Actions with Indirect Effects (Extended Abstract)

Enrico Giunchiglia, G. Neelakantan Kartha and Vladimir Lifschitz
Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712
{enrico,kartha,vl}@cs.utexas.edu

1 Introduction

Our goal is to extend and improve the action language $\mathcal{AR}_0$ proposed in [Kartha and Lifschitz, 1994] and to simplify the method for representing actions with indirect effects in circumscriptive theories described in that paper. The new action language $\mathcal{AR}$ differs from $\mathcal{AR}_0$ in two ways.

First, $\mathcal{AR}$ allows us to represent fluents whose values are non-Boolean, such as $Location(x)$ or $Color(x)$. In the situation calculus, such fluents can be always eliminated in favor of propositional fluents, just as function symbols in first-order logic can be eliminated in favor of predicate symbols. For instance, instead of

$$Brighter(Color(x, s1), Color(x, s2))$$

we can write

$$\forall c_1 c_2 [HasColor(x, c1, s1) \wedge HasColor(x, c2, s2) \supset Brighter(c_1, c_2)].$$

But this may be unnatural and inconvenient. Surprisingly, extending the solution to the frame problem incorporated into the semantics of $\mathcal{AR}_0$ to nonpropositional fluents was not trivial. To justify the new formulation, we prove that the elimination of nonpropositional fluents from a domain description "preserves the meaning" of the description.

Second, "release" propositions of the language $\mathcal{AR}_0$, designed for describing nondeterministic actions, are replaced in $\mathcal{AR}$ by propositions with a similar syntax but a different semantics. This is done in response to the criticism by Fangzhen Lin (personal communication, October 26, 1993), who observed that the behavior of nondeterministic actions in $\mathcal{AR}_0$ may be unintuitive when indirect effects are involved.

Furthermore, we have simplified the representation of action domains by nested abormality theories described in [Kartha and Lifschitz, 1994] and extended it to nonpropositional fluents. The new translation is based on the circumscriptive solution to the frame problem proposed in [Kartha and Lifschitz, 1995] and is related to the ideas of [Winslett, 1988], [Lin and Shoham, 1991] and [Lin and Reiter, 1994].

2 Language $\mathcal{AR}$

2.1 Syntax

An $\mathcal{AR}$ language is characterized by

- a nonempty set of symbols, that are called *fluent names*, or *fluents*,
- a function, associating with every fluent name F a nonempty set Dom_F of symbols that is called the *domain* of F,
- a subset of fluent names, that is called the *frame*,
- a nonempty set of symbols, that are called *action names*, or *actions*.

A *value* is a symbol that belongs to the domain of a fluent.

An *atomic formula* is an expression of the form (F **is** V), where F is a fluent, and $V \in Dom_F$. A *formula* is a propositional combination of atomic formulae. A fluent F is *propositional* if $Dom_F = \{False, True\}$; for a propositional fluent F, we will abbreviate the atomic formula

$$F \textbf{ is } True$$

by F.

There are four types of *propositions* in an $\mathcal{AR}$ language: value propositions, determinate and indeterminate effect propositions, and constraints. A *value proposition* is an expression of the form

$$C \textbf{ after } \overline{A}, \quad (1)$$

where C is a formula, and $\overline{A}$ is a string of actions. If $\overline{A}$ in this proposition is empty, we will write it as

$$\textbf{initially } C.$$

Otherwise, the members of $\overline{A}$ will be separated by semicolons. A *determinate effect proposition* is an expression of the form

$$A \textbf{ causes } C \textbf{ if } P, \quad (2)$$

where A is an action, and C and P are formulae. An *indeterminate effect proposition* is an expression of the form

$$A \textbf{ possibly changes } F \textbf{ if } P, \quad (3)$$

where A is an action, F a fluent that belongs to the frame and P a formula. Intuitively, (3) asserts that the value of F can change nondeterministically when the action A is executed in a situation in which the precondition P is true; such propositions replace the "release" propositions of $\mathcal{AR}_0$. In effect propositions (2), (3), the part

$$\textbf{if } P$$

will be dropped if P is the propositional formula *True*. Finally, a *constraint* is a proposition of the form

$$\textbf{always } C, \tag{4}$$

where C is a formula.

