Indefiniteness and Nondeterminism in Action Description Languages (Draft)

Norman McCain
University of Texas at Austin
mccain@cs.utexas.edu

1 Introduction

According to Lifschitz [1995], the language $\mathcal{A}$ [Gelfond and Lifschitz, 1993] and its successors can be viewed as a combination of two sublanguages. These sublanguages can be distinguished in the following manner. First, there is an *action description language* in which the possible effects of actions in states are described. Secondly, there is a *fact description language* in which facts about situations are described. In $\mathcal{A}$ itself, the action description language consists of "effect propositions" such as

$$\mathit{Shoot}\ \textbf{causes}\ \neg \mathit{Alive}\ \textbf{if}\ \mathit{Loaded}$$

and the fact description language consists of "value propositions" such as

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

The facts described by value propositions in $\mathcal{A}$ are often counterfactual.

The distinction between the two sublanguages, as we have described it, is based on their differing ontologies; specifically, on the difference between states and situations. The following remarks are intended to explain how we understand these two kinds of objects to be related.

Intuitively, by a *situation* we mean a possible world at a moment in time. Thus, a situation is essentially located in a world history, with an actual past and an actual future. By a *state*, on the other hand, we mean the occurrent part of a situation; to reapply a common metaphor, it is the part of a situation that can be captured in a snapshot.[1] Actions are performed in situations and result in situations. The possible effects of an action, however, are determined by the state of the world at the time at which the action is performed. Accordingly, it is natural to describe the possible effects of actions with respect to an ontology of states rather than situations. This is especially

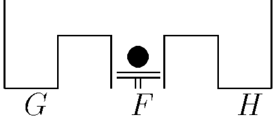
[1] The common description of a situation as a snapshot of the world at a moment in time is, from the present point of view, actually a better description of a state, the past and future being precisely what cannot be captured in a snapshot.

so, since states are simpler objects than situations, in the following sense: Two different situations, belonging either to the same or to different worlds, may realize the same state. (The snapshots of two different situations may be identical.) As a consequence, the number of states may be finite even though the number of situations is not. In the semantics of $\mathcal{A}$ and its successors, states are generally modeled by interpretations of the fluent names, and situations are modeled by strings of action names, the assumption being that a situation can be identified with the string of actions that leads to it.

In this paper, we define two high-level action description languages, called $\mathcal{L}_{\mathrm{C}}$ and $\mathcal{L}_{\mathrm{CN}}$. In the language $\mathcal{L}_{\mathrm{C}}$ we extend the action description sublanguage of $\mathcal{A}$ in two ways. First, in $\mathcal{L}_{\mathrm{C}}$ arbitrary formulas may be used in place of literals in the specification of the effects of actions. Secondly, in $\mathcal{L}_{\mathrm{C}}$ background knowledge can be given in the form of static "causal laws." Intuitively, causal laws describe the causal relations, or more generally the determination relations, that hold among the facts in individual states of the world. (The importance of such background knowledge for an adequate account of the ramifications and derived qualifications of actions has been argued in [McCain and Turner, 1995].) Each of these extensions makes possible a kind of indefiniteness in the description of action domains that is absent in $\mathcal{A}$. Consequently, whereas at most one transition system is described by a set of effect propositions in $\mathcal{A}$, more than one such system may be a model of a domain description in $\mathcal{L}_{\mathrm{C}}$.

In the language $\mathcal{L}_{\mathrm{CN}}$ we extend $\mathcal{L}_{\mathrm{C}}$ to allow the specification of nondeterministic actions, making indefiniteness and nondeterminism both expressible in $\mathcal{L}_{\mathrm{CN}}$. In the literature on reasoning about actions, the distinction between indefiniteness and nondeterminism has not always been clearly drawn.[2] Our primary aim in this paper is to clarify this distinction and to ensure that these two different phenomena are manifested in different ways in the semantics of $\mathcal{L}_{\mathrm{CN}}$. In $\mathcal{L}_{\mathrm{CN}}$ indefiniteness is manifested in multiple models, while nondeterminism is manifested in multiple possible next states within individual models.

The distinction between indefiniteness and nondeterminism can be illustrated by the following examples, which are variants of a domain described in [Kartha and Lifschitz, 1995]. In both examples, we imagine that there is a ball enclosed in a certain device, which is pictured below. There are three locations F, G, and H, and a kicker below location F. Whenever the action *Kick* is performed with the ball in location F, the ball is moved out of location F. Because of the structure of the device, this means that the ball will move either to location G or location H as an indirect effect. The question we must consider is this: In what sense are both of these effects possible? Depending on how we answer this question, we obtain one or the other of the following example domains.



