\sigma,\sigma')] \Rightarrow F \notin \rho(\sigma,\sigma') \\
\Leftrightarrow\ & \{\text{ Propositional logic }\} \\
& \textit{True}
\end{aligned}
$$

To conclude the proof that M satisfies $\mathbf{CIRC}(Z; Ab; Result)$ (recall that Z is the conjunction of the commonsense law of inertia and φ(*Block 2*)), we have to check that M is a minimal model of Z with respect to $\leq^{Ab;Result}$. If it is not a minimal model, then there exists a model M' of Z such that $M' \leq^{Ab;Result} M$ and $Ab^{M'} \subset Ab^M$. That is, there exist F, A, ξ such that

$$\langle F, A, \xi\rangle \in Ab^M, \tag{6.28}$$

but

$$\langle F, A, \xi\rangle \notin Ab^{M'}. \tag{6.29}$$

Let $Result^M(A,\xi) = \xi'$, and let the states associated with ξ and ξ' be σ and σ' respectively. From (6.28) and (6.26),

$$F \in \mathbf{Fr} \wedge F \in \rho(\sigma,\sigma'). \tag{6.30}$$

Let $Result^{M'}(A,\xi) = \xi''$, and the state associated with ξ'' be σ''. From (6.29), since M' is a model of the law of inertia and $Holds^M = Holds^{M'}$, we get

$$F \in FrameFluent^{M'} \Rightarrow [\sigma(F) = \sigma''(F)]. \tag{6.31}$$

From (6.30), $F \in \mathbf{Fr}$. Recall that $FrameFluent^M = \mathbf{Fr}$ and $FrameFluent^M = FrameFluent^{M'}$. Hence $F \in FrameFluent^{M'}$, and $\sigma(F) = \sigma''(F)$ follows by (6.31). Equivalently, we can write this as $F \notin \rho(\sigma, \sigma'')$. Thus, we get that

$$F \in \mathbf{Fr} \wedge F \notin \rho(\sigma, \sigma''). \tag{6.32}$$

There are now two cases to consider, depending on whether or not $Result^M(A, \xi)$ equals $\perp$.

Case 1: $Result^M(A, \xi) = \perp$.

Recall that both M and M' are identical except in the interpretation of Ab and $Result$. In particular, M' satisfies the UNA and DCA for fluents and actions and the Group 3 axiom. Since M' satisfies $\varphi(\textit{Block 2})$, by the corollary to Lemma 6.3, we get that M' is a model of (6.21). That is, we get that $Result^{M'}(A, \xi) = \perp$. Hence $\sigma' = \sigma''$. This contradicts the second conjunctive terms of (6.30) and (6.32).

Case 2: $Result^M(A, \xi) \neq \perp$.

By the definition of $Result^M$, we get that $\sigma' \in Res(A, \sigma)$. Note that $\sigma'' \in Res_0(A, \sigma)$, since $Holds^M = Holds^{M'}$ and M' is a model of the translations of the effect propositions. Let Δ_1 and Δ_2 be defined as follows:

$$\Delta_1 = \rho(\sigma, \sigma') \cap \mathbf{Fr}, \tag{6.33}$$

$$\Delta_2 = \rho(\sigma, \sigma'') \cap \mathbf{Fr}. \tag{6.34}$$

We will show that $\Delta_2 \subset \Delta_1$. This implies that $\sigma' \notin Res(A, \sigma)$, which gives us a contradiction.

Note that (6.30) and (6.32) imply that F is in Δ_1, but not in Δ_2. Hence it suffices to show that Δ_2 is a subset of Δ_1. Let F be any element of Δ_2.

Let F be any element of Δ_2.

$$
\begin{array}{ll}
 & F \in \Delta_2 \\
\Leftrightarrow & \{\text{Definition of } \Delta_2\} \\
 & F \in \rho(\sigma, \sigma'') \wedge F \in \mathbf{Fr} \\
\Leftrightarrow & \{\text{Definition of } Holds^{M'} \ (= Holds^M)\} \\
 & (Holds^{M'}(F, \xi) \not\equiv Holds^{M'}(F, \xi'')) \wedge F \in \mathbf{Fr} \\
\Rightarrow & \{\ M' \text{ is a model of } LI\ \} \\
 & \langle F, A, \xi \rangle \in Ab^{M'} \wedge F \in \mathbf{Fr} \\
\Rightarrow & \{\ Ab^{M'} \subset Ab^M\ \} \\
 & \langle F, A, \xi \rangle \in Ab^M \wedge F \in \mathbf{Fr} \\
\Leftrightarrow & \{\text{ Definition of } Ab^M \text{ from (6.26) }\} \\
 & F \in \rho(\sigma, \sigma') \wedge F \in \mathbf{Fr} \\
\Leftrightarrow & \{\text{Definition of } \Delta_1\} \\
 & F \in \Delta_1
\end{array}
$$

This concludes the proof that M is a model of $\varphi(\mathit{Block\ 3})$. Hence M is a model of $\beta(D)$. That M is similar to Ψ follows because we have already shown that for every value proposition P, P is true in Ψ iff M satisfies βP. $\Box$

Lemma 6.6 *Let D be a finite domain. For any model M of $\beta(D)$, there exists a model Ψ of D such that M is similar to Ψ.*

Proof. The UNA and DCA for fluents and actions allows us to assume (without loss of generality) that

$$\begin{aligned} \mid M \mid_f &= \mathbf{F}, \\ \mid M \mid_a &= \mathbf{A}. \end{aligned}$$

For simplifying the presentation of the proofs to follow, we will extend the language of βD by adding each element of the universe of situations of M as a new object constant. These new objects constants will be called *names*.

For any situation valued term t without variables, define t^* to be a function from fluents to truth values as

$$t^*(F) = \mathsf{T} \Leftrightarrow M \models Holds(F, t).$$

It is easy to check that for any formula C of the domain and any situation valued term t without variables,

$$t^*(C) = \mathsf{T} \Leftrightarrow M \models T(C, t). \tag{6.35}$$

Since M is a model of the translations of the constraints, we get that $M \models T(C, t)$ for any constraint **always** C. It then follows by (6.35) that t^* is a state of D.

Let us now turn to defining Ψ. Ψ is defined on the tuple $\overline{A}$ iff $M \models [\overline{A}] \neq \bot$, and if defined, $\Psi(\overline{A}) = [\overline{A}]^*$.

We now need to check the following:

1. The set $\{\overline{A} \mid \Psi(\overline{A})$ is defined$\}$ is prefix-closed.
2. If $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is not defined, then $Res(A, \Psi(\overline{A})) = \emptyset$.
3. If $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is defined, then $\Psi(\overline{A}; A) \in Res(A, \Psi(\overline{A}))$.
4. Ψ is similar to M.
5. All value propositions of D are true in Ψ.

To check the first of the above, it suffices to note that since M satisfies (6.11), it follows that M satisfies

$$Result(a, s) \neq \bot \supset s \neq \bot.$$

Let us turn to checking item 2. Assume that $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is not defined. We need to check that $Res(A, \Psi(\overline{A})) = \emptyset$. Since $\Psi(\overline{A})$ is defined,

$$M \models [\overline{A}] \neq \bot, \tag{6.36}$$

and

$$\Psi(\overline{A}) = [\overline{A}]^*. \tag{6.37}$$

Note that $M \models T([\overline{A}]^*, [\overline{A}])$ and this in conjunction with (6.37) yields that

$$M \models T(\Psi(\overline{A}), [\overline{A}]). \tag{6.38}$$

Since $\Psi(\overline{A}; A)$ is not defined, we get that

$$M \models [\overline{A}; A] = \bot. \tag{6.39}$$

By the corollary to Lemma 6.3, M satisfies (6.21). This fact, in combination with (6.39) yields that

$$M \models \left[[\overline{A}] = \bot \vee \bigvee_{\langle A', \sigma \rangle \in \Sigma} (A = A' \wedge T(\sigma, [\overline{A}])) \right]. \tag{6.40}$$

Using (6.36) and the UNA for actions, (6.40) simplifies to

$$M \models \bigvee_{\sigma : \langle A, \sigma \rangle \in \Sigma} T(\sigma, [\overline{A}]). \tag{6.41}$$

Note that if $\sigma \neq \sigma'$,

$$M \models T(\sigma, [\overline{A}]) \supset \neg T(\sigma', [\overline{A}]). \tag{6.42}$$

If $\langle A, \Psi(\overline{A}) \rangle \notin \Sigma$, (6.38) in conjunction with (6.42) implies that

$$\bigwedge_{\sigma : \langle A, \sigma \rangle \in \Sigma} \neg T(\sigma, [\overline{A}]),$$

contradicting (6.41). Hence, $\langle A, \Psi(\overline{A}) \rangle \in \Sigma$ or equivalently $Res_0(A, \Psi(\overline{A})) = \emptyset$. This implies that $Res(A, \Psi(\overline{A})) = \emptyset$.

Let us now check that if $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is defined, then $\Psi(\overline{A}; A) \in Res(A, \Psi(\overline{A}))$. We will prove this by contradiction. Since M is a model of $\beta(D)$, it is a model of $\varphi(\textit{Block 3})$. Equivalently, M combined with an appropriate interpretation of Ab (call it Ab^M) satisfies $F_{Block\,3}(Ab)$. Recall that $F_{Block\,3}(Ab)$ stands for **CIRC**$(Z; Ab; Result)$, where Z is the conjunction of the commonsense law of inertia and

$\varphi(\mathit{Block}\ 2)$. We will obtain the desired contradiction by constructing a model M' of Z, identical to M except in the interpretations of $Result$ and Ab and where the extent of Ab is smaller than Ab^M.

Assume that there exist $\overline{A_1}$ and A_2 such that $\Psi(\overline{A_1})$ is defined and $\Psi(\overline{A_1}; A_2)$ is defined and $\Psi(\overline{A_1}; A_2) \notin Res(A_2, \Psi(\overline{A_1}))$. Since $\Psi(\overline{A_1})$ is defined, as before, we see that

$$M \models [\overline{A_1}] \neq \bot, \tag{6.43}$$

$$\Psi(\overline{A_1}) = [\overline{A_1}]^*, \tag{6.44}$$

and

$$M \models T(\Psi(\overline{A_1}), [\overline{A_1}]). \tag{6.45}$$

Since $\Psi(\overline{A_1}; A_2)$ is defined,

$$M \models [\overline{A_1}; A_2] \neq \bot. \tag{6.46}$$

In combination with (6.21) and UNA for actions, (6.46) yields that

$$M \models \left[[\overline{A_1}] \neq \bot \wedge \bigwedge_{\sigma : \langle A_2, \sigma \rangle \in \Sigma} \neg T(\sigma, [\overline{A_1}]) \right]. \tag{6.47}$$

Equations (6.45) and (6.47) imply that $\langle A_2, \Psi(\overline{A_1}) \rangle \notin \Sigma$ or equivalently that $Res_0(A_2, \Psi(\overline{A_1})) \neq \emptyset$. Lemma 6.1 then yields that $Res(A_2, \Psi(\overline{A_1})) \neq \emptyset$. Choose a σ_2' in $Res(A_2, \Psi(\overline{A_1}))$. By the corollary to Lemma 6.2, M is a model of (6.9) and hence

$$M \models \exists s[s \neq \bot \wedge \forall f(Holds(f, s) \equiv \sigma_2'(f))].$$

Let us denote by ξ_2' such an element of $\mid M \mid_s$.

We are now ready to define M'. Let us first introduce some notation to simplify the definition. Let $\xi_1 = [\overline{A_1}]^M$, $\sigma_1 = [\overline{A_1}]^*$, $\xi_2 = [\overline{A_1}; A_2]^M$ and $\sigma_2 = [\overline{A_1}; A_2]^*$.

Let us first define $Result^{M'}$. $Result^{M'}(A_2, \xi_1)$ is defined to be ξ_2'. For all other pairs, $Result^{M'}$ is defined to be equal to $Result^M$.

Let us now define $Ab^{M'}$. Let us first consider triples whose last two components are A_2 and ξ_1. For such triples, we define, for any fluent F,

$$\langle F, A_2, \xi_1 \rangle \in Ab^{M'} \;\Leftrightarrow\; F \in \mathbf{Fr} \wedge F \in \rho(\sigma_1, \sigma_2'). \tag{6.48}$$

Any other triple is in $Ab^{M'}$ iff it is in Ab^M. This concludes the definition of M'.

Let us first check that M' is a model of Z (recall that Z is the conjunction of the commonsense law of inertia LI and $\varphi(\mathit{Block}\ 2)$). Using the corollary to Lemma 6.3, it suffices to check that M is a model of LI, translations of the effect propositions and constraints and formulae (6.9), (6.11) and (6.21).

We will first check that M' is a model of LI. By the definition of $Result^{M'}$ and $Ab^{M'}$, it suffices to check for any F that

$$M' \models FrameFluent(F) \wedge \neg Ab(F, A_2, \xi_1) \supset (Holds(F, \xi_1) \equiv Holds(F, Result(A_2, \xi_1)).$$

$$M' \models FrameFluent(F) \wedge \neg Ab(F, A_2, \xi_1) \supset (Holds(F, \xi_1) \equiv Holds(F, Result(A_2, \xi_1))$$

$\Leftrightarrow$ {Definition }

$M' \models FrameFluent(F) \wedge \neg Ab(F, A_2, \xi_1) \Rightarrow M' \models (Holds(F, \xi_1) \equiv Holds(F, \xi_2'))$

$\Leftrightarrow$ { $FrameFluent^{M'} = FrameFluent^{M}$ ($=$ **Fr**), Definition of $Ab^{M'}$, σ_1 and σ_2' }

$[F \in \mathbf{Fr} \wedge F \notin \rho(\sigma_1, \sigma_2')] \Rightarrow \sigma_1(F) = \sigma_2'(F)$

$\Leftrightarrow$ {Definition of ρ }

$[F \in \mathbf{Fr} \wedge F \notin \rho(\sigma_1, \sigma_2')] \Rightarrow F \notin \rho(\sigma_1, \sigma_2')$

$\Leftrightarrow$ { Propositional logic }

True

We will now check that M' is a model of the translation of the effect propositions. By the definition of $Result^{M'}$, it suffices to check that

$$M' \models [\overline{A_1}; A_2] \neq \perp \wedge T(P, [\overline{A_1}]) \supset T(C, [\overline{A_1}; A_2])$$

for each effect proposition A_2 **causes** C **if** P of the domain D.

$M' \models [\overline{A_1}; A_2] \neq \perp \wedge T(P, [\overline{A_1}]) \supset T(C, [\overline{A_1}; A_2])$

$\Leftrightarrow$ {Definition }

$M' \models \xi_2' \neq \perp \wedge T(P, \xi_1) \supset T(C, \xi_2')$

$\Leftrightarrow$ {$M' \models \xi_2' \neq \perp$, by the definition of ξ_2' }

$M' \models T(P, \xi_1) \supset T(C, \xi_2')$

$\Leftrightarrow$ {Definition }

$M \models T(P, \xi_1) \Rightarrow M \models T(C, \xi_2')$

$\Leftrightarrow$ { Definition of σ_1 and σ_2'}

$\sigma_1(P) = \mathsf{T} \Rightarrow \sigma_2'(C) = \mathsf{T}$

$\Leftrightarrow$ { $\sigma_2' \in Res(A_2, \sigma_1)$ }

True

That M' is a model of the translations of the constraints and of the formula (6.9) follows easily from the fact that M and M' are identical except in the interpretations of Ab and $Result$. To see that M is a model of (6.11), observe that M and M' define $Result(A, \perp)$

identically for all A. (Note that $[\overline{A_1}]^M$ is not equal to $\perp$). Finally, to see that M' is a model of (6.21), observe that for any A and name ξ,

$$M \models Result(A, \xi) = \perp \;\Leftrightarrow\; M' \models Result(A, \xi) = \perp.$$

Let us now turn to showing that $Ab^{M'} \subset Ab^M$. By assumption, $\sigma_2 \notin Res(A_2, \sigma_1)$. (Recall that $\sigma_1 = \Psi(\overline{A_1})$ and $\sigma_2 = \Psi(\overline{A_1}; A_2)$). Since σ_2' is, by definition, an element of $Res(A_2, \sigma_1)$, we get

$$\rho(\sigma_1, \sigma_2') \cap \mathbf{Fr} \subset \rho(\sigma_1, \sigma_2) \cap \mathbf{Fr}. \tag{6.49}$$

Let us first show that $\langle F, A_2, \xi_1\rangle \in Ab^{M'}$ implies that $\langle F, A_2, \xi_1\rangle \in Ab^M$.

$$\begin{array}{ll}
 & \langle F, A_2, \xi_1\rangle \in Ab^{M'} \\
\Leftrightarrow & \{\text{ Eqn (6.48) }\} \\
 & F \in \mathbf{Fr} \wedge F \in \rho(\sigma_1, \sigma_2') \\
\Rightarrow & \{\text{ Eqn (6.49) }\} \\
 & F \in \mathbf{Fr} \wedge F \in \rho(\sigma_1, \sigma_2) \\
\Leftrightarrow & \{\text{ Definition of } \sigma_1 \text{ and } \sigma_2 \ \} \\
 & F \in \mathbf{Fr} \wedge (M \models Holds(F, \xi_1) \not\equiv Holds(F, \xi_2)) \\
\Rightarrow & \{\ M \text{ is a model of } LI\ \} \\
 & \langle F, A_2, \xi_1\rangle \in Ab^M
\end{array}$$

To show that $Ab^{M'}$ is a proper subset of Ab^M, we proceed as follows. By (6.49), there is a fluent name F such that

$$F \in \rho(\sigma_1, \sigma_2) \wedge F \in \mathbf{Fr} \wedge F \notin \rho(\sigma_1, \sigma_2').$$

We see from the above and the definition of $Ab^{M'}$ that $\langle F, A_2, \xi_1\rangle \notin Ab^{M'}$. However, since M is a model of LI, we get that $\langle F, A_2, \xi_1\rangle \in Ab^M$. Thus, we see that $Ab^{M'}$ is a proper subset of Ab^M.

Let us now check that Ψ is similar to M.

$$\begin{array}{ll}
 & C \textbf{ after } \overline{A} \text{ is true in } \Psi \\
\Leftrightarrow & \{\text{ Definition of “true in } \Psi\text{”}\} \\
 & \Psi(\overline{A}) \text{ is defined and } \Psi(\overline{A})(C) = \mathsf{T} \\
\Leftrightarrow & \{\text{ Definition of } \Psi\ \} \\
 & [\overline{A}]^M \neq \perp \text{ and } [\overline{A}]^*(C) = \mathsf{T} \\
\Leftrightarrow & \{\text{ Eqn (6.35) }\} \\
 & M \models [\overline{A}] \neq \perp \wedge T(C, [\overline{A}])
\end{array}$$

Since Ψ is similar to M and M is a model of the translations of the value propositions of the domain, Ψ is a model of the value propositions of the domain. □

Theorem 6.1 follows from Lemmas 6.5 and 6.6. □

Chapter 7

$\mathcal{AR}$—A language for non-Boolean Fluents

7.1 Introduction

In this chapter, we extend $\mathcal{AR}^-$, the language introduced in Chapter 4 in two ways. First, $\mathcal{AR}$ allows us to represent fluents whose values are non-Boolean, such as $\mathit{Location}(x)$ or $\mathit{Color}(x)$. In the situation calculus, such fluents can be always eliminated in favor of propositional fluents, just as function symbols in first-order logic can be eliminated in favor of predicate symbols. For instance, instead of

$$\mathit{Brighter}(\mathit{Color}(x, s1), \mathit{Color}(x, s2))$$

we can write

$$\forall c_1 c_2 [\mathit{HasColor}(x, c1, s1) \wedge \mathit{HasColor}(x, c2, s2) \supset \mathit{Brighter}(c_1, c_2)].$$

But this may be unnatural and inconvenient. Surprisingly, extending the solution to the frame problem incorporated into the semantics of $\mathcal{AR}^-$ to nonpropositional fluents was not trivial. To justify the new formulation, we prove that the elimination of nonpropositional fluents from a domain description "preserves the meaning" of the description (Theorem 7.2).

