Formalizing Narratives using Nested Circumscription

Chitta Baral, Alfredo Gabaldon

Dept. of Computer Science, University of Texas at El Paso, El Paso, Texas 79968
http://cs.utep.edu/csdept/krgroup.html
{chitta,alfredo}@cs.utep.edu

Alessandro Provetti

D.S.I., Università di Milano, Via Comelico 39, I-20135 Milan, Italy
http://eolo.usr.dsi.unimi.it/~provetti/
provetti@dsi.unimi.it

Abstract

Representing and reasoning about narratives together with the ability to do hypothetical reasoning is important for agents in a dynamic world. These agents need to record their observations and action executions as a narrative and at the same time, to achieve their goals against a changing environment, they need to make plans (or re-plan) from the *current* situation. The early action formalisms did one or the other. For example, while the original situation calculus was meant for hypothetical reasoning and planning, the event calculus was more appropriate for narratives. Recently, there have been some attempts at developing formalisms that do both. Independently, there has also been a lot of recent research in reasoning about actions using circumscription. Of particular interest to us is the research on using high-level languages and their logical representation using nested abnormality theories (NATs) –a form of circumscription with blocks that make knowledge representation modular. Starting from from theories in the high-level language $\mathcal{L}$, which is in passing extended to allow concurrent actions, we define a translation to NATs that preserves both narrative and hypothetical reasoning. We initially use the high level language $\mathcal{L}$, and then extend it to allow concurrent actions. In the process, we study several knowledge representation issues such as filtering, and restricted monotonicity with respect to NATs. Finally, we compare our formalization with other approaches, and discuss how our use of NATs makes it easier to incorporate other features of action theories, such as constraints, to our formalization.

1 Introduction

A narrative is a possibly incomplete set of observations about the world in terms of what actions/events occured and the value of fluents at different instants in the past. Initial formulations of narratives, such as Kowalski and Sergot's event calculus [KowSer86] and Allen's temporal logic [All84], were concerned about inferring values of fluents at time instants other than those explicitly given in the narrative and also possibly 'abduce' occurrences of actions not explicitly mentioned in the narrative. These formulations were not concerned [1] with hypothetical and/or counterfactual reasoning about values of fluents in a possible future world reached by executing a sequence of actions. Such hypothetical reasoning is important from the point of view of planning, where an agent needs to construct a plan –normally a sequence of actions, to achieve a particular goal.

The formalism of situation calculus [McCHay69] has been normally used for hypothetical reasoning about actions and forms the basis of classical planning. But in its original form it does not allow narratives; the only actual observations that can be expressed in it are about values of fluents in the initial situation.

1.1 Allowing narratives with hypothetical reasoning

Recently, researchers [MilSha94,KakMil96,PinRei93,McC95,BarGabPro96a,Pro96] have realized the importance of having a formalism that captures both narratives and hypothetical reasoning about actions effects. Such a formalism is necessary to formulate planning and execution of actions of an agent in a dynamic environment, where exogenous actions may occur. The agent has to record observations about occurrences of actions (both its own action executions, and exogenous happenings) or fluent values and sometimes infer them. Also, the agent has to make plans and more importantly may have to dynamically revise its plans or construct new ones when faced with exogenous events. These plans are not from the initial situation as in situation calculus, but from the *current* situation. Although the terms 'dynamic planning', 'planning with execution', and 'reactive planning' have been used in the planning community, a formalism that allows both narratives and hypothetical reasoning is neces-

[1] It was later shown that some of these formalisms can use abduction in the meta-level to do hypothetical reasoning and planning. However, Reiter [Rei96] has argued why it is better to do hypothetical reasoning at the object level. Besides, the action descriptions used in these formalisms were restrictive in the sense that they did not allow features such as constraints.

sary to form the backbone of such planners – the role situation calculus plays for classical planners.

Miller and Shanahan [MilSha94] and Pinto and Reiter [PinRei93] were perhaps the first who considered both narratives and hypothetical reasoning. Both formalisms made many important contributions: the former showed how situation calculus can be extended to incorporate narratives, and the latter introduced the concept of *actual situations* –those that were reached by actual occurrences of actions. Still, both formalisms have several drawbacks. For example, Miller and Shanahan only allow fluent facts about the initial state, and require that all action occurrences be explicitly stated. Also, Pinto and Reiter's [PinRei93] solution suffer from *premature minimization* of occurrences, i.e., while minimizing occurrence of actions the possible existence of new situations –those not dictated by the given axiomatization– is ruled out [2]. This prevents them from being able to make plans. As a result, both approaches lack the re-planning ability of an agent who, for instance, after making a plan of packing his suitcase and driving the car to the airport and doing the packing, observes his car with a demobilizing dent. The agent did not see the exogenous action that made the dent so it cannot put a corresponding action occurrence to its narrative, but it should be able to infer such an occurrence from its observation that the car has a dent.

1.2 Expressiveness of our formalism

In this paper we focus on a more general formalism that allows both narratives and hypothetical reasoning and overcomes many of the limitations of the earlier proposals. In particular, besides being able to express fluent values at the initial state, and effects of actions on fluents –as allowed in $\mathcal{A}$ [GelLif92], the precursor of $\mathcal{L}$– we can also express *observations* such as:

- fluent facts about non-initial states,
- actual occurrence of actions at different time points (labeled by actual-situation constants), and
- ordering among actual situations.

Given such descriptions, which includes observations, we can then:

[2] This term was coined by Reiter in [Rei96] where he discussed this drawback. The later papers of Reiter and Pinto [Rei96,Pin96] avoid this drawback but to prevent it they abandon minimization of action occurrences. We further explain 'premature minimization' and do a detailed comparison of our approach with theirs in Section 11.5.

– plan from the current situation by doing hypothetical reasoning;
– explain observations through inferring action occurrences (not explicitly mentioned);
– infer new fluent facts (that are not in the narrative) about the various situations that explain the observations or are implied by them;
– do counterfactual reasoning about action occurrences [PinRei93,Pro96].

Moreover, our formalism allows a clear distinction between observations and hypothesis, and makes it clear why the situation calculus atom $holds(f, [a_1, \ldots, a_n])$ is a hypothesis and not an observation.

Our formulation is done using a novel form of circumscription called *Nested Abnormality Theories* (NATs) [Lif95]. One important aspect of our approach is that unlike most of the earlier formalizations of narratives [MilSha94,PinRei93] that were done directly in a logical language, our formalization is grounded to the high level language $\mathcal{L}$ developed by Baral, Gelfond and Provetti [BarGelPro95b]. In other words, in the first half of the paper we give a translation of descriptions in $\mathcal{L}$ to nested abnormality theories and show their equivalence. Later, we extend $\mathcal{L}$ to allow concurrent actions and give a translation of descriptions in this extended language to nested abnormality theories. $\mathcal{L}$ is one of a family of high-level action languages recently proposed in literature [GelLif92,San92,KakMil96].

1.3 High-level language approach vs direct formalization approach

In general, researchers that use a high level language to formalize reasoning about actions have normally followed the methodology of:

– developing a high level language – a fairly restricted language with English-like syntax that can be easily followed, but with a precise semantics (that makes common-sense assumptions used in the formulation precise), often defined using an automata that makes it easier to incorporate notions such as the 'frame,' 'qualification' and 'ramification' that are otherwise difficult to encode in a logical language; and

– defining correct (sometimes complete) translations of high-level language theories into classical logic or logic programming so as to apply existing query-answering systems and compare different approaches within a well known setting.

The approaches that directly formalize reasoning about actions in a logical language, which we will refer to as the *direct formalization approach,* also often require the initial descriptions to be given in a particular form and in a restricted subset of the logical language. Unlike high-level language approaches, where queries are also required to have a restricted syntax, the direct formalization approach allow queries to be any sentence in the logical language. Even though this may seem like a big plus point for following the direct formalization approach, often by restricting the query language faster querying mechanisms can be developed. This is the reality in most systems, such as databases and knowledge bases (including expert systems).

The other main difference between the two approaches is that in the direct formalization approach the semantics is directly defined in a standard logical language without grounding it to an independent semantics. Although having an extra semantics may seem superfluous at first sight, it forms the basis to which the logical formalization can be grounded and their correctness becomes a precise mathematical question. In the absence of this extra semantics, the correctness of a direct formalization approach becomes an empirical question where different examples (in English) are formalized and the adequacy of the formalization is decided on whether the formalization of the examples matches our intuition or not. When examples are given in English, the assumptions surrounding them are often partially stated; thus opening up the possibility of multiple interpretations and as a result the adequacy and correctness of a direct formalization approach can never be precisely settled. A new example, often with different underlying assumptions than used in the original formalization, starts a debate about the adequacies of the formalization. *This is partially what happened during the debate about the Yale-Shooting problem.* Also, since the extra semantics in the high level language approach is defined using automata and structures (such as a state being a collection of fluents) that make it easy to formalize notions that are difficult to express in a logical language, by having this semantics we can later precisely determine if a logical formulation of a difficult notion (such as the frame problem) is correct or not. Not having this extra semantics, and the inherent difficulty (for even experts) of the mathematics of the logical languages, makes it hard to evaluate the appropriateness of a formulation in the direct approach. (In contrast, the simplicity of the high level semantics often makes it easier to evaluate its appropriateness with respect to examples.)

Two anecdotal examples make this point. The first is the formalization of the frame problem in the various logical languages that were the subject of debate in the Yale-shooting issues. The formulation of this in the high level language $\mathcal{A}$ is extremely simple; thus, for any example that can be expressed in $\mathcal{A}$, the correctness of a particular logical formulation can easily be settled once and for all by checking whether it matches the semantics of the formulation in $\mathcal{A}$. The second anecdotal example is the case of 'premature minimization' mentioned earlier. This feature was not intended by researchers working directly in situation calculus. In fact, their goal was to capture narratives together with planning. Unfortunately, their direct logical formalization, which was not verified with respect to an independent semantics, had this unintended consequence. Perhaps the high level language approach could have reduced the debate in the former anecdote and prevented the oversight in the latter.

We would like to add that the high level language approach facilitates the comparison of different theories of action. The best example is Kartha's [Kar93] comparison of three action languages by translating them into $\mathcal{A}$ and proving their equivalence. Finally, in this paper by having high-level descriptions with an independent semantics and a corresponding circumscriptive theory, we agree with Hoare's view [Hoa87]:

> For specifications of the highest quality and importance I would recommend complete formalization of requirements in two entirely different styles, together with a proof that they are consistent or even equivalent to each other. [...] A language for which two consistent and complementary definitions are provided may be confidently taken as a secure basis for software engineering.

1.4 Advantages of using nested abnormality theories

One major advantage of using nested abnormality theories in our work is that it allows easy incorporation of additional features, such as constraints, which already have been formalized –in the absence of narratives, with NATs [KarLif94,GiuKarLif95]. Also, although the nesting of blocks in NATs may at first glance suggest loss of declarativeness and elaboration tolerance [McC88], we believe it makes it easier to represent knowledge, particularly in the action domain (see [Lif95,KarLif94,GiuKarLif95] for more on this). This is because we can develop blocks that represent meaningful structural units and use the blocks in other units without worrying about undesired interactions. Besides, a sub-theory, e.g. a predicate definition, can be elaborated by just changing the appropriate block, without any non-local surgery on the rest of the theory. (In Section 9 we show how we can add constraints to our theory by only changing the block that encodes the transition function.) To some extent, using 'flat' circumscription to formalize a large body of knowledge is like writing a large program without using subroutines. We believe that the simplicity of our translation of $\mathcal{L}$ (as presented in this paper) demonstrates the usefulness of NATs for knowledge representation.

1.5 Other contributions

An additional and important contribution of this work is that the formalization presented here constitutes a full, non-trivial example of the use of *value* *minimization,* a technique developed by Baral et. al. [BarGabPro96b] for minimizing the value of a function in a circumscriptive theory. Value minimization is used in our formalization to capture the assumption that the only action occurrences which can be derived from a set of observations are those which are necessary to explain the observations themselves.

In reasoning with narratives, since we allow facts about non-initial states and partial knowledge about action occurrences, we need some form of abduction to entail occurrence of such actions and values of fluents at time points that are not explicitly stated in the narrative. Unlike [KakMil96], where a particular kind of abductive reasoning (through explanations to restore consistency) is done at the meta level, we use filtering [San89,San92] at the object level.

To sum up, our formalization seems to be of independent interest from the point of view of knowledge representation and nonmonotonic logics, since it illustrates the usefulness of NATs in terms of ease of representation and restricted monotonicity [Lif93], the use of value minimization of functions in knowledge representation and the formalization of filtering in NATs,

1.6 Organization of the rest of the paper

We start with an overview of the high level language $\mathcal{L}$ from [BarGelPro95b,BarGelPro97] in Section 2 and an overview of NATs in Section 3. We then give a translation of domain descriptions in $\mathcal{L}$ into NATs in Section 4 and illustrate the NAT characterization with respect to some examples in Section 6. In Section 5 we discuss value minimization of functions and its role in our translation described in Section 4. In Section 7, we formally relate the entailment in $\mathcal{L}$ to the entailment in our translation. In Section 8 we extend $\mathcal{L}$ to allow concurrent actions, give a translation of domain descriptions in this extended language to NATs and relate their entailment relation. In Section 9, we show how our formulation in terms of NATs can be easily extended to take into account constraints and in Section 10 we discuss the restricted monotonicity property of our NAT formulation and show the relation between abductive reasoning, filtering and restricted monotonicity. In Section 11 we compare our work with earlier work on combining narratives and hypothetical reasoning –particularly by Miller, Shanahan, Pinto, Reiter, and McCarthy. Finally, all the proofs are given in the Appendix.

2 Overview of $\mathcal{L}$

The high level language $\mathcal{L}$ was developed by Baral, Gelfond and Provetti [BarGelPro95b,BarGelPro97] to allow representation of and reasoning with narratives together with hypothetical reasoning.

Beside the syntax and semantics of $\mathcal{L}$, [BarGelPro97] defines a translation to logic programs and Prolog for a subclass of this language. Here, we start with a translation of domain descriptions in $\mathcal{L}$ to NATs, and then extend it to allow concurrent actions. *Let us make it clear that the contribution of this paper is not the development of $\mathcal{L}$ but rather the formalization of $\mathcal{L}$ in NATs, which allows easy integration of additional features such as constraints; and the study of our NAT formalization in terms of its impact to knowledge representation in general.*

Let us now give a brief overview of the syntax and semantics of $\mathcal{L}$. To make this paper self-contained, we present motivations for the choice of constructs in this language and give several examples. Additional detailed motivations and examples can be found in [BarGelPro97].

The alphabet of $\mathcal{L}$ consists of three disjoint nonempty sets of symbols $\mathcal{F}$, $\mathcal{A}$ and $\mathcal{S}$, called *fluents*, *actions*, and *actual situations*. Elements of $\mathcal{A}$ and $\mathcal{S}$ will be denoted by (possibly indexed) letters A and S respectively. We will also assume that $\mathcal{S}$ contains two special situations S_0 and S_N called *initial* and *current* situation, respectively.

A *fluent literal* is a fluent possibly preceded by $\neg$. Fluent literals will be usually denoted by letters F, P and Q with indexes. $\neg\neg F$ will be equated to F. For a fluent F, by $\overline{\neg F}$ we mean F, and by $\overline{F}$ we mean $\neg F$.

There are two kinds of propositions in $\mathcal{L}$ called causal laws and facts. Causal laws are the same as effect propositions in $\mathcal{A}$ [GelLif92], i.e. a *causal law* is an expression of the form:

$$A \textbf{ causes } F \textbf{ if } P_1, \ldots, P_n \tag{1}$$

where A is an action, and $F, P_1, \ldots, P_n$ $(n \geq 0)$ are fluent literals. $P_1, \ldots, P_n$ are called *preconditions* of (1). We will read (1) as "F is guaranteed to be true after the execution of an action A in any state of the world where $P_1, \ldots, P_n$ are true." If $n = 0$, we write the causal law as A **causes** F.

An atomic *fluent fact* is an expression of the form

$$F \textbf{ at } S \tag{2}$$

where F is a fluent literal and S is a situationfootnoteUnless otherwise stated, by situations we will mean actual situations.. The intuitive reading of (2) is "F was observed to be true in situation S." An agent in a dynamic world can update its knowledge about the world by adding such fluent facts (based on its observations) to its database.

An atomic *occurrence fact* is an expression of the form

$$\alpha \ \textbf{occurs\_at} \ S \tag{3}$$

where α is a sequence of actions, and S is a situation. It states that "the sequence α of actions was observed to have occurred in situation S." (We assume that actions in a sequence follow each other immediately). An agent in a dynamic world can update its knowledge about action occurrences – both exogenous actions, and actions executed by the agent itself, by adding such occurrence facts to its database.

An atomic *precedence fact* is an expression of the form

$$S_1 \ \textbf{precedes} \ S_2 \tag{4}$$

where S_1 and S_2 are situations. It states that the domain was in situation S_2 after being in situation S_1. The agent can use such precedence facts to record the temporal ordering between its observations about fluent facts and occurrence facts. Precedence facts saying a situation precedes the initial one, e.g., S_1 **precedes** S_0, are of course forbidden.

Since propositions of type (1) express general knowledge about effects of actions, they are referred to as *laws*. Propositions (2), (3) and (4) are called *atomic facts* or *observations*. A *fact* is a propositional combination of atomic facts. A collection of laws and facts is called *domain description*; he sets of laws and facts of a domain description D will be denoted by D_l and D_f respectively. Domain descriptions are required to satisfy the following properties:

– its propositions do not contain the situation constant S_N, and
– for every situation there is at most one occurrence fact.

The first property is very important and allows nonmonotonic interpretation of the situation constant S_N, which represents the current situation. The purpose of the second property is to prohibit concurrent actions in $\mathcal{L}$.

An important feature of an agent associated with an $\mathcal{L}$ theory is the ability to reason with incomplete narratives and make revisable and nonmonotonic conclusions about observations it might have missed [3]. The following example further illustrates our point.

Example 1 *(discovering occurrences)* Consider the following version of the stolen-car example: we know that initially we have a car (in the garage). However, at a later instance of time we observe that we no longer have the car. We also know that after our car gets stolen then we will no longer have the car. The following is a description of this story in $\mathcal{L}$:

$$D_1 = \begin{cases} steal(car) \ \textbf{causes} \ \ \neg has(car) \\ has(car) \ \textbf{at} \ \ S_0 \\ \neg has(car) \ \textbf{at} \ \ S_1 \\ S_0 \ \textbf{precedes} \ \ S_1 \end{cases}$$

Intuitively, from this description we as a smart agent would like to conclude that action $steal(car)$ must have occured in situation S_0 thus explaining the fact that in that situation we had a car but in situation S_1 we no longer have it. In other words, we *abduce* that action $steal(car)$ occurred in the initial situation causing $has(car)$ to become false. The semantics of the language $\mathcal{L}$ as defined in Section 2.2, indeed ensures that D_1 entails $steal(car)$ **occurs_at** s_0.

It is clear that in reaching this conclusion we are making some assumptions about the domain. We are, for example, assuming that action $steal(car)$ is the only action that may cause $has(car)$ to become false. Otherwise, the conclusion would be unsustained as it could be explained by the occurrence of another action. All the assumptions embodied in domain descriptions in $\mathcal{L}$ are discussed in the subsection below. □

Note that fluent facts about situations other than S_0 are allowed and, as it is shown in this example, a semantics of $\mathcal{L}$ must be able to capture some form of *abductive reasoning* to be able to abduce the explanations to these observations.

2.1 Assumptions embodied in $\mathcal{L}$ domain descriptions

Domain descriptions in $\mathcal{L}$ are used in conjunction with the following informal assumptions which restrict the language so that the agent can perform intelligent (perhaps hasty, and thus revisable) reasoning.

[3] Unlike in [MilSha94,KakMil96] we would like our agent to make this conclusion in the object language itself, not through meta level reasoning.

a) changes in the values of fluents can only be caused by execution of actions;
b) there are no actions except those from the language of the domain description;
c) there are no effects of actions except those specified by the causal laws;
d) no actions occur except those needed to explain the facts in the domain description, and
e) actions do not overlap or happen simultaneously.

These assumptions give an intuitive understanding of domain descriptions in $\mathcal{L}$. We now present the semantics of domain descriptions in $\mathcal{L}$ as defined in [BarGelPro97], which precisely specifies the sets of acceptable conclusions which can be reached from such descriptions and assumptions (a)-(e).

2.2 Semantics of $\mathcal{L}$

In $\mathcal{L}$, states of the world are represented by sets of the fluents which are true. Actions executed in a particular state may add/remove such fluents according to causal laws. Thus, the set of all possible evolutions of the world described by the causal laws can be represented by a transition diagram with states corresponding to states of the world and transitions labeled by actions. Satisfying the facts in a domain description intuitively consists of selecting a state as the initial one, and a path in the diagram describing the actual evolution of the domain.

A *state* is a set of fluent names. A *causal interpretation* is a partial function Ψ from sequences of actions to states such that: (i) the empty sequence, $[\,]$, belongs to the domain of Ψ; and (ii) Ψ is prefix-closed. [4]

$\Psi([\,])$ is called the initial state of Ψ. A partial function Ψ serves as an interpretation [5] of the laws of D. If a sequence α belongs to the domain of Ψ, we say that α is *possible* in the initial state of Ψ.

Given a fluent F and a state σ, we say that F *holds* in σ (F is *true* in σ) if

[4] By "prefix closed" we mean that for any sequence of actions α and action A, if $\alpha \cdot A$ is in the domain of Ψ then so is α, where $\alpha \cdot A$ means the sequence of actions where A follows α.

[5] In $\mathcal{A}$ [GelLif92], laws are interpreted using a transition function from states and actions to states, and this function together with the interpretation of the initial state form the interpretation of the domain description. Because of our restricted syntax, we could use this formulation, but decided to stick to the original semantics of $\mathcal{L}$, which at that time was chosen to accommodate future extensions involving triggers and actions with non-deterministic effects.

$F \in \sigma$; $\neg F$ *holds* in σ (F is *false* in σ) if $F \notin \sigma$. The *truth* of a propositional formula with respect to σ is defined as usual.