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

2.2 Examples

Example 1. A disc is divided into N sectors, $N > 1$. At every instant of time, one of the sectors is under the head that reads information from the disk. There is an action which rotates the disc counterclockwise. We can describe this system using one fluent *Orientation*, whose domain is $\{0, \dots, N-1\}$, and which belongs to the frame. There is one action, *Turn*. The domain description consists of N propositions

$$\begin{array}{r} \textit{Turn} \textbf{ causes } (\textit{Orientation} \textbf{ is } (i+1) \bmod N) \\ \textbf{if } (\textit{Orientation} \textbf{ is } i) \quad (0 \le i < N). \end{array} \tag{5}$$

Example 2. (This is an enhancement of the example used by Fangzhen Lin in his criticism.) In the following description of throwing dice, *Top* is a fluent with the domain $\{1, \dots, 6\}$, and *Even* is a propositional fluent. The former is in the frame, and the latter is not; a constraint explicitly defines *Even* in terms of *Top*:

always $\textit{Even} \equiv (\textit{Top} \textbf{ is } 2 \vee \textit{Top} \textbf{ is } 4 \vee \textit{Top} \textbf{ is } 6)$,
Throw **possibly changes** *Top*.

2.3 Semantics: The Transition Function

A *valuation* is a function that is defined on the set of fluents and maps each fluent F to an element of Dom_F. A valuation σ can be extended to atomic formulae in a standard way:

$$\sigma(F \textbf{ is } V) = \begin{cases} \textit{True}, & \text{if } \sigma(F) = V, \\ \textit{False}, & \text{otherwise.} \end{cases}$$

It can be further extended to arbitrary formulae according to the truth tables of propositional logic.

Consider a domain D. A valuation σ is a *state* if, for every constraint (4) in D, $\sigma(C) = \textit{True}$. The semantics of $\mathcal{AR}$ shows how the effect propositions of D define a nondeterministic transition system with this set of states, whose input symbols are actions. We will describe this transition system by a function *Res* that maps an action and a state to a set of states. The elements of $Res(A, \sigma)$ are, intuitively, the states that differ from σ as dictated by the determinate effect propositions for A, but, at the same time, do not differ from σ "too much."

As a preliminary step, define $Res_0(A, \sigma)$ to be the set of states σ' such that, for each determinate effect proposition (2) in D, $\sigma'(C) = \textit{True}$ whenever $\sigma(P) = \textit{True}$. The set $Res(A, \sigma)$ will be defined as the subset of $Res_0(A, \sigma)$ whose elements are "close" to σ.

In order to make this precise, the following notation is needed. For any action A and any states σ, σ', $New_A(\sigma, \sigma')$ is the set of formulae

$$F \textbf{ is } \sigma'(F)$$

such that

- F is in the frame and $\sigma'(F) \neq \sigma(F)$, or
- for some indeterminate effect proposition (3) in D, $\sigma(P) = \textit{True}$.

The condition $\sigma'(F) \neq \sigma(F)$ expresses that F **is** $\sigma'(F)$ is a "new fact" that becomes true if the execution of A in state σ results in state σ'. The set $New_A(\sigma, \sigma')$ includes such "new facts" for all frame fluents F. (If F is not a frame fluent then it is not expected to keep its old value after performing an action, so that the change in its value is disregarded.) On the other hand, if some indeterminate effect proposition allows F to change, we treat its value in state σ' as "new" even if it happens to coincide with the value of F in state σ.

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

2.4 Semantics: Models and Entailment

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

A value proposition (1) is *true* in a structure Ψ if $\overline{A}$ is executable in Ψ and $\Psi(\overline{A})(C) = \textit{True}$. A constraint (4) is *true* in a structure Ψ if for any string $\overline{A}$ executable in Ψ, $\Psi(\overline{A})(C) = \textit{True}$.