Second, a new kind of proposition, called an indeterminate effect proposition, is introduced in $\mathcal{AR}$. These propositions enable us to describe nondeterministic actions.

In the next section, we define $\mathcal{AR}$ and illustrate its use with examples. In Section 7.3, we embed $\mathcal{AR}^-$ in $\mathcal{AR}$. In Section 7.4, we study the properties of $\mathcal{AR}$.

7.2 Language $\mathcal{AR}$

7.2.1 Syntax

An $\mathcal{AR}$ language is characterized by

- a nonempty set of symbols, that are called *fluent names*, or *fluents*,
- a function, associating with every fluent name F a nonempty set Dom_F of symbols that is called the *domain* of F,
- a subset of fluent names, that is called the *frame*,
- a nonempty set of symbols, that are called *action names*, or *actions*.

A *value* is a symbol that belongs to the domain of a fluent.

An *atomic formula* is an expression of the form $(F \textbf{ is } V)$, where F is a fluent, and $V \in Dom_F$. A *formula* is a propositional combination of atomic formulae. A fluent F is *propositional* if $Dom_F = \{False, True\}$; for a propositional fluent F, we will abbreviate the atomic formula

$$F \textbf{ is } True$$

by F.

Note that this is more general than in $\mathcal{AR}^-$, where the domain of every fluent was $\{False, True\}$, and formulae were defined as propositional combinations of fluents. An atomic formula F of $\mathcal{AR}^-$ corresponds to $(F \textbf{ is } True)$ in the syntax of $\mathcal{AR}$.

There are four types of *propositions* in an $\mathcal{AR}$ language: value propositions, determinate and indeterminate effect propositions, and constraints. A *value proposition* is an expression of the form

$$C \textbf{ after } \overline{A}, \tag{7.1}$$

where C is a formula, and $\overline{A}$ is a string of actions.

A *determinate effect proposition* is an expression of the form

$$A \textbf{ causes } C \textbf{ if } P, \tag{7.2}$$

where A is an action, and C and P are formulae.

An *indeterminate effect proposition* is an expression of the form

$$A \textbf{ possibly changes } F \textbf{ if } P, \tag{7.3}$$

where A is an action, F a fluent that belongs to the frame and P a formula. Intuitively, (7.3) asserts that the value of F can change nondeterministically when the action A is executed in a situation in which the precondition P is true.

Finally, a *constraint* is a proposition of the form

$$\textbf{always } C, \tag{7.4}$$

where C is a formula.

A *domain description*, or *domain*, is a set of propositions.

7.2.2 Examples

In our examples, we will make use of the notational conventions that were introduced in Section 4.2.2. Thus for instance, we will drop the part

$$\textbf{if } P$$

in effect propositions (7.2) and (7.3) if P is the propositional formula *True*. Our comments on the "models" of these domain descriptions will turn into precise statements after the semantics of $\mathcal{AR}$ is described in the next section.

Example 1. A disc is divided into N sectors, $N > 1$. At every instant of time, one of the sectors is under the head that reads information from the disk. There is an action which rotates the disc counterclockwise. We can describe this system using one fluent *Orientation*, whose domain is $\{0, \ldots, N-1\}$, and which belongs to the frame. There is one action, *Turn*. The domain description consists of N propositions

$$\begin{array}{r} \textit{Turn} \textbf{ causes } (\textit{Orientation} \textbf{ is } (i+1) \bmod N) \\ \textbf{if } (\textit{Orientation} \textbf{ is } i) \quad (0 \le i < N). \end{array} \tag{7.5}$$

The domain description from Example 1 has N models, corresponding to N possible initial orientations of the disk. By adding the value proposition

$$\textbf{initially } \textit{Orientation} \textbf{ is } 0 \tag{7.6}$$

to the domain description, we would reduce the number of models to one.

Example 2. The "Russian Turkey Shoot" [Sandewall, 1992] is the enhancement of the original shooting story in which there is an additional action of spinning the gun's bullet chamber. More precisely, we think of *Spin* as the action of looking to see whether the chamber is empty, inserting a bullet if it is, and then giving it a spin. We extend (4.1) by the proposition

$$\textit{Spin} \textbf{ possibly changes } \textit{Loaded}.$$

This domain has infinitely many models. In fact, the set of its models has the cardinality of the continuum. To select a specific model, we need to specify, for every string of actions that ends with *Spin*, whether or not *Loaded* is true after these actions are executed sequentially from the initial situation.

Example 3. A part of a table is painted white, and a part black. There are two actions: picking up a block and dropping it onto the table; the latter may put the block entirely within the white region, or entirely within the black region, or touching both the white and black regions. This example was suggested by Ray Reiter (personal communication, March 11, 1992). It can be described in $\mathcal{AR}$ using two frame fluents, *White* and *Black*, which express that the block is, at least partially, on the corresponding part of the table:

$$\begin{array}{l} \textit{PickUp}\ \textbf{causes}\ \neg \textit{White} \wedge \neg \textit{Black}, \\ \textit{Drop}\ \textbf{causes}\ \textit{White} \vee \textit{Black}, \\ \textit{Drop}\ \textbf{possibly changes}\ \textit{White}, \\ \textit{Drop}\ \textbf{possibly changes}\ \textit{Black}. \end{array}$$

The second proposition shows that the values of fluents after the execution of *Drop*, although not predetermined by their values in the previous situation, are not totally arbitrary either. This domain also has infinitely many models and the set of its models has the cardinality of the continuum.

Example 4. (This is an enhancement of an example from Fangzhen Lin.) In the following description of throwing dice, *Top* is a fluent with the domain $\{1, \ldots, 6\}$, and *Even* is a propositional fluent. The former is in the frame, and the latter is not; a constraint explicitly defines *Even* in terms of *Top*:

$$\begin{array}{l} \textbf{always}\ \textit{Even} \equiv (\textit{Top}\ \textbf{is}\ 2 \vee \textit{Top}\ \textbf{is}\ 4 \vee \textit{Top}\ \textbf{is}\ 6), \\ \textit{Throw}\ \textbf{possibly changes}\ \textit{Top}. \end{array}$$

This domain description also has $2^{\aleph_0}$ models, one for each infinite sequence of "outcomes" $1, \ldots, 6$.

7.2.3 Semantics

A *valuation* is a function that is defined on the set of fluents and maps each fluent F to an element of $\textit{Dom}_F$. A valuation σ can be extended to atomic formulae in a standard way:

$$\sigma(F\ \textbf{is}\ V) = \begin{cases} \textit{True}, & \text{if } \sigma(F) = V, \\ \textit{False}, & \text{otherwise.} \end{cases}$$

It can be further extended to arbitrary formulae according to the truth tables of propositional logic.

Consider a domain D. A valuation σ is a *state* if, for every constraint (7.4) in D, $\sigma(C) = \textit{True}$. The semantics of $\mathcal{AR}$ shows how the effect propositions of D define a nondeterministic transition system with this set of states, whose input symbols are actions. We will describe this transition system by a function *Res* that maps an action and a state to a set of states. The elements of *Res*(A, σ) are, intuitively, the states that differ from σ as dictated by the determinate effect propositions for A, but, at the same time, do not differ from σ "too much."

As a preliminary step, define $Res_0(A, \sigma)$ to be the set of states σ' such that, for each determinate effect proposition (7.2) in D, $\sigma'(C) = True$ whenever $\sigma(P) = True$. The set *Res*(A, σ) will be defined as the subset of $Res_0(A, \sigma)$ whose elements are "close" to σ.

In order to make this precise, the following notation is needed. For any action A and any states σ, σ', $New_A(\sigma, \sigma')$ is the set of formulae

$$F \textbf{ is } \sigma'(F)$$

such that

- F is in the frame and $\sigma'(F) \neq \sigma(F)$, or
- for some indeterminate effect proposition (7.3) in D, $\sigma(P) = True$.

The condition $\sigma'(F) \neq \sigma(F)$ expresses that F **is** $\sigma'(F)$ is a "new fact" that becomes true if the execution of A in state σ results in state σ'. The set $New_A(\sigma, \sigma')$ includes such "new facts" for all frame fluents F. (If F is not a frame fluent then it is not expected to keep its old value after performing an action, so that the change in its value is disregarded.) On the other hand, if some indeterminate effect proposition allows F to change, we treat its value in state σ' as "new" even if it happens to coincide with the value of F in state σ. (This definition of $New_A(\sigma, \sigma')$ is based on a suggestion made by Enrico Giunchiglia).

Now the transition function *Res* corresponding to D is defined as follows: *Res*(A, σ) is the set of states $\sigma' \in Res_0(A, \sigma)$ for which $New_A(\sigma, \sigma')$ is minimal relative to set inclusion—in other words, for which there is no $\sigma'' \in Res_0(A, \sigma)$ such that $New_A(\sigma, \sigma'')$ is a proper subset of $New_A(\sigma, \sigma')$.

The definition of a model and of entailment is the same as in Section 4.4.

7.3 Relation between $\mathcal{AR}$ and $\mathcal{AR}^-$

Note that any domain description in $\mathcal{AR}^-$ can be viewed as a propositional domain description (without indirect effect propositions) in $\mathcal{AR}$. The following theorem shows that there is a close correspondence between the models of $\mathcal{AR}^-$ and the models of $\mathcal{AR}$.

Theorem 7.1 *Let D be a propositional domain description without indirect effect propositions in the language $\mathcal{AR}$. M is a model of D in the sense of $\mathcal{AR}$ iff M is a model of D in the sense of $\mathcal{AR}^-$.*

7.4 Some Theorems about Domain Descriptions in $\mathcal{AR}$

Note that all the properties of domain descriptions in $\mathcal{AR}_0$, from Section 4.5, namely replacement (Theorem 4.1), explicit definitions (Theorem 4.2), elimination of constraints (Theorem 4.3), and restricted monotonicity (Theorem 4.4) continue to hold for domain descriptions in $\mathcal{AR}$ (The proofs are similar and are omitted). We will now examine two additional properties of domains in $\mathcal{AR}$. The proofs of these properties can be found in [Giunchiglia *et al.*, 1995].

7.4.1 Elimination of Nonpropositional Fluents

Let D be a domain description, and let S be a subset of fluents in the language of D such that every fluent in S has finitely many values.

The language of the domain description D_S is obtained from the language of D by replacing each fluent F in S with the set of new propositional fluents $\{F^V \mid V \in Dom_F\}$. A new fluent F^V is included in the frame if F was in the frame.

For any fluent F in the language of D, let F_S be the set of fluents in the language of D_S defined as follows:

$$F_S = \begin{cases} \{F^V \mid V \in Dom_F\} & \text{if } F \in S, \\ \{F\} & \text{otherwise.} \end{cases}$$

If C is a formula in the language of D, C_S stands for the formula obtained from C by replacing all atomic parts F **is** V such that $F \in S$ with F^V. If α is a constraint, a value proposition or a determinate effect proposition in the language of D, α_S is obtained from α by the same process.

The propositions of D_S are formed by

- replacing each constraint, value proposition or determinate effect proposition α in D with α_S;
- replacing each indeterminate effect proposition (7.3) in D with the set of indeterminate effect propositions

  $$A \textbf{ possibly changes } F' \textbf{ if } P_S$$

  for all $F' \in F_S$,
- adding, for every fluent $F \in S$ and every pair of distinct values $V_1, V_2 \in Dom_F$, the constraints

  $$\textbf{always } \neg(F^{V_1} \wedge F^{V_2})$$

and

$$\textbf{always} \bigvee_{V \in Dom_F} F^V.$$

Theorem 7.2 *A value proposition*

$$C \textbf{ after } \overline{A}$$

is entailed by D if and only if

$$C_S \textbf{ after } \overline{A}$$

is entailed by D_S.

In this sense, nonpropositional fluents with finite domains are not necessary. Any domain description involving only fluents with finite domains can be effectively reduced to a domain description involving only propositional fluents.

As an illustration of this theorem, let D be the domain description consisting of propositions (7.5) and (7.6), and let $S = \{\mathit{Orientation}\}$. This description D entails

$$\mathit{Orientation}\ \textbf{is}\ (k \bmod N) \quad \textbf{after} \quad \underbrace{\mathit{Turn};\ldots;\mathit{Turn}}_{k}.$$

Consequently, the proposition

$$\mathit{Orientation}^{(k \bmod N)} \quad \textbf{after} \quad \underbrace{\mathit{Turn};\ldots;\mathit{Turn}}_{k}$$

is entailed by the following domain description D_S:

$$\begin{array}{l}
\mathit{Turn}\ \textbf{causes}\ \mathit{Orientation}^{(i+1 \bmod N)}\ \textbf{if}\ \mathit{Orientation}^{i} \\
\qquad\qquad (0 \leq i < N), \\
\textbf{initially}\ \mathit{Orientation}^{0}, \\
\textbf{always}\ \neg(\mathit{Orientation}^{i} \wedge \mathit{Orientation}^{j}) \\
\qquad\qquad (0 \leq i < j < N), \\
\textbf{always}\ \mathit{Orientation}^{0} \vee \ldots \vee \mathit{Orientation}^{N-1}.
\end{array}$$

7.4.2 Indeterminate Effect Propositions

In this section, we give a theorem that shows that the transition function for a domain description that includes an indeterminate effect proposition can be characterized in terms of the transition functions for a set of closely related domain descriptions that do not include that indeterminate effect propostion.

Let D be a domain description and S a subset of the indeterminate effect propositions in D. D_S is defined as the set of domain descriptions whose elements are obtained from D replacing each proposition A **possibly changes** F **if** P in S with a proposition A **causes** $(F$ **is** $V)$**if** P, where $V \in Dom_F$. We will denote Res computed in a particular domain description by superscripting it with the name of the domain description, as in $Res^D(A, \sigma)$.

Theorem 7.3 *Let D be a domain description and S a subset of the indeterminate effect propositions in D. For any action A and state σ of D, $Res^D(A, \sigma) = \cup_{D' \in D_S} Res^{D'}(A, \sigma)$.*

Consider for example the domain in which we have an action of picking up a coin and tossing it. This domain can be described in $\mathcal{AR}$ by the following domain description D:

always *Head* $\equiv \neg$*Tail*,
Toss **possibly changes** *Head*.

If $S = \{$*Toss* **possibly changes** *Head*$\}$, we have that D_S consists of the domain description D_1

always *Head* $\equiv \neg$*Tail*,
Toss **causes** $($*Head* **is** *True*$)$.

and D_2

always *Head* $\equiv \neg$*Tail*,
Toss **causes** $($*Head* **is** *False*$)$.

According to Theorem 7.3, the result of executing *Toss* in state σ in D is the union of the set of states resulting from the execution of *Toss* in state σ in D_1 and in D_2.

Both D_1 and D_2 have two models each (depending on whether *Head* or *Tail* was initially true), while D has infinitely many models. In addition, the models of D_1 and D_2 are also models of D. However, in general, there is no relation between the models of elements of D_S and the models of D. To see this, let D be the domain consisting of the following propositions:

always $\neg(F_1 \wedge F_2)$,
A **causes** $F_1 \wedge \neg F_2$ **if** $\neg F_1 \wedge \neg F_2$,
A **possibly changes** F_1 **if** $F_1 \wedge \neg F_2$,
A **possibly changes** F_2 **if** $F_1 \wedge \neg F_2$.

Let S be the set consisting of both indeterminate effect propositions and let D' (an element of D_S) be

always $\neg(F_1 \wedge F_2)$,
A **causes** $F_1 \wedge \neg F_2$ **if** $\neg F_1 \wedge \neg F_2$,
A **causes** F_1 **if** $F_1 \wedge \neg F_2$,
A **causes** F_2 **if** $F_1 \wedge \neg F_2$.

The following is a model of D': Ψ is defined only on ϵ and A and it is defined on these as follows:

$$\begin{array}{rcl} \Psi(\epsilon) & = & \emptyset \\ \Psi(A) & = & \{F_1\} \end{array}$$

However, this is not a model of D.

7.5 Discussion

It is interesting to compare release propositions [Kartha and Lifschitz, 1994] with indeterminate effect propositions. Note that though release propositions and indeterminate effect propositions are syntactically similar, the semantics given to them in $\mathcal{AR}_0$ and $\mathcal{AR}$ differ. To see this, consider the following two domain descriptions in $\mathcal{AR}_0$ and $\mathcal{AR}$:

always $F \equiv G$,
A **releases** F.

always $F \equiv G$,
A **possibly changes** F.

In both domains, F and G are fluents that belong to the frame. Let σ_0 be the state that maps both F and G to *False* and σ_1 be the state that maps both of them to *True*. For the domain in $\mathcal{AR}_0$,

$$\begin{array}{rcl} Res_0(A, \sigma_0) & = & \{\sigma_0, \sigma_1\} \\ Res(A, \sigma_0) & = & \{\sigma_0\} \end{array}$$

However, for the domain in $\mathcal{AR}$,

$$\begin{array}{rcl} Res_0(A, \sigma_0) & = & \{\sigma_0, \sigma_1\} \\ Res(A, \sigma_0) & = & \{\sigma_0, \sigma_1\} \end{array}$$

This example shows that **possibly changes** and **releases** behave differently in the presence of constraints. Is the situation different for domains without constraints? The following domains illustrate that this is not the case:

A **releases** F,
A **causes** $\neg F \vee G$.

A **possibly changes** F,
A **causes** $\neg F \vee G$.

As before, both F and G belong to the frame (in both domains). Since there are no constraints, the set of states is the set of all functions from the set $\{F, G\}$ to the set

$\{True, False\}$. Let us agree to identify any such function by the subset of the fluents mapped to *True*. Thus, let $\sigma_0, \sigma_1, \sigma_2$ and σ_3 denote respectively the four functions $\emptyset$, $\{F\}$,$\{G\}$ and $\{F, G\}$.

For the domain in $\mathcal{AR}_0$,

$$\begin{array}{rcl} Res_0(A, \sigma_0) & = & \{\sigma_0, \sigma_2, \sigma_3\} \\ Res(A, \sigma_0) & = & \{\sigma_0\}. \end{array}$$

However, for the domain in $\mathcal{AR}$,

$$\begin{array}{rcl} Res_0(A, \sigma_0) & = & \{\sigma_0, \sigma_2, \sigma_3\} \\ Res(A, \sigma_0) & = & \{\sigma_0, \sigma_3\}. \end{array}$$

Thus, we see that propositional $\mathcal{AR}$ domains with indirect effect propositions cannot be identified with $\mathcal{AR}_0$ domains with the indirect effect propositions replaced by release propositions.

7.6 Proofs

7.6.1 Proof of Theorem 7.1

In this proof, we need to talk about the sets Res_0 and Res viewing the domain D as a domain description in $\mathcal{AR}$ and $\mathcal{AR}^-$. For clarity, we will denote these sets when we view D as a domain description in $\mathcal{AR}^-$ by Res_0^- and Res^-.

It is clear that the states D are the same whether it is viewed as a domain description in $\mathcal{AR}$ or $\mathcal{AR}_0$ and that for any action A and state σ, $Res_0(A, \sigma) = Res_0^-(A, \sigma)$. Hence, it suffices to show that $Res(A, \sigma) = Res^-(A, \sigma)$. This is an easy consequence of the following lemma.

Lemma 7.1 *Let σ, σ_1 and σ_2 be states of D. Then*

$$\mathit{New}_A(\sigma, \sigma_1) \subset \mathit{New}_A(\sigma, \sigma_1) \;\Leftrightarrow\; \rho(\sigma, \sigma_1) \cap \mathbf{Fr} \subset \rho(\sigma, \sigma_2) \cap \mathbf{Fr}$$

Proof. Using the fact that D contains no indeterminate effect propositions (viewing it as a domain in $\mathcal{AR}$), we see that for any states σ and σ', $\mathit{New}_A(\sigma, \sigma')$ is the set $\{F \textbf{ is } \sigma'(F) \mid \sigma(F) \neq \sigma'(F)\} \cap \mathbf{Fr}$.