[2]An exception is [Boutilier and Friedman, 1995], which is an important source of ideas for the present paper. We remark that indefiniteness is not expressible in the action description language described by Boutilier and Friedman, and that their explanation of the distinction between indefiniteness and nondeterminism in somewhat different from our's.

Example 1 We assume that the action is deterministic, so that in each state of the world in which the ball is in location F only one of the indirect effects is possible, but we also assume that we do not know which of the two indirect effects this is. So both are possible, so far as we know.

Using fluent names F, G, and H to indicate the position of the ball, the domain can be described in either $\mathcal{L}_{\mathrm{C}}$ or $\mathcal{L}_{\mathrm{CN}}$ as follows.

$$\begin{array}{l} \mathit{True} \Rightarrow (F \vee G \vee H) \wedge (\neg F \vee \neg G) \wedge (\neg F \vee \neg H) \wedge (\neg G \vee \neg H) \\ \mathit{Kick}\ \textbf{causes}\ \neg F\ \textbf{if}\ F \end{array}$$

Let us call this domain description D_1. The first proposition is a "causal law." Intuitively, it says that in every state of the world the ball is caused to be in exactly one of the three locations. In the terminology of Lin and Reiter [1994], the causal law functions as a "ramification constraint." The second proposition is an "effect proposition" of a form provided in the language $\mathcal{A}$. It describes the direct effect of the action *Kick* when the ball is in location F.

As we will see in Section 3, the domain D_1 in $\mathcal{L}_{\mathrm{C}}$ (and $\mathcal{L}_{\mathrm{CN}}$) has exactly two models. In the first model, when performing the action *Kick* with the ball in location F, the ball is caused to go to location G, while in the second model, it is caused to go to location H. The existence of these two models indicates that the domain description is indefinite with respect to the effect of performing the action *Kick* when the ball is in location F.

Example 2 We assume that the indirect effect of the action is undetermined in states of the world in which the ball in location F. Thus, both G and H are possible indirect effects, not only so far as we know, or so far as we have said, but also in reality.

The domain of Example 2 can be described in $\mathcal{L}_{\mathrm{CN}}$ as follows.

$$\begin{array}{l} \mathit{True} \Rightarrow (F \vee G \vee H) \wedge (\neg F \vee \neg G) \wedge (\neg F \vee \neg H) \wedge (\neg G \vee \neg H) \\ \mathit{Kick}\ \textbf{causes}\ \neg F\ \textbf{if}\ F \\ \mathit{Kick}\ \textbf{possibly causes}\ G\ \textbf{if}\ F \\ \mathit{Kick}\ \textbf{possibly causes}\ H\ \textbf{if}\ F \end{array}$$

Let us call this domain description D_2. The two new propositions describe two *possible* direct effects of performing the action *Kick* when the ball is in location F. By contrast, the effect proposition describes a necessary direct effect. Of course, every necessary direct effect is also a possible direct effect, so the domain description intuitively implies that the action has three possible direct effects, namely, $\{\neg F\}$, $\{\neg F, G\}$, and $\{\neg F, H\}$.

As we will see in Section 5, the domain D_2 in $\mathcal{L}_{\mathrm{CN}}$ has a unique model. In this model there are two possible next states that can result from performing the action *Kick* when the ball is in location F.

2 Transition Systems

The signature for an action description language consists of two nonempty sets of symbols: a set $\mathbf{A}$ of action names and a set $\mathbf{F}$ of fluent names. An action description language is interpreted with respect to a structure called a transition system. Definitions similar to that which follows appear in [Boutilier and Friedman, 1995] and [Lifschitz, 1995].

A *transition system* for an action description language with the signature $\langle \mathbf{A}, \mathbf{F} \rangle$ consists of the following elements:

- a set $\mathcal{S}$ of objects called *states*,
- a function R from $\mathbf{A} \times \mathcal{S}$ into the powerset of $\mathcal{S}$, and
- a function V from $\mathbf{F} \times \mathcal{S}$ into the set of truth values.

The function R is called a transition function. Intuitively, the elements of $R(A, S)$ are the states that could result from performing the action A in the state S. If $R(A, S)$ contains exactly one state, then A is deterministic in S. If $R(A, S)$ contains more than one state, then A is nondeterministic in S. If $R(A, S)$ is empty, then A cannot be performed in S.