To better understand the rôle Ψ plays in interpreting domain descriptions let us define *models* of descriptions consisting entirely of causal laws. To this goal, we will attempt to carefully define effects of actions as determined by such a description D and our informal assumptions (a)-(c), and (e).

A fluent F is an *immediate effect* of (executing) A in σ if there is a causal law (1) in D whose preconditions hold in σ. Let us define the following sets:

$$E_A^+(\sigma) = \{F : F \text{ is an immediate effect of } A \text{ in } \sigma\}$$
$$E_A^-(\sigma) = \{F : \neg F \text{ is an immediate effect of } A \text{ in } \sigma\}$$
$$Res(A, \sigma) = \sigma \cup E_A^+(\sigma) \setminus E_A^-(\sigma)$$

The following definition captures the meaning of causal laws of D.

Definition 1 (Causal interpretation) *A causal interpretation Ψ satisfies causal laws of D if for any sequence $\alpha \cdot A$ from the language of D,*

$$\Psi(\alpha \cdot A) = \begin{cases} Res(A, \Psi(\alpha)) & if\ E_A^+(\Psi(\alpha)) \cap E_A^-(\Psi(\alpha)) = \emptyset \\ \\ undefined & otherwise \end{cases}$$

We say that Ψ is a causal model of D if it satisfies all the causal laws of D. $\square$

Causal models are uniquely determined by their initial values, i.e., for any two causal models Ψ_1 and Ψ_2 of a domain description D, if $\Psi_1([\,]) = \Psi_2([\,])$ then $\Psi_1 = \Psi_2$.

We are now ready to discuss how observations are interpreted in $\mathcal{L}$. Let D be an arbitrary domain description and let a causal interpretation Ψ be a causal model of D. To interpret the observations of D we first need to define the meaning of situation constants $S_0, S_1, S_2, \ldots$ from $\mathcal{S}$. To do that we consider a mapping Σ from $\mathcal{S}$ to sequences of actions from the language of D.

Definition 2 (Situation assignment) *A mapping Σ from $\mathcal{S}$ to sequences of actions is called a situation assignment of $\mathcal{S}$ if it satisfies the following* *properties:*

(i) $\Sigma(S_0) = [\,]$;
(ii) $\forall s_i \in \mathcal{S}.\ \Sigma(s_i)$ *is a prefix of* $\Sigma(S_N)$.

□

Intuitively, the first condition ensures that S_0 is indeed the initial situation. The second condition ensures that all the situation constants in $\mathcal{S}$ refer to actual situations – situations that have happened so far. Hence, Σ maps them to action sequences that are prefix of the action sequence that has happened until now, which is denoted by $\Sigma(S_N)$.

Definition 3 (Interpretation) *An interpretation M of $\mathcal{L}$ is a pair (Ψ, Σ), where Ψ is a causal model of D, Σ is a situation assignment of $\mathcal{S}$ and $\Sigma(S_N)$ belongs to the domain of Ψ. $\Sigma(S_N)$ will be called the actual path of M.* □

Now we can define the truth of facts of D with respect to an interpretation M. Facts which are not true in M will be called *false* in M.

Definition 4 *For any interpretation $M = (\Psi, \Sigma)$,*

(i) $(F$ **at** $S)$ *is true in M (or satisfied by M) if F is true in $\Psi(\Sigma(S))$;*
(ii) $(\alpha$ **occurs_at** $S)$ *is true in M if $\Sigma(S) \cdot \alpha$ is a prefix of the actual path of M;*
(iii) $(S_1$ **precedes** $S_2)$ *is true in M if $\Sigma(S_1)$ is a proper prefix of $\Sigma(S_2)$*

Truth of non-atomic facts in M is defined as usual. Of course, a set of facts is true in interpretation M if all its members are true in M. □

To complete the definition of model we need only to formalize assumption (d) on domain descriptions: "no actions occur except those needed to explain the facts in the domain description." (A similar assumption is used by Pinto and Reiter in [PinRei93,Pin94].) This is done by imposing a minimality condition on situation assignments of $\mathcal{S}$, which leads to the following definition.

Definition 5 (Model) *An interpretation $M = (\Psi, \Sigma)$ is a model of a domain description D in $\mathcal{L}$ if the following conditions are satisfied:*

(i) Ψ *is a causal model of D;*
(ii) *facts of D are true in M;*
(iii) *there is no other interpretation $N = (\Psi, \Sigma')$ such that N satisfies condition (ii) and $\Sigma'(S_N)$ is a subsequence*[6] *of $\Sigma(S_N)$.* □

[6] Recall that $\alpha = A_1, \ldots, A_m$ is a subsequence of $\beta = B_1, \ldots, B_n$ if there exists a strictly increasing sequence $i_1, \ldots, i_m$ of indices of β such that for all $j = 1, \ldots, m$,

It is important to note that we are only minimizing the actual action occurrences between the initial situation and the current situation. No such minimization is done about the future. This is instrumental for our formalization to avoid the trap of 'premature minimization' [Rei96], which we will further discuss in Section 11.5. The final definition is matter of course.

Definition 6 *A domain description D is said to be consistent if it has a model. A domain description D entails a fact ϕ (written $D \models \phi$) iff ϕ is true in all models of D.* □

2.2.1 Hypotheses and their entailment in $\mathcal{L}$

Planning from the current situation is necessary for an agent in a dynamic environment where exogenous events may occur; our agent may need to revise its plan and construct new ones starting from the current situation. In fact, once the agent realizes –halfway in the execution of the plan– that the changed conditions make it not effective any more, it no longer makes sense for the agent to make a plan from the initial situation. That is because the agent cannot wish away what has already happened. Hence, it needs to make a plan from the current situation it is in. To describe such plans and also to be able to do counterfactual reasoning, $\mathcal{L}$ has a construct called *hypotheses* which is of the following form:

$$(\neg)F \textbf{ after } [A_1, \ldots, A_n] \textbf{ at } S_i \tag{5}$$

Intuitively, the above hypothesis means that F holds (does not hold) after actions $A_1, \ldots, A_n$ are executed in S_i. Note that $A_1, \ldots, A_n$ may not be the sequence of actions that actually followed from S_i, thus making this a counterfactual statement. When S_i is S_N, we just write:

$$(\neg)F \textbf{ after } [A_1, \ldots, A_n] \tag{6}$$

Statements like (6) denote reasoning about plans starting from the current situation. Intuitively, the above hypothesis means that F holds (does not hold) after actions $A_1, \ldots, A_n$ are executed in S_N. Note that the above construct has a different meaning in the language $\mathcal{A}$. There, it means that F holds (does not hold) after actions $A_1, \ldots, A_n$ are executed in the initial situation S_0.

If $n = 0$, then we write the hypothesis as

$$\textbf{currently } (\neg)F \tag{7}$$

we have $A_j = B_{i_j}$.

Hypotheses are not part of a domain description, rather they are part of the query language.

We say a hypothesis of the form (5) is true in a model (Ψ, Σ) of a domain description D if $(\neg)F$ holds in

$$\Psi(\Sigma(S_i) \cdot A_1 \cdot \ldots \cdot A_n)$$

and, D entails a hypotheses if it is true in all models of D. Truth of hypotheses of the form 6 and 7 is defined accordingly.

We can now define a notion of a plan from the current situation.

Definition 7 ([BarGelPro97]) *Let D be a domain description and G be a set of fluent literals. A sequence α of actions is a plan from the current situation for achieving a goal G if $D \models f$ **after** α for every fluent literal $f \in G$.*

We now describe several examples that illustrate the expressibility of the language $\mathcal{L}$, as earlier mentioned in Section 1.2.

2.3 Examples illustrating expressibility of $\mathcal{L}$

2.3.1 Explaining observations

First let us formally show how the intuition discussed in Example 1 are captured by the semantics of $\mathcal{L}$. In that example an observation is explained by discovering missing action occurrences,

Example 2 (Inferring missing action occurrences) Consider domain description D_1 from Example 1. By fact

$$has(car) \textbf{ at } S_0$$

we have that $has(car) \in \Psi(\Sigma(S_0))$. By the causal law

$$steal(car) \textbf{ causes } \neg has(car)$$

we have $has(car) \notin \Psi([steal(car), \ldots, steal(car)])$ for any sequence of one or more actions $steal(car)$. It is easy to see that $\Sigma(S_N) = \Sigma(S_1) =$

$[steal(car)]$ is the minimal sequence satisfying the facts in the domain description. Thus, $\Sigma(S_0) \cdot steal(car)$ is a prefix of $\Sigma(S_N)$. Therefore, D_1 entails $steal(car)$ **occurs_at** S_0.

Let us now discuss another example where observations are explained by discovering both missing action occurrences and new fluent facts (that were not known before). Suppose our agent has the following domain description consisting of some laws and some observations.

$$\left.\begin{array}{l} shoot\ \textbf{causes}\ \ \neg loaded \\ shoot\ \textbf{causes}\ \ \neg alive\ \textbf{if}\ \ loaded \\ unload\ \textbf{causes}\ \ \neg loaded \\ alive\ \textbf{at}\ \ S_0 \\ \neg alive\ \textbf{at}\ \ S_1 \\ S_0\ \textbf{precedes}\ \ S_1 \end{array}\right\} = D_2$$

Using the entailment relation of $\mathcal{L}$ we can make the following conclusions that illustrate some of the features of $\mathcal{L}$.

- **Missing action Occurrences**

  $D_2 \models shoot$ **occurs_at** S_0

  Thus, while explaining the observation that $alive$ is false in S_1 we conclude that the action $shoot$ must have occured at S_0.

- **Discovering new fluent facts about the past**

  $D_2 \models loaded$ **at** S_0

  Thus while explaining the observation that $alive$ is false in S_1 we conclude that $loaded$ must have been true at S_0.

- **Discovering new fluent facts about the current situation**

  $D_2 \models$ **currently** $\neg loaded$

  Explaining observations not only allow us to discover new facts about the past but also to make new conclusions about the current situation. In this case, we are able to conclude that the gun is not loaded in the current situation.

- **Counterfactual reasoning**

  $D_2 \models alive$ **after** $unload, shoot$ **at** S_0

Earlier we concluded that *shoot* occured at S_0. If we were now to reason about what would have happened if the gun was unloaded at S_0 and then *shoot* had happened, we would conclude that *alive* would be true in the resultant situation. This is the type of counterfactual reasoning that is possible in $\mathcal{L}$.

Since this reasoning is counter to what really happened, this is a form of counterfactual reasoning, and we can do such reasoning in $\mathcal{L}$.

2.3.2 Planning from the current situation

In the following example we use $\mathcal{L}$ to show the planning and re-planning done by an agent in a dynamic world. This is very important from the point of view of an agent architecture where the agent continuously observes the world and adds the observation to its domain description, makes a new plan –from the current situation– to reach its goal and executes part of the plan, records its actions in its domain description, before going back to observe again. The planning in each cycle needs to be done from the situation the agent is in at that moment (referred to as the current situation) rather than the initial situation. In fact, the agent can not wish away what has happened –be it performed by the itself or by the environment– since the initial situation[7].

Example 3 Suppose our agent has the following initial knowledge, in terms of the various actions and their effects and the values of fluents in the initial situation.

$$\left.\begin{array}{l}
load\ \textbf{causes}\ \ loaded\\
unload\ \textbf{causes}\ \ \neg loaded\\
set_target\ \textbf{causes}\ \ target_set\\
shoot\ \textbf{causes}\ \ target_hit\ \textbf{if}\ \ loaded, target_set\\
loaded\ \textbf{at}\ \ S_0\\
\neg target_set\ \textbf{at}\ \ S_0
\end{array}\right\} = D_{\text{hyp0}}$$

Given the agent's goal to have the *target_hit*, it develops the plan $[set\_target, shoot]$ by checking that indeed

$$D_{\text{hyp0}} \models target\_hit\ \textbf{after}\ \ [set\_target, shoot]$$

[7] Note that, given the complexity of planning [Erol et al. 95] such an architecture may not be useful for agents such as mobile robots that need to react in real time –a reactive architecture would be more appropriate in that case– but it may still be useful for Internet agents where there is more time to plan.

holds. Then, the agent proceeds to execute the first action: *set_target*, and adds *set_target* **occurs_at** S_1 and S_0 **precedes** S_1 to the domain description. (Let us call the updated domain description D_{hyp1}.) At this point, the agent's domain description, D_{hyp1}, entails:

currently *loaded*

currently *target_set*

Before executing the next action, however, it observes that the gun is no longer loaded, and updates its domain description by adding $\neg loaded$ **at** S_2 and S_1 **precedes** S_2. (Let us call the updated domain description D_{hyp2}.) It concludes that:

$D_{\text{hyp2}} \models$ **currently** *target_set*

$D_{\text{hyp2}} \models$ **currently** $\neg loaded$

$D_{\text{hyp2}} \models [set\_target, unload]$ **occurs_at** $S_0 \vee [unload, set\_target]$ **occurs_at** S_0

The agent then notices that it can no longer continue with its original plan, since

$D_{\text{hyp2}} \not\models target\_hit$ **after** *shoot*

Hence, the agent proceeds to re-planning from the current situation S_N, rather than from the initial situation, S_0. While the initial plan starting from S_0 is still $[set\_target, shoot]$, the new plan is $[load, shoot]$ since

$D_{\text{hyp2}} \models target\_hit$ **after** $[load, shoot]$

Assuming that no other untoward incident takes place, the agent can proceed with its current plan to reach the goal. □

3 Overview of Nested Circumscription

Nested Abnormality Theories (NATs) is a novel circumscription [McC86,Lif94] technique introduced by Lifschitz [Lif95]. With NATs it is possible to circumscribe several predicates each with respect to only parts of the theory of interest, as opposed to previous techniques such as parallelized and circumscription theories where the circumscription must be done with respect to all of the axioms in the underlying theory. Furthermore, all the complications arising from the interaction of multiple circumscription axioms in a theory are avoided in NATs with the introduction of blocks. A *block* is characterized by a set of axioms $A_1, \ldots, A_n$—possibly containing the abnormality predicate Ab –which 'describe' a set of predicate/function constants $C_1, \ldots, C_m$. The notation for such a theory is

$$\{C_1, \ldots, C_m \ : \ A_1, \ldots, A_n\} \tag{8}$$

where each A_i may itself be a block of form (8). The 'description' of $C_1, \ldots, C_m$ by a block may depend on other descriptions in embedded blocks.

Interference between circumscription in different blocks is prevented by replacing a predicate Ab with an existentially quantified variable. Lifschitz's idea is to make Ab 'local' to the block where it is used, since abnormality predicates play only an auxiliary role, i.e. the interesting consequences of the theory are those which do not contain Ab. The next section contains the formal definitions of this concepts.

3.1 Syntax and semantics of NATs

The following definitions are from [Lif95]. Let L be a second order language which does not include Ab. For every natural number k, let L_k be the language obtained by adding the k-ary predicate constant Ab to L. $\{C_1, ..., C_m \ : \ A_1, ..., A_n\}$ is a *block* if each $C_1, \ldots, C_m$ is a predicate or a function constant of L, and each $A_1, \ldots, A_n$ is a formula of L_k or a block.

A *Nested Abnormality Theory* is a set of blocks. The semantics of NATs is characterized by a mapping φ from blocks into sentences of L. If A is a formula of language L_k, φA stands for the universal closure of A, otherwise

$$\varphi\{C_1, ..., C_m \ : \ A_1, ..., A_n\} \ = \ (\exists ab)F(ab)$$

where

$$F(Ab) = \mathrm{CIRC}[\varphi A_1 \wedge ... \wedge \varphi A_n; Ab; C_1, ..., C_m]$$

Recall that $\mathrm{CIRC}[T; P; Q]$, means circumscription of the theory T, by minimizing the predicates in P, and varying the objects in Q.

For any NAT T, φT stands for $\{\varphi A | A \in T\}$. A *model* of T is a model of φT in the sense of classical logic. A *consequence* of T is a sentence ϕ of language

L that is true in all models of T. In this paper, as suggested in [Lif95], we use the abbreviation

$$\{C_1, \ldots, C_m,\ \min P : A_1, \ldots, A_n\}$$

to denote blocks of the form

$$\{C_1, \ldots, C_m, P : P(x) \supset Ab(x), A_1, \ldots, A_n\}$$

As the notation suggests, this type of block is used when it is necessary to circumscribe a particular predicate P in a block. In [Lif95] it is shown that

$$\varphi\{C_1, \ldots, C_m,\ \min P : A_1, \ldots, A_n\}$$

is equivalent to the formula

$$\mathrm{CIRC}[A_1 \wedge \ldots \wedge A_n; P; C_1, \ldots, C_m]$$

when each A_i is a sentence.

4 From domain descriptions in $\mathcal{L}$ to NATs

We are now ready to present our translation from $\mathcal{L}$ to NATs. We will start by formally describing some rather typical relations on sequences which we will use later in our proofs.

4.1 Relations on sequences

The relations we are interested in are prefix, subsequence and concatenate. These relations are captured by the NAT blocks $B_{\text{prefix\_eq}}$, $B_{\text{subsequence}}$, and $B_{\text{concatenate}}$, respectively, which are presented in a later section. Given the formal description of these relations, it is important to ensure that all and only the intended instances of these relations are included in models of the theory described in the following sections. Let us start by defining the prefix relation which will be denoted by $\preceq$:

$$A_n \circ \ldots \circ A_1 \preceq B_m \circ \ldots \circ B_1 \ \Leftrightarrow\ \forall i, i \leq n.\ A_i = B_i$$

where above and in the rest of this section by "=" we mean syntactic identity. The second relation we need to describe is subsequence:

$$A_n \circ \ldots \circ A_1 \ll B_m \circ \ldots \circ B_1 \;\Leftrightarrow\; \exists \mu \; \forall i, A_i = B_{\mu(i)} \;\wedge [i \leq j \Rightarrow \mu(i) \preceq \mu(j)]$$

Relation $\ll$ formalizes the notion of *subsequence:* if $\alpha \ll \beta$ then intuitively β contains all the elements of α, in the same order, but it may contain more elements. This relation is defined by block $B_{\text{subsequence}}$ in the theory.

The third relation needed for dealing with sequences is concatenation. We will *concatenate* sequences of actions in reverse, i.e. $\alpha \cdot \beta = \beta\alpha$:

$$(A_n \circ A_{n-1} \circ \ldots \circ A_1) \cdot (B_m \circ B_{m-1} \circ \ldots \circ B_1) = B_m \circ B_{m-1} \circ \ldots \circ B_1 \circ A_n \circ A_{n-1} \circ \ldots \circ A_1$$

Block $B_{\text{concatenate}}$ defines this relation.

4.2 The target language

The language of the theory T is many-sorted and borrows many notations from the standard situation calculus. The sorts are: *actions*, *fluents*, *situations*, and *sequences*. The variables for the first three sorts will be denoted by possibly indexed letters a, f and s, respectively, unless otherwise stated. Variables for sequences will be denoted by possibly indexed Greek letters α, β and γ. The language includes the elements of $\mathcal{A}$ as action constants, of $\mathcal{F}$ as fluent constants, and $\mathcal{S}$ as situation constants. It also includes the sequence constant ϵ which stands for the empty sequence.

Sequences of actions are defined by means of a function $\circ$ from sequences and actions to sequences. Sequences of actions are then constructed by multiple application of the function $\circ$. We will use infix notation for $\circ$ –given its similarity to the *Res* function widely used in situation calculus– and ignore parenthesis without ambiguity. The following is an example of a sequence:

$$A_n \circ A_{n-1} \circ \ldots \circ A_1 \circ \epsilon$$

In addition to $\circ$, we have the following function and predicate constants:

- Sit_map: a function that maps situations into sequences;
- $Prefix\_eq(\alpha, \beta)$: both arguments of sort situation, this predicate captures relation $\preceq$ on sequences, i.e. $\alpha \preceq \beta$;
- $Subsequence(\alpha, \beta)$: captures relation $\ll$, i.e. $\alpha \ll \beta$;

- $Holds(f, \alpha)$: with sorts fluent and sequence, meaning that f is true in the state resulting from executing α in the initial situation;
- $Concatenate(\alpha, \beta, \gamma)$: captures relation $\cdot$ on sequences, i.e. $\gamma = \alpha \cdot \beta$;
- $Causes^{+(-)}(a, f, \alpha)$: with sorts action, fluent and sequence, meaning that executing a in the state resulting from executing α in the initial situation makes f true(false);

4.3 Framework axioms

To yield the expected results, the theory includes a set of extra axioms which represent the domain-closure assumption of $\mathcal{L}$ theories, and in particular assumption *b)*: "there are no actions except those from the language of the domain description." Also unique-name assumptions for actions, fluents and situations are in the theory.

$$(\forall a).a = A_1 \vee \ldots \vee a = A_l$$

$$(\forall f).f = F_1 \vee \ldots \vee f = F_m$$

$$(\forall s).s = S_0 \vee \ldots \vee s = S_N$$

$$UNA[actions],\ UNA[fluents],\ UNA[situations]$$

$$(\forall a, \alpha).\epsilon \neq a \circ \alpha$$

$$(\forall a, b, \alpha, \beta).a \circ \alpha = b \circ \beta \supset a = b\ \wedge \alpha = \beta$$

where $UNA(sort)$ is the standard set of inequalities between each distinct pair of constants from $sort$, e.g. $UNA[situations]$ stands for $S_0 \neq S_1,\ S_1 \neq S_2 \ldots$ etc. In the rest of the paper we will refer to the above sentences as *Framework axioms*.

4.4 Translation of Facts

Atomic facts are translated as follows:

$$\begin{aligned}
(\neg)F\ \textbf{at}\ \ S \quad &\Rightarrow (\neg)Holds(F, Sit_map(S)) \\
\alpha\ \textbf{occurs_at}\ \ S \ &\Rightarrow (\exists\beta).Concatenate(Sit_map(S), \alpha, \beta) \\
&\quad\ \wedge Prefix_eq(\beta, Sit_map(S_N)) \\
S_1\ \textbf{precedes}\ \ S_2 &\Rightarrow Prefix_eq(Sit_map(S_1), Sit_map(S_2)) \wedge \\
&\quad\ \neg Prefix_eq(Sit_map(S_2), Sit_map(S_1))
\end{aligned}$$

Non-atomic facts are translated in the obvious way. For any fact ϕ, we will use $\tau(\phi)$ to denote its translation. For the collection of facts D_f in a domain description D, the set of formulae $\tau(\phi)$ for each $\phi \in D_f$ will be denoted by $\tau(D_f)$.