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

- if $\overline{A}A$ is executable in Ψ, then

$$\Psi(\overline{A}A) \in Res(A, \Psi(\overline{A})),$$

- otherwise, $Res(A, \Psi(\overline{A})) = \emptyset$.

For instance, the domain description from Example 1 has N models, corresponding to N possible initial orientation of the disk. By adding the value proposition

$$\textbf{initially } \textit{Orientation} \textbf{ is } 0 \tag{6}$$

to the domain description, we would reduce the number of models to one. The domain description from Example 2 has $2^{\aleph_0}$ models, one for each infinite sequence of "outcomes" $1, \dots, 6$.

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

2.5 Elimination of Nonpropositional Fluents

Let D be a domain description, and let S be a subset of fluents in the language of D such that every fluent in S has finitely many values.

The language of the domain description D_S is obtained from the language of D by replacing each fluent F in S with the set of new propositional fluents

$\{F^V \mid V \in Dom_F\}$. A new fluent F^V is included in the frame if F was in the frame.

For any fluent F in the language of D, let F_S be the set of fluents in the language of D_S defined as follows:

$$F_S = \begin{cases} \{F^V \mid V \in Dom_F\} & \text{if } F \in S, \\ \{F\} & \text{otherwise.} \end{cases}$$

If C is a formula in the language of D, C_S stands for the formula obtained from C by replacing all atomic parts F is V such that $F \in S$ with F^V. If α is a constraint, a value proposition or a determinate effect proposition in the language of D, α_S is obtained from α by the same process.

The propositions of D_S are formed by

- replacing each constraint, value proposition or determinate effect proposition α in D with α_S;
- replacing each indeterminate effect proposition (3) in D with the set of indeterminate effect propositions

  $$A \textbf{ possibly changes } F' \textbf{ if } P_S$$

  for all $F' \in F_S$,
- adding, for every fluent $F \in S$ and every pair of distinct values $V_1, V_2 \in Dom_F$, the constraints

  $$\textbf{always } \neg(F^{V_1} \wedge F^{V_2})$$

  and

  $$\textbf{always } \bigvee_{V \in Dom_F} F^V.$$

Theorem 1 *A value proposition*

$$C \textbf{ after } \overline{A}$$

is entailed by D if and only if

$$C_S \textbf{ after } \overline{A}$$

is entailed by D_S.

In this sense, nonpropositional fluents with finite domains are not necessary. Any domain description involving only fluents with finite domains can be effectively reduced to a domain description involving only propositional fluents.

As an illustration of this theorem, let D be the domain description consisting of propositions (5) and (6), and let $S = \{Location\}$. This description D entails

$$Orientation \textbf{ is } (k \bmod N) \quad \textbf{after} \quad \underbrace{Turn;\ldots;Turn}_{k}.$$

Consequently, the proposition

$$Orientation^{(k \bmod N)} \quad \textbf{after} \quad \underbrace{Turn;\ldots;Turn}_{k}$$

is entailed by the following domain description D_S:

$Turn$ **causes** $Orientation^{(i+1 \bmod N)}$ **if** $Orientation^i$
$(0 \leq i < N)$,
initially $Orientation^0$,
always $\neg(Orientation^i \wedge Orientation^j)$
$(0 \leq i < j < N)$,
always $Orientation^0 \vee \ldots \vee Orientation^{N-1}$.

3 Translation from $\mathcal{AR}$ into Circumscription

In this section, we present a new translation γ from $\mathcal{AR}$ domain descriptions to "nested abnormality theories" which is simpler than the translation β described in [Kartha and Lifschitz, 1994]. We restrict attention to *finite* domain descriptions, that is, to the domain descriptions with finitely many fluents, values, actions and propositions.