By an *interpretation* of a set X of atoms we mean a function that maps each element of X to a truth value. The function V associates an interpretation of $\mathbf{F}$ with every state.

3 The Language $\mathcal{L}_{\text{C}}$

In this section we define the syntax and semantics of the high-level action description language $\mathcal{L}_{\text{C}}$.

Throughout this section, we will assume a fixed signature $\langle \mathbf{A}, \mathbf{F} \rangle$, where $\mathbf{A}$ is a set of action names and $\mathbf{F}$ is a set of fluent names.

3.1 The Syntax of $\mathcal{L}_{\text{C}}$

By a *fluent literal* we mean an expression of either the form F or $\neg F$, where F is a fluent name. By a *fluent formula* we mean a propositional combination of fluent names. In $\mathcal{L}_{\text{C}}$ we generalize the syntax of $\mathcal{A}$ by allowing fluent formulas in place of fluent literals in effect propositions. We also introduce an additional proposition type which we call a "causal law."

An *effect proposition* is an expression of the form

$$A \ \textbf{causes} \ \phi \ \textbf{if} \ \psi \tag{1}$$

where A is an action name, and ϕ and ψ are fluent formulas. If ψ is *True*, we write simply A **causes** ϕ. Intuitively, (1) says that in every state that satisfies ψ, doing A causes or preserves the truth of ϕ. We say that ψ is a *fluent precondition* for the effect of A on ϕ.

We write the expression

$$\textbf{impossible} \ A \ \textbf{if} \ \psi \tag{2}$$

where A is an action name and ψ is a fluent formula, as an abbreviation for an effect proposition of the form A **causes** *False* **if** ψ. Intuitively, (2) expresses an *action precondition* for A. It says that A cannot be performed in any state in which ψ is true. The terminology of action and fluent preconditions is due to Reiter [1991]. The syntax of effect propositions and the method of expressing action preconditions as abbreviations in the manner described is due to Kartha and Lifschitz [1994].

A *causal law* is an expression of form

$$\phi \Rightarrow \psi \tag{3}$$

where ϕ and ψ are fluent formulas. One possible reading of (3) is that ϕ's being true *causes* or *determines* the truth of ψ.

A *domain description* (or *domain*) in $\mathcal{L}_{\mathrm{C}}$ is a set of effect propositions and causal laws.

3.2 The Semantics of $\mathcal{L}_{\mathrm{C}}$

In this section we define the conditions under which a transition system for $\mathcal{L}_{\mathrm{C}}$ is a model of a domain description.

We treat causal laws as inference rules.[3] Let X be a set of formulas of propositional logic, and C be a set of inference rules. We say that X is *closed under* C, if for every inference rule $\phi \Rightarrow \psi$ in C, if $\phi \in X$ then $\psi \in X$. By $Cn(X)$ we designate the set of formulas ϕ such that $X \vdash \phi$.

The derivability relation $\vdash$ in propositional logic is easily extended to take account of inference rules. Given a set Γ of formulas, a set C of inference rules, and a formula ϕ, we write

$$\Gamma \vdash_C \phi$$

to mean that ϕ is an element of the smallest set X of formulas such that $\Gamma \subseteq X$, $X = Cn(X)$, and X is closed under C.

In the following, we will identify an interpretation I of $\mathbf{F}$ with the set of fluent literals L such that $I \models L$. Thus, for example, given an interpretation I, an expression such as $Cn(I)$ is well-formed. (It designates the set of fluent formulas that are true in I.) Throughout this paper, we use the symbol L to stand exclusively for fluent literals.

Let $\langle \mathcal{S}, R, V \rangle$ be a transition system for $\mathcal{L}_{\mathrm{C}}$, D be a domain description in $\mathcal{L}_{\mathrm{C}}$, A be an action name in $\mathbf{A}$, and S be a state in $\mathcal{S}$. By the *explicit effect* of A in S according to D — in symbols, $E_D(A, S)$ — we designate the set of fluent formulas ϕ such that for some fluent formula ψ, A **causes** ϕ **if** ψ is in D and $S \models \psi$. By $C(D)$ we designate the set of causal laws in D.