4.5 The resulting NAT

We now present our translation of domain descriptions in $\mathcal{L}$ into NATs (universal quantification with the highest scope is implicit on free variables):

$$
\begin{array}{l}
T(D) = \\
\quad \{\ Sit\_map\ : \\
(a)\ Sit\_map(S_0) = \epsilon \\
(b)\ Prefix\_eq(Sit\_map(s), Sit\_map(S_N)) \\
(c)\ Subsequence(\alpha, Sit\_map(S_N)) \supset Ab(\alpha) \\
\qquad \tau(D_f),\ SC(D_l) \\
\qquad B_{\text{prefix\_eq}},\ B_{\text{subsequence}},\ B_{\text{concatenate}} \\
\qquad \text{Framework axioms} \\
\quad \}
\end{array}
$$

where:

$SC(D_l) =$

$$\left\{\begin{array}{l}
\neg Causes^{+}(a,f,\alpha) \;\wedge \neg Causes^{-}(a,f,\alpha) \supset [Holds(f,\alpha) \equiv Holds(f,a \circ \alpha)] \\
Causes^{+}(a,f,\alpha) \supset Holds(f,a \circ \alpha) \\
Causes^{-}(a,f,\alpha) \supset \neg Holds(f,a \circ \alpha) \\
\{ \text{ min } Causes^{+} \;: \\
\quad H(P_1,\alpha) \;\wedge \ldots \;\wedge H(P_n,\alpha) \supset Causes^{+}(A,F,\alpha) \\
\qquad\qquad\qquad\qquad\qquad (\text{for each } A \textbf{ causes } \; F \textbf{ if } \; P_1,\ldots,P_n \in D) \\
\} \\
\{ \text{ min } Causes^{-} \;: \\
\quad H(P_1,\alpha) \;\wedge \ldots \;\wedge H(P_m,\alpha) \supset Causes^{-}(A,F,\alpha) \\
\qquad\qquad\qquad\qquad\qquad (\text{for each } A \textbf{ causes } \; \neg F \textbf{ if } \; P_1,\ldots,P_m \in D) \\
\}
\end{array}\right.$$

In the above translation, for a positive fluent literal F, $H(F,\alpha)$ denotes $Holds(F,\alpha)$; while for a negative fluent literal $\neg G$, $H(\neg G,\alpha)$ denotes $\neg Holds(G,\alpha)$. Predicate $Causes^{+(-)}$ in the form used above comes originally from Lifschitz [Lif95].

$B_{\text{prefix\_eq}} =$

$\{\text{min } Prefix\_eq \;:$

$$Prefix\_eq(\alpha,\alpha)$$

$$Prefix\_eq(\alpha,\beta) \supset Prefix\_eq(\alpha,a \circ \beta)$$

$\}$

$B_{\text{subsequence}} =$

$\{\text{min } Subsequence \;:$

$$Subsequence(\alpha,\alpha)$$

$$Subsequence(\alpha,\beta) \supset Subsequence(\alpha,a \circ \beta)$$

$$Subsequence(\alpha,\beta) \supset Subsequence(a \circ \alpha,a \circ \beta)$$

$\}$

$B_{\text{concatenate}} =$

$\{\min \mathit{Concatenate}$:

$$Concatenate(\alpha, \epsilon, \alpha)$$

$$Concatenate(\alpha, \beta, \gamma) \supset Concatenate(\alpha, a \circ \beta, a \circ \gamma)$$

$\}$

Let us now compare the axioms of $T(D)$ with the definitions in the section on semantics.

- The blocks $B_{\text{prefix\_eq}}$, $B_{\text{subsequence}}$ and $B_{\text{concatenate}}$ define the predicates $Prefix\_eq$, *Subsequence* and *Concatenate* respectively. The axioms inside each of these blocks are definite clauses and in the appendix we use results from [MarSub89,Lif94] to show that indeed they are a correct representation of their corresponding relations.
- $SC(D_l)$ is the part of $T(D)$ most similar to standard situation calculus and captures the meaning of causal laws. Models of $SC(D_l)$ correspond to the causal models of D. (We show this in Section 7.) Inside $SC(D_l)$ we have two blocks that minimize the predicates $Causes^+$ and $Causes^-$. Intuitively, they encode the informal assumption (c) in Section 2.1, saying that there are no effects of actions except those specified by the causal laws. The first axiom of $SC(D_l)$ encodes the frame axiom –the assumption (a) in Section 2.1 as a first order statement while the second and the third axioms in $SC(D_l)$, encode the effect of actions on fluents.
- The axioms (a) and (b) encode Definition 2 in a straightforward manner.
- The axioms in $\tau(D_f)$ encode Definition 4 in a straightforward way.
- Finally, the minimality in condition (iii) of Definition 5 (which corresponds to the assumption (d) of Section 2.1) is encoded by axiom (c) plus circumscription of *Ab*. Intuitively, here we would like to minimize the *value of the term* $Sit\_map(S_N)$ is accomplished through axiom (c) by minimizing *Ab* while varying Sit_map. A discussion on this axiom and the technique it exemplifies follows in Section 5.
- Before moving on, let us quickly review how the various informal assumptions described in Section 2.1 are encoded in our NAT. We already discussed the assumptions (a), (c), and (d); assumption (b) is captured through the 'Framework axioms.' Verifying that assumption (e) is captured by the translation is slightly more involved: suppose that there is a situation S s.t. A **occurs_at** S and B **occurs_at** S, are in D, i.e. A and B occurred concurrently at situation S. As a result, $Concatenate(Sit\_map(S), A, \alpha)$, $Prefix\_eq(\alpha, Sit\_map(S_N))$, $Concatenate(Sit\_map(S), B, \beta)$ and $Prefix\_eq(\beta, Sit\_map(S_N))$ hold for some α and β. It is not hard to prove that this implies $A = B$.

4.6 Translation of hypotheses

Even though hypotheses do not appear in domain descriptions, we define a translation so that we can check whether they are entailed by the NAT theory. For hypotheses of the form (5), the translation, $\tau((\neg)F$ **after** $[A_1,\ldots,A_m]$ **at** $S_i)$, is defined as follows:

$$(\exists\beta).Concatenate(Sit\_map(S_i), A_n \circ \ldots \circ A_1 \circ \epsilon, \beta) \supset (\neg)Holds(F,\beta) \quad (9)$$

5 Value minimization of functions

In [BarGabPro96b], we introduced the concept of value minimizing a function. That is, forcing a function to map the elements of its domain onto minimal elements of its range, where the minimality criterion is with respect to an arbitrary partial order defined on the range. This is completely different from all earlier formulations of circumscription where function and predicates (or formulas) were allowed to vary but only the latter could be minimized.

As we mentioned in the introduction, our formalization of narrative applies this technique. In order to capture condition (iii) of Definition 5, we minimize the value of function Sit_map on term S_N, w. r. t. the partial order on sequences defined by predicate $Subsequence$. Therefore, this is an instance of 'term minimization.' Intuitively, this forces Sit_map to map S_N onto the 'minimal' possible sequence such that the facts and all the axioms remain satisfied.

By using the syntactic definition of value minimization with respect our application, we can express the minimization of $Sit\_map(S_N)$ as follows: Let T_{subseq} stand for $T(D)$ minus axiom (c), and let $T_{\text{subseq}}(\alpha)$ denote $T_{\text{subseq}} \wedge Sit\_map(S_N) = \alpha$, for an arbitrary sequence α. Term $Sit\_map(S_N)$ is minimized with respect to $Subsequence$ by postulating:

$$\begin{aligned}&(\forall\alpha).T_{\text{subseq}}(\alpha) \supset \\ &\quad (\forall\alpha').(T_{\text{subseq}}(\alpha') \wedge Subsequence(\alpha',\alpha)) \supset Subsequence(\alpha,\alpha')\end{aligned} \quad (10)$$

In [BarGabPro96b] we show that value minimization of a function ϕ in a theory T with respect to an ordering $\mathcal{R}$ (which itself is defined in a block inside T) while varying the predicate/function constants Z, can be achieved by the following NAT:

$$\{\phi, Z \ : \quad (\forall x, y).\ \mathcal{R}(y, \phi(x)) \supset Ab(x, y) \quad T \quad \}$$

Moreover, to minimize ϕ at only one term σ, the corresponding NAT characterization becomes:

$$\{\phi, Z \ : \quad (\forall y).\ \mathcal{R}(y, \phi(\sigma)) \supset Ab(y) \quad T \quad \}$$

By applying this result and observing that in our case ϕ is the function *Sit_map*, $\mathcal{R}$ is the predicate *Subsequence*, and σ is S_N, we obtain the following NAT characterization of the formula (10).

$$\{Sit\_map \ : \quad (\forall y).\ Subsequence(y, Sit\_map(S_N)) \supset Ab(y) \quad T_{\mathrm{subseq}} \quad \}$$

This is exactly our translation $T(D)$.

6 NAT characterization of some simple domain descriptions

In this section we illustrate our NAT characterization of domain description in $\mathcal{L}$ through some examples.

Example 4 Consider again the domain description D_1 from Example 1, the translation corresponds to

$$\tau(D_f) = \begin{cases} Holds(has(car), Sit\_map(S_0)) \\ \neg Holds(has(car), Sit\_map(S_1)) \\ Prefix\_eq(Sit\_map(S_0), Sit\_map(S_1)) \wedge \\ \qquad\qquad \neg Prefix\_eq(Sit\_map(S_0), Sit\_map(S_1)) \end{cases}$$

and the sub-block of $SC(D_{1l})$ defining $Causes^-$:

$\{\min Causes^- :$

$\qquad Causes^-(steal(car), has(car), \alpha)$

$\}$

which is equivalent to

$$(\forall a, f, \alpha).Causes^-(a, f, \alpha) \equiv a = steal(car) \ \wedge f = has(car) \qquad (11)$$

The sub-block defining $Causes^+$ is empty, i.e. $Causes^+(a, f, \alpha)$ is false for all a, f, α. From this, (11), the axiom $Causes^-(a, f, \alpha) \supset \neg Holds(f, a \circ \alpha)$, and the fact that $steal(car)$ is the only action in our language; it follows that for every sequence β,

$\neg Holds(has(car), \beta)$ iff $\beta = steal(car) \circ \alpha$, for some α.

From $\neg Holds(has\_car, Sit\_map(S_1))$ we have $Sit\_map(S_1) = steal(car) \circ \alpha$ for some sequence α. By value minimization and axiom (b),

$$Sit\_map(S_N) = Sit\_map(S_1) = steal(car) \circ \epsilon$$

and from this and the blocks $B_{\text{concatenate}}$ and $B_{\text{prefix\_eq}}$ we get that

$$Concatenate(Sit\_map(S_0), steal(car) \circ \epsilon, steal(car) \circ \epsilon)$$

and

$$Prefix\_eq(steal(car) \circ \epsilon, Sit\_map(S_N))$$

Hence, $\tau(steal(car)$ **occurs_at** $S_0)$ holds. □

We now consider a slightly different example where by using the observations we can 'abduce' new truth values of fluents.

Example 5 *(abduction of fluent-values)* Consider another scenario where there is a gun that causes Fred to die if fired at him when loaded. Suppose we know that Fred had been observed to be alive at some moment of time and to be dead at a later moment. Description D_2 below captures this scenario:

$$D_2 = \begin{cases} shoot \ \mathbf{causes} \ \neg alive \ \mathbf{if} \ loaded \\ alive \ \mathbf{at} \ S_0 \\ \neg alive \ \mathbf{at} \ S_1 \\ S_0 \ \mathbf{precedes} \ S_1 \end{cases}$$

Let us consider the corresponding NAT, $T(D_2)$, starting from the inner blocks. Since the sub-block defining $Causes^+$ is empty we have

$$(\forall a, f, \alpha).\neg Causes^+(a, f, \alpha) \tag{12}$$

The sub-block defining $Causes^-$ contains only the axiom

$$Holds(loaded, \alpha) \supset Causes^-(shoot, alive, \alpha)$$

By minimization of $Causes^-$ in this block we obtain

$$(\forall a, f, \alpha).Causes^-(a, f, \alpha) \equiv (a = shoot \ \wedge f = alive \ \wedge Holds(loaded, \alpha)) \tag{13}$$

Now, $\tau(D_f)$ includes $Holds(alive, Sit\_map(S_0))$ and $\neg Holds(alive, Sit\_map(S_1))$. Let us reason about the value of the fluent *loaded* in the initial state. There are two cases:

a) *loaded* was initially false $\neg Holds(loaded, \epsilon)$ and since there is no action that affects this fluent, i.e. for all a, α, $\neg Causes^+(a, loaded, \alpha)$ and $\neg Causes^-(a, loaded, \alpha)$, we have that

$$(\forall \alpha)\neg Holds(loaded, \alpha) \tag{14}$$

From this and (13) it follows that

$$(\forall a, f, \alpha)\neg Causes^-(a, f, \alpha) \tag{15}$$

From the fact $\tau(alive$ **at** $S_0)$ and axiom (a) we obtain $Holds(alive, \epsilon)$. From this, (12), (15), and axiom

$$\neg Causes^+(a, f, \alpha) \wedge \neg Causes^-(a, f, \alpha) \supset [Holds(f, \alpha) \equiv Holds(f, a \circ \alpha)]$$

we conclude that $(\forall \alpha) Holds(alive, \alpha)$. However, from fact $\neg alive$ **at** S_1 we have $\neg Holds(alive, Sit\_map(S_1))$. This is a contradiction, therefore the initial assumption of loaded being initially false is ruled out.

b) The other possibility is that *loaded* was initially true, i.e. $Holds(loaded, \epsilon)$ holds. From (13) this implies that $Causes^-(shoot, alive, \epsilon)$ and by axiom

$$Causes^-(a, f, \alpha) \supset \neg Holds(f, a \circ \alpha)$$

we obtain $\neg Holds(alive, shoot \circ \epsilon)$. Clearly, an interpretation of Sit_map such that S_1 and S_N are mapped into sequence $shoot \circ \epsilon$ satisfies $\tau(D_f)$ and minimization of $Sit\_map(S_N)$. It remains easy to prove that this interpretation is the only model of the theory and that it entails $\tau(shoot$ **occurs_at** $S_0)$. □

7 Correctness of the NAT formalization

In this section we formulate the results about the correctness of our NAT formalization of domain descriptions in $\mathcal{L}$. In the process, for a domain description D, we show the equivalence between models of $SC(D_l)$ and causal models of D. This illustrates the modularity of our NAT formulation; later, (in Section 9) we discuss how this allows us to easily extend the language with additional features, such as constraints, by only replacing $SC(D_l)$. The proofs of the lemmata and the theorem of this section are given in the appendix.

Throughout the paper, we will use α, β to denote sequences both in the context of the semantics of $\mathcal{L}$ and its NAT formalization. For instance, if α stands for a sequence with actions $A_1, \ldots, A_m$, we may write $Holds(F, \alpha) \Leftrightarrow F \in \Psi(\alpha)$ meaning $Holds(F, A_m \circ \ldots \circ A_1 \circ \epsilon) \Leftrightarrow F \in \Psi([A_1, \ldots, A_m])$.

The equivalence of SC and Ψ is first proved under the assumption that Ψ if defined for every sequence of actions. [8]. This restriction is removed in Section 8 when concurrent actions are considered.

Proposition 1 *(Causal Equivalence)*

[8] This conditions is satisfied by disallowing contradictory causal laws; two axioms A **causes** F **if** $P_1, \ldots, P_n$ and A **causes** $\neg F$ **if** $Q_1, \ldots, Q_m$ are said to be contradictory if $\{P_1, \ldots, P_n\} \cap \{\overline{Q_1}, \ldots, \overline{Q_m}\} = \emptyset$).

Part 1: *For every causal model Ψ of D_l there exists a model $\mathcal{M}_{\rm SC}$ of $SC(D_l) \cup$ Framework Axioms such that for all F,α:*

$$F \in \Psi(\alpha) \ \Leftrightarrow \ \mathcal{M}_{\rm SC} \models Holds(F, \alpha). \tag{16}$$

Part2: *For every model $\mathcal{M}_{\rm SC}$ of $SC(D_l)$ there exists a causal model Ψ of D_l such that (16) holds.* □

Proposition 2 *(Equivalence of Models) For any domain description D, if the interpretation (Ψ, Σ) is a model of D then there exists a model $\mathcal{M}$ of $T(D)$ such that for every fact ϕ in the language of D:*

$$(\Psi, \Sigma) \models_{\mathcal{L}} \phi \ \Leftrightarrow \ \mathcal{M} \models \tau(\phi) \tag{17}$$

and if $\mathcal{M}$ is a model of $T(D)$ then there exists a model (Ψ, Σ) of D such that (17) holds.

□

Proposition 3 *(Equivalence) For any domain description D and fact ϕ in the language of D:*

$$D \models_{\mathcal{L}} \phi \ \Leftrightarrow \ T(D) \models \tau(\phi)$$

□

The following proposition shows the correctness of our theory with respect to hypotheses.

Proposition 4 *For any domain description D and hypothesis ϕ in the language of D:*

$$D \models_{\mathcal{L}} \phi \ \Leftrightarrow \ T(D) \models \tau(\phi)$$

□

8 Narratives with Concurrent Actions

In this section we consider reasoning about concurrent actions together with narratives.This means that the informal assumption (e) of Section 2.1 is weakened to allow concurrent contemporaneous actions. Still, we do not allow non-contemporaneous overlapping actions. The semantics of $\mathcal{L}$ is extended to allow concurrent actions. We will only consider concurrent actions consisting of the the simultaneous execution of a finite number of basic actions with non-contradictory effects. For a more sophisticated treatment of concurrent actions in a non-narrative setting see e.g., [BarGel93] and [LinSho92]. We will refer to the language with the extended semantics as $\mathcal{L}_c$.

8.1 Extended semantics

Concurrent actions will be characterized by finite non-empty sets of basic actions. Let us consider the changes that are required in the semantics of $\mathcal{L}$ to account for concurrent actions. First, we need to redefine what the effect of an action is since we now have a more general notion of action. Since a singleton concurrent action $\{A\}$ is basically the same as basic action A, from now on by action we will mean concurrent action.

We say that a fluent F is an *effect* of an action A in a state σ if there is an action $B \in A$ such that F is an immediate effect of B in σ. Furthermore, we replace the definitions of sets $E_A^+(\sigma)$ and $E_A^-(\sigma)$ with the following:

$$E_A^+(\sigma) = \{F : F \text{ is an effect of } A \text{ in } \sigma\}$$

$$E_A^-(\sigma) = \{F : \neg F \text{ is an effect of } A \text{ in } \sigma\}$$

We will also need the following notions. Let $\alpha = [A_1, \ldots, A_k]$ and $\beta = [B_1, \ldots, B_k]$. We say β is *embedded* in α (written as $\beta \subseteq \alpha$) if $B_i \subseteq A_i$ for each i. Intuitively, $\beta \subseteq \alpha$ means that we can obtain β from α by removing some basic actions from each (concurrent) action in the action sequence α.

Furthermore, we say that β is an *embedded-subsequence* of γ if there is a subsequence α of γ such that β is embedded in α. The definitions of causal interpretation, situation assignment, interpretation, and consistent domain description remain the same. We only need to modify Definitions 4 and 5.

Definition 8 *For any interpretation $M = (\Psi, \Sigma)$.*

(i) (F **at** S) *is true in M (or satisfied by M) if F is true in $\Psi(\Sigma(S))$;*
(ii) (α **occurs_at** S) *is true in M if there is β such that $\Sigma(S) \cdot \beta$ is a prefix of the actual path of M and $\alpha \subseteq \beta$.*
(iii) (S_1 **precedes** S_2) *is true in M if $\Sigma(S_1)$ is a proper prefix of $\Sigma(S_2)$*

□

The above definition differs from its counterpart Definition 4 on point (ii), where truth of action occurrences is defined for concurrent actions.

Definition 9 *An interpretation* $M = (\Psi, \Sigma)$ *will be called a model of a domain description* D *in* $\mathcal{L}$ *if the following conditions are satisfied:*

(i) Ψ *is a causal model of* D*;*
(ii) facts of D *are true in* M*;*
(iii) There is no other interpretation $N = (\Psi, \Sigma')$ *such that* N *satisfies condition* (ii) *and* $\Sigma'(S_N)$ *is an embedded-subsequence of* $\Sigma(S_N)$*.*

□

This definition differs from Definition 5 on point (iii) where the minimality condition is now in terms of the relation embedded-subsequence.

8.2 Translation into NATs

Let us now modify the NAT translation of Section 4 to accommodate the semantics of $\mathcal{L}_c$.

In our translation, we will use sets to represent concurrent actions. We will not, however, axiomatize sets and their operations but assume that their standard interpretation[9]. For concurrent actions we introduce a new sort called *c-actions*. Function $\circ$ will now be of sorts *c-action* $\times$ *sequence* $\mapsto$ *sequence*. Some new predicates will be introduced below.

We need to change the sub-theory $SC(D_l)$ to capture the effects of concurrent actions. The kind of concurrent actions considered here are such that they *inherit* the effects of their constituent actions. Thus, we include two blocks, one defining predicate $Inherits^+(a, f, \alpha)$ and one defining predicate $Inherits^-(a, f, \alpha)$, both with arguments of the sorts, c-action, fluent and sequence respectively:

$$\{\min\ Inherits^+ :$$

$$b \in a\ \wedge Causes^+(b, f, \alpha) \supset Inherits^+(a, f, \alpha)$$

$$\}$$

The block for $Inherits^-$ is defined similarly. The intuitive meaning of

[9] For a formulation of $\mathcal{L}_c$ in NAT without set theory please see
http://cs/utep/edu/chitta/papers/L-NAT-ext.ps

$Inherits^{+(-)}(a, f, \alpha)$ is that concurrent action a inherits the effect f $(\neg f)$, from its sub-actions, in the situation after α is executed.