3.1 Nested Abnormality Theories

The formalism of nested abnormality theories is introduced in [Lifschitz, 1994]. Its use is demonstrated there by recasting several familiar applications of circumscription in its framework—inheritance hierarchies, the domain closure assumption, and the causal minimization approach to the frame problem.

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

$$C_1, \ldots, C_m : A_1, \ldots, A_n. \tag{7}$$

The circumscription operator allows us to translate (7) into the language of classical logic by forming the circumscription of the abnormality predicate Ab relative to the conjunction of the axioms $A_1 \wedge \ldots \wedge A_n$ with $C_1, \ldots, C_m$ allowed to vary; we denote this circumscription by

$$\text{CIRC}[A_1 \wedge \ldots \wedge A_n; Ab; C_1, \ldots, C_m]. \tag{8}$$

(See [Lifschitz, 1993] for the definition of the circumscription operator.) However, this is not general enough for the purpose of representing defaults, and "prioritized circumscription" is proposed in [McCarthy, 1986] as a more general representation tool. In nested abnormality theories, we generalize (7) in a different way: each A_i in (7) is allowed to be a "block" of form (7), so that axioms become "nested." Intuitively, each block can be viewed as a group of axioms that describes a certain collection of predicates and functions $C_1, \ldots, C_m$, and the embedding of blocks reflects the dependence of these descriptions on each other.

It is also convenient to turn the predicate constant Ab in (8) into an existentially quantified predicate variable. The abnormality predicate usually plays an auxiliary role in a formalization; what we are actually interested in are the logical consequences of (8) that do not include Ab. To put it differently, if (8) is denoted by $F(Ab)$, and ab is a predicate variable of the same arity as Ab, then what we are interested in are the consequences of the sentence $\exists ab F(ab)$. The effect of this modification in the context of nested abnormality theories is that the abnormality predicate becomes local to the block in which it is used.

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

The syntax of simple abnormality theories is defined in such a way that Ab may even have different arities in different blocks.

We turn now to the formal treatment of the subject. The definitions below are reproduced from [Lifschitz, 1994].

Consider a first-order language L, possibly many sorted. We assume that L does *not* include Ab among its symbols. For any list $\tau_1, ..., \tau_k$ $(k \geq 0)$ of sorts of variables available in L, by $L_{\tau_1 ... \tau_k}$ we denote the language obtained from L by adding Ab as a k-ary predicate constant taking arguments of the sorts $\tau_1, ..., \tau_k$. *Blocks* are defined recursively as follows: For any list of sorts $\tau_1, ..., \tau_k$ and any list of function and/or predicate constants $C_1, \ldots, C_m$ $(m \geq 0)$ of L, if each of $A_1, \ldots, A_n$ $(n \geq 0)$ is a formula of $L_{\tau_1 ... \tau_k}$ or a *block*, then

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

is a *block*. A *nested abnormality theory* is a set of blocks, called its *axioms*.

The semantics of nested abnormality theories is characterized by a map φ that translates blocks into second-order formulae. It is convenient to make φ defined also on formulae of the languages $L_{\tau_1 ... \tau_k}$. If A is such a formula, then φA stands for the universal closure of A. For blocks we define, recursively:

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

where

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

Note that, for any block A, φA is a sentence not containing Ab.

A sentence A of L will be identified with the block $\{: A\}$. It is easy to see that $\varphi\{: A\}$ is equivalent to A.

For any nested abnormality theory T, φT stands for $\{\varphi A \mid A \in T\}$. A *model* of T is a model of φT in the sense of classical logic. A *consequence* of T is a sentence of L that is true in all models of T.

3.2 Language

In the nested abnormality theories described below, the underlying language L has variables of four sorts: for values $v, v_1, v_2, \ldots$, actions $a, a_1, a_2, \ldots$, situations $s, s_1, s_2, \ldots$ and "aspects." As in [McCarthy, 1986], aspects will be used to distinguish between different kinds of abnormality.