A transition system $\langle \mathcal{S}, R, V \rangle$ is a *model* of a domain D in $\mathcal{L}_{\mathrm{C}}$ if it satisfies the following three conditions:

(1) The set $\mathcal{S}$ of states is the set of interpretations I of $\mathbf{F}$ such that $Cn(I)$ is closed under $C(D)$.[4]

[3] In [McCain and Turner, 1995], causal laws are treated in two ways; first, as inference rules, and later as sentences in a modal, conditional logic. There are advantages to the latter point of view, but it is more complex. For present purposes, the treatment in terms of inference rules will suffice.

[4] To put it in another way, the set $\mathcal{S}$ of states is the set of interpretations I of $\mathbf{F}$ such that for every causal law $\phi \Rightarrow \psi$ in D, if $I \models \phi$ then $I \models \psi$.

(2) For all action names A, and states S and S',

   (a) $R(A, S) = \{S'\}$ only if $S' = \{L : (S \cap S') \cup E_D(A, S) \vdash_{C(D)} L\}$, and

   (b) $R(A, S) = \emptyset$ if and only if there is no state S' that satisfies the fixpoint equation in (a).

(3) For every fluent name F and state S, $V(F, S) = S(F)$.

We refer to (1)–(3) as the *model conditions* for $\mathcal{L}_{\mathrm{C}}$. According to model condition (2), $R(A, S)$ is required to be at most a singleton. This requirement reflects the assumption in $\mathcal{L}_{\mathrm{C}}$ that actions are deterministic. Model condition (2a) implements the "common sense law of inertia," which says that changes occur only when they are caused. (This aspect of the fixpoint equation in (2a) is discussed at length in [McCain and Turner, 1995].) According to model condition (2b), actions are assumed to be executable, unless the supposition that they are so conflicts with (2a).

Example 1 (cont'd) The reader may now verify our earlier claim that the domain description D_1 of Example 1 has two models. In both models the set $\mathcal{S}$ contains the following three states.

$$S_1 = \{F, \neg G, \neg H\} \qquad S_2 = \{\neg F, G, \neg H\} \qquad S_3 = \{\neg F, \neg G, H\}$$

In one model (let us call it M_1), R is defined as follows.

$$R(Kick, S_1) = \{S_2\} \qquad R(Kick, S_2) = \{S_2\} \qquad R(Kick, S_3) = \{S_3\}$$

In the other model (let us call it M_2), R is the same as in M_1 except wrt state S_1, where instead $R(Kick, S_1) = \{S_3\}$.

4 The Language $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$

In this section, we augment $\mathcal{L}_{\mathrm{C}}$ by the fact description language MPL of [Boutilier and Friedman, 1995]. We call the combined language $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$.

4.1 The Syntax of $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$

The signature for MPL is the same as for $\mathcal{L}_{\mathrm{C}}$. It consists of a set $\mathbf{A}$ of action names and a set $\mathbf{F}$ of fluent names.

The set of formulas of MPL is the smallest set satisfying the following conditions: (1) if $\phi \in \mathbf{F}$ then ϕ is a formula, and (2) if ϕ and ψ are formulas and α is a string of elements of $\mathbf{A}$ then $\neg\phi$, $\phi \wedge \psi$, $\mathsf{E}\phi$, and $\langle\langle\alpha\rangle\rangle\phi$ are formulas.[5] Following Boutilier and Friedman, we define $\langle\alpha\rangle\phi$ as an abbreviation for $\mathsf{E}\langle\langle\alpha\rangle\rangle\phi$, and $[\alpha]\phi$ as an abbreviation for $\neg\langle\alpha\rangle\neg\phi$.

A *domain description* (or *domain*) in $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$ is a set of formulas belonging either to $\mathcal{L}_{\mathrm{C}}$ or MPL.

[5] Actually, in the full MPL langauge α may be any program term in the sense of PDL [Fischer and Ladner, 1979].

4.2 The Semantics of $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$

The semantics of MPL are described in [Boutilier and Friedman, 1995]. In reviewing the semantics below, we employ some minor differences in terminology.

A model structure for MPL (and $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$) is a pair (M, x), consisting of a transition system $M = \langle \mathcal{S}, R, V \rangle$ and a distinguished path x through it. Intuitively, x represents the actual future. A path may be finite or infinite. A finite path has the form

$$S_0 \xrightarrow{A_1} S_1 \xrightarrow{A_2} \cdots \xrightarrow{A_n} S_n$$

where $S_0, \ldots, S_n$ are states, $A_1, \ldots, A_n$ are action names, and for all i, $1 \leq i \leq n$, $S_i \in R(A_i, S_{i-1})$. We will call such a path an *execution* of $A_1 \ldots A_n$.