We also add the following block defining the predicate $Undef$ characterizing action sequences where the causal interpretation Ψ is undefined.

$$
\begin{aligned}
&\{\min\ Undef\ : \\
&\qquad Inherits^{+}(a, f, \alpha)\ \wedge Inherits^{-}(a, f, \alpha) \supset Undef(a \circ \alpha) \\
&\qquad Undef(\alpha) \supset Undef(a \circ \alpha) \\
&\}
\end{aligned}
$$

Next, we replace the first three axioms of $SC(D_l)$ with the following axioms:

$$\neg Inherits^{+}(a, f, \alpha)\ \wedge \neg Inherits^{-}(a, f, \alpha) \supset [Holds(f, \alpha) \equiv Holds(f, a \circ \alpha)]$$

$$\neg Undef(a \circ \alpha)\ \wedge Inherits^{+}(a, f, \alpha) \supset Holds(f, a \circ \alpha)$$

$$\neg Undef(a \circ \alpha)\ \wedge Inherits^{-}(a, f, \alpha) \supset \neg Holds(f, a \circ \alpha)$$

Next, we add a new block, B_{embedded}, at the same level of nesting as blocks defining $B_{\text{prefix\_eq}}$ etc., which defines *Embedded,* with two arguments of sort sequence:

$$
\begin{aligned}
&B_{\text{embedded}} = \\
&\{\min\ Embedded\ : \\
&\qquad Embedded(\epsilon, \epsilon) \\
&\qquad [a \in b \supset a \in c] \supset Embedded(b \circ \epsilon, c \circ \epsilon) \\
&\qquad [Embedded(b \circ \epsilon, a \circ \epsilon)\ \wedge Embedded(\beta, \alpha)] \supset Embedded(b \circ \beta, a \circ \alpha) \\
&\}
\end{aligned}
$$

The new ordering *embedded-subsequence* is captured by predicate *Embsubseq,* defined in terms of the relations *Embedded* and *Subsequence* in the following block:

$$
\begin{aligned}
&B_{\text{embsubseq}} = \\
&\{\min\ Embsubseq\ : \\
&\qquad Embedded(\alpha, \beta)\ \wedge Subsequence(\beta, \gamma) \supset Embsubseq(\alpha, \gamma) \\
&\}
\end{aligned}
$$

At the top level of nesting we add the following axiom

$$\neg Undef(Sit\_map(S_N))$$

which restricts actual situations to be mapped onto defined states. (We did not need this in $T(D)$, because our restriction on D not to allow contradictory causal laws guarantee that the causal interpretation Ψ is defined for all action sequences.)

Finally, $Sit\_map(S_N)$ is now minimized w.r.t. the partial order *Embsubseq* by replacing axiom (c) with axiom

$$Embsubseq(\alpha, Sit\_map(S_N)) \supset Ab(\alpha) \tag{18}$$

We will denote the new theory by T_c and the modified block SC by SC_c.

$T_c(D) =$

$\{ \; Sit\_map \; :$

$\neg Undef(Sit\_map(S_N))$

$(a) \; Sit\_map(S_0) = \epsilon$

$(b) \; Prefix\_eq(Sit\_map(s), Sit\_map(S_N))$

$(c') \; Embsubseq(\alpha, Sit\_map(S_N)) \supset Ab(\alpha)$

$\tau(D_f), \;\; SC_c(D_l)$

$B_{\text{prefix\_eq}}, \;\; B_{\text{subsequence}}, \;\; B_{\text{concatenate}}, \;\; B_{\text{embedded}}, \;\; B_{\text{embsubseq}}$

Framework axioms

$\}$

where:

$SC_c(D_l) =$

$$
\left\{
\begin{array}{l}
\neg Inherits^{+}(a,f,\alpha)\ \wedge \neg Inherits^{-}(a,f,\alpha) \supset [Holds(f,\alpha) \equiv Holds(f, a\circ\alpha)] \\
\neg Undef(a\circ\alpha)\ \wedge Inherits^{+}(a,f,\alpha) \supset Holds(f, a\circ\alpha) \\
\neg Undef(a\circ\alpha)\ \wedge Inherits^{-}(a,f,\alpha) \supset \neg Holds(f, a\circ\alpha) \\
\{\ \min\ Causes^{+(-)}\ : \\
\quad H(P_1,\alpha)\ \wedge \ldots\ \wedge H(P_n,\alpha) \supset Causes^{+(-)}(A,F,\alpha) \\
\qquad\qquad\qquad\qquad\qquad (\text{for each } A \textbf{ causes } (\neg)F \textbf{ if } P_1,\ldots,P_n \in D) \\
\} \\
\{\min\ \ Inherits^{+(-)}\ : \\
\qquad b \in a\ \wedge Causes^{+(-)}(b,f,\alpha) \supset Inherits^{+(-)}(a,f,\alpha) \\
\} \\
\{\min\ \ Undef\ : \\
\qquad Inherits^{+}(a,f,\alpha)\ \wedge Inherits^{-}(a,f,\alpha) \supset Undef(a\circ\alpha) \\
\qquad Undef(\alpha) \supset Undef(a\circ\alpha) \\
\}
\end{array}
\right.
$$

The translation of facts from D is the same except for occurrence facts; let $\tau(\alpha$ **occurs_at** $S)$ stands for:

$$
\begin{array}{l}
(\exists\beta,\gamma).Embedded(\alpha,\beta)\ \wedge Concatenate(Sit\_map(S),\beta,\gamma) \\
\qquad\qquad\qquad\qquad\qquad \wedge Prefix\_eq(\gamma, Sit\_map(S_N)))
\end{array}
$$

We now state the correctness of our NAT formulation of $\mathcal{L}_c$ with propositions similar to those in Section 7. The proofs are given in the Appendix.

Proposition 5 *(Causal Equivalence)*

Part 1: *For every causal model Ψ of D_l there exists a model $\mathcal{M}_{\mathrm{SC}_c}$ of $SC_c(D_l)$ such that for all F,α:*

$$
\begin{array}{l}
F \in \Psi(\alpha)\ \Leftrightarrow\ \mathcal{M}_{\mathrm{SC}} \models_{\mathcal{L}} Holds(F,\alpha),\ \text{and} \\
\Psi(\alpha)\ \textit{is undefined}\ \Leftrightarrow\ \mathcal{M}_{\mathrm{SC}} \models Undef(\alpha)
\end{array}
\tag{19}
$$

Part2: *For every model $\mathcal{M}_{\mathrm{SC}}$ of $SC_c(D_l)$ there exists a causal model Ψ of D_l*

such that (19) holds. □

Proposition 6 *For any domain description D, if interpretation (Ψ, Σ) is a model of D then there exists a model $\mathcal{M}$ of $T_c(D)$ such that for every fact ϕ in the language of D:*

$$(\Psi, \Sigma) \models_{\mathcal{L}} \phi \;\Leftrightarrow\; \mathcal{M} \models \tau(\phi) \tag{20}$$

and if $\mathcal{M}$ is a model of $T_c(D)$ then there exists a model (Ψ, Σ) of D such that (20) holds. □

Proposition 7 *For any domain description D and fact ϕ in the language of D:*

$$D \models \phi \;\Leftrightarrow\; T_c(D) \models \tau(\phi)$$

□

Example 6 Consider a scenario where there are two guns which cause Fred to die if fired at him when loaded. Suppose all we know is that initially either gun was loaded and that both were fired at him. The following domain description captures this story:

$$\left.\begin{array}{l}
shoot(1)\ \textbf{causes}\ \ \neg alive\ \textbf{if}\ \ loaded(1) \\
shoot(2)\ \textbf{causes}\ \ \neg alive\ \textbf{if}\ \ loaded(2) \\
alive\ \textbf{at}\ \ S_0 \\
(loaded(1)\ \textbf{at}\ \ S_0) \vee (loaded(2)\ \textbf{at}\ \ S_0) \\
\{shoot(1), shoot(2)\}\ \textbf{occurs_at}\ \ S_0
\end{array}\right\} D_{\text{guns}}$$

It is easy to see that D_{guns} entails $\neg alive$ **at** S_N. □

9 Further extension of $\mathcal{L}$ –allowing actions with indirect effects

In this section we discuss how the NAT formalization of $\mathcal{L}$ can be easily extended to allow *ramification constraints, and axioms about executability of actions.* Giunchiglia, Kartha and Lifschitz [KarLif94,GiuKarLif95] introduced the high level language $\mathcal{AR}$ with the above mentioned features (and some additional ones, such as actions with non-deterministic effects) with non-propositional fluents, but in the absence of narratives. They then give a formulation of temporal projection problems in $\mathcal{AR}$ using NATs. Essentially, if we restrict their formulation to boolean fluents, then we can say that their NAT characterization corresponds to a causal model Ψ for an extended language that allows ramification constraints, executability conditions and non-deterministic effects.

Since NATs are nestings of independent blocks, we can replace the $SC(D_l)$ part of the NAT $T(D)$ of Section 4.5 – that characterizes the causal model of our original but restricted language, by a new theory that captures the causal model of the extended language. Moreover, the resultant NAT gives a characterization of an extension of $\mathcal{L}$ where constraints and executability conditions are allowed.

We now present the syntax of (ramification) constraints, and executability conditions and touch upon how their addition to $\mathcal{L}$ necessitates some changes in the semantics.

Constraint are of the form

$$\textbf{always} \;\; C \tag{21}$$

where C is a propositional fluent formula. Intuitively, a constraint **always** $(in\_lake \supset wet)$. asserts that if someone is in the lake then he/she is wet.

The semantics of $\mathcal{L}$ extended with constraints is defined by requiring $Res(A, \sigma)$ to be the set of valid states *closest* (in the sense of set difference) to σ which contain $E_A^+(\sigma)$ and do not contain any element of $E_A^-(\sigma)$) –where valid states are states that satisfy all the constraints; and requiring $\Psi(\alpha \cdot A)$ to be an element of $Res(A, \Psi(\alpha))$. This guarantees that for all $alpha$, either C holds in $\Psi(\alpha)$ or $\Psi(\alpha)$ is undefined and captures the indirect effect of actions due to constraints.

An executability condition is of the form

$$\textbf{impossible} \;\; A \; \textbf{if} \;\; Q_1, \ldots, Q_n \tag{22}$$

where $Q_1, \ldots, Q_n$ are fluent literals. From the semantics point of view, a proposition of this type is satisfied by a causal interpretation Ψ if $\Psi(\alpha \cdot A)$ is undefined when $Q_1, \ldots, Q_n$ are satisfied by $\Psi(\alpha)$. For instance,

impossible $get\_out\_of\_the\_lake$ **if** $\neg in\_lake$

means that it is impossible to execute the action $get\_out\_of\_the\_lake$ in a state where in_lake is false.

We are now ready to discuss the theory, T^+ obtained from T (of Section 4.5) by replacing SC with SC^+ which encodes the transition function Ψ in the presence of constraints and executability conditions. We now present the theory SC^+.

$SC^+ =$

$(h)\ \gamma_\alpha C$ (for each constraint (21))

$(i)\ Holds^*\ (f, a \circ \alpha) \equiv Holds(f, a \circ \alpha)$

$\{Holds^*$:

$(j)\ \neg Undef(a \circ \alpha)\ \wedge \neg Ab(f, a, \alpha) \supset$
$[Holds(f, \alpha) \equiv Holds^*(f, a \circ \alpha)]$

$\{Holds^*$, min $Undef$:

$(k)\ H(P_1, \alpha)\ \wedge \ldots\ \wedge H(P_m, \alpha) \wedge$
$\neg Undef(A \circ \alpha) \supset H^*(F, A \circ \alpha)$
(for each causal law (1))

$(l)\ H(Q_1, \alpha)\ \wedge \ldots\ \wedge H(Q_n, \alpha) \supset Undef(A \circ \alpha)$
(for each proposition (22))

$(m)\ Undef(\alpha) \supset Undef(a \circ \alpha)$

$(n)\ \gamma^*_{a \circ \alpha} C$ (for each constraint (21))

$\}$

$\}$

where, $\gamma_\alpha C$ stands for the universally closed formula obtained from a constraint (21) by replacing each fluent literal F by $H(f, \alpha)$. Note also that

- SC^+ is a slightly modified and simplified version of the main part of the NAT in [GiuKarLif95,Kar95]. The main differences are: our use of boolean fluents instead of non-boolean ones as in [Kar95] and our use of conjunction of fluent literals in the **if** part of (1) and (22) and in the **causes** part of (1) instead of fluent formulas, as in [Kar95]. We make these restrictions for simplicity, and to stay as close to $\mathcal{L}$ as possible, while still being able to make our point about easy incorporation of constraints and executability conditions into $\mathcal{L}$.
- The outer block of SC^+ encodes $Res(a, \sigma)$, for all actions a, and for all states σ that are mapped to by some action sequence. While σ is encoded using $Holds$, $Res(a, \sigma)$ is encoded using $Holds^*$. The inner block defines which actions are undefined (or not executable) in which state.

The reason behind having two different predicates $Holds$ and $Holds^*$ and only varying the latter when minimizing Ab, is to individually minimize the difference between σ and $Res(a, \sigma)$; without globally minimizing all such differences as a whole. The latter precludes many models and may result in intuitive interpretations being ruled out. Consider the example where we have a **causes** f **if** p and **always** $\neg f$ in our domain description. If we had replaced $Holds^*$ in SC^+ by $Holds$. Then minimizing $Undef$ while varying $Holds$ in the inner block would result in $\neg Holds(p, \alpha)$ being true in all models of the inner block. This is prevented by using separate $Holds$ and $Holds^*$.

– Although at first glance it seems that only (l) and (m) are necessary in the inner block, the necessity of (k) and (n) are to specify undefined action sequences that may arise because of possible interaction between effect axioms and constraints, as it might happen when the domain is constrained to have P initially true.

  Note that our constraints are meant for ramification; i.e. if we have **always** $\neg(f \wedge g)$ and a **causes** f, then application of a in a state where g is true (directly) makes f true in the resulting situation and (indirectly, because of ramifications) makes g false in the resulting state. On the other hand, if we have
  always $\neg(f \wedge g)$, a **causes** f and a **causes** g **if** p,
  then a is not executable in any situation where p is true. To capture this we need (k) and (n) in the inner block.

– axiom (j) encodes the law of inertia.

– axiom (h) restricts all states to be valid, making sure they satisfy all the constraints.

– axiom (i) relates $Holds$ and $Holds^*$. Once the minimizations are done appropriately in the inner and outer blocks, axiom (i) selects only those models where $Holds$ and $Holds^*$ agree.

– Another difference between SC^+ and the corresponding theory in [KarLif94,Kar95] is that in our case the second argument of $Holds$ is a sequence of actions while in the other ones it is a state. This is because all situation constants in our language are mapped onto a sequence of actions; there is no such requirement in [KarLif94,Kar95].

– Finally, although we did not discuss allowing indeterminate propositions [Kar95], they can be easily incorporated. An indeterminate proposition of the form A **possibly_changes** F **if** $P_1, \ldots, P_k$ might be incorporated into SC^+ by adding

$$H(P_1, \alpha) \wedge \ldots \wedge H(P_k, \alpha) \supset Ab(F, A, \alpha)$$

  just before (j) inside the outer block defining $Holds^*$.

Although we do not formally state and prove the equivalence between the causal interpretation Ψ of a domain description D with constraints and executability conditions and the corresponding SC^+, the equivalence directly follows from [Kar95]. Moreover, the correspondence between domain descriptions in the extended $\mathcal{L}$ (with constraints and executability conditions) and the corresponding NATs, with SC replaced by SC^+, follows (with slight modifications) from [Kar95] and the proof of Lemma 2.

We would like to stress that the ability of NATs to allow us to treat blocks in a similar way as subroutines in structured programming makes NATs a strong tool for knowledge representation. This allowed us to take a block from the formalization of actions of Giunchiglia et. al. and "insert" it into our theory with practically no complications.

10 Filtering and restricted monotonicity in T and T^+

To the best of our knowledge, T and T^+ are among the largest circumscriptive theories in the literature, in terms of the number of levels of minimization and filtering [San89]. They are thus good testbed examples for analyzing the expressibility of NATs (and circumscription in general) in terms of the various KR features they incorporate. In Section 5 we already discussed how *value minimization* [BarGabPro96b] of functions was expressed using NATs in the theory T. In this section we focus on two additional aspects – filtering and restricted monotonicity w.r.t. T and T^+.

10.1 Filtering

The notion of filtering was first introduced by Sandewall in [San89]. We define it as follows:

Definition 10 *Let T be a (possibly nonmonotonic) theory and Q (an observation) be a set of sentences in first-order logic. By $Filter(T, Q)$, we refer to the theory whose models are the models of T that are first-order models of Q.* $\square$

Proposition 8 *$Filter(T, Q)$ is monotonic with respect to Q.* $\square$

Proof: Follows directly from the definition of $Filter(T, Q)$; adding sentences to Q can only decrease the models of $Filter(T, Q)$.

10.2 Restricted monotonicity in filtering

The concept of restricted monotonicity was introduced by Lifschitz [Lif93]. In this subsection we first recall some of his definitions and then show how filtering and restricted monotonicity are related and how restricted monotonicity is captured in NATs.

Definition 11 *[Lif93] A* declarative formalism *is defined by a set S of symbolic expressions called* sentences*, a set P of symbolic expressions called* postulates*, and a map Cn from sets of postulates to sets of sentences.*

A set of postulates is referred to as a theory*; and a sentence A is a consequence of a theory T is $A \in Cn(T)$.* □

Definition 12 *[Lif93] Let $\langle S, P, Cn\rangle$ be a declarative formalism. Let a subset S_0 of S be designated as the set of assertions, and a set P_0 of P as the set of parameters. A theory T is said to satisfy the restricted monotonicity condition with respect to S_0 and P_0 if, for any sets $p, q \subset P_0$,*

$p \subset q \Rightarrow Cn(T \cup p) \cap S_0 \subset Cn(T \cup q) \cap S_0$ □

In [Lif93], Lifschitz gives several examples of restricted monotonicity and in [KarLif94,GiuKarLif97] Kartha, Giunchiglia and Lifschitz prove restricted monotonicity of $\mathcal{AR}$ and $\mathcal{AR}_0$. Here we generalize some of the above results and discuss the restricted monotonicity associated with filtering in general.

In case of filtering, we can say that the theory $Filter(T, Q)$ has the restricted monotonicity property w.r.t. Q. This follows directly by considering parameters as statements of the language of observations, defining $Cn(T \cup p)$ to be the set $\{f \mid f$ is true in all models of $Filter(T, Q)\}$ and considering S_0 to be the set of all sentences.

Filtering is easily achieved in NATs by means of two blocks representing Q and T; thus making it easy [10] to express restricted monotonicity in NATs. This is exactly what happens in T and T^+, respectively.

Before moving on to analyze block of T and T^+. let us say a few more words about NATs as a knowledge representation language. Besides the issue of recent advances of automated circumscriptive reasoning [Doherty et al. 95] that benefit NATs, *NATs are good for representing knowledge because they allow*

[10] Among other knowledge representation languages, filtering is easily done in logic programming by representing observations as integrity constraints. On the other hand, logic programs do not easily allow a hierarchy of integrity constraints. Thus, multiple levels of filtering are not easily represented in logic programs.

us to encode nonmonotonicity about certain aspects, and at the same time guarantee encoding of restricted monotonicity with respect to others. Moreover, nesting allows several levels of filtering. Prima facie, the equivalence of $Filter(Filter(T, Q), Q')$ and $Filter(T, Q \cup Q')$ may suggest that several levels of filtering are not needed On the other hand, often (as in [San89]), we need to compute $Filter(Min(Filter(T, Q), \leq), Q')$, where we minimize $Filter(T, Q)$ with respect to an ordering $\leq$ before filtering the result by Q'. Assuming that $\leq$ can be encoded by varying $P_1, \ldots, P_k$ and minimizing $P_{k+1}, \ldots, P_m$, and T itself is a NAT, we can represent $Filter(Min(Filter(T, Q), \leq), Q')$ by the following:

$$
\begin{array}{l}
\{ \\
\quad Q' \\
\quad \{ \quad P_1, \ldots, P_k, \quad min P_{k+1}, \ldots, P_n \\
\quad\quad\; Q \\
\quad\quad\; T \\
\quad \} \\
\}
\end{array}
$$

10.3 Analyzing blocks and sub-blocks of T and T^+

- Consider the inner block of SC^+. Let us call it B_{inner}. Intuitively, this block (among other things) encodes a function from $Holds$ literals to $Undef$ literals. The axioms $(k), (l), (m)$, and (n) partially define this function, which is further refined by the minimization of $Undef$ while varying $Holds^*$. This leaves open the possibility to add observations about the successful execution of sequences. So, if sequence β has succeeded, we can add $\neg Undef(\beta)$ as a new block B_1 consisting only of $\neg Undef(\beta)$, added at the same level of axiom (j).

  In the NAT B_2 consisting of B_{inner} and B_1 we are basically filtering B_{inner} by B_1 and the resultant models are the models of B_{inner} that are also models of B_1. Thus B_2 is monotonic w.r.t. addition of observations to B_1 but nonmonotonic w.r.t. addition of causal laws, constraints, or executability conditions, which change B_{inner}.

- Let us call B_{outer} the outer block of SC^+, which intuitively encodes a function from $Holds$ literals to $Undef$ and $Holds^*$ literals. Observations about $Holds^*$ are put outside of B_{outer}, as axioms (h) and (i) in SC^+, and axioms $FACTS$ and (e) in T^+. Here also, the observations filter the NAT B_{outer}. The resulting theory has the restricted monotonicity property about the observations.

– The block $T(D)$, represents a function from literals of predicates At, $Occurs$, $Precceds$, $Holds$, $Subsequence$, $Prefix\_eq$, $Concatenate$ to mappings of the function Sit_map. Observations about the value of Sit_map can be incorporated by putting them outside $T(D)$. These observations will then filter the models of $T(D)$ and keep only those that agree with them.

– In [BarGelPro97] hypothesis of the form F **after** α **at** S were treated as observations on top of the domain description in $\mathcal{L}$. We can incorporate them to our NAT consisting of $T(D)$, by adding $\forall\beta Concatenate(Sit\_map(S), \alpha, \beta) \supset Holds(F, \beta)$ to the NAT –not inside $T(D)$. These observations will filter $T(D)$, and select only those models of $T(D)$ that agree with the observations. The theory $Filter(T(D), Obs)$ will thus be monotonic with respect to the hypotheses, while it is nonmonotonic with respect to facts, which are incorporated by changing $T(D)$.