Consider a finite domain description D. By $\mathbf{F}$ we will denote the set of fluents in the language of D, by $\mathbf{Fr}$ the set of frame fluents, by $\mathbf{V}$ the set of values, and by $\mathbf{A}$ the set of actions. The language L has the following object constants:

- the elements of $\mathbf{V}$, representing values,
- the elements of $\mathbf{A}$, representing actions,
- S_0 and $\perp$, representing situations,
- for all $F \in \mathbf{Fr}$, the constants $\widetilde{F}$, representing aspects.

Intuitively, the situation constant $\perp$ represents the value "undefined", so that the assertion that a is not executable in situation s can be expressed by

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

Every $F \in \mathbf{F}$ is included in L as a constant representing a function from situations to values. Also, L includes the binary function *Result*, as usual in the situation calculus. Finally, L has the constants F_R for all $F \in \mathbf{F}$ and *Poss* that will be "explicitly defined" by the axioms

$$F_R(a, s) = F(Result(a, s)),$$
$$Poss(a, s) \equiv Result(a, s) \neq \perp.$$

Note that "formulae" as defined in Section 2.1 are not among the formulae of the language L. To avoid confusion, we will refer to the formulae in the sense of Section 2.1 as "domain formulae." For any domain formula C and any situation term ζ, by $T(C, \zeta)$ we will denote the formula obtained from C as the result of replacing each atomic part F **is** V by $F(\zeta) = V$. For instance,

$$T(\neg(Orientation \textbf{ is } 0), s)$$

stands for

$$Orientation(s) \neq 0.$$

If α is an action term then $T_R(C, \alpha, \zeta)$ will stand for the formula obtained from C as the result of replacing each atomic part F **is** V by $F_R(\alpha, \zeta) = V$.

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

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

3.3 Two Defaults

Our translation γ from $\mathcal{AR}$ into nested abnormality theories uses two defaults. The first default says that an action is normally executable in a situation. We express this by

$$\neg Ab(a, s) \supset Poss(a, s). \tag{9}$$

The second default is the *commonsense law of inertia*, which expresses that the value of a frame fluent normally remains unchanged after performing an action. The formulae describing how actions change the world represent exceptions to this default. The commonsense law of inertia is expressed by the set of formulae

$$Poss(a, s) \wedge v = F_R(a, s) \wedge \neg Ab(\widetilde{F}, v, a, s) \supset v = F(s),$$

one for each F in $\mathbf{Fr}$. This set of formulae will be denoted by LI.

As we will see, the nested abnormality theory γD for a finite domain D consists of several formulae, including LI and (9) arranged into a system of blocks. We should not worry about by the fact that Ab has four arguments in LI and only two arguments in (9): these formulae will appear in two different blocks.

3.4 Translating Propositions

We will define how to construct, for each proposition P in D, its translation. These formulae will be included in the translation γD of D.

If P is a value proposition (1), then γP is

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

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

$$\textit{Orientation} \textbf{ is } 5 \textbf{ after } \textit{Turn}$$

is translated as

$$\begin{aligned} &\textit{Result}(\textit{Turn}, S_0) \neq \bot \\ &\quad \wedge \textit{Orientation}(\textit{Result}(\textit{Turn}, S_0)) = 5. \end{aligned}$$

If P is a determinate effect proposition (2), then γP is

$$T(P, s) \wedge \textit{Poss}(A, s) \supset T_R(C, A, s).$$

For instance, the proposition

$$\textit{Turn} \textbf{ causes } (\textit{Orientation} \textbf{ is } 6) \textbf{ if } (\textit{Orientation} \textbf{ is } 5)$$

is translated as

$$\begin{aligned} \textit{Orientation}(s) = 5 \wedge \textit{Poss}(\textit{Turn}, s) \supset \\ \textit{Orientation}_R(\textit{Turn}, s) = 6. \end{aligned}$$

If P is an indeterminate effect proposition (3), then γP is

$$T(P, s) \supset Ab(\widetilde{F}, F_R(A, s), A, s).$$

This formula will accompany the commonsense laws of inertia LI, so that the law of inertia will be disabled in application to F when A is executed in the presence of the precondition P. For instance, the proposition

$$\textit{Throw} \textbf{ possibly changes } \textit{Top}$$

is translated as

$$Ab(\widetilde{\textit{Top}}, \textit{Top}_R(\textit{Throw}, s), \textit{Throw}, s)$$

(if we disregard the trivial antecedent $T(\textit{True}, s)$, which equals $\textit{True}$).