Thus, we need to show that

$$\begin{array}{rcl} \{F \textbf{ is } \sigma_1(F) \mid \sigma(F) \neq \sigma_1(F)\} \cap \mathbf{Fr} & \subset & \{F \textbf{ is } \sigma_2(F) \mid \sigma(F) \neq \sigma_2(F)\} \cap \mathbf{Fr} \\ & \Leftrightarrow & \\ \{F \mid \sigma(F) \neq \sigma_1(F)\} \cap \mathbf{Fr} & \subset & \{F \mid \sigma(F) \neq \sigma_2(F)\} \cap \mathbf{Fr} \end{array} \tag{7.7}$$

Note that (7.7) follows from

$$\{F \textbf{ is } \sigma_1(F) \mid \sigma(F) \neq \sigma_1(F)\} \quad \subseteq \quad \{F \textbf{ is } \sigma_2(F) \mid \sigma(F) \neq \sigma_2(F)\} \tag{7.8}$$

$$\Leftrightarrow$$

$$\{F \mid \sigma(F) \neq \sigma_1(F)\} \quad \subseteq \quad \{F \mid \sigma(F) \neq \sigma_2(F)\} \tag{7.9}$$

and

$$\begin{array}{rcl} \{F \textbf{ is } \sigma_1(F) \mid \sigma(F) \neq \sigma_1(F)\} & = & \{F \textbf{ is } \sigma_2(F) \mid \sigma(F) \neq \sigma_2(F)\} \\ & \Leftrightarrow & \\ \{F \mid \sigma(F) \neq \sigma_1(F)\} & = & \{F \mid \sigma(F) \neq \sigma_2(F)\} \end{array}$$

In fact, it suffices to prove the former equivalence, since the latter follows from it by symmetry. We now turn to a proof of the former.

Let the sets A, B, A' and B' be defined as follows:

$$\begin{array}{rcl} A & = & \{F \textbf{ is } \sigma_1(F) \mid \sigma(F) \neq \sigma_1(F)\} \\ B & = & \{F \textbf{ is } \sigma_2(F) \mid \sigma(F) \neq \sigma_2(F)\} \\ A' & = & \{F \mid \sigma(F) \neq \sigma_1(F)\} \\ B' & = & \{F \mid \sigma(F) \neq \sigma_2(F)\} \end{array}$$

Left to Right: Assume (7.8). Let F be an element of A'. Then $\sigma(F) \neq \sigma_1(F)$. Thus, F **is** $\sigma_1(F)$ is an element of A and hence of B. That is, $\sigma_1(F) = \sigma_2(F)$. This, in combination with $\sigma(F) \neq \sigma_1(F)$ yields that $\sigma(F) \neq \sigma_2(F)$, and hence F is an element of B'.

Right to Left: Assume (7.9). Let F **is** $\sigma_1(F)$ be an element of A. Then $\sigma(F) \neq \sigma_1(F)$. Thus F in an element of A' and hence of B'. That is, $\sigma(F) \neq \sigma_2(F)$ and hence F **is** $\sigma_2(F)$ is an element of B. Since F is propositional, $\sigma_1(F) = \sigma_2(F)$ and it follows that F **is** $\sigma_1(F)$ is an element of B. □

Chapter 8

A sound and complete translation from $\mathcal{AR}$

8.1 Introduction

In this chapter, we give a sound and complete translation from $\mathcal{AR}$ based on a novel formalization of actions using nested abnormality theories. This formalization makes essential use of the main feature of nested abnormality theories—that the effects of various circumscriptions are confined to enclosing blocks. We also investigate the applicability of this formalization to automating reasoning about actions.

The key idea underlying this formalization is closest to that of Winslett [1988] and can be explained roughly as follows. This formalization is such that there are only two relevant situations that are taken into account while considering the effects of an action A in a situation s—s itself and the situation that results by performing the action A in s. In particular, we do not consider the actions which lead to s from the initial situation, nor do we consider what happens after A is performed.

Section 8.2 presents the translation from $\mathcal{AR}$ using the new formalization. Section 8.3 explores the applicability of this formalization to automating reasoning about actions.

8.2 A Sound and Complete Translation from $\mathcal{AR}$

In this section, we show how to transform a domain description in $\mathcal{AR}$ into a nested abnormality theory. We restrict attention to *finite* domain descriptions, that is, to domain descriptions with finitely many fluents, actions and propositions.

8.2.1 Language

In the nested abnormality theories described below, the underlying language L has variables of four sorts: for values $v, v_1, v_2, \ldots$, actions $a, a_1, a_2, \ldots$, situations $s, s_1, s_2, \ldots$ and "aspects." As in [McCarthy, 1986], aspects will be used to distinguish between different kinds of abnormality.

Consider a finite domain description D. By **F** we will denote the set of fluents in the language of D, by **Fr** the set of frame fluents, by **V** the set of values, and by **A** the set of actions. The language L has the following object constants:

- the elements of **V**, representing values,
- the elements of **A**, representing actions,
- S_0 and $\perp$, representing situations,
- for all $F \in$ **Fr**, the constants $\widetilde{F}$, representing aspects.

Intuitively, the situation constant $\perp$ represents the value "undefined", so that the assertion that a is not executable in situation s can be expressed by

$$Result(a, s) = \perp.$$

Every $F \in$ **F** is included in L as a constant representing a function from situations to values. Also, L includes the binary function $Result$, as usual in the situation calculus. Finally, L has the constants F_R for all $F \in$ **F** and $Poss$ that will be "explicitly defined" by the axioms

$$F_R(a, s) = F(Result(a, s)),$$
$$Poss(a, s) \equiv Result(a, s) \neq \perp.$$

Note that "formulae" in the sense of $\mathcal{AR}$ are not among the formulae of the language L. To avoid confusion, we will refer to the formulae in the sense of $\mathcal{AR}$ as "domain formulae." For any domain formula C and any situation term ζ, by $T(C, \zeta)$ we will denote the formula obtained from C as the result of replacing each atomic part F **is** V by $F(\zeta) = V$. For instance,

$$T(\neg(\textit{Orientation}\ \mathbf{is}\ 0), s)$$

stands for

$$\textit{Orientation}(s) \neq 0.$$

If α is an action term then $T_R(C, \alpha, \zeta)$ will stand for the formula obtained from C as the result of replacing each atomic part F **is** V by $F_R(\alpha, \zeta) = V$.

For any string of actions $A_1 \ldots A_m$, by $[A_1 \ldots A_m]$ we will denote the ground term

$$Result(A_m, Result(A_{m-1}, \ldots, Result(A_1, S_0), \ldots)).$$

8.2.2 Two Defaults

Our translation γ from $\mathcal{AR}$ into nested abnormality theories uses two defaults. The first default says that an action is normally executable in a situation. We express this by

$$\neg Ab(a, s) \supset Poss(a, s). \tag{8.1}$$

The second default is the *commonsense law of inertia*, which expresses that the value of a frame fluent normally remains unchanged after performing an action. The formulae describing how actions change the world represent exceptions to this default. The commonsense law of inertia is expressed by the set of formulae

$$Poss(a, s) \wedge v = F_R(a, s) \wedge \neg Ab(\widetilde{F}, v, a, s) \supset v = F(s),$$

one for each F in **Fr**. This set of formulae will be denoted by *LI*.

As we will see, the nested abnormality theory γD for a finite domain D consists of several formulae, including *LI* and (8.1) arranged into a system of blocks. We should not worry about the fact that *Ab* has four arguments in *LI* and only two arguments in (8.1): these formulae will appear in two different blocks.

8.2.3 Translating Propositions

We will define how to construct, for each proposition P in D, its translation. These formulae will be included in the translation γD of D.

If P is a value proposition C **after** $\overline{A}$, then γP is

$$[\overline{A}] \neq \perp \wedge T(C, [\overline{A}]).$$

The first conjunct expresses that the sequence of actions $\overline{A}$ is executable. For instance, the proposition

Orientation **is** 5 **after** *Turn*

is translated as

$$Result(Turn, S_0) \neq \perp \wedge Orientation(Result(Turn, S_0)) = 5.$$

If P is a determinate effect proposition A **causes** C **if** P, then γP is

$$T(P, s) \wedge Poss(A, s) \quad \supset \; T_R(C, A, s).$$

For instance, the proposition

Turn **causes** $($*Orientation* **is** $6)$ **if** $($*Orientation* **is** $5)$

is translated as

$$\mathit{Orientation}(s) = 5 \wedge \mathit{Poss}(\mathit{Turn}, s) \supset \mathit{Orientation}_R(\mathit{Turn}, s) = 6.$$

If P is an indeterminate effect proposition A **possibly changes** F **if** P, then γP is

$$T(P, s) \supset \mathit{Ab}(\widetilde{F}, F_R(A, s), A, s).$$

This formula will accompany the commonsense laws of inertia *LI*, so that the law of inertia will be disabled in application to F when A is executed in the presence of the precondition P. For instance, the proposition

Throw **possibly changes** *Top*

is translated as

$$\mathit{Ab}(\widetilde{\mathit{Top}}, \mathit{Top}_R(\mathit{Throw}, s), \mathit{Throw}, s)$$

(if we disregard the trivial antecedent $T(\mathit{True}, s)$, which equals *True*).

It remains to describe how constraints are translated. In $\mathcal{AR}$, constraints play two different roles: they determine the set of states, and they also determine the indirect effects of actions. Accordingly, in the nested abnormality theory γD every constraint of D will be represented by two formulae. If P is a constraint **always** C, then we will denote the formula $T(C, s)$ by γP, and the formula $T_R(C, a, s)$ by $\gamma_R P$. For instance, if P is

always *Even* $\equiv$ (*Top* **is** $2 \vee$ *Top* **is** $4 \vee$ *Top* **is** 6)

then γP is

$$\begin{array}{l} \mathit{Even}(s) = \mathit{True} \\ \qquad \equiv (\mathit{Top}(s) = 2 \vee \mathit{Top}(s) = 4 \vee \mathit{Top}(s) = 6), \end{array}$$

and $\gamma_R P$ is

$$\begin{array}{l} \mathit{Even}_R(a, s) = \mathit{True} \\ \equiv (\mathit{Top}_R(a, s) = 2 \vee \mathit{Top}_R(a, s) = 4 \vee \mathit{Top}_R(a, s) = 6). \end{array}$$

8.2.4 Definition of γD

Now we are ready to define the representation γD of a finite domain D as a nested abnormality theory.

Let D_v, D_d and D_i be the parts of D consisting of its value propositions, determinate effect propositions and indeterminate effect propositions. By D_c we denote the set of constraints that consists of all constraints of D and the constraints

$$\textbf{always} \bigvee_{V \in \mathit{Dom}_F} F \textbf{ is } V$$

for all $F \in \mathbf{F}$. $\mathbf{F}_R$ stands for the list of all fluents F_R.

The axioms of γD are as follows.

Group 1. Unique names axioms: $c_1 \neq c_2$ for all pairs c_1, c_2 of distinct object constants of the same sort. For instance, this group includes the axiom $S_0 \neq \bot$.

Group 2. Domain closure axioms:

$$\bigvee_{V \in \mathbf{V}} v = V,$$

$$\bigvee_{A \in \mathbf{A}} a = A.$$

Group 3. Translations of the constraints:

$$\gamma P \qquad (P \in D_c).$$

Group 4. Definitions of *Poss* and $\mathbf{F}_R$:

$$\begin{array}{l} Poss(a,s) \equiv Result(a,s) \neq \bot, \\ F_R(a,s) = F(Result(a,s)) \qquad (F \in \mathbf{F}). \end{array}$$

Group 5. Translations of the value propositions:

$$\gamma P \qquad (P \in D_v).$$

Group 6. Characterization of the effects of actions:

$$\begin{array}{l} \{\mathbf{F}_R : \\ \quad \mathit{LI}, \\ \quad \gamma P \qquad (P \in D_i), \\ \quad \{\mathit{Poss}, \mathbf{F}_R : \\ \qquad \neg \mathit{Ab}(a,s) \supset \mathit{Poss}(a,s), \\ \qquad \neg \mathit{Poss}(a,\bot), \\ \qquad \gamma P \qquad (P \in D_d), \\ \qquad \gamma_R P \qquad (P \in D_c) \\ \quad \} \\ \}. \end{array}$$

The two nested circumscriptions represented by this block are, of course, the main part of the theory. The inner circumscription tells us that an action can be executed unless this is prohibited by the determinate effect propositions and constraints of D. The outer circumscription encodes the idea of inertia. The nesting of blocks reflects our intention to decide first which actions can be executed, and then what the effects of actions are.

Note the use of $\mathbf{F}_R$ instead of $Result$ in the range of the outer circumscription. The intuition behind this style of describing actions can roughly be explained as follows: when we want to consider the effect of an action a on a fluent in a situation s, the only two situations that are of interest are the situation s and the situation obtained by performing the action a in s—for instance, we do not want to consider the sequence of actions that lead to situation s nor do we wish to consider what happens afterward. In other words, we wish to reason in a "local" fashion, without considering the history. The new fluents achieve this by "removing" the dependence of the $Result$ function. In other words, in the range of the circumscription, we accept the local "theory update" view, as in Winslett's work [Winslett, 1988].

The simplification achieved here in comparison with the formalization presented in Chapter 6 is that the existence of situations principle is not needed any more. This is an important advantage, because the formalization of that principle required the use of higher-order circumscription.

8.2.5 Soundness and Completeness

Consider a finite domain D. Let Ψ be a model of D, and let M be a model of γD. We say that M is *similar* to Ψ if, for every value proposition P, P is true in Ψ if and only if M satisfies γP.

Theorem 8.1 *Let D be a finite domain. For any model Ψ of D, there exists a model M of γD similar to Ψ. For any model M of γD, there exists a model Ψ of D such that M is similar to Ψ.*

The following corollary expresses the soundness and completeness of the translation.

Corollary. *For any finite domain D and any value proposition P, γD entails γP if and only if D entails P.*

8.3 Automating Reasoning About Actions

In this section, we investigate the applicability of the formalization presented in Section 8.2 to automating reasoning about actions. Since the formalization makes use of the main feature of nested abnormality theories—that the effects of various circumscriptions are confined to enclosing blocks—the circumscriptions involved become rather simple. We present examples to illustrate that in many cases, we will be able to eliminate all their second order quantifiers using existing techniques for computing circumscriptions. One of the two methods for computing circumscriptions employed here is a generalization of predicate completion [Lifschitz, 1993a] and the other is based on the SCAN algorithm [Gabbay and Ohlbach, 1992].

We will formalize the examples using NATs. The formalization is essentially the same as that given in Section 8.2. The main differences are that in these examples, we focus attention on propositional action domains and that we do not try to characterize $Poss$ using another NAT.

Example 1. Consider a blocks world domain in which the only actions possible are painting blocks with different colours. To describe this domain, we first introduce an action term $\mathit{paint}(x, y)$ that stands for the action of painting block x with colour y. The following effect axiom describes what this action does:

$$Poss(\mathit{paint}(x, y), s) \supset \mathit{colour}(x, y, Result(\mathit{paint}(x, y), s)).$$

The only ramification constraint that we have for this domain is that a block can have just one colour. This constraint is expressed by the simple state formula

$$\mathit{colour}(x, y_1, s) \wedge \mathit{colour}(x, y_2, s) \supset y_1 = y_2. \tag{8.2}$$

This action domain can be formalized as the following nested abnormality theory.

$$
\begin{array}{l}
(i) \;\; \mathit{paint}(x_1, y_1) = \mathit{paint}(x_2, y_2) \supset (x_1 = x_2 \wedge y_1 = y_2), \\
(ii) \;\; \mathit{Aspect}_{colour}(x_1, y_1) = \mathit{Aspect}_{colour}(x_2, y_2) \supset (x_1 = x_2 \wedge y_1 = y_2), \\
(iii) \; \mathit{colour}(x, y_1, s) \wedge \mathit{colour}(x, y_2, s) \supset y_1 = y_2, \\
(iv) \;\; \mathit{colour}_R(x, y, a, s) \equiv \mathit{colour}(x, y, Result(a, s)), \\
\{\mathit{colour}_R \; : \\
\quad (v) \;\; Poss(a, s) \wedge \neg \mathit{Ab}(\mathit{Aspect}_{colour}(x, y), a, s) \supset \\
\qquad\qquad\qquad\qquad\qquad \mathit{colour}_R(x, y, a, s) \equiv \mathit{colour}(x, y, s), \\
\quad (vi) \;\; Poss(a, s) \supset [a = \mathit{paint}(x, y) \supset \mathit{colour}_R(x, y, a, s)], \\
\quad (vii) \; \mathit{colour}_R(x, y_1, a, s) \wedge \mathit{colour}_R(x, y_2, a, s) \supset y_1 = y_2 \\
\}.
\end{array}
$$

Example 2. The following NAT formalizes an extended version of the Yale shooting problem from [Baker, 1991].

$$
\begin{array}{l}
\mathit{UNA}, \\
\mathit{Walking}(s) \supset \mathit{Alive}(s), \\
\mathit{Walking}_R(a,s) \equiv \mathit{Walking}(\mathit{Result}(a,s)), \\
\mathit{Alive}_R(a,s) \equiv \mathit{Alive}(\mathit{Result}(a,s)), \\
\mathit{Loaded}_R(a,s) \equiv \mathit{Loaded}(\mathit{Result}(a,s)), \\
\{\mathit{Walking}_R, \mathit{Alive}_R, \mathit{Loaded}_R : \\
\quad \mathit{Poss}(a,s) \wedge \neg \mathit{Ab}(Aspect_{Walking}, a, s) \supset \mathit{Walking}_R(a,s) \equiv \mathit{Walking}(s), \\
\quad \mathit{Poss}(a,s) \wedge \neg \mathit{Ab}(Aspect_{Alive}, a, s) \supset \mathit{Alive}_R(a,s) \equiv \mathit{Alive}(s), \\
\quad \mathit{Poss}(a,s) \wedge \neg \mathit{Ab}(Aspect_{Loaded}, a, s) \supset \mathit{Loaded}_R(a,s) \equiv \mathit{Loaded}(s), \\
\quad \mathit{Poss}(\mathit{Load}, s) \supset \mathit{Loaded}_R(\mathit{Load}, s), \\
\quad \mathit{Poss}(\mathit{Shoot}, s) \wedge \mathit{Loaded}(s) \supset \neg \mathit{Alive}_R(\mathit{Shoot}, s), \\
\quad \mathit{Poss}(\mathit{Shoot}, s) \supset \neg \mathit{Loaded}_R(\mathit{Shoot}, s), \\
\quad \mathit{Walking}_R(a,s) \supset \mathit{Alive}_R(a,s) \\
\}.
\end{array}
$$

In the NAT above, *UNA* stands for the uniqueness names axioms for aspects and actions. For instance, it includes axioms such as

$$
\begin{array}{l}
Aspect_{Walking} \neq Aspect_{Alive}, \\
\mathit{Load} \neq \mathit{Shoot}.
\end{array}
$$

Example 3. Consider a table divided into three sectors F, G and H. A block is always in exactly one of these three locations. There is an action A, which, if performed when the block is in location F, moves it out of that location. Hence, after the action is performed, the block is in location G or H, but we do not know which. Thus, this is a domain where the indirect effects are nondeterministic.

The corresponding NAT is as follows:

UNA,
$F(s) \vee G(s) \vee H(s)$,
$\neg F(s) \vee \neg G(s)$,
$\neg F(s) \vee \neg H(s)$,
$\neg G(s) \vee \neg H(s)$,
$F_R(a, s) \equiv F(\mathit{Result}(a, s))$,
$G_R(a, s) \equiv G(\mathit{Result}(a, s))$,
$H_R(a, s) \equiv H(\mathit{Result}(a, s))$,
$\{F_R, G_R, H_R :$
$\quad \mathit{Poss}(a, s) \wedge \neg \mathit{Ab}(Aspect_F, a, s) \supset F_R(a, s) \equiv F(s)$,
$\quad \mathit{Poss}(a, s) \wedge \neg \mathit{Ab}(Aspect_G, a, s) \supset G_R(a, s) \equiv G(s)$,
$\quad \mathit{Poss}(a, s) \wedge \neg \mathit{Ab}(Aspect_H, a, s) \supset H_R(a, s) \equiv H(s)$,
$\quad \mathit{Poss}(A, s) \wedge F(s) \supset \neg F_R(A, s)$,
$\quad F_R(a, s) \vee G_R(a, s) \vee H_R(a, s)$,
$\quad \neg F_R(a, s) \vee \neg G_R(a, s)$,
$\quad \neg F_R(a, s) \vee \neg H_R(a, s)$,
$\quad \neg G_R(a, s) \vee \neg H_R(a, s)$
$\}$.