11 Related work

Until recently there were two distinct directions in reasoning about actions; one based on situation calculus which considered hypothetical reasoning and planning, normally simple actions (i.e., no continuous actions, no actions with durations, etc.), while the other (in particular, Allen's temporal logic [All84], Kowalski and Sergot's event calculus [KowSer86]) focused on reasoning with narratives and often allowed actions with durations, and continuous actions, but did not consider hypothetical reasoning. In the late eighties and early nineties several influential works [GelLifRab91,GelLif92,San92,Rei91] have had a big impact in this field, and triggered a flurry of new research. Some of these appear in the special journal issues [Lif97,Geo94], in the workshops AAAI'96 Workshop on Reasoning about actions and AAAI'95 Spring symposium on Extending Theories of Action, and in the recent AAAI, IJCAI and KR conferences.

In general, our work has been influenced by the approach of using high level action description languages [GelLif92,GiuKarLif97,BarGel93,BarGelPro97], and their formalization (particularly of $\mathcal{AR}$ [GiuKarLif97]) using nested circumscription. In the previous sections we argued why we believe nested circumscription is an excellent KR language, and listed some of its features. Also, we acknowledge Sandewall's idea of filtering [San89] with which NATs has much in common.

In this section we first compare NATs with Sandewall's filter preferential entailment and then give a detailed comparison of our work with other proposals that do both narrative and hypothetical reasoning; particularly, those by McCarthy [McC95], Kakas and Miller [KakMil96], Miller and Shanahan [MilSha94], and Pinto and Reiter [PinRei93,Pin96].

11.1 Sandewall's filter preferential entailment

The notion of filtering was first introduced by Sandewall [San89] to be able to formally *obtain explanations in terms of action occurrences* given action descriptions (in terms of conditions and effects), physical laws, and observations about values of fluent at specific time instants. He argued that his approach can also be used to obtain a plan for a specific goal, by considering the goal as an observation which needs to be explained, and considering an explanation as a plan that achieves the goal.

Sandewall's notion of filtering is a general one. The NATs formalism that we use in this paper is an instance of it, where circumscription is used for minimizations, with local abnormal predicates.

He motivates the intuition behind using filtering of the models of a theory containing action descriptions, and natural laws (No observations) by observations instead of the models of the theory containing action descriptions, natural laws and observations by appealing to an example involving a moving object. *We give a different justification in Section 10; we show that by using filtering the resulting theory has the restricted monotonicity property with respect to the observations.*

Although Sandewall's formalism [San89] considers one of the important aspects of narratives (i.e., explaining values of fluents at different time instants by action occurrences), it does not cover the whole issue of hypothetical reasoning in presence of narratives and unlike our does not have a notion of planning from the current situation, which we do in this paper. On the other hand, Sandewall considers actions with durations, which are not considered in this paper.

11.2 McCarthy's formulation of situation calculus with concurrent events and narratives

In this section we relate our work with McCarthy's draft titled 'Situation Calculus with concurrent events and Narrative' [McC95] available through his

Web page. In our comparison we make several quotes from his draft.

- McCarthy says:

  > Situations in a narrative are partially ordered in time. The real situations are totally ordered, but the narrative does not include full information about this ordering.

  We agree with the above statement, but in our formalization of a narrative description with partially ordered situations, rather than having models encoding this partial ordering,each model encodes a possible total ordering of the situations.

- He also writes:

  > In a narrative, it is not necessary that what is said to hold in a situation be a logical consequence (even nonmonotonically) of what was said to hold about a previous situation and known common sense facts about the effects of events. ... Nevertheless, some narratives are anomalous. ... We want to introduce a concept of *proper narrative*, but it isn't clear exactly what it should be. The fluents holding in a new situation should be reasonable outcomes of the events that have been reported, except for those fluents that are newly asserted, e.g. [...]

  In our formulation, if something is said to hold in a situation, it may not logically follow from the rest of the description itself, though we incorporate the assumption that it must have an explanation. Since this assumption is part of our formulation, when something is said to hold in a situation, it is a logical consequence of action occurrences leading up to this situation. Yet, these action occurrences need not be all explicitly stated in the original description, some may be abduced from the observations that are made. Of course, a description may have several models, each suggesting a different action occurrence.

  We believe that our formulation suggests a definition for *proper narrative*, which is not defined in [McC95], and in our formulation we require that models of the description encode proper narratives, and lack of information leading to not having a unique proper narrative results in multiple models of the description.

- McCarthy goes on writing:

  > Perhaps narrative seems easy, since it is not yet clear what facts must be included in a narrative and what assertions should be inferable from a narrative.

  We hope that in the previous sections we have given a satisfactory answer to this.

- McCarthy also discusses elaborations of actions in a narrative by finer actions and sub-narratives; we do not consider them here.

– McCarthy informally discusses an interesting example involving narratives. The example has two versions, one consisting of two narratives occuring concurrently and independently, and another where the same two narratives have a few points of interaction. In the following example, we show how the second version of the example can be represented in $\mathcal{L}$. (The representation of the first version is a subset of the representation of the second version.)

Example 7 *(Glasgow, London, Moscow and New York [McC95])* The story is the following: there is a person called Daddy in New York who is stacking blocks. At the same time, there is a person called Junior who is in Glasgow and has tickets for and is traveling from Glasgow to Moscow via London. Arriving into London, Junior loses his ticket and sends a telegram to Daddy asking for money. Daddy then sells one of his blocks to get the money and sends it to Junior. Junior then gets the money, buys a London-Moscow ticket and finishes his trip.

Part of a domain description in $\mathcal{L}$ which captures this story would be the following:

Causal law schemata:

$Flies(Prsn, X, Y)$ **causes** $At(Prsn, Y)$ **if** $AtPprsn, X), Has(Prsn, tkt(X, Y))$

$Flies(Prsn, X, Y)$ **causes** $\neg At(Prsn, X)$ **if** $At(Prsn, X), Has(Prsn, tkt(X, Y))$

$Lose(Prsn, X)$ **causes** $\neg Has(Prsn, X)$

$Buys(Prsn, X)$ **causes** $Has(Prsn, X)$ **if** $Has(Prsn, money)$

$Sells(Prsn, blk1)$ **causes** $Has(Prsn, money)$

$Sells(Prsn, blk2)$ **causes** $Has(Prsn, money)$

$Sends(Prsn1, Prsn2, X)$ **causes** $Sent(Prsn1, Prsn2, X)$ **if** $Has(Prsn1, X)$

$Receives(Prsn1, Prsn2, money)$ **causes** $Has(Prsn1, money)$ **if** $Sent(Prsn2, Prsn1, money)$

$Receives(Prsn1, Prsn2, tlgrm)$ **causes** $Rcvd(Prsn1, tlgrm)$ **if** $Sent(Prsn2, Prsn1, tlgrm)$

$Stack(Prsn, X, Y)$ **causes** $On(X, Y)$ **if** $Clear(X), Clear(Y)$

Fluent facts about the initial situation:

Junior:	**Daddy:**
$At(jr, gw)$ **at** S_0	$At(ddy, ny)$ **at** S'_0
$Has(jr, gw\_ldn\_tkt)$ **at** S_0	$\neg Has(ddy, money)$ **at** S'_0
$Has(jr, ldn\_mscw\_tkt)$ **at** S_0	$Has(ddy, blk1)$ **at** S'_0
	$Has(ddy, blk2)$ **at** S'_0
	$On(blk1, tbl)$ **at** S'_0
	$On(blk2, tbl)$ **at** S'_0

Occurrences of actions:

Junior:	**Daddy:**
$Flies(jr, gw, ldn)$ **occurs_at** S_0	$Stacks(ddy, blk2, blk1)$ **occurs_at** S'_0
$Lose(jr, tkt(ldn, mscw))$ **occurs_at** S_1	
$Sends(jr, ddy, tlgrm)$ **occurs_at** S_2	
	$Receives(ddy, jr, tlgrm)$ **occurs_at** S'_1
	$Sells(ddy, blk1)$ **occurs_at** S'_2
	$Sends(ddy, jr, money)$ **occurs_at** S'_3
$Receives(jr, ddy, money)$ **occurs_at** S_3	
$Buys(jr, ldn\_mscw\_tkt)$ **occurs_at** S_4	
$Flies(jr, ldn, mscw)$ **occurs_at** S_5	

Ordering on situations:

S_0 **precedes** S_1		S'_0 **precedes** S'_1
S_1 **precedes** S_2		
	S_2 **precedes** S'_1	
		S'_1 **precedes** S'_2
S_2 **precedes** S_3		
		S'_2 **precedes** S'_3
	S'_3 **precedes** S_3	
S_3 **precedes** S_4		
S_4 **precedes** S_5		

For this domain description, the semantics of $\mathcal{L}$ tells us among other things that $Has(ddy, money)$ **at** S_3', $Has(jr, money)$ **at** S_4, **currently** $At(jr, mscw)$; and that neither S_1 **precedes** S_0' nor S_0' **precedes** S_1 are entailed by the domain description. Intuitively, we have two sub-narratives occurring simultaneously with few points of interaction. Although in reality there is a total order of all the events, the domain description does not entail such an order because it contains only partial information about it. $\square$

11.3 Miller and Shanahan's circumscriptive approach

In [MilSha94] Miller & Shanahan introduced a formalization of narratives using the situation calculus and circumscription. Their formalization of narrative has many similarities to our work. Their function $State$ which maps time points (real numbers) to situations (constructed using the function Res, an initial situation S_0 and action constants) is similar to our Σ which maps situation constants to sequences of actions.

However, their formalization requires [11] that the domain description include all occurrences of actions and fluent facts are restricted to be about the initial situation only.

Our approach is then more general with respect to the above restrictions. And further, propositional combinations of fluent facts, occurrence facts and precedence facts are also allowed in our formalization. Our semantics incorporates the abductive reasoning necessary to make conclusions regarding occurrences of actions and values of fluents in different situations, even if they are not explicitly stated in the domain description. On the other hand, [MilSha94] contains discussions on allowing divisible and overlapping actions, which we do not discuss in this paper.

11.4 Kakas and Miller's $\mathcal{E}$

Kakas and Miller [KakMil96] have introduced a high level language $\mathcal{E}$ based on event-calculus. A domain description in their language consists of c-propositions, h-propositions and i-propositions, which are similar to causal laws, atomic occurrence facts, and atomic fluent facts in our language. Their time points correspond to situation constants in our language. The major

[11] They do point out that these restrictions can be weakened using abduction.

differences between their work and that of ours are:

– In our language the ordering between situation constants is part of the domain description while in their language it is rather part of the domain language.

– Hypothetical reasoning in $\mathcal{E}$ is done by defining Δ-sequences (which are hypothetical time points) corresponding to each hypothetical situation, and for a time point between two consecutive hypothetical situations; and then defining the ordering between these Δ-sequences. These definitions are included in the domain language. In addition hypothetical action occurrences are added to the domain description. The following example illustrates how we can reason about the hypothetical situation $Res(Shoot, Res(Load, S_0))$ in $\mathcal{E}$.

  · First the domain language will contain the Δ-sequences:

    (i) $\langle\langle\rangle\rangle$,
    (ii)$\langle\langle|load|\rangle\rangle$,
    (iii)$\langle\langle|load|, load\rangle\rangle$,
    (iv)$\langle\langle|load|, load, |shoot|\rangle\rangle$, and
    (v) $\langle\langle|load|, load, |shoot|, shoot\rangle\rangle$.

    The Δ-sequences (i), (iii) and (v) above correspond to the situations S_0, $Res(Load, S_0)$, and $Res(Shoot, Res(Load, S_0))$, respectively. The Δ-sequences (ii), and (iv) above correspond to the time point between the situations S_0 and $Res(Load, S_0)$, and $Res(Load, S_0)$ and $Res(Shoot, Res(Load, S_0))$, respectively. The Δ-sequences (ii), and (iv) are necessary to be able to specify that the action *Load* happened in the Δ-sequences (ii) and that the action *Shoot* happened in the Δ-sequences (iv).

  · Next the domain language will also contain the following ordering between the Δ sequences:
    $\langle\langle\rangle\rangle$ precedes $\langle\langle|load|\rangle\rangle$ precedes $\langle\langle|load|, load\rangle\rangle$ precedes $\langle\langle|load|, load, |shoot|\rangle\rangle$ precedes $\langle\langle|load|, load, |shoot|, shoot\rangle\rangle$.

  · Finally, the domain description will contain action occurrences
    load happened at $\langle\langle|load|\rangle\rangle$, and
    shoot happened at $\langle\langle|load|, load, |shoot|\rangle\rangle$.

  It should be noted that the number of δ sequences is infinite, thus making the domain language infinite; also the number of action occurrences will be infinite. However, both can be finitely expressed in a logical language such as logic programming.

It seems to us –and hopefully the above example illustrates it– that the additional formulations in $\mathcal{L}$ (as opposed to $\mathcal{A}$) that is used for incorporating narratives into hypothetical reasoning (i.e. going from $\mathcal{A}$ to $\mathcal{L}$) is much less than that is necessary in simulating hypothetical reasoning in the narrative based language $\mathcal{E}$. Nevertheless, it is important to know, and Kakas and Miller [KakMil96] show us, how hypothetical reasoning can be done in a narrative-based language.

- Unlike than in $\mathcal{L}$, in $\mathcal{E}$ information about action occurrences (called h-propositions) is assumed to be complete. Although this seems very restrictive, they later simulate incompleteness by introducing a notion of explanation that restores inconsistency in a theory where the completeness assumption makes it contradictory. They first start with explanation consisting of action occurrences (h-explanation), and then incorporate explanation consisting of c-propositions (our causal laws). *In our formalization we assume our set of causal laws to be complete.*

  Kakas and Miller then consider projection domain descriptions, i.e. those where t-propositions (our atomic fluent facts) are only allowed about the initial situation (such t-propositions are referred to as i-propositions), and discuss how to incorporate observations about fluent values non-initial time points (such t-propositions are referred to as o-propositions). They incorporate observations by first explaining in terms of additional i-propositions (called i-explanations) and then explain in terms of both i-propositions and h-propositions (called ih-explanations), where the h-propositions are used to restore consistency.

  In contrast, our domain descriptions allow incomplete information about action occurrences, values of fluents at both initial and non-initial situations, and a partial specification of the ordering between the situation constants. *Our formulation is such that each model of the domain description fills in the missing action* occurrences, and has a total ordering between the situation constants. (The ordering between time-points in $\mathcal{E}$ cannot be total as it also orders the hypothetical time-points.)

  Thus, we incorporate the concept of ih-explanation, i-explanation and h-explanation in the object language itself without resorting to restoring consistency or meta-level explanation. Still, we do not allow explanations through c-propositions/causal laws, which we consider as fixed.

- Finally, we believe that our notions of *current situation* and *planning from the current situation* to be very important. These notions are not considered in [KakMil96].

11.5 Pinto and Reiter's actual and legal situations

Pinto and Reiter in [PinRei93] were among the first ones to introduce a time line into situation calculus, allowing not only the hypothetical reasoning inherent of situation calculus but also the expression of event occurrences and values of fluents at different time instants. *Their approach, based on Reiter's [Rei91] solution to the frame problem, was restricted to non-concurrent actions, and appealed to circumscription to minimize occurrences.* In their formalization, they introduced a predicate *actual* on situations to characterize situations that lie on the path that describes the world's real evolution. Unlike the formalism in $\mathcal{L}$ [BarGelPro97], they do not have the notion of a *current* situation and of *planning from the current situation.* Their notion of planning [12] then boils down to finding actual situations that satisfy a given goal. By minimization of *occurs* in the actual line, the possible existence of new situations –ones not dictated by the given axiomatization, is ruled out; this results in not being able to find new situations where the goal is satisfied. Reiter refers to this as the 'premature minimization problem' [Rei96] and points out that Miller and Shanahan's formalism also suffers from it. Even though we also minimize action occurrences, *our formalism does not suffer from this problem*, because the minimization is only used to define the current situation while planning (by hypothetical reasoning) is still possible from the current situation.

Pinto in his thesis [Pin94] builds up on the work in [PinRei93], and discusses many new issues including concurrent actions, continuous actions and natural events; since the core of the approach: –the use of Reiter's solution to the frame problem, definition of actual, and minimization of occurs, remains unchanged from [PinRei93], his work also suffers from the same problem of premature minimization and inability to make executable plans.

In their later work [PinRei95,Rei96,Pin96], Pinto and Reiter *abandon minimization of occurrences*, and in [Rei96] and [Pin96] they allow the possibility of several 'hypothetical actual branches' – whose situations are now referred to as 'legal' instead of 'actual', that *restores the capability of making plans.* But even here, there approach to planning is based on finding a legal situation where the goal is satisfied. It is not clear to us if this notion of planning is useful in the scenario of an agent in a dynamic world. The agent does not need a plan from the initial situation –as Pinto and Reiter's approach would give us; rather, it needs a plan from the current one.

Let us now try to list the major differences between our approach and that of Pinto and Reiter's.

– Our approach is based on a high level language, $\mathcal{L}$, which is restrictive compared to that of Pinto and Reiter's. For example, our domain description does not allow a statement that has both holds and occurs in them. Moreover, we do not allow triggers, and natural events, which are considered in

[12] The issue of planning is not directly mentioned in [PinRei93], but based on the discussion in [Rei96] on 'premature minimization' and the notion of planning there, it seems that this is what Pinto and Reiter had in mind.

[Pin94,Rei96,Pin96].

- In our formalism the only restriction on the legality of actions in a branch is due to the executability conditions. Thus our branches consists of 'legal' situations and are 'hypothetical actual branches' in their sense.
- In [Rei96,Pin96] 'legal situations' replace the notion of 'actual situation.' Such is not the case in our formalism. While all situations in our branches are legal, only a finite set of situations in a finite line –called the *actual line,* from the initial situation and ending at the 'current' situation are considered to be 'actual'. *We minimize the occurrences of actions in this actual line.* In contrast,Pinto and Reiter [PinRei93,Pin94] face the problem of premature minimization while minimizing occurs, in later works [Pin96,Rei96] they abandon [13] minimization of occurrences. Also, it seems [14] that Pinto in [Pin96] *does not allow statements about values of fluents at non-initial situations;* allowing such statements and abducing additional information from them in the object language itself is an important aspect of our work.

  We believe –and so do Pinto and Reiter in [PinRei93,Pin94]– on the intuitiveness of minimizing action occurrences on the description of the evolution of the world up until the current situation. We avoid the problem of 'premature minimization' because of carefully minimizing only the action occurrences in the 'actual line', while they minimize action occurrences without restraint. Our notion of current situation comes in handy.
- Pinto and Reiter stay within first-order logic as much as possible. In [PinRei93,Pin94] all their axiomatization except the induction axiom, and the minimization of occurs at the end, are in first order logic; and later [Pin96,Rei96] all their minimization except the induction axiom is in first order logic.

  Our formalization here is based on circumscription, and seemingly makes it easier to allow constraints and even actions with uncertain effects together with effect axioms. Pinto concurs and in [Pin96] says:

  > An important advantage of the circumscriptive approach is that the solution to the frame problem is more general and can be applied to theories that include state constraints as well as effect axioms. However, the correctness of solutions based on circumscription is hard to assess.

  In our case since we show our circumscriptive formulation to be correct with

[13] In [Pin96], Pinto shows how to characterize within the first-order logical language preferences between action sequences based on some minimality criteria. Although, as he shows, such a characterization can be used to define particular kind of minimal plans, it is not clear how such minimizations can be used to minimize action occurrences in the actual line.

[14] This is based on the remark Pinto makes in the Conclusion section of [Pin96], where he says he would like to extend his work to study the effect of adding statements regarding the value of fluents at different points in time.

respect to the semantics of the high level language $\mathcal{L}$; it makes it easier to assess the correctness of solutions based on our approach.

12 Conclusion and Future Work

In this paper we discussed the necessity of allowing both narratives and hypothetical reasoning and presented a sound and complete translation of domain descriptions in the action description language $\mathcal{L}$ into nested abnormality theories. Our translation uses a new formulation of circumscription where values of functions are minimized. We then extended the language $\mathcal{L}$ to allow concurrent execution of actions and showed that the earlier NAT translation can be easily elaborated for this case. We continued further by adding actions with indirect effects, and discussing the relation between filtering, NATs and restricted monotonicity. Finally, we give a detailed comparison of our approach with other approaches in the literature.

In the future, we would like to continue on several fronts. In particular, we would like to

- develop algorithms that will allow us to discover missing action occurrences based on our observation. Also, we would like to explore learning of environment interaction patterns based on the past occurrences and use that information to make future plans.
- carefully extend our language to allow additional constructs such as natural events, explicit time, continuous actions, and various kinds of action occurrences beyond the simple ones that we have – such as non preventable occurrences, conditional occurrences, eventual occurrences, and triggered occurrences. One of us [BarLob96] has already participated in extending $\mathcal{L}$ to allow triggers so as to formulate active databases; but this high level formulation has not been axiomatized using circumscription.
- introduce the notion of 'current situation' and 'planning from the current situation' to Pinto and Reiter's formalism; and have separate notions of 'legal' and 'actual' situations, and allow minimization of occurrences of actions in the actual line. With these additional notions we would then like to compare their axiomatization and our circumscriptive axiomatization on domain descriptions in $\mathcal{L}$, similar to the comparison in [Kar93].
- finally, in Section 10 we discussed the relation between filtering and restricted monotonicity in general and instances of filtering and restricted monotonicity in T and T^+. We believe that each filtering step in T and T^+ also encodes abductive reasoning, i.e., abductive reasoning is done in T and T^+ through filtering. We need to further study this issue and identify all

instances of abduction in T and T^+, and discuss under what circumstances abduction can be done through filtering.

Appendix: Correctness of translation T

In proofs we will use the following notation:

$$\mathcal{M}[\![\pi]\!]$$

will stand for the set of tuples which belong to the extent of predicate π in interpretation $\mathcal{M}$. With functions we use

$$\mathcal{M}[\![\varphi]\!](\tau)$$

to denote the object which function φ maps τ into in interpretation $\mathcal{M}$.