Let x and x' be paths. We write $\mathit{first}(x)$ to designate the first state in x, and we write $x \approx x'$ to mean $\mathit{first}(x) = \mathit{first}(x')$.

Let x be a finite path $S_0 \xrightarrow{A_1} S_1 \xrightarrow{A_2} \cdots \xrightarrow{A_n} S_n$, and y be a finite or infinite path $S'_0 \xrightarrow{A'_1} S'_1 \xrightarrow{A'_2} \cdots$ such that $S_n = S'_0$. By $x \cdot y$ we designate the path $S_0 \xrightarrow{A_1} S_1 \xrightarrow{A_2} \cdots \xrightarrow{A_n} S_n \xrightarrow{A'_1} S'_1 \xrightarrow{A'_2} \cdots$.

We define when a formula ϕ of MPL is true in a structure $(\langle \mathcal{S}, R, V \rangle, x)$ as follows. (We suppress mention of the transition system, which is constant through the recursion, writing simply $x \models \phi$.)

If $\phi \in \mathbf{F}$, then $x \models \phi$ iff $V(\phi, \mathit{first}(x)) = \mathit{True}$. Otherwise, for all formulas ϕ and ψ, and any string α of action names,

$$\begin{array}{lll}
x \models \neg\phi & \text{iff} & x \not\models \phi, \\
x \models \phi \wedge \psi & \text{iff} & x \models \phi \text{ and } x \models \psi, \\
x \models \mathsf{E}\phi & \text{iff} & \text{for some path } x',\ x \approx x' \text{ and } x' \models \phi, \\
x \models \langle\langle \alpha \rangle\rangle \phi & \text{iff} & \text{there exist paths } y \text{ and } x' \text{ s.t. } y \text{ is an execution of } \alpha, \\
 & & x = y \cdot x', \text{ and } x' \models \phi\,.
\end{array}$$

Intuitively, $\langle\langle \alpha \rangle\rangle \phi$ says that α actually will be executed and that ϕ will hold afterwards. The abbreviation $\langle \alpha \rangle \phi$ ($[\alpha]\phi$) intuitively says that ϕ holds in some (every) possible future in which α is executed.

A structure (M, x) is a *model* of a set T of MPL formulas if for every formula $\phi \in T$, $(M, x) \models \phi$.

Let D be a domain in $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$. By $D^{\mathcal{L}_{\mathrm{C}}}$ we designate the set of effect propositions and causal laws in D. By D^{MPL} we designate the set of MPL formulas in D. Thus, D is the union of these two sets.

A structure (M, x) is a *model* of a domain D of $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$ if M is a model of $D^{\mathcal{L}_{\mathrm{C}}}$ and (M, x) is a model of D^{MPL}.

Example 1 (cont'd) Suppose we now augment the domain description D_1 of Example 1 by the following MPL formulas.

$$F \tag{4}$$

$$\langle\langle \mathit{Kick} \rangle\rangle G \tag{5}$$

Let us call the augmented domain description D_3. Then $D_3^{\mathcal{L}_{\mathrm{C}}}$ is D_1, and D_3^{MPL} is the set of newly added MPL formulas. Intuitively, (4) says that F is true in the (actual) initial situation, and (5) says that the action *Kick* is performed in the initial situation and G is true afterwards.

Recall that D_1 has two models, the transition systems M_1 and M_2. Notice, however, that there is no path through M_2 that satisfies the two formulas in D_3^{MPL}. Consequently, every model of D_3 must contain the transition system M_1. (There are in fact countably many such models, differing from one another only wrt how many times the action *Kick* will actually be performed.) Thus, the indefiniteness in D_1 is resolved in D_3; intuitively, *Kick* **causes** G **if** F is true in every model of D_3.[6] It seems natural that our knowledge of the effects of actions should at least sometimes be affected by known facts in this way.[7]

It is important to notice the central role of model condition (1) in the phenomenon just described. To see this, suppose we were to modify this condition to allow models to contain "copies" of the presently allowed states. (We say that S' is a *copy* of S if for all $F \in \mathbf{F}$, $V(F,S) = V(F,S')$.) Let M_1' be M_1 but with $\mathcal{S}$ augmented by a copy S_1' of S_1 and R extended so that $R(\mathit{Kick}, S_1') = \{S3\}$. Then

$$(M_1', S_1 \xrightarrow{\mathit{Kick}} S_2)$$

would be a "model" of D_3, in this modified sense, but *Kick* **causes** G **if** F would not be true in it. So, in contrast to what occurs in $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$ (with its actual semantics), the indefiniteness in D_1 would not be resolved in D_3.