Here *UNA* stands for the uniqueness names axioms for aspects.

8.3.1 Completion

Formalizing actions using nested abnormality theories has the important advantage that the syntactic methods for computing circumscriptions [Lifschitz, 1993a] can be applied. In particular, it is possible in many cases to compute the effects of ramifications using such syntactic methods. We now illustrate this with an example.

Let Φ be the conjunction of the universal closures of three formulae (v), (vi) and (vii) of the nested abnormality theory T_{ab}^c corresponding to Example 1. By the definition of the semantics of nested abnormality theories, the block at the end of T_{ab}^c stands for the formula $\exists ab\ C(ab)$ where $C(Ab)$ stands for **CIRC**$[\Phi; Ab; \mathit{colour}_R]$. By Proposition 3 from the Appendix, this circumscription is equivalent to the conjunction of Φ and
CIRC$[\exists \mathit{colour}_R\ \Phi; Ab]$.[1]

Now, it is easy to see that the formula $\exists \mathit{colour}_R\ \ \Phi$ can be rewritten in the form $\exists \mathit{colour}_R\ [E \leq \mathit{colour}_R \wedge B(\mathit{colour}_R)]$ where E stands for the predicate expression

$$\lambda xyas\ [(Poss(a, s) \wedge a = \mathit{paint}(x, y)) \vee \\ \qquad (Poss(a, s) \wedge \mathit{colour}(x, y, s) \wedge \neg Ab(\mathit{Aspect}_{colour}(x, y), a, s))],$$

[1] To simplify notation, we use colour_R both as a predicate constant and predicate variable.

and $B(colour_R)$ stands for the conjunction of the following formulae negative relative to $colour_R$:

$$colour_R(x, y, a, s) \supset colour(x, y, s) \vee Ab(Aspect_{colour}(x, y), a, s) \vee \neg Poss(a, s),$$
$$colour_R(x, y_1, a, s) \wedge colour_R(x, y_2, a, s) \supset y_1 = y_2.$$

Using Proposition 4 from the Appendix, we get that

$$\exists colour_R[E \leq colour_R \wedge B(colour_R)]$$

is equivalent to $B(E)$. We can simplify the expression $B(E)$ as follows. Substituting the expression E into the first conjunct of $B(colour_R)$, we get the following two formulae:

$$\begin{aligned} &Poss(a, s) \wedge a = paint(x, y) \\ &\qquad \supset colour(x, y, s) \vee Ab(Aspect_{colour}(x, y), a, s) \vee \neg Poss(a, s) \end{aligned}$$

and

$$\begin{aligned} &Poss(a, s) \wedge colour(x, y, s) \wedge \neg Ab(Aspect_{colour}(x, y), a, s) \\ &\qquad \supset colour(x, y, s) \vee Ab(Aspect_{colour}(x, y), a, s) \vee \neg Poss(a, s). \end{aligned}$$

The first of these can be rewritten as

$$Poss(a, s) \wedge a = paint(x, y) \wedge \neg colour(x, y, s) \supset Ab(Aspect_{colour}(x, y), a, s) \tag{8.3}$$

and the second one reduces to *true*.

Substituting E in the second conjunct of $B(colour_R)$, we get the following four formulae:

$$Poss(a, s) \wedge a = paint(x, y_1) \wedge a = paint(x, y_2) \supset y_1 = y_2, \tag{8.4}$$

$$\begin{aligned} &Poss(a, s) \wedge a = paint(x, y_1) \wedge colour(x, y_2, s) \wedge \neg Ab(Aspect_{colour}(x, y_2), a, s) \\ &\qquad \supset y_1 = y_2, \end{aligned} \tag{8.5}$$

$$\begin{aligned} &Poss(a, s) \wedge a = paint(x, y_2) \wedge colour(x, y_1, s) \wedge \neg Ab(Aspect_{colour}(x, y_1), a, s) \\ &\qquad \supset y_1 = y_2, \end{aligned} \tag{8.6}$$

$$\begin{aligned} &Poss(a, s) \wedge colour(x, y_1, s) \wedge \neg Ab(Aspect_{colour}(x, y_1), a, s)) \wedge colour(x, y_2, s) \\ &\wedge \neg Ab(Aspect_{colour}(x, y_2), a, s) \supset y_1 = y_2. \end{aligned} \tag{8.7}$$

Note that (8.4) reduces to *true* on account of formula (i) of the NAT T^c_{ab}, and that (8.7) reduces to *true* in the presence of the constraint (iii). Also, (8.5) and (8.6) are equivalent to each other and each can be simplified to

$$Poss(a, s) \wedge a = \mathit{paint}(x, y') \wedge \mathit{colour}(x, y, s) \wedge y \neq y' \supset Ab(\mathit{Aspect}_{colour}(x, y), a, s) \tag{8.8}$$

Thus, in the presence of the other axioms of the NAT, $B(E)$ reduces to the conjunction of (8.3) and (8.8). Using Proposition 2 from the Appendix, we get that the axioms of the NAT entail that $C(Ab)$ is equivalent to the conjunction of Φ and the following explicit definition for Ab:

$$\begin{aligned} Ab(z, a, s) \equiv Poss(a, s) \wedge \exists xy\, [z = \mathit{Aspect}_{colour}(x, y) \wedge \\ ((\neg \mathit{colour}(x, y, s) \wedge a = \mathit{paint}(x, y)) \vee \\ (\mathit{colour}(x, y, s) \wedge \exists\, y'(a = \mathit{paint}(x, y') \wedge y \neq y')))]\,. \end{aligned}$$

It follows that T^c_{ab} is equivalent to the conjunction of formulae (i)–(iv), (vi), (vii) and the result of substituting this expression for *Ab* in (v). The result of this substitution is essentially the conjunction of all "frame axioms" that are needed in conjunction with the causal theory of Example 1. We see that syntactic methods of computing circumscriptions can be employed to generate the necessary frame axioms.

Similar syntactic methods can be employed for computing the the effect of the circumscriptions in Examples 2 and 3.

8.3.2 SCAN

SCAN [Gabbay and Ohlbach, 1992] is an algorithm for eliminating second-order quantifiers over predicate variables $P_1, \ldots, P_n$ in formulae of the form $\exists P_1 \ldots P_n\ \Phi$, where Φ is a first-order formula. If SCAN terminates, then the resulting formula is equivalent to the original formula.

Recall that **CIRC**$[A; P; Z]$, the circumscription of the predicate P in formula A with the functions and/or predicates in the tuple Z varied, is the sentence

$$A(P, Z) \wedge \neg \exists pz[A(p, z) \wedge p < P].$$

Hence to compute circumscriptions using SCAN, we input the formula $\exists pz[A(p, z) \wedge p < P]$ to it, negate the resulting formula and take its conjunction with the formula $A(P, Z)$. Note that if z includes function constants, SCAN cannot be applied . For instance, since Baker's circumscriptive approach [Baker, 1991] involves varying the $Result$ function, it cannot be used directly with SCAN.

An implementation of SCAN has been developed by Engel and Ohlbach[2]. We can apply this system for computing the circumscriptions for Examples 1, 2 and 3.

In Example 1, the following set of formulae is taken as input to SCAN; the variables p and R are eliminated.

```
(all x all y all xa all xs (
  (-p(asp(x,y),xa,xs) &  Poss(xa,xs))  ->
                          (c(x,y,xs) <-> R(x,y,xa,xs)))).
(all x all y all xa all xs  (
  (Poss(xa,xs) &  (xa = paint(x,y)))   -> R(x,y,xa,xs))).
(all x all y1  all y2 all xa all xs (
      ((R(x,y1,xa,xs) & R(x,y2,xa,xs)) ->    (y1 = y2))).
(all xf all xa all xs (p(xf,xa,xs)    ->  Ab(xf,xa,xs))).
(exists xf exists xa exists xs (
                          -p(xf,xa,xs) &  Ab(xf,xa,xs))).
(all x1 all y1 all x2  all y2  (
 (paint(x1,y1) =  paint(x2,y2)) ->
                               ((x1 = x2) & (y1 = y2)))).
(all x1 all y1 all x2  all y2  (
           (asp(x1,y1) =  asp(x2,y2))  ->
                               ((x1 = x2) & (y1 = y2)))).
(all x all y1  all y2  all xs (
               (c(x,y1,xs) & c(x,y2,xs)) ->  (y1 = y2))).
(all x (x = x)).
```

In the formulae above, R stands for *colour*$_R$ and c stands for *colour*. The first three formulae are direct encodings of (v), (vi) and (vii). The next two formulae express that $p < Ab$. The next four are not a part of the circumscription formula; they are included to help SCAN simplify its output. Of these, the first three are formulae (i), (ii) and (iii) of the NAT.

SCAN produces the following 14 clauses as its output (in 1.6 seconds on a Sun 4).

```
1   paint(x,y)!=paint(z,u)|x=z.
2   paint(x,y)!=paint(z,u)|y=u.
3   asp(x,y)!=asp(z,u)|x=z.
4   asp(x,y)!=asp(z,u)|y=u.
5  -c(x,y,z)| -c(x,u,z)|y=u.
```

[2]For more information on the implementation, see
http: //www.mpi-sb.mpg.de/guide/staff/ohlbach/scan/scan.html

```
6   Ab($c3,$c2,$c1).
7   x=x.
8  -Poss(paint(x,y),z)|c(x,y,z)|
    Ab(asp(x,y),paint(x,y),z).
9  -Poss(paint(x,y),$c1)| -Poss($c2,$c1)|
    c(x,y,$c1)|$c3!=asp(x,y)|$c2!=paint(x,y).
10 -Poss(paint(x,y),z)| -Poss(paint(x,u),z)|
    y=u|paint(x,u)!=paint(x,y).
11 -Poss(paint(x,y),z)| -c(x,u,z)|
    Ab(asp(x,u),paint(x,y),z)|y=u.
12 -Poss(paint(x,y),z)| -c(x,u,z)|
    Ab(asp(x,u),paint(x,y),z)|u=y.
13 -Poss($c2,$c1)| -c(x,y,$c1)|$c3!=asp(x,y)|
   -Poss(paint(x,z),$c1)|z=y|paint(x,z)!=$c2.
14 -Poss($c2,$c1)| -c(x,y,$c1)|$c3!=asp(x,y)|
   -Poss(paint(x,z),$c1)|y=z|paint(x,z)!=$c2.
```

In the set of clauses above, the expressions beginning with `$` are Skolem constants introduced by SCAN in the process of converting the input into clauses, and the symbol `|` stands for $\vee$.

By unskolemizing (that is, appending an existential quantifier to) this set of clauses, negating them, conjoining them with the axioms outside the block and simplifying them, we can get an explicit definition of *Ab* identical to the formula obtained in Section 8.3.1, and consequently the frame axioms.

Some of this process of simplification of the set of clauses and elimination of *Ab* can be automated. For instance, to eliminate the redundant clauses, SCAN can be directed to try to derive each newly generated clause from the others for a fixed amount of time. In addition, SCAN can eliminate *Ab* from the simplified set of clauses.

When the formulae corresponding the the NAT of Example 2 were input to SCAN, it produced 16 clauses in 1.8 seconds. Here again, we can further eliminate Ab and produce the frame axioms. For this example, SCAN helped in the automation of this process even better than in the previous case. For instance, consider the following clauses among the ones generated by SCAN:

```
x=L|Lo(y)| -Poss(x,y)| -Lor(x,y).
Lo(x)| -Poss(y,x)|Alr(y,x)| -Al(x).
```

In these clauses, `L` stands for *Load*, `Lo` for *Loaded*, `Lor` for $Loaded_R$, `Al` for *Alive* and `Alr` for $Alive_R$. Rewriting the first one as

$$Poss(a,s) \wedge a \neq Load \wedge \neg Loaded(s) \supset \neg Loaded(Result(a,s))$$

makes it clear that this a negative frame axiom for *Loaded*. Similarly, the second one can be rewritten as

$$Poss(a, s) \wedge \neg \mathit{Loaded}(s) \wedge \mathit{Alive}(s) \supset \mathit{Alive}(Result(a, s)),$$

a positive frame axiom for *Alive*. Frame axioms of a similar and easy to comprehend form were generated for the other fluents also.

In Example 3, SCAN was used to generate the frame axioms in a similar way.

8.4 Discussion

The translation to NATS in Section 8.2 is based on the formalization presented in [Kartha and Lifschitz, 1995]. As noted before, the idea behind this formalization is closest to that of Winslett [1988]. The main difference is that Winslett's formalization is in terms of theory update, whereas this formalization includes the full expressive power of the situation calculus. Combining the ideas of [Winslett, 1988] with the situation calculus is achieved by the use of NATs.

Another closely related direction of research is described in [Lin and Shoham, 1991] and [Lin and Reiter, 1994]. Central to the formalization presented there is a minimality condition formulated in terms of models. To obtain meaningful conclusions with their formalization, Lin, Shoham and Reiter need to impose certain consistency conditions and to include certain "tree axioms" (that impose a tree structure to the universe of situations). In contrast, the use of circumscription here is different and allows us to dispense with the consistency conditions and the tree axioms. In addition, we investigate the applicability of symbolic methods to compute the circumscription, which is a problem not considered by Winslett or by Lin, Shoham and Reiter.

We have presented evidence that this approach to formalizing actions leads, in some cases, to the circumscriptions that can be evaluated using the predicate completion method or the SCAN algorithm. One of these examples (Example 3) involves a "ternary state constraint" in the sense of Section 3.1 of [Pinto, 1994] and thus is not amenable to the methods developed there for the generation of successor state axioms in the case when all constraints are binary.

Our plans for the future include the development of an experimental program that uses SCAN for the completely automated generation of frame axioms. For this purpose, the simplification procedure used currently in SCAN will need to be enhanced.

The set of formulae produced by SCAN is equivalent to the formula input to SCAN, *if the algorithm terminates*. Hence, even if we know that a given second-order formula is equivalent to some first-order formula, there is no guarantee that SCAN will ever find it. It would be interesting to isolate classes of action domains for which it can be proved that SCAN terminates when applied to the corresponding circumscriptive theory.

Another interesting direction for future work is to use the SET-VAR method of [Bledsoe and Feng, 1993] instead of SCAN for computing the circumscriptions.

8.5 Proofs

8.5.1 Proof of Theorem 8.1

The following lemma about domains in $\mathcal{AR}$ will be useful later.

Lemma 8.1 *For a domain in $\mathcal{AR}$ with a finite number of fluent names, for any action A and state σ,*

$$Res_0(A, \sigma) = \emptyset \;\Leftrightarrow\; Res(A, \sigma) = \emptyset.$$

Proof. The left to right direction follows easily since $Res(A, \sigma)$ is a subset of $Res_0(A, \sigma)$. We will prove the contrapositive of the right to left direction. Assume that $Res_0(A, \sigma) \neq \emptyset$. Since the domain has only a finite number of fluent names, for any σ' in $Res_0(A, \sigma)$, the cardinality of $New_A(\sigma, \sigma')$ is finite. Choose a state σ' in $Res_0(A, \sigma)$ for which the cardinality of $New_A(\sigma, \sigma')$ is minimal. Clearly, σ' is an element of $Res(A, \sigma)$ and thus $Res(A, \sigma) \neq \emptyset$. □

For any valuation α of the domain D, α can be identified with the domain formula

$$\bigwedge_{F \in \mathbf{F}} F \textbf{ is } \alpha(F).$$

Then, for any situation term ζ, $T(\alpha, \zeta)$ is

$$\bigwedge_{F \in \mathbf{F}} F(\zeta) = \alpha(F).$$

The following lemma is easy to check.

Lemma 8.2 *The UNA for values and the Group 3 axiom entail that for any domain formula C of D,*

$$T(C, s) \equiv \bigvee_{\alpha : \alpha(C) = True} T(\alpha, s)$$

where α ranges over valuations of D.

Recall that for a block $\{C_1, \ldots, C_m \,:\, A_1, \ldots, A_n\}$, $F(Ab)$ denotes the formula

$$\textbf{CIRC}[\varphi A_1 \wedge \ldots \wedge \varphi A_n; Ab; C_1, \ldots, C_m].$$

When the block is referred to by a name, say B, we will denote this formula by $F_B(Ab)$.

Let *Block 1* denote the inner block of Group 6 and *Block 2* denote the outer block of Group 6. Thus, *Block 1* is a sub-block of *Block 2*.

Recall that this block includes the following formulae:

$$\neg Ab(a, s) \supset Poss(a, s), \tag{8.9}$$

$$\neg Poss(a, \bot), \tag{8.10}$$

$$\gamma P \qquad (P \in D_d), \tag{8.11}$$

and

$$\gamma_R P \qquad (P \in D_c). \tag{8.12}$$

Let

$$\Sigma = \{\langle A, \sigma\rangle \mid Res_0(A, \sigma) = \emptyset\}.$$

Define G as the following predicate expression:

$$G = \lambda a\ s[s = \bot \vee \bigvee_{\langle A,\sigma\rangle\in\Sigma} (a = A \wedge T(\sigma, s))].$$

Lemma 8.3 *Assume that the constraints are consistent. The UNA and DCA for values and actions and the Group 3 axioms entail that $F_{Block\ 1}(Ab)$ is equivalent to the conjunction of (8.9), (8.10), (8.11), (8.12) and the formula $Ab = G$.*

Proof. We will apply Proposition 5 with E chosen as G. Let B denote the conjunction of the UNA and DCA for values and actions and the Group 3 axiom. Let $A(Ab, Poss, \mathbf{F}_R)$ denote the conjunction of (8.9), (8.10), (8.11) and (8.12).

We will first show that B entails

$$A(Ab, Poss, \mathbf{F}_R) \supset G \leq Ab.$$

That is, we need to show that B and $A(Ab, Poss, \mathbf{F}_R)$ entail

$$\forall a\ s\left[s = \bot \vee \bigvee_{\langle A,\sigma\rangle\in\Sigma} (a = A \wedge T(\sigma, s)) \supset Ab(a, s)\right].$$

We will show that B and $A(Ab, Poss, \mathbf{F}_R)$ entail

$$\forall a\ s\left[s = \bot \vee \bigvee_{\langle A,\sigma\rangle\in\Sigma} (a = A \wedge T(\sigma, s)) \supset \neg Poss(a, s)\right]$$

from which the desired conclusion follows, by (8.9).

The formula $\forall a\ s(s = \bot \supset \neg Poss(a, s))$ follows from the formula (8.10) of $A(Ab, Poss, \mathbf{F}_R)$. Hence, assuming B and $A(Ab, Poss, \mathbf{F}_R)$, we only need to check that

$$T(\sigma, s) \supset \neg Poss(A, s),$$

where $\langle A, \sigma\rangle \in \Sigma$. By the definition of Σ, $Res_0(A, \sigma) = \emptyset$. That is, the set of domain formulae consisting of all constraints and all formula C such that there is a determinate effect proposition A **causes** C **if** P where $\sigma(P) = \mathsf{T}$ is inconsistent. Since $A(Ab, Poss, \mathbf{F}_R)$ includes the translations of the determinate effect propositions and constraints (namely, (8.11) and (8.12)), this means that B and $A(Ab, Poss, \mathbf{F}_R)$ entail the following formula:

$$T(\sigma, s) \wedge Poss(A, s) \supset False.$$

This formula is equivalent to

$$T(\sigma, s) \supset \neg Poss(A, s).$$

We will now show that B entails

$$A(Ab, Poss, \mathbf{F}_R) \supset \exists P\ W\ A(G, P, W). \tag{8.13}$$

Here P stands for a predicate variable identical in arity and sorts of arguments to *Poss* and W stands for a list of function variables identical in arity and sorts of arguments to members of the list $\mathbf{F}_R$. Our first task is to define P and W.