In what follows, by a model of a single block or a sub-theory we will mean a model of a block or sub-theory plus the *Framework Axioms*.

Readers who are familiar with logic programming might have noticed that the blocks defining $Prefix\_eq$ and the other relations on sequences are very similar to the typical definition of these relations in logic programming. The following lemma, and those which are similar, are analogous to a lemma due to Marek and Subrahmanian [MarSub89] which is very useful in logic programming. Our lemmata consider only particular predicates, though.

Lemma 1 Let $\mathcal{M}$ be a model of $B_{\text{prefix\_eq}}$. Then, for all $a, \alpha, \beta,$ [15]

$$\langle \alpha, \epsilon \rangle \in \mathcal{M}[\![Prefix\_eq]\!] \Rightarrow \alpha = \epsilon \tag{23}$$

$$\begin{array}{c} [\ \alpha \neq a \circ \beta,\ \langle \alpha, a \circ \beta \rangle \in \mathcal{M}[\![Prefix\_eq]\!]\ ] \\ \Rightarrow \\ [\ \langle \alpha, \beta \rangle \in \mathcal{M}[\![Prefix\_eq]\!]] \end{array} \tag{24}$$

Proof:

[15] Recall that constant ϵ denotes the empty sequence.

Note that the axioms in $B_{\text{prefix\_eq}}$ are definite clauses,[16] so their conjunction, which we will denote by $A(Prefix\_eq)$, is a definite formula [Lif94, Sec. 3.5] in $Prefix\_eq$. By Corollary 3.5.3 from [Lif94],

$$B_{\text{prefix\_eq}} \equiv CIRC[A(Prefix\_eq); Prefix\_eq] \tag{25}$$

is equivalent to

$$(\forall \alpha, \beta).Prefix\_eq(\alpha, \beta) \equiv (\forall p).A(p) \supset p(\alpha, \beta). \tag{26}$$

Let us show that (23) holds. Assume that there is a model $\mathcal{M}$ of (25) s. t. $\mathcal{M} \models Prefix\_eq(\alpha, \epsilon)$ for some α. Suppose that $\alpha \neq \epsilon$. It is easy to see that the model $\mathcal{M}'$ differing from $\mathcal{M}$ only in that $\langle \alpha, \epsilon \rangle \notin \mathcal{M}[\![Prefix\_eq]\!]$ satisfies $A(Prefix\_eq)$ and is therefore preferable to $\mathcal{M}$. But this contradicts our assumption that $\mathcal{M}$ is a model of (25). Therefore, $\alpha = \epsilon$.

Let us show that (24) holds. Assume that there is a model $\mathcal{M}$ of (25) s. t. $\mathcal{M} \models Prefix\_eq(\alpha, a \circ \beta)$ for some sequences $\alpha \neq a \circ \beta$. Suppose that $\mathcal{M} \models \neg Prefix\_eq(\alpha, \beta)$. Consider a predicate P with extent $\mathcal{M}[\![Prefix\_eq]\!] \setminus \langle \alpha, a \circ \beta \rangle$. Clearly, $A(P)$ is satisfied while $A(P) \supset P(\alpha, a \circ \beta)$ is not. Hence our assumption is wrong and $\mathcal{M} \models Prefix\_eq(\alpha, \beta)$. Therefore, (24) is satisfied by all models of (25). $\square$

Lemma 2 Let $\mathcal{M}$ be a model of $B_{\text{prefix\_eq}}$ and α, β be sequences of actions. Then $\mathcal{M} \models Prefix\_eq(\alpha, \beta)$ iff $\alpha \preceq \beta$.

Proof:

($\Rightarrow$)

Let $\mathcal{M}$ be a model of $B_{\text{prefix\_eq}}$. We will show that for all α, β, if $\mathcal{M} \models Prefix\_eq(\alpha, \beta)$ then $\alpha \preceq \beta$ by induction on the length of β. Let $n \geq 0$ be the length of α and $m \geq 0$ be the length of β.

Base case: $m = 0$. Then, $\beta = \epsilon$. Suppose that $n > m$. Then, $\alpha \neq \epsilon$ and by (23) we get a contradiction. Thus, $n \leq m$. Since $m = 0$ and $n \geq 0$, we have that $n = 0$, i.e. $\alpha = \beta = \epsilon$. Therefore, $\alpha \preceq \beta$.

Inductive hypothesis: for all β of length $m \leq k$ and for all α, if $\mathcal{M} \models Prefix\_eq(\alpha, \beta)$ then $\alpha \preceq \beta$.

[16] A clause is said to be *definite* if exactly one of its literals is positive.

Induction: let $a \circ \beta$ be an arbitrary sequence of length $k+1$. Let us show that if $\mathcal{M} \models Prefix\_eq(\alpha, a \circ \beta)$ then $\alpha \preceq a \circ \beta$. From $\mathcal{M} \models Prefix\_eq(\alpha, a \circ \beta)$ and (24) we have that $\mathcal{M} \models Prefix\_eq(\alpha, \beta)$. By the inductive hypothesis, this implies that $\alpha \preceq \beta$. By definition of $\preceq$, $\alpha \preceq a \circ \beta$ follows from $\alpha \preceq \beta$.

$(\Leftarrow)$

Let $\mathcal{M}$ be a model of $B_{\text{prefix\_eq}}$. For all α, β, if $\alpha \preceq \beta$ then, by definition of $\preceq$, $\beta = B_n \circ \ldots \circ B_1 \circ \alpha$ for some $B_1, \ldots, B_n$ and $n \geq 0$.

Now, let us show by induction on n that $\mathcal{M} \models Prefix\_eq(\alpha, B_n \circ \ldots \circ B_1 \circ \alpha)$ for all α.

Base case: $n = 0$. We need to show $\mathcal{M} \models Prefix\_eq(\alpha, \alpha)$. But this is an axiom of $B_{\text{prefix\_eq}}$.

Induction hypothesis: for any $n \leq k$ and α, $\mathcal{M} \models Prefix\_eq(\alpha, B_n \circ \ldots \circ B_1 \circ \alpha)$.

Induction step: we need to show that $\mathcal{M} \models Prefix\_eq(\alpha, B_{k+1} \circ B_k \circ \ldots \circ B_1 \circ \alpha)$. By the induction hypothesis we have that $\mathcal{M} \models Prefix\_eq(\alpha, B_k \circ \ldots \circ B_1 \circ \alpha)$. Then, by axiom

$$(\forall \alpha, \beta, a). Prefix\_eq(\alpha, \beta) \supset Prefix\_eq(\alpha, a \circ \beta)$$

of $B_{\text{prefix\_eq}}$, we have that $\mathcal{M} \models Prefix\_eq(\alpha, B_{k+1} \circ B_k \circ \ldots \circ B_1 \circ \alpha)$. □

Lemma 3 Let $\mathcal{M}$ be a model of $B_{\text{subsequence}}$. Then for all α, β,

$$\begin{array}{c} a \neq b,\ \langle a \circ \alpha, b \circ \beta \rangle \in \mathcal{M}[\![Subsequence]\!] \\ \Rightarrow \\ \langle a \circ \alpha, \beta \rangle \in \mathcal{M}[\![Subsequence]\!] \end{array} \tag{27}$$

$$\begin{array}{c} \langle a \circ \alpha, a \circ \beta \rangle \in \mathcal{M}[\![Subsequence]\!] \\ \Rightarrow \\ \langle \alpha, \beta \rangle \in \mathcal{M}[\![Subsequence]\!] \end{array} \tag{28}$$

Proof:

Similar to the proof of Lemma 1. □

Lemma 4 Let $\mathcal{M}$ be a model of $B_{\text{subsequence}}$. Then, for all α, β, $\mathcal{M} \models Subsequence(\alpha, \beta)$ iff $\alpha \ll \beta$.

Proof:

This can be proved by induction on the lengths of α and β in a manner similar to the proof of Lemma 2. $\square$

Lemma 5 Let $\mathcal{M}$ be a model of $B_{\text{concatenate}}$. Then, for all α, β, γ,

$$[\langle \alpha, \epsilon, \gamma \rangle \in \mathcal{M}[\![Concatenate]\!]\ ] \Rightarrow \alpha = \gamma \tag{29}$$

$$\begin{gathered} [\langle \alpha, a \circ \beta, b \circ \gamma \rangle \in \mathcal{M}[\![Concatenate]\!]\ ] \\ \Rightarrow \\ [\langle \alpha, \beta, \gamma \rangle \in \mathcal{M}[\![Concatenate]\!]\ \wedge a = b] \end{gathered} \tag{30}$$

Proof: Similar to proof of Lemma 1.

$\square$

Lemma 6 Let $\mathcal{M}$ be a model of $B_{\text{concatenate}}$. Then, for all α,β,γ, $\mathcal{M} \models Concatenate(\alpha, \beta, \gamma)$ iff $\gamma = \alpha \cdot \beta$.

Proof: by induction on the length of the second argument in a manner similar to the proof of Lemma 2. $\square$

In the lemma below we will use the following notation: let A be an action and F be a fluent, formula $H^{+(-)}_{A,F}(\alpha)$ will stand for the disjunction of all the premises of axioms of the form:

$$H(p_1, \alpha)\ \wedge \ldots\ \wedge H(p_n, \alpha) \supset Causes^{+(-)}(a, f, \alpha)$$

such that $a = A$ and $f = F$. For instance, if axioms $Holds(P, \alpha) \supset Causes^{+}(A, F, \alpha)$ and $Holds(Q, \alpha) \supset Causes^{+}(A, F, \alpha)$ are all the axioms with consequent $Causes^{+}(A, F, \alpha)$, then $H^{+}_{A,F}(\alpha)$ stands for $Holds(P, \alpha) \vee Holds(Q, \alpha)$.

Lemma 7 Let D be a domain description. Block

$$
\begin{aligned}
&\{\min \; Causes^{+(-)} : \\
&\qquad H(P_1,\alpha) \wedge \ldots \wedge H(P_n,\alpha) \supset Causes^{+(-)}(A,F,\alpha) \\
&\qquad\qquad\qquad (\text{for each } A \textbf{ causes } F \textbf{ if } P_1,\ldots,P_n \in D) \\
&\}
\end{aligned}
\tag{31}
$$

is equivalent to the conjunction of formulas of the form:

$$(\forall\alpha).H^{+(-)}_{A,F}(\alpha) \equiv Causes^{+(-)}(A,F,\alpha) \tag{32}$$

one for each pair of action-fluent constants that appear in (31).

Proof: follows from Proposition 3.1.1. of [Lif94] which shows under what conditions a circumscription formula can be simplified by *predicate completion*.

□

Proposition 1

Part 1: For every causal model Ψ of D_l there exists a model $\mathcal{M}_{\rm SC}$ of $SC(D_l)$ such that for all F,α:

$$F \in \Psi(\alpha) \;\Leftrightarrow\; \mathcal{M}_{\rm SC} \models Holds(F,\alpha) \tag{33}$$

Part2: For every model $\mathcal{M}_{\rm SC}$ of $SC(D_l)$ there exists a causal model Ψ of D_l such that (33) holds.

Proof:

Part 1: Let Ψ be a causal model of D_l and let us consider an interpretation $\mathcal{M}_{\rm SC}$ of $SC(D_l)$ such that,

$$\mathcal{M}_{\rm SC}[\![Holds]\!] = \{\langle F,\alpha\rangle : \; F \in \Psi(\alpha)\}$$

and

$$\mathcal{M}_{\rm SC}[\![Causes^{+(-)}]\!] = \{\langle A,F,\alpha\rangle : \; F \in E^{+(-)}_A(\Psi(\alpha))\}.$$

We will show that $\mathcal{M}_{\mathrm{SC}}$ is a model of $SC(D_l)$. First, let us show that the blocks defining $Causes^{+(-)}$ are satisfied. By Lemma 7 it is sufficient to show that

$$H^{+(-)}_{A,F}(\alpha) \equiv Causes^{+(-)}(A, F, \alpha)$$

Suppose $H^{+(-)}_{A,F}(\alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$ for arbitrary A, F, α. Then, by construction of $\mathcal{M}_{\mathrm{SC}}$, there is a causal law A **causes** F **if** $P_1, \ldots, P_n \in D$ such that $H(P_i, \alpha)$ holds for $i = 1, \ldots, n$. Again, by construction of $\mathcal{M}_{\mathrm{SC}}$, this implies that $P_i \in \Psi(\alpha)$ for $i = 1, \ldots, n$, which in turn implies that $F \in E^{+(-)}_A(\Psi(\alpha))$. By construction of $\mathcal{M}_{\mathrm{SC}}$, $F \in E^{+(-)}_A(\Psi(\alpha))$ implies that $Causes^{+(-)}(A, F, \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$.

Suppose now that $Causes^{+(-)}(A, F, \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$ for arbitrary A, F, α. Then, by construction of $\mathcal{M}_{\mathrm{SC}}$, $F \in E^{+(-)}_A(\Psi(\alpha))$. This implies that there is a causal law A **causes** F **if** $P_1, \ldots, P_n \in D$ such that $P_i \in \Psi(\alpha)$ for $i = 1, \ldots, n$. Again, by construction of $\mathcal{M}_{\mathrm{SC}}$, it follows that $H(P_i, \alpha)$ holds for $i = 1, \ldots, n$. Therefore, $H^{+(-)}_{A,F}(\alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$.

Now, let us show that axiom

$$\neg Causes^{+}(a, f, \alpha) \ \wedge \neg Causes^{-}(a, f, \alpha) \supset [Holds(f, \alpha) \equiv Holds(f, a \circ \alpha)]$$

is satisfied by $\mathcal{M}_{\mathrm{SC}}$. Let A, F, α be an arbitrary action, fluent and sequence in D, respectively, such that LHS of the axiom holds in $\mathcal{M}_{\mathrm{SC}}$. By construction of $\mathcal{M}_{\mathrm{SC}}$, we have that $F \notin E^{+}_A(\Psi(\alpha)) \cup E^{-}_A(\Psi(\alpha))$. By definition of causal model, this implies that $F \in \Psi(\alpha)$ iff $F \in \Psi(\alpha \cdot A)$. Therefore, by construction of $\mathcal{M}_{\mathrm{SC}}$, $Holds(F, \alpha) \equiv Holds(F, A \circ \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$.

Now, consider the axioms

$$Causes^{+}(a, f, \alpha) \supset Holds(f, a \circ \alpha)$$
$$Causes^{-}(a, f, \alpha) \supset \neg Holds(f, a \circ \alpha)$$

Suppose that $Causes^{+}(A, F, \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$ for arbitrary A, F, α. Then, by construction of $\mathcal{M}_{\mathrm{SC}}$, $F \in E^{+}_A(\Psi(\alpha))$. Given our assumption that Ψ is defined on all sequences (Section 7), we have that $F \notin E^{+}_A(\Psi(\alpha)) \cap E^{-}_A(\Psi(\alpha))$, thus by definition of Ψ, $F \in \Psi(\alpha \cdot A)$. Therefore, by construction of $\mathcal{M}_{\mathrm{SC}}$, $Holds(F, A \circ \alpha)$ holds.

A similar argument holds for the second axiom.

Part 2: Let $\mathcal{M}_{\mathrm{SC}}$ be a model of $SC(D_l)$. Let consider a transition function Ψ that satisfies (33) holds. We show Ψ to be a causal model of D_l, i.e., for each A, α

$$\Psi(\alpha \cdot A) = \Psi(\alpha) \cup E_A^+(\Psi(\alpha)) \setminus E_A^-(\Psi(\alpha))$$

We show this is indeed the case by induction on the length of the sequence.

Base Case: The length of α is 0. We need to show that for any A,

$$\Psi([A]) = \Psi([\,]) \cup E_A^+(\Psi([\,])) \setminus E_A^-(\Psi([\,]))$$

where $F \in \Psi([\,])$ iff $Holds(F, \epsilon)$ by construction of Ψ.

a) $\Psi([A]) \supseteq \Psi([\,]) \cup E_A^+(\Psi([\,])) \setminus E_A^-(\Psi([\,]))$.

Let F and A be an arbitrary fluent and action in D resp., such that $F \in \Psi([\,]) \cup E_A^+(\Psi([\,])) \setminus E_A^-(\Psi([\,]))$. Let us show that $F \in \Psi([A])$. Clearly, $F \notin E_A^-(\Psi([\,]))$, hence there is no causal law A **causes** $\neg F$ **if** $Q_1, \ldots, Q_m \in D_l$ such that the preconditions hold in $\Psi([\,])$. By construction of Ψ this means that $\mathcal{M}_{\mathrm{SC}} \models H(Q_i, \epsilon)$ does not hold for some $1 \leq i \leq m$, for all such causal laws. By Lemma 7, this implies that

$$\mathcal{M}_{\mathrm{SC}} \models \neg Causes^-(A, F, \epsilon). \tag{34}$$

There are two cases of interest

i) $F \in E_A^+(\Psi([\,]))$
ii) $F \notin E_A^+(\Psi([\,]))$

If (i) is the case, then there must be a causal law A **causes** F **if** $P_1, \ldots, P_n \in D_l$ such that the preconditions hold in $\Psi([\,])$. By construction of Ψ, this implies that $\mathcal{M}_{\mathrm{SC}} \models H(P_i, \epsilon)$ for each $i = 1, \ldots, n$. Therefore, $\mathcal{M}_{\mathrm{SC}} \models Causes^+(A, F, \epsilon)$ and by axiom $Causes^+(a, f, \alpha) \supset Holds(f, a \circ \alpha)$ we have that $\mathcal{M}_{\mathrm{SC}} \models Holds(F, A \circ \epsilon)$. Thus, by construction of Ψ, $F \in \Psi([A])$.

If (ii) is the case, then $F \in \Psi([\,])$ which implies, by construction of Ψ, that $\mathcal{M}_{\mathrm{SC}} \models Holds(F, \epsilon)$. Furthermore, $F \notin E_A^+(\Psi([\,]))$ implies there is no causal law A **causes** F **if** $P_1, \ldots, P_n \in D_l$ such that the preconditions hold in $\Psi([\,])$. Hence, $\mathcal{M}_{\mathrm{SC}} \models H(P_i, \epsilon)$ does not hold for some $1 \leq i \leq n$, for all causal laws of this form and $\mathcal{M}_{\mathrm{SC}} \models \neg Causes^+(A, F, \epsilon)$. From this, (34) and axiom

$$\neg Causes^{+}(a,f,\alpha) \ \wedge \neg Causes^{-}(a,f,\alpha) \supset [Holds(f,\alpha) \equiv Holds(f, a \circ \alpha)]$$

we conclude that $\mathcal{M}_{\mathrm{SC}} \models Holds(F, A \circ \epsilon)$, and by construction of Ψ that $F \in \Psi([A])$.

b)$\Psi([A]) \subseteq \Psi([\,]) \cup E^{+}_{A}(\Psi([\,])) \setminus E^{-}_{A}(\Psi([\,]))$.

Let F and A be an arbitrary fluent and action in D resp., such that $F \in \Psi([A])$. By construction of Ψ, $\mathcal{M}_{\mathrm{SC}} \models Holds(F, A \circ \epsilon)$. Thus, $\mathcal{M}_{\mathrm{SC}} \models \neg Causes^{-}(A, F, \epsilon)$, and by Lemma 7, for every causal law A **causes** $\neg F$ **if** $Q_1, \ldots, Q_m \in D_l$ we have $\mathcal{M}_{\mathrm{SC}} \models H(Q_i, \epsilon)$ does not hold for some $1 \leq i \leq m$. By construction of Ψ, for every causal law its preconditions do not hold in $\Psi([\,])$ either. Thus, $F \notin E^{-}_{A}(\Psi([\,]))$. We again have two cases

i) $\mathcal{M}_{\mathrm{SC}} \models Causes^{+}(A, F, \epsilon)$
ii) $\mathcal{M}_{\mathrm{SC}} \models \neg Causes^{+}(A, F, \epsilon)$

If (i) is the case, then by Lemma 7 there is a causal law A **causes** F **if** $P_1, \ldots, P_n \in D_l$ such that $\mathcal{M}_{\mathrm{SC}} \models H(P_i, \epsilon)$ for each $i = 1, \ldots, n$. By construction of Ψ, all the preconditions of such a causal law hold in $\Psi([\,])$. Therefore, $F \in E^{+}_{A}(\Psi([\,]))$, and since we assume that Ψ is defined on every sequence (Section 7), $F \notin E^{-}_{A}(\Psi([\,]))$. Thus, $F \in \Psi([\,]) \cup E^{+}_{A}(\Psi([\,])) \setminus E^{-}_{A}(\Psi([\,]))$.

If (ii) is the case then by virtue of the first axiom of $SC(D_l)$, $\mathcal{M}_{SC} \models Holds(F, \epsilon)$. This implies, by construction of Ψ, that $F \in \Psi([\,])$. We showed above that $F \notin E^{-}_{A}(\Psi([\,]))$. Therefore, $F \in \Psi([\,]) \cup E^{+}_{A}(\Psi([\,])) \setminus E^{-}_{A}(\Psi([\,]))$.

Induction Hypothesis: For each A, α

$$\Psi(\alpha \cdot A) = \Psi(\alpha) \cup E^{+}_{A}(\Psi(\alpha)) \setminus E^{-}_{A}(\Psi(\alpha))$$

Inductive case: it can be proved by following the same steps as in the proof of the base case. □

Proposition 2 For any domain description D, if interpretation (Ψ, Σ) is a model of D then there exists a model $\mathcal{M}$ of $T(D)$ such that for every fact ϕ in the language of D:

$$(\Psi, \Sigma) \models \phi \ \Leftrightarrow \ \mathcal{M} \models \tau(\phi) \tag{35}$$

and if $\mathcal{M}$ is a model of $T(D)$ then there exists a model (Ψ, Σ) of D such that (35) holds.