It remains to describe how constraints are translated. In $\mathcal{AR}$, constraints play two different roles: they determine the set of states, and they also determine the indirect effects of actions. Accordingly, in the nested abnormality theory γD every constraint of D will be represented by two formulae. If P is a constraint (4), then we will denote the formula $T(C, s)$ by γP, and the formula $T_R(C, a, s)$ by $\gamma_R P$. For instance, if P is

$$\textbf{always } \textit{Even} \equiv (\textit{Top} \textbf{ is } 2 \vee \textit{Top} \textbf{ is } 4 \vee \textit{Top} \textbf{ is } 6)$$

then γP is

$$\begin{aligned} &\textit{Even}(s) = \textit{True} \\ &\quad \equiv (\textit{Top}(s) = 2 \vee \textit{Top}(s) = 4 \vee \textit{Top}(s) = 6), \end{aligned}$$

and $\gamma_R P$ is

$$\begin{aligned} &\textit{Even}_R(a, s) = \textit{True} \\ &\equiv (\textit{Top}_R(a, s) = 2 \vee \textit{Top}_R(a, s) = 4 \vee \textit{Top}_R(a, s) = 6). \end{aligned}$$

3.5 Definition of γD

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

Let D_v, D_d and D_i be the parts of D consisting of its value propositions, determinate effect propositions and indeterminate effect propositions. By D_c we denote the set of constraints that consists of all constraints of D and the constraints

$$\textbf{always} \bigvee_{V \in Dom_F} F \textbf{ is } V$$

for all $F \in \mathbf{F}$. $\mathbf{F}_R$ stands for the list of all predicates F_R.

The axioms of γD are as follows.

Group 1. Unique names axioms: $c_1 \neq c_2$ for all pairs c_1, c_2 of distinct object constants of the same sort. For instance, this group includes the axiom $S_0 \neq \bot$.

Group 2. Domain closure axioms:

$$\bigvee_{V \in \mathbf{V}} v = V,$$

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

Group 3. Translations of the constraints:

$$\gamma P \qquad (P \in D_c).$$

Group 4. Definitions of $\textit{Poss}$ and $\mathbf{F}_R$:

$$\begin{aligned} &\textit{Poss}(a, s) \equiv \textit{Result}(a, s) \neq \bot, \\ &F_R(a, s) = F(\textit{Result}(a, s)) \qquad (F \in \mathbf{Fr}). \end{aligned}$$

Group 5. Translations of the value propositions:

$$\gamma P \qquad (P \in D_v).$$

Group 6. Characterization of the effects of actions:

$$\begin{aligned} &\{\mathbf{F}_R : \\ &\quad LI, \\ &\quad \gamma P \qquad (P \in D_i), \\ &\quad \{\textit{Poss}, \mathbf{F}_R : \\ &\qquad \neg Ab(a, s) \supset \textit{Poss}(a, s), \\ &\qquad \neg \textit{Poss}(a, \bot), \\ &\qquad \gamma P \qquad (P \in D_d), \\ &\qquad \gamma_R P \qquad (P \in D_c), \\ &\quad \} \\ &\}. \end{aligned}$$

The two nested circumscriptions represented by this block are, of course, the main part of the theory. The inner circumscription tells us that an action can be executed unless this is prohibited by the determinate effect propositions and constraints of D. The outer circumscription encodes the idea of inertia. The nesting of blocks reflects our intention to decide first which actions can be executed, and then what the effects of actions are.