Define

$$\Sigma' = \{\langle A, \sigma\rangle \mid Res_0(A, \sigma) \neq \emptyset\}.$$

For each $\langle A, \sigma\rangle$ in Σ', let $r(A, \sigma)$ be a fixed element of $Res_0(A, \sigma)$.

The definition of P can now be given:

$$P = \bigvee_{\langle A,\sigma\rangle \in \Sigma'} (a = A \wedge T(\sigma, s) \wedge s \neq \bot),$$

We will now define $W_F(a, s)$, the element of W corresponding to F_R in $\mathbf{F}_R$. As a first step towards defining $W_F(a, s)$, let us define a predicate $U_F(a, s, v)$ as follows:

$$\begin{aligned} U_F(a, s, v) \quad = \quad & \textstyle\bigvee_{\langle A,\sigma\rangle \in \Sigma} (a = A \wedge T(\sigma, s) \wedge v = \sigma(F)) \vee \\ & \textstyle\bigvee_{\langle A,\sigma\rangle \in \Sigma'} (a = A \wedge T(\sigma, s) \wedge v = r(A, \sigma)(F)). \end{aligned}$$

We will first show that B and $A(Ab, Poss, \mathbf{F}_R)$ entail

$$\exists w \forall as\ U_F(a, s, w(a, s)). \tag{8.14}$$

By the axiom of choice, it suffices to show that

$$\forall as \exists v \; U_F(a, s, v).$$

The DCA for actions enables the proof obligation to be written as

$$\forall s \bigwedge_A \exists v \; U_F(A, s, v).$$

Fix a particular action A. Using UNA for actions, we see that we need to prove

$$\begin{array}{l} \exists v \Big[\ \bigvee_{\langle A,\sigma\rangle \in \Sigma} (T(\sigma, s) \wedge v = \sigma(F)) \vee \\ \qquad \bigvee_{\langle A,\sigma\rangle \in \Sigma'} (T(\sigma, s) \wedge v = r(A, \sigma)(F) \Big]. \end{array} \tag{8.15}$$

We will first show that

$$\begin{array}{l} \bigvee_\sigma T(\sigma, s) \quad \supset \quad \exists v \Big[\ \bigvee_{\langle A,\sigma\rangle \in \Sigma} (T(\sigma, s) \wedge v = \sigma(F)) \vee \\ \qquad\qquad\qquad\qquad \bigvee_{\langle A,\sigma\rangle \in \Sigma'} (T(\sigma, s) \wedge v = r(A, \sigma)(F) \Big]. \end{array} \tag{8.16}$$

Choose any arbitrary state σ. If $Res_0(A, \sigma) = \emptyset$, define $v = \sigma(F)$, otherwise define $v = r(A, \sigma)(F)$. This shows (8.16). The desired conclusion (8.15) follows from (8.16), because Lemma 8.2 implies that B entails the formula

$$\bigvee_\sigma T(\sigma, s), \tag{8.17}$$

where σ ranges over the set of states of D.

We define $W_F(a, s)$ as the function w in (8.14). Note that if $\langle A, \sigma\rangle \in \Sigma$

$$T(\sigma, s) \supset W_F(A, s) = \sigma(F). \tag{8.18}$$

To see this, assume that $W_F(A, s) = V$. Then $U_F(A, s, V)$ holds. This in conjunction with the fact

$$T(\sigma, s) \supset \neg T(\sigma', s),$$

for any $\sigma' \neq \sigma$ yields (8.18).

Similarly, if $\langle A, \sigma\rangle \in \Sigma'$,

$$T(\sigma, s) \supset W_F(A, s) = r(A, \sigma)(F). \tag{8.19}$$

For any domain formula C, by $T^*(C, \alpha, s)$, we will denote the formula obtained from C as the result of replacing each atomic part F **is** V by $W_F(\alpha, s) = V$, where α stands for any action term. We now are ready to prove (8.13).

We need to show that B and $A(Ab, \mathit{Poss}, \mathbf{F}_R)$ entail the following four formulae:

$$\neg G(a, s) \supset P(a, s), \tag{8.20}$$

$$\neg P(a, \bot), \tag{8.21}$$

$$T(Q, s) \wedge P(A, s) \supset T^*(C, A, s), \tag{8.22}$$

for each effect proposition in D of the form A **causes** C **if** Q,

$$T^*(C, a, s) \tag{8.23}$$

for each constraint **always** C in D_c.

We will first show that B and $A(Ab, \textit{Poss}, \mathbf{F}_R)$ entail (8.20). Assume B and $A(Ab, \textit{Poss}, \mathbf{F}_R)$. We need to show that

$$s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} (a = A \wedge T(\sigma, s)) \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma'} (a = A \wedge T(\sigma, s) \wedge s \neq \bot).$$

Using the DCA and UNA for actions, it suffices to prove for an arbitrary $A \in \mathbf{A}$ that

$$s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} T(\sigma, s) \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma'} T(\sigma, s). \tag{8.24}$$

But (8.24) directly follows from (8.17).

That B and $A(Ab, \textit{Poss}, \mathbf{F}_R)$ entail (8.21) follows easily using the definition of P.

We now need to check that B and $A(Ab, \textit{Poss}, \mathbf{F}_R)$ entail (8.22). Using the definition of $P(A, s)$ and the UNA for actions, we see that the formula (8.22) simplifies to

$$\bigvee_{\langle A,\sigma\rangle \in \Sigma'} (T(\sigma, s) \wedge s \neq \bot) \wedge T(Q, s) \supset T^*(C, A, s). \tag{8.25}$$

Consider any σ such that $\langle A, \sigma\rangle \in \Sigma'$. and assume $T(\sigma, s)$. If $\sigma(Q) = \mathsf{T}$, we get that $r(A, \sigma)(C) = \mathsf{T}$ (Recall that $r(A, \sigma) \in \textit{Res}_0(A, \sigma)$). Then the formula $T^*(C, A, s)$ follows using (8.19). If $\sigma(Q) = \mathsf{F}$,

$$T(\sigma, s) \supset \neg T(Q, s)$$

and then (8.25) follows trivially.

We need to check that B and $A(Ab, \textit{Poss}, \mathbf{F}_R)$ entail (8.23). We will show that

$$\bigvee_{\sigma} T(\sigma, s) \supset \bigwedge_{A} T^*(C, A, s) \tag{8.26}$$

from which (8.22) follows (using (8.17) and the DCA for actions).

Fix σ and A and assume $T(\sigma, s)$. If $\langle A, \sigma\rangle \in \Sigma$, using (8.18), it is easy to see that $T^*(C, A, s)$ follows (Note that $\sigma(C) = \textit{True}$ since σ is a state of the domain D). Otherwise, $T'(C, A, s)$ again follows easily (using $r(A, \sigma)(C) = \mathsf{T}$ and (8.19)). $\square$

Corollary. *Assume that the constraints are consistent. The UNA and DCA for values and actions and the Group 3 axioms entail that* $\varphi(Block\ 1)$ *is equivalent to the conjunction of (8.10), (8.11), (8.12) and the formula*

$$Poss(a,s) \equiv \neg \left[s = \bot \vee \bigvee_{\langle A,\sigma\rangle \in \Sigma} (a = A \wedge T(\sigma, s)) \right]. \tag{8.27}$$

Lemma 8.4 *Let* D *be a finite domain. For any model* Ψ *of* D*, there exists a model* M *of* $\gamma(D)$ *similar to* Ψ*.*

Proof. In defining the model M, our first task is to describe the universe of values $\mid M \mid_v$, the universe of actions $\mid M \mid_a$, the universe of aspects $\mid M \mid_{asp}$ and the universe of situations $\mid M \mid_s$. They are defined as follows:

$$\begin{array}{lcl} \mid M \mid_v & = & \mathbf{V}, \\ \mid M \mid_a & = & \mathbf{A}, \\ \mid M \mid_{asp} & = & \{\widetilde{F} \mid F \in \mathbf{Fr}\}, \\ \mid M \mid_s & = & \{\overline{A} \mid \Psi(\overline{A}) \text{ is defined }\} \cup \{\bot\}. \end{array}$$

Let us extend Ψ to $\bot$ as follows: $\Psi(\bot) = \Psi(\epsilon)$. With each element ξ of $\mid M \mid_s$, we associate the state $\Psi(\xi)$ of the domain D. These states will be useful in defining the interpretation of certain functions (e.g. the fluents) and predicates (e.g. $Poss$).

We will interpret the action and value constants as themselves. The interpretation of the fluents S_0 and $\bot$, and $Result$ are as follows:

$$\begin{array}{lcll} F^M(\xi) & = & \Psi(\xi)(F) & \quad (F \in \mathbf{F}), \\ S_0^M & = & \epsilon, & \\ \bot^M & = & \bot. & \end{array}$$

The definition of $Result^M$ is by cases

$$Result^M(A,\xi) = \begin{cases} \overline{A}A, & \text{if } \xi = \overline{A} \text{ and } \Psi(\overline{A}A) \text{ is defined,} \\ \bot, & \text{otherwise.} \end{cases}$$

Now, we can define $Poss^M$ and F_R^M.

$$\begin{array}{lcl} \langle A,\xi\rangle \in Poss^M & \Leftrightarrow & Result^M(A,\xi) \neq \bot, \\ F_R^M(A,\xi) & = & F^M(Result^M(A,\xi)). \end{array}$$

Using the definition of $Result^M$, the definition of $Poss^M$ can be equivalently expressed as

$$\langle A,\xi\rangle \in Poss^M \quad \Leftrightarrow \quad \xi \neq \bot \wedge Res(A,\Psi(\xi)) \neq \emptyset.$$

Using Lemma 8.1, the above may be also expressed as

$$\langle A, \xi \rangle \in \mathit{Poss}^M \quad \Leftrightarrow \quad \xi \neq \bot \wedge Res_0(A, \Psi(\xi)) \neq \emptyset. \tag{8.28}$$

This completes the definition of M. We now need to check that M is a model of the axioms in Groups 1–5 and $\varphi(\mathit{Block\ 2})$.

First of all, we will prove by induction that

$$\Psi(\overline{A}) \text{ is defined} \Rightarrow [\overline{A}]^M = \overline{A}. \tag{8.29}$$

The base case

$$\Psi(\epsilon) \text{ is defined} \Rightarrow S_0^M = \epsilon$$

follows trivially from the definition of S_0^M. Let the inductive hypothesis be (8.29). To show the induction step, assume $\Psi(\overline{A}; A)$ is defined. This implies that $\Psi(\overline{A})$ is defined and hence, using the inductive hypothesis that $[\overline{A}]^M = \overline{A}$. In addition, $\overline{A}$ belongs to $\mid M \mid_s \setminus \{\bot\}$. The definition of $Result^M$ gives us that $Result^M(A, \overline{A}) = \overline{A}; A$. That is, $Result^M(A, [\overline{A}]^M) = \overline{A}; A$ or equivalently, $[\overline{A}; A]^M = \overline{A}; A$, as desired.

Thus, we see that any situation ξ in $\mid M \mid_s$ is represented by a ground term. Hence, we will write expressions such as

$$M \models F(\xi) = V,$$

with the understanding that in such expressions ξ will stand for corresponding ground term.

We now turn to checking that M is a model of the axioms in Groups 1–5. That M is a model of the axioms in Groups 1, 2 and 4 is easy to see. Let us now check that M is a model of axioms in Group 3. That is, we need to check that for each constraint of the form **always** C and each situation ξ, $M \models T(C, \xi)$. It is easy to show, by induction, that for any domain formula C and any situation ξ,

$$M \models T(C, \xi) \;\Leftrightarrow\; \Psi(\xi)(C) = \mathsf{T}. \tag{8.30}$$

Since $\Psi(\xi)$ is a state of the domain, $\Psi(\xi)(C) = \mathsf{T}$ and the result follows by (8.30).

Let us check that M is a model of the axioms in Group 5, namely, that it is a model of the translations of the value propositions. Let us first show that

$$\Psi(\overline{A}) \text{ is defined} \;\Leftrightarrow\; M \models [\overline{A}] \neq \bot. \tag{8.31}$$

This proof is also by induction. The base case follows easily since both sides of the equivalence (8.31) are true when $\overline{A} = \epsilon$. Let (8.31) be the inductive hypothesis. We will now prove that

$$\Psi(\overline{A}; A) \text{ is defined} \;\Leftrightarrow\; M \models [\overline{A}; A] \neq \bot.$$

holds by proving each implication separately.

$$\begin{array}{ll}
& \Psi(\overline{A}; A) \text{ is defined} \\
\Rightarrow & \{ \text{ Eqn (8.29) } \} \\
& [\overline{A}; A]^M = \overline{A}; A \\
\Rightarrow & \{ \ \overline{A}; A \neq \bot, \text{ since } \Psi(\overline{A}; A) \text{ is defined } \} \\
& [\overline{A}; A]^M \neq \bot \\
\Leftrightarrow & \{ \text{ Definition } \} \\
& M \models [\overline{A}; A] \neq \bot.
\end{array}$$

$$\begin{array}{ll}
& M \models [\overline{A}; A] \neq \bot \\
\Rightarrow & \{ \ M \text{ is a model of } Result(a, \bot) = \bot \ \} \\
& M \models [\overline{A}] \neq \bot \text{ and } M \models [\overline{A}; A] \neq \bot \\
\Leftrightarrow & \{ \text{ Inductive Hypothesis } \} \\
& \Psi(\overline{A}) \text{ is defined and } M \models [\overline{A}; A] \neq \bot \\
\Rightarrow & \{ \text{ Eqn (8.29) } \} \\
& [\overline{A}]^M = \overline{A} \text{ and } M \models [\overline{A}; A] \neq \bot \\
\Rightarrow & \{ \text{ Defn. of } Result^M \ \} \\
& \Psi(\overline{A}; A) \text{ is defined}
\end{array}$$

We will now show that for every value proposition P, P is true in Ψ iff M satisfies γP. (That M is a model of the translations of the value propositions of the domain D is a direct consequence of this result). Again, we will prove both implications separately.

$$\begin{array}{ll}
& C \textbf{ after } \overline{A} \text{ is true in } \Psi \\
\Leftrightarrow & \{ \text{ Definition } \} \\
& \Psi(\overline{A}) \text{ is defined and } \Psi(\overline{A})(C) = \mathsf{T} \\
\Rightarrow & \{ \text{ Eqn (8.29), Eqn (8.31) } \} \\
& M \models [\overline{A}] \neq \bot \text{ and } \overline{A} = [\overline{A}]^M \text{ and } \Psi(\overline{A})(C) = \mathsf{T} \\
\Rightarrow & \{ \text{ Equation (8.30) } \} \\
& M \models T(C, [\overline{A}]) \wedge [\overline{A}] \neq \bot.
\end{array}$$

$$\begin{array}{ll}
& M \models T(C, [\overline{A}]) \wedge [\overline{A}] \neq \bot \\
\Leftrightarrow & \{ \text{ Eqn (8.31) } \} \\
& M \models T(C, [\overline{A}]) \text{ and } \Psi(\overline{A}) \text{ is defined} \\
\Rightarrow & \{ \text{ Eqn (8.29) } \} \\
& M \models T(C, [\overline{A}]) \text{ and } \Psi(\overline{A}) \text{ is defined and } \overline{A} = [\overline{A}]^M \\
\Rightarrow & \{ \text{ Eqn (8.30), the state associated with } \overline{A} \text{ is } \Psi(\overline{A}) \ \} \\
& \Psi(\overline{A})(C) = \mathsf{T} \text{ and } \Psi(\overline{A}) \text{ is defined} \\
\Leftrightarrow & \{ \text{ Definition } \} \\
& C \textbf{ after } \overline{A} \text{ is true in } \Psi
\end{array}$$

It remains to verify that M is a model of $\varphi(\textit{Block 2})$. To do that, we need to check that M combined with an appropriate interpretation of Ab satisfies $F_{Block\ 2}(Ab)$. That is, we have to show that M (combined with an appropriate interpretation of Ab) satisfies **CIRC**$(Z; Ab; \mathbf{F}_R)$, where Z is the conjunction of *LI*, the translations of the indeterminate effect propositions and $\varphi(\textit{Block 1})$.

For any action A and state σ, we define $C_A(\sigma)$ to be the set of fluents F such that for some indeterminate effect proposition A **possibly changes** F **if** P in D , $\sigma(P) = \textit{True}$. The interpretation of Ab can now be given as follows. Let ξ be an element in the universe of situations and let $Result^M(A, \xi) = \xi'$. Let $\sigma = \Psi(\xi)$ and $\sigma' = \Psi(\xi')$. Then

$$\begin{aligned} \langle \widetilde{F}, V, A, \xi \rangle \in \mathit{Ab}^M \quad \Leftrightarrow \quad & [F \in C_A(\sigma) \vee (\langle A, \xi \rangle \in \mathit{Poss}^M \wedge \sigma(F) \neq \sigma'(F))] \wedge \\ & V = \sigma'(F). \end{aligned} \tag{8.32}$$

Let us first check that M is a model of $\varphi(\textit{Block 1})$. By the corollary to Lemma 8.3, we need to check that M is a model of (8.10), (8.11), (8.12) and (8.27).

That M is model of (8.10) follows easily from the definition of M. That it is a model of (8.27) follows from (8.28). Let us now check that M is a model of (8.11). That is, we need to check for each determinate effect proposition A **causes** C **if** P of D that M is a model of its translation. Let ξ be any element in the universe of situations and let $Result^M(A, \xi)$ be ξ'. Also, denote $\Psi(\xi)$ and $\Psi(\xi')$ by σ and σ' respectively. Assume that $M \models Poss(A, \xi)$. Using the definition of Poss^M, this is equivalent to assuming $M \models \xi' \neq \perp$. We need to show that $M \models T(P, \xi) \supset T'(C, A, \xi)$.

Since $\xi' \neq \perp$, using the definition of $Result^M$, we get that

$$\sigma' \in Res(A, \sigma). \tag{8.33}$$

$$\begin{aligned} & M \models T(P, \xi) \supset T_R(C, A, \xi) \\ \Leftrightarrow\ & \{\text{Definition of } F_R^M, \text{ for } F \in \mathbf{F}\} \\ & M \models T(P, \xi) \supset T(C, \xi') \\ \Leftrightarrow\ & \{\text{Definition }\} \\ & M \models T(P, \xi) \Rightarrow M \models T(C, \xi') \\ \Leftrightarrow\ & \{ \text{ Eqn.(8.30)}\} \\ & \sigma(P) = \mathsf{T} \Rightarrow \sigma'(C) = \mathsf{T} \\ \Leftrightarrow\ & \{ \text{ Eqn (8.33) } \} \\ & \textit{True} \end{aligned}$$

Let us now check that M is a model of (8.12). Since M is a model of the DCA for actions, it suffices to check that for each action A, each constraint **always** C in D and each situation ξ, $M \models T_R(C, A, \xi)$. Using the definition of F_R^M (for each $F \in \mathbf{F}$, this reduces to showing that $M \models T(C, Result(A, \xi))$. Let σ be the state associated with $Result^M(A, \xi)$. Since σ is a state of the domain, $\sigma(C) = \mathsf{T}$ and the result follows by (8.30).