Proof:

$(\Rightarrow)$

Let $M = (\Psi, \Sigma)$ be a model of D. Let $\mathcal{M}_M$ be an interpretation of $T(D)$ such that

i) The universe of *actions*, *fluents*, and *situations* consist of the symbols in the sets $\mathcal{A}$, $\mathcal{F}$ and $\mathcal{S}$ of D respectively. The universe of *sequences* consists of a unique object for each possible sequence of actions from $\mathcal{A}$.
ii) $Holds$ and $Causes^{+(-)}$ are interpreted as follows:

$$\mathcal{M}_M[\![Holds]\!] = \{\langle F, \alpha\rangle : F \in \Psi(\alpha)\}$$

$$\mathcal{M}_M[\![Causes^{+(-)}]\!] = \{\langle A, F, \alpha\rangle : F \in E_A^{+(-)}(\alpha)\}$$

iii) all action, fluent and situation constants are interpreted as themselves. Predicates $Prefix\_eq$, $Subsequence$ and $Concatenate$ are interpreted as the corresponding intended relation.
iv) Sit_map is interpreted as follows: for each S in D, $\mathcal{M}_M[\![Sit\_map]\!](S) = A_m \circ \ldots \circ A_1 \circ \epsilon$ if $\Sigma(S) = [A_1, \ldots, A_m]$.

We will show that $\mathcal{M}_M$ is a model of $T(D)$.

First, note that the framework axioms are trivially satisfied. The blocks defining prefix, subsequence and concatenate have already been proven to correctly capture the intended relations.

Axioms (a) and (b) are satisfied by definition of Σ, condition (iii) and correctness of B_{prefix_eq} (Lemma 2).

Note that condition (ii) on $Holds$ and $Causes^{+(-)}$ is the same condition used to show that such an interpretation of these predicates is a model of $SC(D_l)$ (See the proof of Proposition 1), thus axioms $SC(D_l)$ are satisfied.

It remains to be shown that $\mathcal{M}_M$ satisfies axioms $\tau(D)$ and is a minimal model w.r.t. the circumscription policy.

F **at** $S \in D$: $\tau(D)$ includes the axiom $Holds(F, Sit\_map(S))$.

By the same fact from D, $F \in \Psi(\Sigma(S))$. By cond. (iv) on Sit_map and cond. (ii), $F \in \Psi(\Sigma(S))$ implies that $\mathcal{M}_M \models Holds(F, \alpha)$, where α is the same sequence as $\Sigma(S)$.

α **occurs_at** $S \in D$: $\tau(D)$ includes the axiom

$$(\exists\beta).Concatenate(Sit\_map(S), \alpha, \beta) \;\wedge Prefix\_eq(\beta, Sit\_map(S_N))$$

By the same fact from D, α concatenated with $\Sigma(S)$ is a prefix of $\Sigma(S_N)$. By cond. (iv) on Sit_map, $Sit\_map(S)$ and $Sit\_map(S_N)$ are the same sequences as $\Sigma(S)$ and $\Sigma(S_N)$. By correctness of $B_{\text{concatenate}}$ and $B_{\text{prefix\_eq}}$ the axiom is satisfied.

S_1 **precedes** $S_2 \in D$: $\tau(D)$ includes the axiom

$$Prefix\_eq(Sit\_map(S_1), Sit\_map(S_2) \;\wedge \neg Prefix\_eq(Sit\_map(S_2), Sit\_map(S_1))$$

By the same fact from D, $\Sigma(S_1)$ is a proper prefix of $\Sigma(S_2)$. By condition (iv) and correctness of $B_{\text{prefix\_eq}}$, the axiom is satisfied.

Finally, the value minimization axiom (c) captures exactly the minimality condition (See Section 5 for a detailed discussion), with respect to ordering *subsequence*, imposed on $\Sigma(S_N)$ by the definition of models of domain descriptions in $\mathcal{L}$.

($\Leftarrow$)

Let $\mathcal{M}$ be a model of $T(D)$. By Proposition 1 there is a causal model Ψ such that for all F, α,

$$\Psi(\alpha) = \{F : \; \mathcal{M} \models Holds(F, \alpha)\} \tag{36}$$

Let Ψ be such a causal model and let Σ be a situation assignment such that for each S,

$$\Sigma(S) = [A_1, \ldots, A_n] \;\Leftrightarrow\; \mathcal{M}[\![Sit\_map]\!](S) = A_n \circ \ldots \circ A_1 \circ \epsilon \tag{37}$$

Let us show $M = (\Psi, \Sigma)$ is a model of D.

First of all, by axiom

$$Sit\_map(S_0) = \epsilon$$

we have $\Sigma(S_0) = [\,]$, and by axiom

$$Prefix\_eq(Sit\_map(s), Sit\_map(S_N))$$

we have that, for all situations S in the language of D, $\Sigma(S)$ is a prefix of $\Sigma(S_N)$; thus, Σ is a situation assignment. Since we assume that Ψ is defined on all sequences, $\Psi(\Sigma(S_N))$ is defined and therefore Ψ and Σ form an interpretation of D.

It only remains to be shown that facts in D are true in M and that there is no $N' = (\Psi, \Sigma')$ such that $\Sigma'(S_N) \ll \Sigma(S_N)$ and N' is a model of D.

(F **at** S) $\in D$.
Then, $\mathcal{M} \models Holds(F, Sit\_map(S))$. By (37), $Sit\_map(S)$ in $\mathcal{M}$ is the same sequence as $\Sigma(S)$, thus, by (36), $F \in \Psi(\Sigma(S))$. Therefore, (F **at** S) is true in M. The same argument holds for negated fluent facts.

(α **occurs_at** S) $\in D$.
Then, $\mathcal{M} \models (Concatenate(Sit\_map(S), \alpha, \beta) \wedge Prefix\_eq(\beta, Sit\_map(S_N))$, for some β. Let α_S and α_N be the same sequences as $Sit\_map(S)$ and $Sit\_map(S_N)$ in $\mathcal{M}$. By correctness of $B_{\text{concatenate}}$ (Lemma 6), β is the same sequence as α_S concatenated with α. By correctness of $B_{\text{prefix\_eq}}$ (Lemma 2), β is a prefix of sequence α_N. By condition (37) on Σ, β is equal to $\Sigma(S)$ concatenated with α and β is a prefix of $\Sigma(S_N)$. Therefore, (α **occurs_at** S) is true in M.

(S_1 **precedes** S_2) $\in D$: Then,

$$\mathcal{M} \models Prefix\_eq(Sit\_map(S_1), Sit\_map(S_2)) \wedge \\ \neg Prefix\_eq(Sit\_map(S_2), Sit\_map(S_1))$$

By correctness of $B_{\text{prefix\_eq}}$ and (37), $\Sigma(S_1)$ is a proper prefix of $\Sigma(S_2)$. Therefore, (S_1 **precedes** S_2) is true in M.

Finally, we need to show there is no interpretation $N' = (\Psi, \Sigma')$ of D such that $\Sigma'(S_N)$ is a subsequence (not equal) of $\Sigma(S_N)$ and N' is a model of D. Assume that there is such a model N'. Then, by part ($\Rightarrow$) of this lemma, there would be a model $\mathcal{M}'$ of $T(D)$ which differs from $\mathcal{M}$ only on the interpretation of function Sit_map and such that sequence $\mathcal{M}'[\![Sit\_map]\!](S_N)$ is subsequence (not equal) of $\mathcal{M}[\![Sit\_map]\!](S_N)$. This contradicts our assumption that $\mathcal{M}$ is a model of $T(D)$ since $T(D)$ includes value minimization of $Sit\_map(S_N)$ with respect to ordering subsequence (See Section 5). $\square$

Proposition 4 For any domain description D and hypothesis ϕ in the language of D:

$$D \models_{\mathcal{L}} \phi \;\Leftrightarrow\; T(D) \models \tau(\phi)$$

Proof (sketch):

Let F **after** α be an arbitrary hypothesis in the language of D. We need to show that for all models (Ψ, Σ) of D, $F \in \Psi(\Sigma(S_N) \cdot \alpha)$ iff for all models $\mathcal{M}$ of $T(D)$, $(\exists\beta).Concatenate(Sit\_map(S_N), \alpha, \beta) \supset H(F, \beta)$.

This can be proved by following the proof of Proposition 2 where we show how to construct a model $\mathcal{M}$ of $T(D)$ from a model (Ψ, Σ) of D, and vice-versa, such that the following holds: for all F, α

$$F \in \Psi(\alpha) \;\Leftrightarrow\; \mathcal{M} \models Holds(F, \alpha),$$

$$\Sigma(S_N) = \alpha \;\Leftrightarrow\; \mathcal{M} \models (Sit\_map(S_N) = \alpha)$$

$\square$

Appendix: Correctness of translation T_c

Lemma 8 Let $\mathcal{M}$ be a model of B_{embedded}. Then, for all a_1, a_2,α and β,

$$\begin{aligned} &\langle a_1 \circ \epsilon, a_2 \circ \epsilon\rangle \in \mathcal{M}[\![Embedded]\!] \\ &\qquad\Rightarrow \\ &(\forall a).a \in a_1 \Rightarrow a \in a_2 \end{aligned} \tag{38}$$

$$\begin{gathered} \langle a_1 \circ \beta, a_2 \circ \alpha\rangle \in \mathcal{M}[\![Embedded]\!] \\ \Rightarrow \\ [\langle \beta, \alpha\rangle \in \mathcal{M}[\![Embedded]\!],\ \langle a_1 \circ \epsilon, a_2 \circ \epsilon\rangle \in \mathcal{M}[\![Embedded]\!]] \end{gathered} \tag{39}$$

Proof:

Similar to the proof of Lemma 1. $\square$

Lemma 9 Let $\mathcal{M}$ be a model of B_{embedded}. Then, for all α, β, $\mathcal{M} \models Embedded(\alpha, \beta)$ iff $\alpha \subseteq \beta$.

Proof:

This can be proved by induction on the length of the sequences and Lemma 8. □

Lemma 10 Let $\mathcal{M}$ be a model of $B_{\text{embsubseq}}$. Then, for all α, β,

$$\begin{aligned} \langle \alpha, \beta \rangle \in \mathcal{M}[\![Embsubseq]\!] \\ \Rightarrow \\ (\exists \gamma).\langle \alpha, \gamma \rangle \in \mathcal{M}[\![Embedded]\!] \wedge \langle \gamma, \beta \rangle \in \mathcal{M}[\![Subsequence]\!] \end{aligned} \tag{40}$$

Proof:

Similar to the proof of Lemma 1. □

Lemma 11 Let $\mathcal{M}$ be a model of $B_{\text{embsubseq}}$. Then, for all α, β, $\mathcal{M} \models Embsubseq(\alpha, \beta)$ iff α is an embedded-subsequence in β.

Proof:

($\Rightarrow$)

Let $\mathcal{M}$ be a model of $B_{\text{embsubseq}}$ and let α, β be sequences such that $\mathcal{M} \models Embsubseq(\alpha, \beta)$. Then, by (40) we have $\mathcal{M} \models Embedded(\alpha, \gamma) \wedge Subsequence(\gamma, \beta)$ for some sequence γ. By correctness of B_{embedded} (Lemma 9) we have that α is embedded in γ, and by correctness of $B_{\text{subsequence}}$ (Lemma 4) that γ is a subsequence of β. Therefore, α is an embedded-subsequence of β.

($\Leftarrow$)

Let $\mathcal{M}$ be a model of $B_{\text{embsubseq}}$ and let α, β be sequences such that α is an embedded-subsequence of β. Then, by definition of embedded-subsequence, there is a sequence γ such that γ is a subsequence of β and α is embedded in γ. By correctness of $B_{\text{subsequence}}$ (Lemma 4) we have that $\mathcal{M} \models Subsequence(\gamma, \beta)$ and by correctness of B_{embedded} (Lemma 9) that $\mathcal{M} \models Embedded(\alpha, \gamma)$. Therefore, by axiom

$$Embedded(\alpha, \beta) \wedge Subsequence(\beta, \gamma) \supset Embsubseq(\alpha, \gamma)$$

of $B_{\text{embsubseq}}$, $\mathcal{M} \models Embsubseq(\alpha, \beta)$. $\square$

Lemma 12 The block which defines $Inherits^{+(-)}$:

$$
\begin{aligned}
&\{\min \;\; Inherits^{+(-)} \;:\\
&\qquad b \in a \;\wedge Causes^{+(-)}(b, f, \alpha) \supset Inherits^{+(-)}(a, f, \alpha)\\
&\}
\end{aligned}
$$

is equivalent to the following formula

$$(\forall a, f, \alpha).[(\exists b).b \in a \;\wedge Causes^{+(-)}(b, f, \alpha)] \equiv Inherits^{+(-)}(a, f, \alpha) \quad (41)$$

Proof:

It follows from Proposition 3.1.1. of [Lif94] which shows under what conditions a circumscription formula can be simplified by *predicate completion.* $\square$

Lemma 13 Let $\mathcal{M}$ be a model of

$$
\begin{aligned}
&\{\min \;\; Undef \;:\\
&\qquad Inherits^{+}(a, f, \alpha) \;\wedge Inherits^{-}(a, f, \alpha) \supset Undef(a \circ \alpha)\\
&\qquad Undef(\alpha) \supset Undef(a \circ \alpha)\\
&\}
\end{aligned}
\quad (42)
$$

Then, for all a, α,

$$
\begin{gathered}
\langle a \circ \alpha \rangle \in \mathcal{M}[\![Undef]\!]\\
\Rightarrow\\
[\; ((\exists f).\langle a, f, \alpha \rangle \in \mathcal{M}[\![Inherits^{+}]\!] \;\wedge \langle a, f, \alpha \rangle \in \mathcal{M}[\![Inherits^{-}]\!]) \vee\\
\langle \alpha \rangle \in \mathcal{M}[\![Undef]\!] \;]
\end{gathered}
\quad (43)
$$

Proof:

Similar to proof of Lemma 1. $\square$

Proposition 5

Part 1: For every causal model Ψ of D_l there exists a model $\mathcal{M}_{\mathrm{SC}}$ of $SC_c(D_l)$ such that for all F, α,

$$\begin{aligned} &F \in \Psi(\alpha) \;\Leftrightarrow\; \mathcal{M}_{\mathrm{SC}} \models Holds(F, \alpha), \text{ and} \\ &\Psi(\alpha) \text{ is undefined } \;\Leftrightarrow\; \mathcal{M}_{\mathrm{SC}} \models Undef(\alpha) \end{aligned} \tag{44}$$

Part2: For every model $\mathcal{M}_{\mathrm{SC}}$ of $SC_c(D_l)$ there exists a causal model Ψ of D_l such that (44) holds.

Proof:

Part 1: Let Ψ be a causal model of D_l. Let $\mathcal{M}_{\mathrm{SC}}$ be an interpretation of $SC_c(D_l)$ such that

i) $\mathcal{M}_{\mathrm{SC}}[\![Undef]\!] = \{\alpha : \ \Psi(\alpha) \text{ is undefined}\}$,
ii) $\mathcal{M}_{\mathrm{SC}}[\![Causes^{+(-)}]\!] = \{\langle A, F, \alpha\rangle : \ F \text{ is an immediate effect of } A \text{ in } \Psi(\alpha)\}$,
iii) $\mathcal{M}_{\mathrm{SC}}[\![Inherits^{+(-)}]\!] = \{\langle A, F, \alpha\rangle : \ F \in E_A^{+(-)}(\Psi(\alpha))\}$ [17]
iv) $\mathcal{M}_{\mathrm{SC}}[\![Holds]\!] = \{\langle F, \alpha\rangle : \ F \in \Psi(\alpha)\}$.

Note that (i) and (iv) correspond to condition (44). We will show $\mathcal{M}_{\mathrm{SC}}$ is a model of $SC_c(D_l)$.

We showed in Proposition 1 that a model built as above satisfies the blocks which define predicates $Causes^+$ and $Causes^-$.

Given the definition of $E_A^{+(-)}$, it is clear that condition (iii) corresponds exactly to formula (41), therefore the blocks defining predicates $Inherits^+$ and $Inherits^-$ are satisfied.

Let us show the block defining $Undef$ is satisfied. Assume that there are A, F, α such that $Inherits^+(A, F, \alpha) \wedge Inherits^-(A, F, \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$. Then by cond. (iii) we have that $F \in E_A^+(\alpha) \cap E_A^-(\alpha)$ and thus this intersection is not empty. This implies that $\Psi(\alpha)$ is undefined. By this and cond. (i) we have that $\mathcal{M} \models Undef(A \circ \alpha)$. Therefore the first axiom in the block is satisfied by $\mathcal{M}$. Now, suppose $\Psi(\alpha)$ is undefined for some α. Then, since Ψ is prefix closed, for every action A, $\Psi(\alpha \cdot A)$ is also undefined and the second axiom is satisfied. It is easy to see that the extent of $Undef$ is not minimal only if there exist A, α such that $Undef(A \circ \alpha)$ holds while there is no F such that $Inherits^+(A, F, \alpha) \ \wedge Inherits^-(A, F, \alpha)$ holds, nor $Undef(\alpha)$ holds in

[17] Recall that the meaning of $E_A^{+(-)}(\sigma)$ is now different from that in previous proofs. It stands for the set of fluents that become true (false) after executing a concurrent action A.

$\mathcal{M}$. This however contradicts Lemma 13, thus the extent of $Undef$ must be minimal and therefore this block is satisfied by $\mathcal{M}$.

It is easy to see that the axioms

$$\neg Inherits^{+}(a,f,\alpha)\ \wedge \neg Inherits^{-}(a,f,\alpha) \supset [Holds(f,\alpha) \equiv Holds(f,a\circ\alpha)]$$
$$\neg Undef(a\circ\alpha)\ \wedge Inherits^{+}(a,f,\alpha) \supset Holds(f,a\circ\alpha)$$
$$\neg Undef(a\circ\alpha)\ \wedge Inherits^{-}(a,f,\alpha) \supset \neg Holds(f,a\circ\alpha)$$

of $SC_c(D_l)$ are satisfied by $\mathcal{M}_{\mathrm{SC}}$.

Part 2: Let $\mathcal{M}_{\mathrm{SC}}$ be a model of $SC_c(D_l)$. Let Ψ be a transition function such that (44) holds. We show Ψ to be a causal model of D_l.

Consider the axioms in the block for $Undef$. Note that the consequences of both axioms necessarily involve a non-empty sequence. Hence for any interpretation of $Undef$ that contains ϵ and satisfies the axioms we can obtain another interpretation that satisfies the axioms and is a subset. Since the block chooses a minimal interpretation of $Undef$, $Undef(\epsilon)$ does not hold in $\mathcal{M}_{\mathrm{SC}}$. Thus, $\Psi([\,])$ is defined.

Let us show that Ψ is prefix-closed. Assume the contrary and let α and β be sequences such that α is a prefix of β and $\Psi(\beta)$ is defined while $\Psi(\alpha)$ is undefined. By construction of Ψ, it follows that $Undef(\alpha)$ and $\neg Undef(\beta)$ hold in $\mathcal{M}_{\mathrm{SC}}$. However, from axiom

$$(\forall\alpha,a).Undef(\alpha) \supset Undef(a\circ\alpha)$$

and the fact that α is a prefix of β, we conclude that $Undef(\beta)$ holds in $\mathcal{M}_{\mathrm{SC}}$, which contradicts our assumption. Therefore, Ψ is prefix-closed.

Now, let us show that for all A,α, if $E^{+}_{A}(\alpha)\cap E^{-}_{A}(\alpha) \neq \emptyset$ then Ψ is undefined on sequence $A\circ\alpha$.

Suppose $F\in E^{+}_{A}(\alpha)$ and $F\in E^{-}_{A}(\alpha)$ for fluent F, action A and sequence α. This means that F and $\neg F$ are effects of A in $\Psi(\alpha)$, i.e., there exist causal laws

$$\begin{array}{l} B \ \textbf{causes} \ \ F \ \textbf{if} \ \ P_1,\ldots,P_k \\ B' \ \textbf{causes} \ \ \neg F \ \textbf{if} \ \ Q_1,\ldots,Q_l \end{array} \tag{45}$$

in D_l, where $B \in A$ and $B' \in A$ and such that P_i $(i = 1, \ldots, k)$ and Q_i $(i = 1, \ldots, l)$ are true in $\Psi(\alpha)$. From this and construction of Ψ it follows that $\mathcal{M}_{\mathrm{SC}} \models H(P_i, \alpha)$ $(i = 1, \ldots, k)$ and $\mathcal{M}_{\mathrm{SC}} \models H(Q_i, \alpha)$ $(i = 1, \ldots, l)$, for any pair of causal laws of the form (45). This implies that $Causes^{+}(B, F, \alpha)$ and $Causes^{-}(B', F, \alpha)$ hold in $\mathcal{M}_{\mathrm{SC}}$. By axiom

$$b \in a \ \wedge Causes^{+(-)}(b, f, \alpha) \supset Inherits^{+(-)}(a, f, \alpha)$$

we have that $Inherits^{+}(A, F, \alpha)$ and $Inherits^{-}(A, F, \alpha)$ hold in $\mathcal{M}_{\mathrm{SC}}$, and by axiom

$$Inherits^{+}(a, f, \alpha) \ \wedge Inherits^{-}(a, f, \alpha) \supset Undef(a \circ \alpha)$$

we have that $Undef(A \circ \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$. Therefore, by construction of Ψ, $\Psi(\alpha \cdot A)$ is undefined.

Now, let us show that for all A, α, if $E_A^{+}(\alpha) \cap E_A^{-}(\alpha) = \emptyset$, then $\Psi(\alpha \cdot A) = \Psi(\alpha) \cup E_A^{+}(\alpha) \setminus E_A^{-}(\alpha)$. This proof is very similar to that of Proposition 1. We will prove the base case only.

Base Case: Length of α is 0. We will show

$$\Psi([A]) = \Psi([\,]) \cup E_A^{+}(\Psi([\,])) \setminus E_A^{-}(\Psi([\,]))$$

a) $\Psi([A]) \supseteq \Psi([\,]) \cup E_A^{+}(\Psi([\,])) \setminus E_A^{-}(\Psi([\,]))$.

Assume that $F \in \Psi([\,]) \cup E_A^{+}(\Psi([\,])) \setminus E_A^{-}(\Psi([\,]))$. Let us show $F \in \Psi([A])$. Clearly, $F \notin E_A^{-}(\Psi([\,]))$. This implies that there does not exist an atomic action B in A such that $\neg F$ is an immediate effect of B. Again, this implies that $\mathcal{M}_{\mathrm{SC}} \models \neg Causes^{-}(B, F, \epsilon)$. And since this holds for all b such that $b \in A$, $\mathcal{M}_{\mathrm{SC}} \models \neg Inherits^{-}(A, F, \epsilon)$.