Since in $\mathcal{L}_{\mathrm{C}}$ all actions are assumed to be deterministic, the full expressive capacity of MPL is not needed in $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$. In particular, in $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$ the modalities $\langle\alpha\rangle$ and $[\alpha]$ are indistinguishable in any situation in which α is executable. In the next section, we will extend the language $\mathcal{L}_{\mathrm{C}}$ to allow the specification of nondeterministic actions. The extended language will make better use of MPL.

5 The Language $\mathcal{L}_{\mathrm{CN}}$

In this section we define the syntax and semantics of the high-level action description language $\mathcal{L}_{\mathrm{CN}}$ as an extension of $\mathcal{L}_{\mathrm{C}}$. In $\mathcal{L}_{\mathrm{CN}}$, unlike $\mathcal{L}_{\mathrm{C}}$, actions are assumed to be deterministic only by default. Explicit syntax is provided for saying otherwise. We again assume a fixed signature $\langle \mathbf{A}, \mathbf{F} \rangle$, where $\mathbf{A}$ is a set of action names and $\mathbf{F}$ is a set of fluent names.

5.1 The Syntax of $\mathcal{L}_{\mathrm{CN}}$

The language $\mathcal{L}_{\mathrm{CN}}$ extends $\mathcal{L}_{\mathrm{C}}$ by including a single new kind of proposition. A *possible-effect proposition* is an expression of form

$$A \textbf{ possibly causes } \phi \textbf{ if } \psi \tag{6}$$

[6]This claim can be made precise by assigning truth conditions to effect propositions as follows: $(\langle \mathcal{S}, R, V \rangle, x) \models A$ **causes** ϕ **if** ψ iff for all $S \in \mathcal{S}$, if $S \models \psi$ then for all $S' \in R(A,S)$, $S' \models \phi$.

[7]This general point was argued by Tom Costello at the IJACI-95 Workshop on Nonmonotonic Reasoning, Action and Change.

where A is an action name, and ϕ and ψ are fluent formulas. If ψ is *True*, we write simply A **possibly causes** ϕ. Intuitively, (6) says that in every state that satisfies ψ, doing A possibly results in a next state in which ϕ is true.[8]

A *domain description* (or simply a *domain*) in $\mathcal{L}_{\text{CN}}$ is a set of effect propositions, possible-effect propositions, and causal laws.

5.2 The Semantics of $\mathcal{L}_{\text{CN}}$

In this section we define the conditions under which a transition system for $\mathcal{L}_{\text{CN}}$ is a model of a domain description.

Let $\langle \mathcal{S}, R, V \rangle$ be a transition system for $\mathcal{L}_{\text{CN}}$, D be a domain description in $\mathcal{L}_{\text{CN}}$, A be an action name in $\mathbf{A}$, and S be a state in $\mathcal{S}$. By the *possible explicit effects* of A in S according to D — in symbols, $P_D(A, S)$ — we designate the set of sets $Cn(E)$, where E is either $E_D(A, S)$ or $E_D(A, S) \cup \{\phi\}$, for some ϕ such that there is a possible-effect proposition in D of the form A **possibly causes** ϕ **if** ψ and $S \models \psi$.

As an example, if F is true in a state S, $P_{D_2}(Kick, S)$ is

$$\{Cn(\{\neg F\}), Cn(\{\neg F, G\}), Cn(\{\neg F, H\})\}\,.$$

Notice that by ensuring that the sets of formulas in $P_{D_2}(Kick, S)$ are deductively closed, we avoid the situation in which $P_{D_2}(Kick, S)$ contains two distinct but logically equivalent sets of formulas.

Why are the possible-effect propositions considered only singly when calculating the possible explicit effects of an action? For example, in D_2 why do we not also consider the possiblity that G and H are both directly caused by *Kick*? The basic reason is that possibility does not distribute over conjunction. Thus, the fact that *Kick* possibly causes G and possibly causes H does not imply that it possibly causes both G and H. Indeed, in D_2 we know that it cannot cause both G and H, for there is no state in which both G and H are true. Our policy is a conservative one which counts as possible explicit effects only those that are specified as such in the domain description. (Recall that intuitively the explicit effect $E_D(A, S)$ is necessary and, therefore, also possible.)