Let us now check that M is a model of LI and the translations of the indeterminate effect propositions. As before, let ξ be any element in the universe of situations and let $Result^M(A, \xi)$ be ξ'. Also, denote $\Psi(\xi)$ and $\Psi(\xi')$ by σ and σ' respectively. Fix an arbitrary frame fluent F, action A and value V.

$$
\begin{aligned}
& M \models \mathit{Poss}(A,\xi) \land V = F_R(A,\xi) \land \neg Ab(\widetilde{F}, V, A, \xi) \supset (V = F(\xi)) \\
\Leftrightarrow\ & \{\text{Definition}\} \\
& M \models \mathit{Poss}(A,\xi) \land V = F_R(A,\xi) \land \neg Ab(\widetilde{F}, V, A, \xi) \Rightarrow M \models (V = F(\xi)) \\
\Leftrightarrow\ & \{\text{ Defn. of } Ab^M, F^M, F_R^M \ \} \\
& [\langle A,\xi\rangle \in \mathit{Poss}^M \land V = \sigma'(F) \land F \notin \boldsymbol{C}_A(\sigma) \land \sigma(F) = \sigma'(F)] \Rightarrow V = \sigma(F) \\
\Leftrightarrow\ & \{\text{ Propositional logic }\} \\
& \mathit{True}
\end{aligned}
$$

Let us now check that M is a model of the translation of any indeterminate effect proposition A **possibly changes** F **if** P of the domain. Let σ and σ' be defined as before and let $\sigma'(F) = V$.

$$
\begin{aligned}
& M \models T(P,\xi) \supset Ab(\widetilde{F}, F_R(A,\xi), A, \xi) \\
\Leftrightarrow\ & \{\text{Definition}\} \\
& M \models T(P,\xi) \Rightarrow M \models Ab(\widetilde{F}, F_R(A,\xi), A, \xi) \\
\Leftrightarrow\ & \{\text{ Eqn.(8.30), Definition of } F_R^M \ \} \\
& \sigma(P) = \mathsf{T} \Rightarrow M \models Ab(\widetilde{F}, V, A, \xi) \\
\Leftrightarrow\ & \{\text{ Definition of } Ab^M \ \} \\
& \sigma(P) = \mathsf{T} \Rightarrow [F \in \boldsymbol{C}_A(\sigma) \lor (\langle A,\xi\rangle \in \mathit{Poss}^M \land \sigma(F) \neq \sigma'(F))] \land \\
& \qquad\qquad\qquad\qquad V = \sigma'(F) \\
\Leftrightarrow\ & \{\ \sigma(P) = \mathsf{T} \Rightarrow F \in \boldsymbol{C}_A(\sigma) \text{ by definition of } \boldsymbol{C}_A(\sigma)\ \} \\
& \mathit{True}
\end{aligned}
$$

To conclude the proof that M satisfies **CIRC**$(Z; Ab; \mathbf{F}_R)$ (recall that Z is the conjunction of *LI*, the translations of the indeterminate effect propositions and φ(*Block 2*)), we have to check that M is a minimal model of Z with respect to $\leq^{Ab;\mathbf{F}_R}$. If it is not a minimal model, then there exists a model M' of Z such that $M' \leq^{Ab;\mathbf{F}_R} M$ and $Ab^{M'} \subset Ab^M$. That is, there exist $\widetilde{F}, V', A, \xi$ such that

$$\langle \widetilde{F}, V', A, \xi\rangle \in Ab^M, \tag{8.34}$$

but

$$\langle \widetilde{F}, V', A, \xi\rangle \notin Ab^{M'}. \tag{8.35}$$

Let $Result^M(A,\xi) = \xi'$, and let the states associated with ξ and ξ' be σ and σ' respectively. From (8.34) and (8.32),

$$[F \in \boldsymbol{C}_A(\sigma) \vee (\langle A,\xi\rangle \in \mathit{Poss}^M \wedge \sigma(F) \neq \sigma'(F))] \wedge V' = \sigma'(F) \tag{8.36}$$

Define a function σ'' from fluents to values as follows: for any fluent F in $\mathbf{F}$,

$$\sigma''(F) = F_R^{M'}(A,\xi). \tag{8.37}$$

Since M' is a model of (8.12), we see that σ'' is a state of the domain D. Since M' is a model of (8.11), we see that $\sigma'' \in Res_0(A,\sigma)$.

We will first show that for any fluent G in $\boldsymbol{C}_A(\sigma)$,

$$\sigma'(G) = \sigma''(G). \tag{8.38}$$

Assume $G \in \boldsymbol{C}_A(\sigma)$. Then there is an indeterminate effect proposition

A **possibly changes** G **if** P,

such that $\sigma(P) = \mathsf{T}$. Since M' is a model of its translation, we get that

$$M' \models T(P,\xi) \Rightarrow M' \models Ab(\widetilde{G}, G_R(A,\xi), A, \xi).$$

Since M and M' interpret $\mathbf{F}$ identically, we conclude (using (8.30)) that

$$\sigma(P) = \mathsf{T} \Rightarrow \langle \widetilde{G}, \sigma''(G), A, \xi\rangle \in Ab^{M'}$$

Since the antecedent of the above formula is true, so is the consequent. Thus,

$$\langle \widetilde{G}, \sigma''(G), A, \xi\rangle \in Ab^{M'}, \tag{8.39}$$

for any $G \in \boldsymbol{C}_A(\sigma)$. Since $Ab^{M'} \subset Ab^M$, (8.39) implies that

$$\langle \widetilde{G}, \sigma''(G), A, \xi\rangle \in Ab^{M}.$$

Using (8.32), we get from the last formula that $\sigma'(G) = \sigma''(G)$.

We will now show that $F \notin \boldsymbol{C}_A(\sigma)$. Assume $F \in \boldsymbol{C}_A(\sigma)$. Then, by (8.38), we get that $\sigma'(F) = \sigma''(F)$. However, $\sigma'(F) = V'$ from (8.36). Hence $\sigma''(F) = V'$ and using (8.39), we get that

$$\langle \widetilde{F}, V', A, \xi\rangle \in Ab^{M'},$$

contradicting (8.35).

Hence $F \notin \boldsymbol{C}_A(\sigma)$. Using this fact and (8.36), we get that

$$F \notin \boldsymbol{C}_A(\sigma) \wedge \langle A,\xi\rangle \in \mathit{Poss}^M \wedge \sigma(F) \neq \sigma'(F) \wedge V' = \sigma'(F). \tag{8.40}$$

By the above, $M \models \mathit{Poss}(A, \xi)$ and hence $M \models Result(A, \xi) \neq \perp$. Using the definition of $Result^M$, we get that

$$\sigma' \in Res(A, \sigma). \tag{8.41}$$

We can now show that for any fluent G such that $G \in \mathbf{Fr}$ and $\sigma(G) \neq \sigma''(G)$,

$$\sigma'(G) = \sigma''(G). \tag{8.42}$$

Assume that $G \in \mathbf{Fr}$ and $\sigma(G) \neq \sigma''(G)$. By (8.40), $M \models \mathit{Poss}(A, \xi)$ and hence $M' \models \mathit{Poss}(A, \xi)$. Since M' is a model of *LI*, we get that

$$\langle \widetilde{G}, \sigma''(G), A, \xi \rangle \in Ab^{M'},$$

and $\sigma'(G) = \sigma''(G)$ follows, as before.

We can now show that

$$\sigma''(F) \neq \sigma'(F). \tag{8.43}$$

Assume $\sigma''(F) = \sigma'(F)$. Then $\sigma''(F) = V'$. That is, $F_R^{M'}(A, \xi) = V'$. Using (8.35), this implies that $M' \models \neg Ab(F, F_R(A, \xi), A, \xi)$. Since M' is a model of *LI*, this in turn yields

$$\sigma(F) = \sigma''(F) = V',$$

which contradicts (8.40).

We are now in a position to obtain a contradiction. We consider two cases, depending on whether or not $\sigma(F)$ equals $\sigma''(F)$.

Case 1: $\sigma(F) \neq \sigma''(F)$.

Then, using (8.42), we get that $\sigma'(F) = \sigma''(F)$, which contradicts (8.43).

Case 2: $\sigma(F) = \sigma''(F)$.

In this case, we will show that

$$\mathit{New}_A(\sigma, \sigma'') \subset \mathit{New}_A(\sigma, \sigma'),$$

thereby contradicting (8.41). Using (8.40), we see that F **is** $V' \in \mathit{New}_A(\sigma, \sigma')$, and using the case assumption that F **is** $V' \notin \mathit{New}_A(\sigma, \sigma'')$. Hence it suffices to show that $\mathit{New}_A(\sigma, \sigma'') \subseteq \mathit{New}_A(\sigma, \sigma')$.

$$
\begin{aligned}
& G \textbf{ is } V \in \mathit{New}_A(\sigma, \sigma'') \\
\Leftrightarrow\ & \{\text{Definition of } \mathit{New}_A(\sigma, \sigma'')\} \\
& [G \in \boldsymbol{C}_A(\sigma) \vee (G \in \mathbf{Fr} \wedge \sigma(G) \neq \sigma''(G))] \wedge \sigma''(G) = V \\
\Rightarrow\ & \{\text{Facts (8.38) and (8.42)}\} \\
& [G \in \boldsymbol{C}_A(\sigma) \vee (G \in \mathbf{Fr} \wedge \sigma(G) \neq \sigma''(G))] \wedge \sigma''(G) = V \wedge \sigma'(G) = \sigma''(G) \\
\Rightarrow\ & \{\text{Equality}\} \\
& [G \in \boldsymbol{C}_A(\sigma) \vee (G \in \mathbf{Fr} \wedge \sigma(G) \neq \sigma'(G))] \wedge \sigma'(G) = V \\
\Leftrightarrow\ & \{\text{Definition of } \mathit{New}_A(\sigma, \sigma')\} \\
& G \textbf{ is } V \in \mathit{New}_A(\sigma, \sigma')
\end{aligned}
$$

This concludes the proof that M is a model of $\varphi(\textit{Block 2})$. Hence M is a model of $\gamma(D)$. That M is similar to Ψ follows because we have already shown that for every value proposition P, P is true in Ψ iff M satisfies γP. $\square$

Lemma 8.5 *Let D be a finite domain. For any model M of $\gamma(D)$, there exists a model Ψ of D such that M is similar to Ψ.*

Proof. The UNA and DCA for values and actions allows us to assume (without loss of generality) that

$$\begin{aligned} \mid M \mid_v &= \mathbf{V}, \\ \mid M \mid_a &= \mathbf{A}. \end{aligned}$$

For simplifying the presentation of the proofs to follow, we will extend the language of γD by adding each element of the universe of situations of M as a new object constant. These new object constants will be called *names*.

For any situation valued term t without variables, let t^* to be a function from fluents to values defined as follows:

$$t^*(F) = F^M(t^M).$$

This function can be extended to domain formulae in the standard way. It is then easy to check that for any domain formula C and any situation valued term t without variables,

$$t^*(C) = \mathsf{T} \Leftrightarrow M \models T(C, t). \tag{8.44}$$

Since M is a model of the Group 3 axioms, we get that $M \models T(C, t)$ for any constraint **always** C and that $t^*(F) \in \textit{Dom}_F$ for any fluent F. It then follows by (8.44) that t^* is a state of D.

Let us now turn to defining Ψ. Ψ is defined on the tuple $\overline{A}$ iff $M \models [\overline{A}] \neq \perp$, and if defined, $\Psi(\overline{A}) = [\overline{A}]^*$.

We now need to check the following:

1. The set $\{\overline{A} \mid \Psi(\overline{A}) \text{ is defined}\}$ is prefix-closed.

2. If $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is not defined, then $Res(A, \Psi(\overline{A})) = \emptyset$.

3. If $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is defined, then $\Psi(\overline{A}; A) \in Res(A, \Psi(\overline{A}))$.

4. Ψ is similar to M.

5. All value propositions of D are true in Ψ.

To check the first of the above, note that since M is a model of (8.10) and the Group 4 axioms, we get the M is a model of

$$Result(a, \perp) = \perp.$$

Equivalently, M satisfies

$$Result(a, s) \neq \bot \supset s \neq \bot.$$

This shows that the set $\{\overline{A} \mid \Psi(\overline{A}) \text{ is defined}\}$ is prefix-closed.

Assume that $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is not defined. We need to check that $Res(A, \Psi(\overline{A})) = \emptyset$. Since $\Psi(\overline{A})$ is defined,

$$M \models [\overline{A}] \neq \bot, \tag{8.45}$$

and

$$\Psi(\overline{A}) = [\overline{A}]^*. \tag{8.46}$$

Note that $M \models T([\overline{A}]^*, [\overline{A}])$ and this in conjunction with (8.46) yields that

$$M \models T(\Psi(\overline{A}), [\overline{A}]). \tag{8.47}$$

Since $\Psi(\overline{A}; A)$ is not defined, we get that

$$M \models [\overline{A}; A] = \bot. \tag{8.48}$$

Using the definition of *Poss* from Group 4, we then get that

$$M \models \neg \mathit{Poss}(A, [\overline{A}]). \tag{8.49}$$

By the corollary to Lemma 8.3, M satisfies (8.27). This fact, in combination with (8.49) yields that

$$M \models \left[[\overline{A}] = \bot \vee \bigvee_{\langle A', \sigma \rangle \in \Sigma} (A = A' \wedge T(\sigma, [\overline{A}])) \right]. \tag{8.50}$$

Using (8.45) and the UNA for actions, (8.50) simplifies to

$$M \models \bigvee_{\sigma : \langle A, \sigma \rangle \in \Sigma} T(\sigma, [\overline{A}]). \tag{8.51}$$

Note that if $\sigma \neq \sigma'$,

$$M \models T(\sigma, [\overline{A}]) \supset \neg T(\sigma', [\overline{A}]). \tag{8.52}$$

If $\langle A, \Psi(\overline{A}) \rangle \notin \Sigma$, (8.47) in conjunction with (8.52) implies that

$$\bigwedge_{\sigma : \langle A, \sigma \rangle \in \Sigma} \neg T(\sigma, [\overline{A}]),$$

contradicting (8.51). Hence, $\langle A, \Psi(\overline{A}) \rangle \in \Sigma$ or equivalently $Res_0(A, \Psi(\overline{A})) = \emptyset$. This implies that $Res(A, \Psi(\overline{A})) = \emptyset$.

Let us now check that if $\Psi(\overline{A})$ is defined and $\Psi(\overline{A}; A)$ is defined, then $\Psi(\overline{A}; A) \in Res(A, \Psi(\overline{A}))$. We will prove this by contradiction. Since M is a model of $\gamma(D)$, it is a model of $\varphi(\textit{Block 2})$. Equivalently, M combined with an appropriate interpretation of Ab (call it Ab^M) satisfies $F_{Block\ 2}(Ab)$. Recall that $F_{Block\ 2}(Ab)$ stands for $\mathbf{CIRC}(Z; Ab; \mathbf{F}_R)$, where Z is the conjunction of *LI*, the translations of the indeterminate effect propositions and $\varphi(\textit{Block 1})$. We will obtain the desired contradiction by constructing a model M' of Z, identical to M except in the interpretations of $\mathbf{F}_R$ and Ab and where the extent of Ab is smaller than Ab^M.

Assume that there exist $\overline{A_1}$ and A_2 such that $\Psi(\overline{A_1})$ is defined and $\Psi(\overline{A_1}; A_2)$ is defined and $\Psi(\overline{A_1}; A_2) \notin Res(A_2, \Psi(\overline{A_1}))$. Since $\Psi(\overline{A_1})$ is defined, as before, we see that

$$M \models [\overline{A_1}] \neq \perp, \tag{8.53}$$

$$\Psi(\overline{A_1}) = [\overline{A_1}]^*, \tag{8.54}$$

and

$$M \models T(\Psi(\overline{A_1}), [\overline{A_1}]). \tag{8.55}$$

Since $\Psi(\overline{A_1}; A_2)$ is defined,

$$M \models [\overline{A_1}; A_2] \neq \perp\ . \tag{8.56}$$

Using the definition of *Poss* from Group 4, (8.56) implies that

$$M \models \mathit{Poss}(A_2, [\overline{A_1}]). \tag{8.57}$$

In combination with (8.27) and UNA for actions, (8.56) yields that

$$M \models \left[[\overline{A_1}] \neq \perp \wedge \bigwedge_{\sigma:\langle A_2, \sigma\rangle \in \Sigma} \neg T(\sigma, [\overline{A_1}]) \right]. \tag{8.58}$$

Equations (8.55) and (8.58) imply that $\langle A_2, \Psi(\overline{A_1})\rangle \notin \Sigma$ or equivalently that $Res_0(A_2, \Psi(\overline{A_1})) \neq \emptyset$. Lemma 8.1 then yields that $Res(A_2, \Psi(\overline{A_1})) \neq \emptyset$. Choose a σ_2' in $Res(A_2, \Psi(\overline{A_1}))$.

We are now ready to define M'. Let us first introduce some notation to simplify the definition. Let $\xi_1 = [\overline{A_1}]^M$, $\sigma_1 = [\overline{A_1}]^*$ and $\sigma_2 = [\overline{A_1}; A_2]^*$.

Let us first define $F_R^{M'}$ for any $F_R \in \mathbf{F}_R$ as follows: $F_R^{M'}(A_2, \xi_1) = \sigma_2'(F)$. For all other tuples, $F_R^{M'}$ is defined to be equal to F_R^M.

Let us now define $Ab^{M'}$. Let us first consider quadruples whose last two components are A_2 and ξ_1. For such quadruples, we define, for any frame fluent F and value V,

$$\begin{aligned} \langle \widetilde{F}, V, A_2, \xi_1\rangle \in Ab^{M'} \quad \Leftrightarrow \quad & [\sigma_1(F) \neq \sigma_2'(F) \vee F \in C_{A_2}(\sigma_1)] \wedge \\ & V = \sigma_2'(F). \end{aligned} \tag{8.59}$$

(Recall that for any action A and state σ, $C_A(\sigma)$ is the set of fluents F such that for some indeterminate effect proposition A **possibly changes** F **if** P in D , $\sigma(P) = \mathit{True}$.) Any other quadruple is in $Ab^{M'}$ iff it is in Ab^M. This concludes the definition of M'.

Let us first check that M' is a model of Z (recall that Z is the conjunction of *LI*, the translations of the indeterminate effect propositions and $\varphi(\mathit{Block\ 1})$). Using the corollary to Lemma 8.3, it suffices to check that M is a model of LI, translations of the indeterminate effect propositions, and formulae (8.10), (8.11), (8.12) and (8.27).