3.6 Discussion

The use of circumscription in Group 6 leads to intuitively expected results in "difficult" cases, such as the shooting scenario from [Hanks and McDermott, 1987] or the two bus example from [Kartha, 1994]. Moreover, the soundness and completeness theorem stated below shows that we can expect as few unpleasant surprises from this approach to the use of circumscription as we expect from the semantics of $\mathcal{AR}$. How is this achieved?

Note that, in the circumscriptions in Group 6, we have completely eliminated the function constant *Result* in favor of $\mathbf{F}_R$ and *Poss*. When we want to consider the effect of an action a on a fluent in a situation s, the only two situations that are of interest are the situation s and the situation obtained by performing the action a in s. For instance, we do not want to consider the sequence of actions that leads to s, nor do we wish to consider what happens afterward. Accordingly, we switch to the "local language" of $\mathbf{F}_R$ and *Poss*, which represents the "theory update view" of [Winslett, 1988], rather than the "situation calculus view" of [McCarthy, 1986].

The simplification achieved here in comparison with [Kartha and Lifschitz, 1994] is that the existence of situations principle is not needed any more. This is an important advantage, because the formalization of that principle required the use of higher-order circumscription.

3.7 Soundness and Completeness

Consider a finite domain D. Let Ψ be a model of D, and let M be a model of γD. We say that M is *similar* to Ψ if, for every value proposition P, P is true in Ψ if and only if M satisfies γP.

Theorem 2 *Let D be a finite domain. For any model Ψ of D, there exists a model M of γD similar to Ψ. For any model M of γD, there exists a model Ψ of D such that M is similar to Ψ.*

The following corollary expresses the soundness and completeness of the translation.

Corollary. *For any finite domain D and any value proposition P, γD entails γP if and only if D entails P.*

Acknowledgements

We would like to thank Fangzhen Lin, Norman McCain and Hudson Turner for useful discussions related to the subject of this paper. This work was partially supported by National Science Foundation under grant IRI-9306751.

References

[Hanks and McDermott, 1987] Steve Hanks and Drew McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence*, 33(3):379–412, 1987.

[Kartha and Lifschitz, 1994] G. Neelakantan Kartha and Vladimir Lifschitz. Actions with indirect effects (preliminary report). In *Proc. of the Fourth Int'l Conf. on Principles of Knowledge Representation and Reasoning*, pages 341–350, 1994.

[Kartha and Lifschitz, 1995] G. Neelakantan Kartha and Vladimir Lifschitz. A simple formalization of actions using circumscription. Manuscript, 1995.

[Kartha, 1994] G. Neelakantan Kartha. Two counterexamples related to Baker's approach to the frame problem. *Artificial Intelligence*, 69:379–391, 1994.

[Lifschitz, 1993] Vladimir Lifschitz. Circumscription. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *The Handbook of Logic in AI and Logic Programming*, volume 3, pages 298–352. Oxford University Press, 1993.

[Lifschitz, 1994] Vladimir Lifschitz. Nested abnormality theories. Manuscript, 1994.

[Lin and Reiter, 1994] Fangzhen Lin and Raymond Reiter. State constraints revisited. *Journal of Logic and Computation, Special Issue on Actions and Processes*, 4(5):655–678, 1994.

[Lin and Shoham, 1991] Fangzhen Lin and Yoav Shoham. Provably correct theories of action (preliminary report). In *Proc. AAAI-91*, pages 349–354, 1991.

[McCarthy, 1986] John McCarthy. Applications of circumscription to formalizing common sense knowledge. *Artificial Intelligence*, 26(3):89–116, 1986. Reproduced in [McCarthy, 1990].

[McCarthy, 1990] John McCarthy. *Formalizing common sense: papers by John McCarthy.* Ablex, Norwood, NJ, 1990.

[Winslett, 1988] Marianne Winslett. Reasoning about action using a possible models approach. In *Proc. AAAI-88*, pages 89–93, 1988.