Now, there are two cases of interest:

i) $F \in E_A^{+}(\Psi([\,]))$
ii) $F \notin E_A^{+}(\Psi([\,]))$

Consider case (i). There must be an action B in A such that

$$B \ \textbf{causes} \ \ F \ \textbf{if} \ \ P_1, \ldots, P_n \in D_l$$

and the preconditions hold in $\Psi([\,])$. As shown in the first part of this proof, this implies that $\mathcal{M}_{\mathrm{SC}} \models Causes^{+}(B, F, \epsilon)$ and by axiom

$$b \in a \ \wedge Causes^{+}(b, f, \alpha) \supset Inherits^{+}(a, f, \alpha)$$

$\mathcal{M}_{\mathrm{SC}} \models Inherits^{+}(A, F, \epsilon)$. Since we are considering the case when $\neg Undef(A \circ \alpha)$ holds in $\mathcal{M}_{\mathrm{SC}}$ for all α, by axiom

$$\neg Undef(a \circ \alpha) \ \wedge Inherits^{+}(a, f, \alpha) \supset Holds(f, a \circ \alpha)$$

we get that $\mathcal{M}_{\mathrm{SC}} \models Holds(F, A \circ \epsilon)$, which by construction of Ψ implies that $F \in \Psi([A])$.

Now consider case (ii). Since $F \notin E_A^{+}(\Psi([\,]))$, we have $F \in \Psi([\,])$, which by construction of Ψ implies that $\mathcal{M}_{\mathrm{SC}} \models Holds(F, \epsilon)$. We showed above that from $F \notin E_A^{-}(\Psi([\,]))$ it follows that $\mathcal{M}_{\mathrm{SC}} \models \neg Inherits^{-}(A, F, \epsilon)$. The same can be shown for positive effects, i.e., that from $F \notin E_A^{+}(\Psi([\,]))$ it follows that $\mathcal{M}_{\mathrm{SC}} \models \neg Inherits^{+}(A, F, \epsilon)$.

Thus, we have that $Holds(F, \epsilon)$, $\neg Inherits^{+}(A, F, \epsilon)$, and $\neg Inherits^{-}(A, F, \epsilon)$ hold in $\mathcal{M}_{\mathrm{SC}}$. Then, by the axiom

$$\neg Inherits^{+}(a, f, \alpha) \ \wedge \neg Inherits^{-}(a, f, \alpha) \supset [Holds(f, \alpha) \equiv Holds(f, a \circ \alpha)]$$

it follows that $\mathcal{M}_{\mathrm{SC}} \models Holds(F, A \circ \epsilon)$ and –by construction of Ψ– $F \in \Psi([a])$.

b) $\Psi([a]) \subseteq \Psi([\,]) \cup E_a^{+}(\Psi([\,])) \setminus E_a^{-}(\Psi([\,]))$.

Let $F \in \Psi([A])$. Let us show that $F \in \Psi([\,]) \cup E_A^{+}(\Psi([\,])) \setminus E_A^{-}(\Psi([\,]))$. By construction of Ψ, $\mathcal{M}_{\mathrm{SC}} \models Holds(F, A \circ \epsilon)$. Since we are considering the case when $\mathcal{M}_{\mathrm{SC}} \models \neg Undef(A \circ \epsilon)$, $\mathcal{M}_{\mathrm{SC}} \models \neg Inherits^{-}(A, F, \epsilon)$ would imply, by the axiom

$$\neg Undef(a \circ \alpha) \ \wedge Inherits^{-}(a, f, \alpha) \supset \neg Holds(f, a \circ \alpha)$$

that $\mathcal{M}_{\mathrm{SC}} \models \neg Holds(F, A \circ \epsilon)$. Therefore, $\mathcal{M}_{\mathrm{SC}} \models \neg Inherits^{-}(A, F, \epsilon)$. Then, by Lemma 12, we have that for every $B \in A$ $\mathcal{M}_{\mathrm{SC}} \models \neg Causes^{-}(B, F, \epsilon)$. Consequently, for every $B \in A$ such that B **causes** $\neg F$ **if** $Q_1, \ldots, Q_m \in D_l$, the preconditions do not hold in $\Psi([\,])$. From this it follows that $F \notin E_A^{-}(\Psi([\,]))$. Now, there are two cases of interest:

i) $\mathcal{M}_{\mathrm{SC}} \models Inherits^{+}(A, F, \epsilon)$

ii) $\mathcal{M}_{\text{SC}} \models \neg Inherits^{+}(A, F, \epsilon)$

Consider case (i). By Lemma 12 we have that there exists an action $B \in A$ such that $\mathcal{M}_{\text{SC}} \models Causes^{+}(B, F, \epsilon)$ and by Lemma 7 that there is a causal law B **causes** F **if** $P_1, \ldots, P_n \in D_l$ such that $\mathcal{M}_{\text{SC}} \models H(P_i, \epsilon)$ for $i = 1, \ldots n$. By construction of Ψ, all P_i must hold in $\Psi([\,])$. Therefore, $F \in E_A^{+}(\Psi([\,]))$. From this and $F \notin E_A^{-}(\Psi([\,]))$, shown above, we conclude that $F \in \Psi([\,]) \cup E_A^{+}(\Psi([\,])) \setminus E_A^{-}(\Psi([\,]))$.

Now consider case (ii). By virtue of the first axiom of $SC_c(D_l)$, $Holds(F, \epsilon)$. By construction of Ψ, $F \in \Psi([\,])$. Since $F \notin E_A^{-}(\Psi([\,]))$, shown above, $F \in \Psi([\,]) \cup E_A^{+}(\Psi([\,])) \setminus E_A^{-}(\Psi([\,]))$. $\square$

Proposition 6

For any domain description D, if interpretation (Ψ, Σ) is a model of D then there exists a model $\mathcal{M}$ of $T_c(D)$ such that for every fact ϕ in the language of D:

$$(\Psi, \Sigma) \models_{\mathcal{L}_c} \phi \;\Leftrightarrow\; \mathcal{M} \models \tau(\phi) \tag{46}$$

and if $\mathcal{M}$ is a model of $T_c(D)$ then there exists a model (Ψ, Σ) of D such that (46) holds.

Proof:

$(\Rightarrow)$

Let $M = (\Psi, \Sigma)$ be a model of D. Let $\mathcal{M}_M$ be an interpretation of $T_c(D)$ such that:

i) The universe of *fluents* and *situations* consist of the symbols in the sets $\mathcal{F}$ and $\mathcal{S}$ of D respectively. The universe of *c-actions* is $2^{\mathcal{A}} - \emptyset$. The universe of *sequences* consists of a unique object for each possible sequence of *c-actions*.
ii) $\mathcal{M}_M[\![Undef]\!] = \{\alpha : \Psi(\alpha) \text{ is undefined}\}$
iii) $\mathcal{M}_M[\![Causes^{+(-)}]\!] = \{\langle A, F, \alpha\rangle : F \text{ is an immediate effect of } A \text{ in } \Psi(\alpha)\}$,
iv) $\mathcal{M}_M[\![Inherits^{+(-)}]\!] = \{\langle A, F, \alpha\rangle : F \in E_A^{+(-)}(\Psi(\alpha))\}$,
v) $\mathcal{M}_M[\![Holds]\!] = \{\langle A, F, \alpha\rangle : F \in \Psi(\alpha)\}$,
vi) all action, fluent and situation constants are interpreted as their corresponding universe element. Predicates $Prefix\_eq$, $Subsequence$, $Concatenate$, $Embedded$ and $Embsubseq$ are interpreted as the intended relation.

vii) for each S, $\mathcal{M}_M[\![Sit\_map]\!](S) = A_m \circ \ldots \circ A_1 \circ \epsilon$ if $\Sigma(s) = [A_1, \ldots, A_m]$.

We will show that $\mathcal{M}_M$ is a model of $T_c(D)$.

The blocks defining relations on sequences ($Prefix\_eq$, etc.) have been shown to capture the intended relations. The framework axioms are trivially satisfied.

By definition of models of D, Ψ is defined on $\Sigma(S_N)$, thus by conditions (ii) and (vii) on $\mathcal{M}_M$, axiom $\neg Undef(Sit\_map(S_N))$ is satisfied.

By definition of situation assignment, $\Sigma(S_0) = [\,]$ and for each S, $\Sigma(S)$ is a prefix of $\Sigma(S_N)$. Thus, by condition (vii) on $\mathcal{M}_M$, axioms

$$Sit\_map(S_0) = \epsilon$$

$$Prefix\_eq(Sit\_map(s), Sit\_map(S_N))$$

are satisfied.

Note that conditions (ii), (iii), (iv), and (v) on predicates $Undef$, $Causes^{+(-)}$, $Inherits^{+(-)}$ and $Holds$, are the same as those used to show that such an interpretation of these predicates is a model of $SC_c(D_l)$ (see the proof of Proposition 5), therefore $SC_c(D_l)$ is satisfied.

Fluent facts and precedence facts are translated into the same sentences as in T and can be shown to be satisfied as in the proof of Proposition 2.

α **occurs_at** $S \in D$: $\tau(D)$ includes an axiom:

$$(\exists \beta, \gamma).Embedded(\alpha, \beta) \wedge Concatenate(Sit\_map(S), \beta, \gamma) \wedge Prefix\_eq(\gamma, Sit\_map(S_N)))$$

By the same fact in D, there is β s.t. $\Sigma(S)\cdot\beta$ is a prefix of $\Sigma(S_N)$ and $\alpha \subseteq \beta$. Let γ stand for the sequence $\Sigma(S)\cdot\beta$. By cond. (vii) if $\Sigma(S) = \alpha_S$ and $\Sigma(S_N) = \alpha_N$ then $Sit\_map(S) = \alpha_S$ and $Sit\_map(S_N) = \alpha_N$ hold in $\mathcal{M}_M$. By correctness of B_{embedded} (Lemma 9), $Embedded(\alpha, \beta)$ holds and by correctness of $B_{\text{concatenate}}$ (Lemma 6), $Concatenate(\alpha_S, \beta, \gamma)$ holds. Moreover, by correctness of $B_{\text{prefix\_eq}}$ (Lemma 2), $Prefix\_eq(\gamma, \alpha_N)$ holds. Therefore, axiom (f') is satisfied by $\mathcal{M}_M$.

Finally, value minimization axiom (c') exactly captures the minimality condition, with respect to ordering *embedded-subsequence*, imposed on sequence $\Sigma(S_N)$ by the definition of models of domain descriptions in $\mathcal{L}_c$.

($\Leftarrow$)

Let $\mathcal{M}$ be a model of $T_c(D)$. By Proposition 5 there exists a causal model Ψ such that for all F, α,

$$\begin{aligned} &F \in \Psi(\alpha) \text{ iff } \mathcal{M} \models Holds(F, \alpha), \text{ and} \\ &\Psi(\alpha) \text{ is undefinediff} \mathcal{M}_{\mathrm{SC}} \models Undef(\alpha) \end{aligned} \tag{47}$$

Let Ψ be such a causal model and Σ be a situation assignment such that for every S,

$$\Sigma(S) = \alpha_S \quad \Leftrightarrow \quad \mathcal{M}[\![Sit\_map]\!](S) = \alpha_S. \tag{48}$$

We will show $M = (\Psi, \Sigma)$ is a model of D. As before, by (48) and axiom

$$Sit\_map(S_0) = \epsilon$$

we have $\Sigma(S_0) = [\,]$, and by axiom

$$Prefix\_eq(Sit\_map(s), Sit\_map(S_N))$$

we have $\Sigma(S)$ is a prefix of $\Sigma(S_N)$ for any situation symbol S in the language of D. Thus, Σ is a situation assignment. By (47) and axiom

$$\neg Undef(Sit\_map(S_N))$$

of $T_c(D)$ we have that $\Sigma(S_N)$ belongs to the domain of Ψ. Therefore, Σ and Ψ form an interpretation of D.

It only remains to be shown that facts in D are true in M and that there is no $N' = (\Psi, \Sigma')$ such that $\Sigma'(S_N)$ is an embedded-subsequence of $\Sigma(S_N)$ and N' is a model of D.

Fluent and precedence facts are translated into the same sentences as in T and it can be shown as in the proof of Proposition 2 that M satisfies these types of fact.

We need to show M satisfies facts of the form α **occurs_at** S. Suppose (α **occurs_at** S) $\in D$. Then $\mathcal{M}$ entails

$$(\exists \beta, \gamma).Embedded(\alpha, \beta) \wedge Concatenate(Sit\_map(S), \beta, \gamma) \wedge \\ Prefix\_eq(\gamma, Sit\_map(S_N)))$$

Let β, γ be sequences such that the above sentence holds and consider α_S, α_N s.t. $Sit\_map(S) = \alpha_S$ and $Sit\_map(S_N) = \alpha_N$ hold in $\mathcal{M}$. Then, by correctness of B_{embedded} (Lemma 9), we have that $\alpha \subseteq \beta$, by correctness of $B_{\text{concatenate}}$ (Lemma 6), γ is equal to α_S concatenated with β, and by correctness of $B_{\text{prefix\_eq}}$ (Lemma 2), γ is a prefix of α_N. By (48), $\Sigma(S) = \alpha_S$ and $\Sigma(S_N) = \alpha_N$. Hence $\Sigma(S) \cdot \beta$ is a prefix of $\Sigma(S_N)$. By this and $\alpha \subseteq \beta$, M satisfies (α **occurs_at** S).

Finally, we need to show there is no interpretation $N' = (\Psi, \Sigma')$ of D such that $\Sigma'(S_N)$ is an embedded-subsequence (not equal) of $\Sigma(S_N)$ and N' is a model of D. Suppose there is such a model N'. Then, by part ($\Rightarrow$) of this lemma, there is a model $\mathcal{M}'$ of $T_c(D)$ which differs from $\mathcal{M}$ only on the interpretation of function Sit_map and such that sequence $\mathcal{M}'[\![Sit\_map]\!](S_N)$ is an embedded-subsequence (not equal) of $\mathcal{M}[\![Sit\_map]\!](S_N)$. This contradicts our assumption that $\mathcal{M}$ is a model of $T_c(D)$, since $T_c(D)$ includes value minimization of $Sit\_map(S_N)$ w.r.t. ordering *embedded-subsequence* (See Section 5). $\square$

References

[All84] Allen J.F. 1984. Towards a general theory of action and time. *Artificial Intelligence* 23:123–154.

[Bar95] Baral C. 1995. Reasoning about actions: Non-deterministic effects, Constraints and Qualification. In *Proc. of 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, pp. 2017–2023.

[BarGel93] Baral C. and Gelfond M. 1993. Representing concurrent actions in extended logic programming. In *Proc. of 13th International Joint Conference on Artificial Intelligence (IJCAI'93)*, pp. 866–871.

[BarGel94] Baral C. and Gelfond M. 1994. Logic programming and knowledge representation. *Journal of Logic Programming*, 19,20:73–148.

[BarGabPro96a] Baral C., Gabaldon A. and Provetti A. 1996. Formalizing Narratives using Nested Circumscription. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI'96)*, pp. 652–657.

[BarGabPro96b] Baral C., Gabaldon A. and Provetti A. 1996. Value Minimization in Circumscription. In L.C. Aiello, J. Doyle and S. Shapiro, *Proceedings of Knowledge Representation and Reasoning (KR'96)*, pp. 474–481.

[BarGelPro95b] Baral C., Gelfond M. and Provetti A. 1995. Representing Actions - I: Laws, Observations and Hypothesis. In the Working Notes of AAAI Spring Symposium *Extending Theories of Actions: Formal Theories and Practical Applications.*

[BarGelPro97] Baral C., Gelfond M. and Provetti A. 1997. Representing Actions : Laws, Observations and Hypothesis. In *Journal of Logic Programming*, 31(1-3), pp. 201-243.

[BarLob96] Baral, C. and Lobo, J. 1996. Formal Characterization of Active Databases. In *Proc. of International Workshop on Logic in Databases.*

[Doherty et al. 95] Doherty, P. and Lukaszewicz, W. and Szalas, A. 1995. Computing circumscription revisited: a reduction algorithm. To appear in *Journal of Automated Reasoning.*

[Erol et al. 95] Erol K., Nau D., and Subrahmanian V.S., 1995. Complexity, decidability and undecidability results for domain-independent planning. *Artificial Intelligence*, 76(1-2):75–88

[GelLif92] Gelfond M. and Lifschitz V. 1992. Representing actions in extended logic programs. In *Joint International Conference and Symposium on Logic Programming*, 559–573.

[GelLifRab91] Gelfond M., Lifschitz V., and Rabinov A. 1991. What are the limitations of the situation calculus? In Boyer R., editor, *Automated Reasoning: Essays in Honor of Woody Bledsoe.* Kluwer Academic, Dordrecht.

[Geo94] Georgeff, M. 1994. Editor, Special issue of Journal of Logic and Computation on *Action and Processes*, 4(5), Oxford University Press.

[GiuKarLif95] Giunchiglia E., Kartha N. and Lifschitz V. 1995. Actions with indirect effects (Extended Abstract). Working Notes of AAAI Spring Symposium *Extending Theories of Actions: Formal Theories and Practical Applications.* Stanford.

[GiuKarLif97] Giunchiglia E., Kartha N. and Lifschitz V. 1997. Representing Actions: Indeterminacy and Ramifications. *Artificial Intelligence*, 97(2):409–443.

[Hoa87] Hoare C. 1987. An overview of some formal methods for program design. *IEEE Computer*, 85-91, Sept 87.

[KakKowTon92] Kakas A., Kowalski, R. and Toni F. 1992. Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770.

[KakMil96] Kakas A. and Miller, R. 1996. A simple declarative language for describing narratives with actions. *Journal of Logic Programming,* 31(1-3):157–200.

[Kar93] Kartha, G.N. 1993 Soundness and completeness theorems for three formalizations of action. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence IJCAI93*, pp 712–718.

[Kar95] Kartha, G.N. 1995. A mathematical investigation of reasoning about actions. Ph.D dissertation. University of Texas at Austin.

[KarLif94] Kartha N. and Lifschitz V., 1994. Actions with indirect effects (preliminary report). In *KR'94*, 341–350.

[KowSer86] Kowalski R. and Sergot M. 1986. A logic-based calculus of events. *New Generation Computing*, 4:67–95.

[Lif95] Lifschitz V. 1995. Nested Abnormality Theories. *Artificial Intelligence*, 74:351–365.

[Lif94] Lifschitz V., 1994. 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 298-352. Oxford University Press.

[Lif93] Lifschitz V., 1993. Restricted Monotonicity. In *Proc. of AAAI'93*, pp. 432–437.

[Lif97] Lifschitz V. (editor) 1997. Special issue on Reasoning about Actions and Change. *Journal of Logic Programming*, 31(1-3).

[LinSho92] Lin F., and Shoham Y., 1992. Concurrent actions in the situation calculus. In *Proc. AAAI-92.* pages 590-595.

[MarSub89] Marek W. and Subrahmanian V.S., 1989. The Relationship Between Stable, Supported, Default and Auto-Epistemic Semantics for General Logic Programs. In G. Levi and M. Martelli, editors, *Proceedings of Sixth International Conference in Logic Programming*, pp 600–620.

[McC86] McCarthy J. 1986. Applications of circumscription to formalizing common sense knowledge. Artificial Intelligence, 26(3):89–116.

[McC88] McCarthy J 1988. Mathematical logic in Artificial Intelligence. *Dædalus*, pp. 297-311.

[McC92] McCarthy J. 1992. Overcoming an unexpected obstacle. Manuscript.

[McC95] McCarthy J. 1995. Situation calculus with concurrent events and narrative. Preliminary manuscript. (Available only at http://www-formal.stanford.edu/jmc/narrative.dvi)

[McCHay69] McCarthy J. and Hayes P. 1986. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer B. and Michie D., editors, *Machine Intelligence*, 4:463–502. Edinburgh University Press.

[MilSha94] Miller R. and Shanahan M. 1994. Narratives in the Situation Calculus. Journal of Logic and Computation, 4(5):513–530.

[Pro96] Provetti A., 1996. Hypothetical Reasoning: From Situation Calculus to Event Calculus. *Computational Intelligence Journal*, 12(3):478–498.

[Pin94] Pinto, J., 1994. *Temporal Reasoning in the Situation Calculus*, PhD thesis, Department of Computer Science, University of Toronto, Canada.

[Pin96] Pinto, J. 1996. Occurrences and narratives as constraints in the branching structure of the situation calculus. Manuscript (http://lyrcc.ing.puc.cl/jpinto/).

[PinRei93] Pinto J. and Reiter R. 1993. Temporal reasoning in logic programming: A case for the situation calculus. In *Proc. of 10th International Conference in Logic Programming*, pages 203–221.

[PinRei95] Pinto J. and Reiter R., 1995. Reasoning about time in the situation calculus. In *Annals of Mathematics and AI*, 14, 2-4, pp. 251–268.

[Rei91] Reiter R., 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz V. (editor), *Artificial Intelligence and Mathematical Theory of Computation*, 359–380.

[Rei96] Reiter, R. 1996. Natural actions, concurrency and continuous time in the situation calculus. Proc. of *KR'96* Conference, 2–13.

[Levesque et al. 97] Levesque, H., Reiter, R., Lesperance, Y., Lin, F. and Scherl, R. 1997. Golog: A Logic programming language for dynamic domains. In *Journal of Logic Programming*, 31(1–3).

[San89] Sandewall E. 1989. Filter preferential entailment for the logic of action in almost continuous worlds. In *Proc. of 11th International Joint Conference on Artificial Intelligence*, 894–899.

[San92] Sandewall E., 1992. Features and fluents: A systematic approach to the representation of knowledge about dynamical systems. Oxford Un. Press.

[Sha95] Shanahan, M. 1995. A Circumscriptive Calculus of Events. *Artificial Intelligence,* 75(2):249–284.

[Sha97] Shanahan M., 1997. Solving the frame Problem: A Mathematical Investigation of the Commonsense Law of Inertia. The MIT Press.