We first check that M is a model of LI. By the definition of $F_R^{M'}$ and $Ab^{M'}$, it suffices to check for any frame fluent F and value V that

$$\begin{aligned} M' &\models \mathit{Poss}(A_2, \xi_1) \wedge V = F_R(A_2, \xi_1) \wedge \neg Ab(\widetilde{F}, V, A_2, \xi_1) \supset \\ &\quad (V = F(\xi_1). \end{aligned}$$

Now,

$$\begin{aligned} & M' \models \mathit{Poss}(A_2, \xi_1) \wedge V = F_R(A_2, \xi_1) \wedge \neg Ab(\widetilde{F}, V, A_2, \xi_1) \supset \\ & \quad (V = F(\xi_1). \\ \Leftrightarrow\ & \{\text{Definition}\} \\ & M' \models \mathit{Poss}(A_2, \xi_1) \wedge V = F_R(A_2, \xi_1) \wedge \neg Ab(\widetilde{F}, V, A_2, \xi_1) \Rightarrow \\ & \quad M' \models V = F_R(A_2, \xi_1) \\ \Leftrightarrow\ & \{\text{ Defn. of } Ab^{M'}, F_R^{M'}, \sigma_1 \} \\ & [\langle A_2, \xi_1\rangle \in \mathit{Poss}^M \wedge V = \sigma_2'(F) \wedge \sigma_1(F) = \sigma_2'(F) \wedge F \notin C_{A_2}(\sigma_1)] \Rightarrow \\ & \quad V = \sigma_1(F) \\ \Leftrightarrow\ & \{\text{ Propositional logic }\} \\ & \mathit{True} \end{aligned}$$

We will now check that M' is a model of the translations of the indeterminate effect propositions. As before, it suffices to check that

$$M \models T(P, \xi_1) \supset Ab(\widetilde{F}, F_R(A_2, \xi_1), A_2, \xi_1)$$

for any indeterminate effect proposition A_2 **possibly changes** F **if** P of the domain. Let $\sigma_2'(F) = V$.

$$\begin{aligned} & M \models T(P, \xi_1) \supset Ab(\widetilde{F}, F_R(A_2, \xi_1), A_2, \xi_1) \\ \Leftrightarrow\ & \{\text{Definition}\} \\ & M \models T(P, \xi_1) \Rightarrow M \models Ab(\widetilde{F}, F_R(A_2, \xi_1), A_2, \xi_1) \\ \Leftrightarrow\ & \{\text{ Eqn.(8.30), Definition of } F_R^{M'} \} \\ & \sigma_1(P) = \mathsf{T} \Rightarrow M \models Ab(\widetilde{F}, V, A_2, \xi_1) \\ \Leftrightarrow\ & \{\text{ Definition of } Ab^{M'} \} \\ & \sigma_1(P) = \mathsf{T} \Rightarrow [\sigma_1(F) \neq \sigma_2'(F) \vee F \in C_{A_2}(\sigma_1)] \wedge V = \sigma_2'(F) \\ \Leftrightarrow\ & \{ \sigma_1(P) = \mathsf{T} \Rightarrow F \in C_{A_2}(\sigma_1) \text{ by definition of } C_{A_2}(\sigma_1) \} \\ & \mathit{True} \end{aligned}$$

Since M and M' interpret *Poss* and **F** identically, we see that M' is a model of (8.10) and (8.27). Since σ_2' is a state of D, we get that M' is a model of (8.12). Hence our task reduces to checking that M' is a model of (8.11). For this, it suffices to show that

$$M' \models \mathit{Poss}(A_2, [\overline{A_1}]) \wedge T(P, [\overline{A_1}]) \supset T'(C, A_2, [\overline{A_1}])$$

for each determinate effect proposition A_2 **causes** C **if** P of the domain D.

$$\begin{array}{ll}
 & M' \models \mathit{Poss}(A_2, [\overline{A_1}]) \wedge T(P, [\overline{A_1}]) \supset T'(C, A_2, [\overline{A_1}]) \\
\Leftrightarrow & \{\text{Definition}\} \\
 & M' \models \mathit{Poss}(A_2, [\overline{A_1}]) \Rightarrow M' \models T(P, [\overline{A_1}]) \supset T'(C, A_2, [\overline{A_1}]) \\
\Leftrightarrow & \{\ \mathit{Poss}^M = \mathit{Poss}^{M'}\ \} \\
 & M \models \mathit{Poss}(A_2, [\overline{A_1}]) \Rightarrow M' \models T(P, [\overline{A_1}]) \supset T'(C, A_2, [\overline{A_1}]) \\
\Leftrightarrow & \{\text{Eqn (8.57)}\} \\
 & M' \models T(P, \xi_1) \supset T'(C, A_2, \xi_1) \\
\Leftrightarrow & \{\text{Definition}\} \\
 & M' \models T(P, \xi_1) \Rightarrow M' \models T'(C, A_2, \xi_1) \\
\Leftrightarrow & \{\ \text{Definition of } \sigma_1 \text{ and } \sigma_2'\} \\
 & \sigma_1(P) = \mathsf{T} \Rightarrow \sigma_2'(C) = \mathsf{T} \\
\Leftrightarrow & \{\ \sigma_2' \in Res(A_2, \sigma_1)\ \} \\
 & \mathit{True}
\end{array}$$

This concludes the proof that M' is a model of Z. Let us now turn to showing that $Ab^{M'} \subset Ab^M$. By assumption, $\sigma_2 \notin Res(A_2, \sigma_1)$. (Recall that $\sigma_1 = \Psi(\overline{A_1})$ and $\sigma_2 = \Psi(\overline{A_1}; A_2)$). Since σ_2' is, by definition, an element of $Res(A_2, \sigma_1)$, we get

$$\mathit{New}_{A_2}(\sigma_1, \sigma_2') \subset \mathit{New}_{A_2}(\sigma_1, \sigma_2). \tag{8.60}$$

Let us first show that $Ab^{M'} \subseteq Ab^M$. Assume $\langle \widetilde{F}, V, A_2, \xi_1 \rangle \in Ab^{M'}$. It follows by (8.59) that

$$[\sigma_1(F) \neq \sigma_2'(F) \vee F \in C_{A_2}(\sigma_1)] \wedge V = \sigma_2'(F) \tag{8.61}$$

If $F \in C_{A_2}(\sigma_1)$, then F **is** $\sigma_2'(F) \in \mathit{New}_{A_2}(\sigma_1, \sigma_2')$ and using (8.60), we conclude that F **is** $\sigma_2'(F) \in \mathit{New}_{A_2}(\sigma_1, \sigma_2)$. However, from $F \in C_{A_2}(\sigma_1)$, we also conclude that F **is** $\sigma_2(F) \in \mathit{New}_{A_2}(\sigma_1, \sigma_2)$. Hence,

$$\sigma_2(F) = \sigma_2'(F) = V. \tag{8.62}$$

Also, since $F \in C_{A_2}(\sigma_1)$, there exists an indeterminate effect proposition

A_2 **possibly changes** F **if** P, where $\sigma_1(P) = \mathsf{T}$. Since M is a model of the translation of this proposition, we get that $\langle \widetilde{F}, \sigma_2(F), A_2, \xi_1\rangle \in Ab^M$, and using (8.62), we see that $\langle \widetilde{F}, V, A_2, \xi_1\rangle \in Ab^M$

Hence we can assume that $F \notin C_{A_2}(\sigma_1)$. Hence, by (8.61), we see that

$$\sigma_1(F) \neq \sigma_2'(F) \wedge F \notin C_{A_2}(\sigma_1) \wedge V = \sigma_2'(F) \tag{8.63}$$

$$\begin{aligned}
& \langle \widetilde{F}, V, A_2, \xi_1\rangle \in Ab^{M'} \\
\Rightarrow & \{ \text{ Eqn (8.63), Definition of } \mathit{New}_{A_2}(\sigma_1, \sigma_2') \ \} \\
& F \textbf{ is } V \in \mathit{New}_{A_2}(\sigma_1, \sigma_2') \\
\Rightarrow & \{ \text{ Eqn (8.60) } \} \\
& F \textbf{ is } V \in \mathit{New}_{A_2}(\sigma_1, \sigma_2) \\
\Rightarrow & \{ F \notin C_{A_2}(\sigma_1) \text{ , Definition of } \mathit{New}_{A_2}(\sigma_1, \sigma_2) \ \} \\
& \sigma_2(F) = V \wedge \sigma_1(F) \neq \sigma_2(F) \\
\Leftrightarrow & \{ \text{Definition of } \sigma_1 \text{ and } \sigma_2, \} \\
& M \models F_R(A_2, \xi_1) = V \wedge F(\xi_1) \neq F_R(A_2, \xi_1) \\
\Rightarrow & \{ \ M \text{ is a model of } LI \text{, Eqn (8.57) } \} \\
& \langle \widetilde{F}, V, A_2, \xi_1\rangle \in Ab^M
\end{aligned}$$

To show that $Ab^{M'}$ is a proper subset of Ab^M, we proceed as follows. By (8.60), there is a fluent name F such that

$$F \textbf{ is } V \in \mathit{New}_{A_2}(\sigma_1, \sigma_2), \text{ but } F \textbf{ is } V \notin \mathit{New}_{A_2}(\sigma_1, \sigma_2'). \tag{8.64}$$

We first prove the following:

$$\text{For any } V' \in \mathit{Dom}_F,\ F \textbf{ is } V' \notin \mathit{New}_{A_2}(\sigma_1, \sigma_2'). \tag{8.65}$$

For $V' = V$, this follows from (8.64). For any other values V', if F **is** $V' \in \mathit{New}_{A_2}(\sigma_1, \sigma_2')$, it follows by (8.60) that F **is** $V' \in \mathit{New}_{A_2}(\sigma_1, \sigma_2)$, contradicting the first part of (8.64).

From (8.65), we get that

$$\sigma_1(F) = \sigma_2'(F) \tag{8.66}$$

(otherwise F **is** $\sigma_2'(F) \in \mathit{New}_{A_2}(\sigma_1, \sigma_2')$, contradicting (8.65)).

Note that it follows from (8.64) that $\sigma_2(F) = V$. Also, $F \notin C_{A_2}(\sigma_1)$ (otherwise F **is** $\sigma_2'(F) \in \mathit{New}_{A_2}(\sigma_1, \sigma_2')$, contradicting (8.65)). We can then conclude from the definition of $\mathit{New}_{A_2}(\sigma_1, \sigma_2)$ that $\sigma_1(F) \neq \sigma_2(F)$. From this, (8.66) and $\sigma_2(F) = V$, we conclude that

$$\sigma_2'(F) \neq V \tag{8.67}$$

Using the the definition of $Ab^{M'}$ and (8.67), we can conclude that $\langle \widetilde{F}, V, A_2, \xi_1\rangle \notin Ab^{M'}$. However, we have already shown that $\sigma_1(F) \neq \sigma_2(F)$ and $\sigma_2(F) = V$. Using (8.57) and that M is a model of LI, we get that $\langle \widetilde{F}, V, A_2, \xi_1\rangle \in Ab^M$. Thus, we see that $Ab^{M'}$ is a proper subset of Ab^M.

Let us now check that Ψ is similar to M.

$$
\begin{array}{ll}
 & C \textbf{ after } \overline{A} \text{ is true in } \Psi \\
\Leftrightarrow & \{ \text{Definition of "true in } \Psi\text{"} \} \\
 & \Psi(\overline{A}) \text{ is defined and } \Psi(\overline{A})(C) = \mathsf{T} \\
\Leftrightarrow & \{ \text{Definition of } \Psi \} \\
 & [\overline{A}]^{M} \neq \bot \text{ and } [\overline{A}]^{*}(C) = \mathsf{T} \\
\Leftrightarrow & \{ \text{Eqn (8.44)} \} \\
 & M \models [\overline{A}] \neq \bot \wedge T(C, [\overline{A}])
\end{array}
$$

Since Ψ is similar to M and M is a model of the translations of the value propositions of the domain, Ψ is a model of the value propositions of the domain. $\square$

Chapter 9

Conclusions

9.1 Summary of Contributions

In this thesis, we presented a systematic study of reasoning about actions, with special focus on the frame and ramification problems. The main contributions of this thesis are as follows.

- We introduced three translations from $\mathcal{A}$, each based on a different formalization of actions, and proved the soundness and completeness of these translations.
- We introduced an action language $\mathcal{AR}^-$ in which domain constraints can be expressed, and studied its properties.
- We gave two counterexamples that demonstrate certain shortcomings of Baker's approach to the frame problem.
- We gave a sound and complete translation from $\mathcal{AR}^-$ using nested abnormality theories.
- We introduced an action language $\mathcal{AR}$ capable of expressing non-Boolean fluents and nondeterministic actions and studied its properties.
- We gave a sound and complete translation from $\mathcal{AR}$ using a novel formalization of actions and illustrated via examples the potential of using this formalization in automating reasoning about actions.

9.2 Future Work

9.2.1 Extensions to More Expressive Formalisms

The study presented in this thesis has been in the context of the formalism of the situation calculus. Though the situation calculus is conceptually simple and clear, it does not seem easy to study some aspects of reasoning about actions within this formalism. The following example, similar to the ferryboat connection problem from Sandewall [1992], illustrates that there are situations that cannot be conveniently represented in the standard situation calculus framework.

Example 1. If Mary arrives at the bus-stop at or before 7.00 am, she will catch the first bus. If she does not catch the first bus, she will take a cab. It is given that she arrives at the bus-stop between 6:59am and 7.01am.

Conclusion : Mary will either catch the first bus or take a cab.

To represent this example, we need to enhance the situation calculus by adding a time line to the situation calculus, as is done in [Pinto, 1994]. There, other enhancements for representing concurrent actions and continuous change are also proposed. The systematic study of reasoning about actions needs to be extended to handle these additional expressive capabilities.

9.2.2 Qualification and Ramification Constraints

As pointed out by [Ginsberg and Smith, 1988b] and [Lin and Reiter, 1994], formulae which are syntactically indistinguishable from domain constraints (that are responsible for the indirect effect of actions) can also be used to express indirect restrictions on the performability of an action. For instance, consider the fact that two blocks cannot be at the same place at the same time. This constraint can be interpreted in two ways. According to the first interpretation, moving a block to a position already occupied by another block causes the second block to move to another position. In this case, the constraint is said to be a *ramification constraint*, since it leads to an indirect effect of the action of moving the first block. In the second interpretation, the domain constraint can be taken to mean that the action of moving the first block to the position occupied by the second block cannot be performed. In this case, the domain constraint is said to be a *qualification constraint*, because it gives conditions under which an action can be performed. Another example of a qualification constraint is the one in the missionaries and cannibals puzzle [McCarthy, 1986]: the number of missionaries on each side of the river is at least the number of cannibals on the same side.

In general, we will have to represent both ramification and qualification constraints and reason with them. Since both of them often have a syntactically identical form, it is not clear how to distinguish between them. It seems that alternate syntax (and semantics) to account for this difference may have to be developed. Recently, several researchers [McCain and Turner, 1995; Lin, 1995; Zhang and Foo, 1995] have suggested that representing the notion of causality may be crucial in properly reasoning with such constraints. This needs to be investigated further.

9.2.3 Dealing with Unwanted Persistence of Fluents

Though we usually want the values of fluents to persist, there are situations where we do not want this to happen. The following example [Myers and Smith, 1988] illustrates this.

Example 2. If an object is on the table, it is baby-proof. Initially, Block A is on the table. Picking up an object causes it to be not on the table.

Question: Is Block A baby-proof after picking it up?

Intuitively, the answer to this question is 'Unknown'. But, if we represent baby-proof as a fluent and insist that it persists when the action pick up is performed, we will get the unintuitive answer "Yes" to the question. How to prevent such overly strong applications of the commonsense law of inertia is another topic that deserves further study. The notion of dependent fluents [Giunchiglia and Lifschitz, 1995] is a step in this direction.

9.3 Final Words

The basic approach in this thesis has been to propose a syntax and semantics for action languages and to view different approaches to reasoning about action as translations from such languages. How do we know that the semantics we propose for the action languages correspond to what is sanctioned by our commonsense intuitions? Note that this question asks us to establish a correspondence between a formal notion (the semantics) and an informal one (our intuitions). (The situation is not unlike asking about the relationship between the formal and informal notions of computability). We cannot hope for a *proof* of such a correspondence. However, we can provide evidence by encoding several commonsense domains within our formalism and showing that the conclusions sanctioned by the formal system correspond to our intuitions. Additional evidence can be provided by observing that the encoding is simple and perspicuous and that we can prove properties about the formal system that correspond to our intuitions.

Another question that might be raised concerns the need for introducing action languages. Can we not employ a standard (say first order) language, define a semantics for it that corresponds to our intuitions and carry out the study of actions within such a framework? This is the approach taken by Sandewall [1992], Lin, Shoham and Reiter [Lin and Shoham, 1991; Lin and Reiter, 1994]. The answer to this question is twofold. Firstly, the syntax of action languages corresponds more closely to how we describe action domains and hence it is easier to encode various action domains using action languages. Secondly, some of the underlying notions that arise in action domains (such as causality) may be difficult to express in a standard first order language.

This thesis takes the first few steps in the systematic study of theories of action using action languages. It is hoped that this study will serve as the foundation on which more comprehensive theories of action can be built.

Appendix A

A.1 Some Results on Circumscription

In this section, we collect together a number of results on circumscription that are used in the thesis. The terminology and presentation follows [Lifschitz, 1993a]. The reader is referred there for motivating examples and proofs of the theorems. In each of the propositions below, the number in parantheses identifies the section where the proposition can be found in [Lifschitz, 1993a].

Our first task is to define circumscription. For any predicate symbols P, Q of the same arity, let $P = Q$ stand for $\forall x(P(x) \equiv Q(x))$ and $P \leq Q$ stand for $\forall x(P(x) \supset Q(x))$. Let Z stand for a tuple $Z_1, Z_2, \ldots Z_m$ of object, function, and/or predicate constants. Let $A(P, Z)$ be a sentence containing Z and a predicate constant P.

The *circumscription of* P *in* A *with* Z *varied*, denoted by **CIRC**$(A; P; Z)$ is the sentence

$$A(P, Z) \wedge \neg \exists p\ z[A(p, z) \wedge p < P].$$

Here p is a predicate variable of the same arity as P, z stands for an m-tuple of variables which matches the m-tuple Z in arity and type, and $p < P$ stands for $(p \leq P) \wedge \neg(p = P)$.

The models of **CIRC**$(A; P; Z)$ are the models of A where the extent of P is minimal. To make this precise, let us first define a relation $\leq^{P;Z}$ on structures. For structures M_1 and M_2, $M_1 \leq^{P;Z} M_2$ means M_1 differs from M_2 only in the interpretation of P and Z, and the extent of P in M_1 is a subset of its extent in M_2. We write $M_1 <^{P;Z} M_2$ if $M_1 \leq^{P;Z} M_2$, but not $M_2 \leq^{P;Z} M_1$.

It is easy to see that the relation $\leq^{P;Z}$ is reflexive and transitive. Hence, given a class of structures, we can talk about the structures in the class that are minimal relative to the relation. A model M of A is *minimal* relative to $\leq^{P;Z}$ if there is no model M' of A such that $M' <^{P;Z} M$.

Proposition 1 (2.5) *A structure M is a model of* **CIRC**$(A; P; Z)$ *iff M is minimal with respect to the partial order $\leq^{P;Z}$.*

We now give some results that will be helpful for computing circumscriptions.

In the following proposition, let x be a tuple of variables and let $F(x)$ be a formula with all parameters explicitly shown.

Proposition 2 (3.1) *If $F(x)$ does not contain P, then the circumscription* **CIRC**$[\forall x(F(x) \supset P(x)); P]$ *is equivalent to* $\forall x(F(x) \equiv P(x))$.

The following proposition shows how to reduce circumscription with varied constants to the simpler case of circumscription where nothing is varied.

Proposition 3 (3.2) *The circumscription* **CIRC**$[A(P, Z); P; Z]$ *is equivalent to*

$$A(P, Z) \wedge \textbf{CIRC}[\exists z A(P, z); P].$$

An occurrence of an predicate symbol in a formula is *positive* if it is in the range of an even number of negations and *negative* otherwise (assuming that formula is the conjunctive normal form). A formula $A(P)$ is positive (relative to P) if all occurrences of P in it are positive, and negative if all occrrences of P are negative.

To express some results on circumscription concisely, we introduce the notion of *predicate expressions*. If $F(x_1, \ldots, x_n)$ is a formula, then the expression

$$\lambda x_1, \ldots, x_n \;\; F(x_1, \ldots, x_n) \tag{A.1}$$

is called a predicate expression. The free variables (if any) of the formula $F(x_1, \ldots, x_n)$, other than $x_1, \ldots, x_n$ are called parameters of (A.1). Thus $\lambda x P(x, y)$ is a predicate expression with the parameter y.

Predicate expressions are a convenient way to describe formulae obtained by substitution. If Q is (A.1) and $t_1, \ldots t_n$ are terms, then $Q(t_1, \ldots, t_n)$ stands for the formula $F(t_1, \ldots, t_n)$. Thus $\lambda x P(x, y)(a)$ stands for $P(a, y)$. If $A(P)$ is a formula containing a predicate constant P, and Q is a predicate expression of the same arity as P, then $A(Q)$ stands for the result of replacing each atomic part $P(t_1, \ldots t_n)$ in $A(P)$ by $Q(t_1, \ldots, t_n)$.

The following proposition is useful in simplifying formulae:

Proposition 4 (3.3) *For any formula $B(p)$ negative relative to p, and any predicate expression E that does not have p among its parameters, $\exists p[E \leq p \wedge B(p)]$ is equivalent to $B(E)$.*

The following proposition is used to verify conjectures about circumscription.

Proposition 5 (3.4) *Let E be a predicate expression without parameters and let Γ be a set of sentences. Both E and Γ do not contain P or Z. If Γ entails the two sentences*

$$A(P, Z) \supset E \leq P,$$

$$A(P, Z) \supset \exists z A(E, z),$$

then it entails the sentence

$$\mathbf{CIRC}(A(P, Z); P; Z) \equiv A(P, Z) \wedge P = E.$$

A.2 Nested Abnormality Theories

The following definitions are reproduced from [Lifschitz, 1995].