Let S an interpretation of $\mathbf{F}$, E be a set of fluent formulas, and C be a set of inference rules. In [McCain and Turner, 1995], $Res_C(E, S)$ was defined as the set of interpretations S' such that

$$S' = \{L : (S \cap S') \cup E \vdash_C L\}.$$

Intuitively, $Res_C(E, S)$ is the set of next states that might, so far as we know, result from performing an action with the explicit effect E in the state S given background knowledge C. Hence, a multiplicity of states in $Res_C(E, S)$ is an indication of indefiniteness, not nondeterminism.

A transition system $\langle \mathcal{S}, R, V \rangle$ is a *model* of a domain D in $\mathcal{L}_{\text{CN}}$ if it satisfies the following three conditions:

[8]The syntax of possible-effect propositions is similar to that used in [Denecker and De Schreye, 1993] and [Thielscher, 1994].

(1) The set $\mathcal{S}$ of states is the set of interpretations I of $\mathbf{F}$ such that $Cn(I)$ is closed under $C(D)$.

(2) For all action names A and states S,

   (a) if $R(A,S) \neq \emptyset$ then there exists a function f from $P_D(A,S)$ onto $R(A,S)$ such that for each $E \in P_D(A,S)$, $f(E) \in Res_{C(D)}(E,S)$ (the purpose of the function f is explained below), and

   (b) if $R(A,S) = \emptyset$ then for some $E \in P_D(A,S)$, $Res_{C(D)}(E,S) = \emptyset$.

(3) For every fluent name F and state S, $V(F,S) = S(F)$.

We refer to (1)–(3) as the *model conditions* for $\mathcal{L}_{\text{CN}}$. Model conditions (1) and (3) are the same as for $\mathcal{L}_{\text{C}}$. The role of the function f in (2a) is to select (for inclusion in $R(A,S)$) one state $f(E)$ from $Res_{C(D)}(E,S)$, for each $E \in P_D(A,S)$. Intuitively, each such selection settles the indefiniteness (if any) associated with E in S in one way. According to model condition (2b), an action is executable, unless one of its "possible effects" is impossible.

The following proposition shows that $\mathcal{L}_{\text{CN}}$ properly extends $\mathcal{L}_{\text{C}}$.

Proposition 1 *Let D be a domain description in $\mathcal{L}_{\text{C}}$. The semantics for $\mathcal{L}_{\text{C}}$ and $\mathcal{L}_{\text{CN}}$ specify the same set of models for D.*

Example 2 (cont'd) The reader may now verify our earlier claim that the domain description of Example 2 has a unique model $\langle \mathcal{S}, R, V \rangle$. The set $\mathcal{S}$ contains the following three states.

$$S_1 = \{F, \neg G, \neg H\} \qquad S_2 = \{\neg F, G, \neg H\} \qquad S_3 = \{\neg F, \neg G, H\}$$

The nondeterministic transition function R is defined as follows.

$$R(Kick, S_1) = \{S_2, S_3\} \qquad R(Kick, S_2) = \{S_2\} \qquad R(Kick, S_3) = \{S_3\}$$

6 The Language $\mathcal{L}_{\text{CN}}^{\text{MPL}}$

In this section we combine $\mathcal{L}_{\text{CN}}$ and MPL in the same manner in which $\mathcal{L}_{\text{C}}$ and MPL were combined in Section 4. The syntax and semantics of $\mathcal{L}_{\text{CN}}^{\text{MPL}}$ are briefly described in the following two paragraphs.

A *domain description* (or *domain*) D in $\mathcal{L}_{\text{CN}}^{\text{MPL}}$ is a set of formulas belonging either to $\mathcal{L}_{\text{CN}}$ or MPL. By $D^{\mathcal{L}_{\text{CN}}}$ we designate the set of effect propositions, causal laws, and possible-effect propositions in D. By D^{MPL} we designate the set of MPL formulas in D. Thus, D is the union of these two sets.

A structure (M, x) for $\mathcal{L}_{\text{CN}}^{\text{MPL}}$ is a *model* of a domain D of $\mathcal{L}_{\text{CN}}^{\text{MPL}}$ if M is a model of $D^{\mathcal{L}_{\text{CN}}}$ and (M, x) is a model of D^{MPL}.

Example 2 (cont'd) Suppose we add to the domain description D_2 of Example 2 the formulas (4) and (5), which previously we added to D_1. Let us call the augmented domain description D_4. In contrast to what we observed in D_3, adding the two MPL formulas does not resolve any indefiniteness with respect to the possible effects of actions in D_2. Indeed, there is no such indefiniteness in D_2. The two formulas do, however, contribute to the following interesting combination of modal consequences of D_4.

- Kicking twice can lead to a situation in which the ball is in location H.

$$\langle Kick \rangle \langle Kick \rangle H$$

- In fact, we will kick, but doing so will leave the ball in location G.

$$\langle\langle Kick \rangle\rangle G$$

- Given how kicking the first time will turn out, kicking a second time cannot lead to a situation in which the ball is in location H.

$$\langle\langle Kick \rangle\rangle \neg \langle Kick \rangle H$$

According to our common sense view of the world, the present and past are now settled and cannot be other than they are, while the future in at least some respects is unsettled and could turn out in more than one way. As time marches on, however, future possibilities tend to become closed off; what was still possible at an earlier time may become impossible later on, and what was merely possible may become inevitable. This general phenomenon is illustrated by the above mentioned consequences of D_4. One reason for describing nondeterministic actions is to capture such common sense judgments as these.

7 Conclusion

We have defined two high-level action description languages, $\mathcal{L}_{\mathrm{C}}$ and $\mathcal{L}_{\mathrm{CN}}$. In $\mathcal{L}_{\mathrm{C}}$ actions are assumed to be deterministic, but domain descriptions may be indefinite. Indefiniteness is reflected in the existence of multiple models for a domain. In $\mathcal{L}_{\mathrm{CN}}$ the situation wrt indefiniteness is the same, but it is also possible to specify that actions are nondeterministic. Nondeterminism is reflected in $\mathcal{L}_{\mathrm{CN}}$ in the existence of multiple next states within individual models.

In both $\mathcal{L}_{\mathrm{C}}^{\mathrm{MPL}}$ and $\mathcal{L}_{\mathrm{CN}}^{\mathrm{MPL}}$, indefiniteness in the specification of the effects of actions in states can sometimes be resolved by assertions about facts in situations. The importance of model condition (1) in this regard was discussed in Section 4. In the course of this discussion, we briefly considered an alternative semantics which allowed domains to be indefinite wrt the set of states. Indefiniteness of this kind might in some ways resemble nondeterminism. For instance, it would allow an action to have different effects in states that agree on the values of the fluent names. (Nondeterminism allows even more; namely, different possible effects in the very same state.) Indefiniteness wrt the set of states may turn out to be very important in formalizing action domains, perhaps even more important than nondeterminism. This is a subject for further study.

Acknowledgments

The author would like to thank Vladimir Lifschitz and Hudson Turner for many useful discussions related to the subject of this paper. This work was partially supported by National Science Foundation grant #IRI-9306751.

References

[Boutilier and Friedman, 1995] Craig Boutilier and Nir Friedman. Nondeterministic actions and the frame problem. In *Working Notes: AAAI Spring Symposium on Extending Theories of Action: Formal Theory and Practical Applications*, pages 39–44, 1995.

[Denecker and De Schreye, 1993] Mark Denecker and Danny De Schreye. Representing incomplete knowledge in abductive logic programming. In Dale Miller, editor, *Logic Programming: Proceedings of the 1993 Int'l Symposium*, pages 147–163, 1993.

[Fischer and Ladner, 1979] M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *J. Comp. and Sys. Sci.*, 18:194–211, 1979.

[Gelfond and Lifschitz, 1993] Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17:301–322, 1993.

[Kartha and Lifschitz, 1994] G. Neelakantan Kartha and Vladimir Lifschitz. Actions with indirect effects. 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. In *Proc. of IJCAI-95*, pages 1970–1975, 1995.

[Lifschitz, 1995] Vladimir Lifschitz. Two components of an action language. Unpublished manuscript, 1995.

[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.

[McCain and Turner, 1995] Norman McCain and Hudson Turner. A causal theory of ramifications and qualifications. In *Proc. of IJCAI-95*, pages 1978–1984, 1995.

[Reiter, 1991] Raymond Reiter. The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 359–380. Academic Press, 1991.

[Thielscher, 1994] Michael Thielscher. Representing actions in equational logic programming. In *Proc. ICLP-94*, pages 207–224, 1994.