Consider a language L of classical logic which may have variables of several sorts and higher-order variables. We assume that L does *not* include *Ab* among its symbols. For any list $\tau_1, ..., \tau_k$ of sorts of variables available in L, by $L_{\tau_1 \ldots \tau_k}$ we denote the language obtained from L by adding *Ab* as a k-ary predicate constant taking arguments of the sorts $\tau_1, ..., \tau_k$. *Blocks* are defined recursively as follows: For any list of sorts $\tau_1, ..., \tau_k$ and any list of function and/or predicate constants $C_1, \ldots, C_m$ ($m \geq 0$) of L, if each of $A_1, \ldots, A_n$ ($n \geq 0$) is a formula of $L_{\tau_1 \ldots \tau_k}$ or a *block*, then

$$\{C_1, \ldots, C_m \, : \, A_1, \ldots, A_n\}$$

is a *block*. A *nested abnormality theory* is a set of blocks, called its *axioms*.

The semantics of nested abnormality theories is characterized by a map φ that translates blocks into formulae. It is convenient to make φ defined also on formulae of the languages $L_{\tau_1 \ldots \tau_k}$. If A is such a formula, then φA stands for the universal closure of A. For blocks we define, recursively:

$$\varphi\{C_1, \ldots, C_m \, : \, A_1, \ldots, A_n\} = \exists ab F(ab),$$

where

$$F(Ab) = \mathbf{CIRC}[\varphi A_1 \wedge \ldots \wedge \varphi A_n; Ab; C_1, \ldots, C_m].$$

Note that, for any block A, φA is a sentence not containing *Ab*.

A sentence A of L will be identified with the block $\{: \, A\}$. It is easy to see that $\varphi\{: \, A\}$ is equivalent to A.

For any nested abnormality theory T, φT stands for $\{\varphi A \mid A \in T\}$. A *model* of T is a model of φT in the sense of classical logic. A *consequence* of T is a sentence of L that is true in all models of T.

Bibliography

[Allen, 1984] James Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.

[Apt and Bezem, 1990] Krzysztof Apt and Marc Bezem. Acyclic programs. In David Warren and Peter Szeredi, editors, *Logic Programming: Proc. of the Seventh Int'l Conf.*, pages 617–633, 1990.

[Baker, 1991] Andrew Baker. Nonmonotonic reasoning in the framework of situation calculus. *Artificial Intelligence*, 49:5–23, 1991.

[Baral and Gelfond, 1993] Chitta Baral and Michael Gelfond. Representing concurrent actions in extended logic programming. In *Proc. of IJCAI-93*, pages 866–871, 1993.

[Bidoit and Froidevaux, 1988] Nicole Bidoit and Christine Froidevaux. Negation by default and nonstratifiable logic programs. Technical Report 437, Université Paris XI, 1988.

[Bledsoe and Feng, 1993] Woodrow Bledsoe and Guohui Feng. Set-Var. *Journal of Automated Reasoning*, 11:293–314, 1993.

[Boyer and Moore, 1988] Robert Boyer and Strother Moore. *A Computational Logic Handbook*. Academic Press, 1988.

[Brewka and Hertzberg, 1993] Gerhard Brewka and Joachim Hertzberg. How to do things with worlds–on formalizing actions and plans. *Journal of Logic and Computation*, 3(5):517–532, 1993.

[Chen and Warren, 1992] Weidong Chen and David S. Warren. A goal-oriented approach to computing well-founded semantics. In Krzysztof Apt, editor, *Proc. Joint Int'l Conf. and Symp. on Logic Programming*, pages 589–603, 1992.

[Crawford and Etherington, 1992] James Crawford and David Etherington. Formalizing reasoning about change: A qualitative reasoning approach. In *Proc. AAAI-92*, pages 577–583, 1992.

[Davis, 1991] Ernest Davis. *Representations of Commonsense Knowledge*. Morgan Kaufmann, San Mateo, CA, 1991.

[Denecker and De Schreye, 1993] Mark Denecker and Danny De Schreye. Representing incomplete knowledge in abductive logic programming. In Dale Miller, editor, *Logic Programming: Proceedings of the 1993 Int'l Symposium*, pages 147–163, 1993.

[Dijkstra and Scholten, 1990] E. W. Dijkstra and C. Scholten. *Predicate Calculus and Program Semantics*. Springer Verlag, New York, 1990.

[Dung, 1993] Phan Minh Dung. Representing actions in logic programming and its applications in database updates. In *Logic Programming: Proceedings of the Tenth Int'l Conf. on Logic Programming*, pages 222–238, 1993.

[Elkan, 1992] Charles Elkan. Reasoning about action in first-order logic. In *Proceedings of the Ninth Biennial Conference of the Canadian Society for Computational Studies of Intelligence (CSCSI'92), Vancouver, Canada*. Morgan Kaufmann, 1992.

[Etherington *et al.*, 1985] David Etherington, Robert Mercer, and Raymond Reiter. On the adequacy of predicate circumscription for closed world reasoning. *Computational Intelligence*, 1:11–15, 1985.

[Etherington, 1987] David Etherington. A semantics for default logic. In *Proc. IJCAI-87*, pages 495–498, 1987.

[Evans, 1989] Chris Evans. Negation-as-failure as an approach to the Hanks and McDermott problem. In *Proc. of the Second Int'l Symp. on Artificial Intelligence*, 1989.

[Fikes and Nilsson, 1971] Richard Fikes and Nils Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3–4):189–208, 1971.

[Finger, 1986] Jeffrey Finger. Exploiting constraints in design synthesis. PhD thesis, Stanford University, 1986.

[Gabbay and Ohlbach, 1992] Dov Gabbay and Hans J. Ohlbach. Quantifier elimination in second-order predicate logic. In Bernhard Nebel, Charles Rich, and William Swartout, editors, *Proc. of the Third Int'l Conf. on Principles of Knowledge Representation and Reasoning*, 1992.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3,4):365–385, 1991.

[Gelfond and Lifschitz, 1993] Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17:301–322, 1993.

[Gelfond *et al.*, 1991a] Michael Gelfond, Vladimir Lifschitz, Halina Przymusińska, and Miroslaw Truszczyński. Disjunctive defaults. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Principles of Knowledge Representation and Reasoning: Proc. of the Second Int'l Conf.*, pages 230–237, 1991.

[Gelfond *et al.*, 1991b] Michael Gelfond, Vladimir Lifschitz, and Arkady Rabinov. What are the limitations of the situation calculus? In Robert Boyer, editor, *Automated Reasoning: Essays in Honor of Woody Bledsoe*. Kluwer Academic, Dordrecht, 1991.

[Gelfond, 1989] Michael Gelfond. Autoepistemic logic and formalization of commonsense reasoning. In Michael Reinfrank, Johan de Kleer, Matthew Ginsberg, and Erik Sandewall, editors, *Non-Monotonic Reasoning: 2nd International Workshop (Lecture Notes in Artificial Intelligence 346)*, pages 176–186. Springer-Verlag, 1989.

[Ginsberg and Smith, 1988a] Matthew L. Ginsberg and David E. Smith. Reasoning about action I: a possible worlds approach. *Artificial Intelligence*, 35(2):165–195, 1988.

[Ginsberg and Smith, 1988b] Matthew L. Ginsberg and David E. Smith. Reasoning about action II: the qualification problem. *Artificial Intelligence*, 35(3):311–342, 1988.

[Giunchiglia and Lifschitz, 1995] Enrico Giunchiglia and Vladimir Lifschitz. Dependent fluents. Manuscript, 1995.

[Giunchiglia *et al.*, 1995] Enrico Giunchiglia, G. Neelakantan Kartha, and Vladimir Lifschitz. Actions with indirect effects. Manuscript, 1995.

[Green, 1969] C Green. Applications of theorem proving to problem solving. In *Proc. of IJCAI-69*, pages 219–239, 1969.

[Gries and Schneider, 1993] David Gries and Fred B. Schneider. *A Logical Approach to Discrete Math*. Springer Verlag, New York, 1993.

[Guerreiro and Casanova, 1990] Ramiro Guerreiro and Marco Casanova. An alternative semantics for default logic. Preprint, The Third International Workshop on Nonmonotonic Reasoning, South Lake Tahoe, 1990.

[Hanks and McDermott, 1987] Steve Hanks and Drew McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence*, 33(3):379–412, 1987.

[Kartha and Lifschitz, 1994] G. Neelakantan Kartha and Vladimir Lifschitz. Actions with indirect effects. In Jon Doyle, Erik Sandewall, and Piero Torasso, editors, *Proc. of the Fourth Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 341–350, 1994.

[Kartha and Lifschitz, 1995] G. Neelakantan Kartha and Vladimir Lifschitz. A simple formalization of actions using circumscription. In *Proc. of IJCAI-95*, 1995. To appear.

[Kartha, 1993] G. Neelakantan Kartha. Soundness and completeness theorems for three formalizations of action. In *Proc. of IJCAI-93*, pages 724–729, 1993.

[Kartha, 1994] G. Neelakantan Kartha. Two counterexamples related to Baker's approach to the frame problem. *Artificial Intelligence*, 69:379–391, 1994.

[Kautz, 1986] Henry Kautz. The logic of persistence. In *Proc. of AAAI-86*, pages 401–405, 1986.

[Konolige, 1988] Kurt Konolige. On the relation between default and autoepistemic logic. *Artificial Intelligence*, 35:343–382, 1988.

[Konolige, 1993] Kurt Konolige. Autoepistemic logic. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *The Handbook of Logic in AI and Logic Programming*, volume 3, pages 217–295. Oxford University Press, 1993.

[Kowalski and Sadri, 1994] Robert Kowalski and Fariba Sadri. The situation calculus and event calculus compared. In Maurice Bruynooghe, editor, *Proc. of ILPS-94*, pages 539–553, 1994.

[Kowalski and Sergot, 1986] Robert Kowalski and Marek Sergot. A logic-based calculus of events. *New Generation Computing*, 4:67–95, 1986.

[Lifschitz *et al.*, 1993] Vladimir Lifschitz, Norman McCain, and Hudson Turner. Automated reasoning about action: A logic programming approach. In Dale Miller, editor, *Proc. of ILPS-93*, page 641, 1993.

[Lifschitz, 1987a] Vladimir Lifschitz. Formal theories of action. In Frank M. Brown, editor, *The Frame Problem in Artificial Intelligence, Proc. of the 1987 Workshop*, pages 35–58, 1987.

[Lifschitz, 1987b] Vladimir Lifschitz. On the semantics of STRIPS. In Michael Georgeff and Amy Lansky, editors, *Reasoning about Actions and Plans*, pages 1–9. Morgan Kaufmann, San Mateo, CA, 1987.

[Lifschitz, 1989] Vladimir Lifschitz. Benchmark problems for formal non-monotonic reasoning, version 2.00. In Michael Reinfrank, Johan de Kleer, Matthew Ginsberg, and Erik Sandewall, editors, *Non-Monotonic Reasoning: 2nd International Workshop (Lecture Notes in Artificial Intelligence 346)*, pages 202–219. Springer-Verlag, 1989.

[Lifschitz, 1990] Vladimir Lifschitz. Frames in the space of situations. *Artificial Intelligence*, 46:365–376, 1990.

[Lifschitz, 1991] Vladimir Lifschitz. Towards a metatheory of action. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Proc. of the Second Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 376–386, 1991.

[Lifschitz, 1993a] Vladimir Lifschitz. Circumscription. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *The Handbook of Logic in AI and Logic Programming*, volume 3, pages 297–352. Oxford University Press, 1993.

[Lifschitz, 1993b] Vladimir Lifschitz. A language for describing actions. In *Working Papers of the Second Symposium on Logical Formalizations of Commonsense Reasoning*, pages 103–113, 1993.

[Lifschitz, 1993c] Vladimir Lifschitz. Restricted monotonicity. In *Proc. of the Eleventh National Conference of Artificial Intelligence*, pages 432–437, 1993.

[Lifschitz, 1995] Vladimir Lifschitz. Nested abnormality theories. *Artificial Intelligence*, 1995. To appear.

[Lin and Reiter, 1994] Fangzhen Lin and Raymond Reiter. State constraints revisited. *Journal of Logic and Computation, Special Issue on Actions and Processes*, 4(5):655–678, 1994.

[Lin and Shoham, 1991] Fangzhen Lin and Yoav Shoham. Provably correct theories of action. In *Proc. of the Ninth National Conference of Artificial Intelligence*, pages 349–354, 1991.

[Lin, 1995] Fangzhen Lin. Embracing causality in theories of action. Manuscript, 1995.

[McCain and Turner, 1995] Norman McCain and Hudson Turner. A causal theory of ramifications and qualifications. In *Working Notes of the AAAI Spring Symposium on Extending Theories of Action*, pages 130–135, 1995.

[McCarthy and Hayes, 1969] John McCarthy and Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 4, pages 463–502. Edinburgh University Press, Edinburgh, 1969.

[McCarthy, 1977] John McCarthy. Epistemological problems of artificial intelligence. In *Proc. of IJCAI-77*, pages 1038–1044, 1977.

[McCarthy, 1980] John McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, 13(1–2):27–39, 1980.

[McCarthy, 1986] John McCarthy. Applications of circumscription to formalizing common sense knowledge. *Artificial Intelligence*, 28(1):89–116, 1986.

[McDermott and Doyle, 1980] Drew McDermott and Jon Doyle. Nonmonotonic logic I. *Artificial Intelligence*, 13(1–2):41–72, 1980.

[McDermott, 1982a] Drew McDermott. Nonmonotonic logic II: Nonmonotonic modal theories. *Journal of the ACM*, 29(1):33–57, 1982.

[McDermott, 1982b] Drew McDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.

[Miller and Shanahan, 1994] Rob Miller and Murray Shanahan. Narratives in the situation calculus. *Journal of Logic and Computation, Special Issue on Actions and Processes*, 4(5):511–530, 1994.

[Moore, 1984] Robert Moore. Possible-world semantics for autoepistemic logic. In *Non-Monotonic Reasoning Workshop*, pages 344–354, 1984.

[Moore, 1985] Robert Moore. Semantical considerations on nonmonotonic logic. *Artificial Intelligence*, 25(1):75–94, 1985.

[Morris, 1988] Paul Morris. The anomalous extension problem in default reasoning. *Artificial Intelligence*, 35(3):383–399, 1988.

[Myers and Smith, 1988] Karen Myers and David Smith. The persistence of derived information. In *Proc. AAAI-88*, pages 496–500, 1988.

[Pednault, 1989] Edwin Pednault. ADL: Exploring the middle ground between STRIPS and the situation calculus. In Ronald Brachman, Hector Levesque, and Raymond Reiter, editors, *Proc. of the First Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 324–332, 1989.

[Pinto and Reiter, 1993] Javier A. Pinto and Raymond Reiter. Temporal reasoning in logic programming: A case for the situation calculus. In *Proc. ICLP-93*, pages 203–221, 1993.

[Pinto, 1994] Javier A. Pinto. Temporal reasoning in the situation calculus. Ph. D. thesis, University of Toronto (available as Technical Report KRR-TR-94-1), 1994.

[Poole, 1993] David Poole. Default logic. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *The Handbook of Logic in AI and Logic Programming*, volume 3, pages 189–215. Oxford University Press, 1993.

[Reiter, 1980] Raymond Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1–2):81–132, 1980.

[Reiter, 1991] Raymond 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*, pages 359–380. Academic Press, 1991.

[Reiter, 1993] Raymond Reiter. Proving properties of states in the situation calculus. *Artificial Intelligence*, 64(2):337–351, 1993.

[Sandewall, 1989] Erik Sandewall. Filter preferential entailment for the logic of action in almost continuous worlds. In *Proc. of IJCAI-89*, pages 894–899, 1989.

[Sandewall, 1992] Erik Sandewall. Features and fluents. Technical Report LiTH-IDA-R-92-30, Linkoping University, Sweden, September 1992.

[Schubert, 1990] L.K. Schubert. Monotonic solution of the frame problem in the situation calculus: an efficient method for worlds with fully specified actions. In H.E Kyberg, R.P Loui, and G.N Carlson, editors, *Knowledge Representation and Defeasible Reasoning*, pages 23–67. Kluwer Academic Press, Boston, Mass., 1990.

[Shoham, 1987] Yoav Shoham. *Reasoning about change*. MIT Press, Boston, MA, 1987.

[Shoham, 1988] Yoav Shoham. Chronological ignorance: experiments in nonmonotonic temporal reasoning. *Artificial Intelligence*, 36:279–331, 1988.

[Shvarts, 1990] Grigori Shvarts. Autoepistemic modal logics. In Rohit Parikh, editor, *Theoretical Aspects of Reasoning about Knowledge: Proc. of the Third Conf.*, pages 97–110, 1990.

[Siklóssy and Roach, 1975] L. Siklóssy and J. Roach. Model verification and improvement using DISPROVER. *Artificial Intelligence*, 6:41–52, 1975.

[Subramanian, 1993] Sakthi Subramanian. *A Mechanized Framework for Specifying Problem Domains and Verifying Plans*. PhD thesis, University of Texas, Austin, Department of Computer Science, 1993.

[Thielscher, 1994a] Michael Thielscher. An Analysis of Systematic Approaches to Reasoning about Actions and Change. In P. Jorrand, editor, *International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA)*, Sofia, Bulgaria, 1994. World Scientific Publishing Co. Singapore.

[Thielscher, 1994b] Michael Thielscher. Representing actions in equational logic programming. In Pascal Van Hentenryck, editor, *Proc. ICLP-94*, pages 207–224, 1994.

[Turner, 1994] Hudson Turner. Signed logic programs. In Maurice Bruynooghe, editor, *Proc. of ILPS-94*, pages 61–75. MIT Press, 1994.

[Winslett, 1988] Marianne Winslett. Reasoning about action using a possible models approach. In *Proc. of AAAI-88*, pages 89–93, 1988.

[Zhang and Foo, 1995] Yan Zhang and Norman Foo. DNC: Representing nondeterminism and causality in reasoning about action. Manuscript, 1995.

Vita

G. Neelakantan Kartha was born in Alappuzha, Kerala, India on July 1, 1963, the son of R. Gopalakrishna Kurup and K.B. Subhadra Kunjamma. He received his Secondary School Leaving Certificate in March, 1978 from Sanathana Dharma Vidyalaya, Alappuzha, a Pre Degree certificate in May, 1980 from Sanathana Dharma College, Alappuzha, a Bachelor of Engineering degree in Electronics and Communication in March, 1985 from Regional Engineering College, Tiruchirappally and a Master of Science degree in Computer Science in May, 1988 from Clarkson University, New York. In September, 1988, he joined the Department of Computer Sciences at the University of Texas at Austin as a doctoral student. He married Rohini Nair on June 17, 1990. He has published the following papers.

1. G. Neelakantan Kartha. Soundness and Completeness Theorems for Three Formalizations of Action. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence, pages 712-718, 1993*. A preliminary version appeared in *Proceedings of Common Sense 93: The Second Symposium on Logical Formalizations of Commonsense Reasoning, pages 85-93, 1993*.

2. G. Neelakantan Kartha and Vladimir Lifschitz. Actions with Indirect Effects (preliminary report). In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning, pages 341-350, 1994.*

3. G. Neelakantan Kartha. Two Counterexamples Related to Baker's Approach to the Frame Problem. In *Artificial Intelligence, volume 69, pages 379-391, 1994*.

4. Enrico Giunchiglia, G. Neelakantan Kartha and Vladimir Lifschitz. Actions with Indirect Effects (extended abstract). In *Working Notes of the AAAI Spring Symposium Series on Extending Theories of Action: Formal Theories and Practical Applications, pages 80-85, 1995*.

5. G. Neelakantan Kartha and Vladimir Lifschitz. A Simple Formalization of Actions Using Circumscription. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1995*. To appear.

Permanent Address: Sri Nilayam,
Sanathanam Ward,
Alappuzha-688001,
Kerala, India.

This dissertation was typeset with $\LaTeX\,2_\varepsilon$[1] by the author.

[1] $\LaTeX\,2_\varepsilon$ is an extension of $\LaTeX$. $\LaTeX$ is a collection of macros for $\TeX$. $\TeX$ is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin.