Representing Actions - I:
(Laws, Observations and Hypotheses)

Chitta Baral\(^1\), Michael Gelfond\(^1\) and Alessandro Provetti\(^2\)

\(^1\)Department of Computer Science
University of Texas at El Paso
El Paso, Texas 79968 U.S.A.
{chitta,mgefond}@cs.utep.edu
915-747-6952/5030 (voice/fax)

\(^2\)C.I.R.F.I.D.
Universitá di Bologna
I-40121 Bologna ITALY
provetti@cirfid.unibo.it

Abstract

We propose extensions \(\mathcal{L}_0\) and \(\mathcal{L}_1\) of the action description language \(\mathcal{A}\) that can express both actual and hypothetical situations, observations of the truth values of fluents in these situations (as opposed to hypothetical values of fluents expressible in \(\mathcal{A}\)), and observations of actual occurrences of actions. The corresponding entailment relation formalizes various types of common-sense reasoning about actions and their effects not modeled by the previous approaches. We then formalize the notion of planning from the current situation using \(\mathcal{L}_1\).

Introduction

To perform nontrivial reasoning an intelligent agent situated in a changing domain needs the knowledge of causal laws that describe effects of actions changing the domain, the ability to observe and record occurrences of these actions and the truth values of fluents\(^1\) at particular moments of time. Discovery of methods of representing this kind of information in a form allowing various types of reasoning about the dynamic world and at the same time tolerant to future updates is one of the central problems of knowledge representation.

Recently, there has been several efforts towards systematic development of provably correct methods (Gelfond & Lifschitz 1992; Sandewall 1992; Lesperance \textit{et al.} 1994) to reason about actions. Our approach is an extension of (Gelfond & Lifschitz 1992) where the authors introduced the high-level action description language \(\mathcal{A}\) capable of expressing causal laws describing effects of actions as well as statements about values of fluents in possible states of the world.

In the last two years the syntax and semantics of \(\mathcal{A}\) were expanded to allow descriptions of the effects of concurrent and non-deterministic actions as well as descriptions of global constraints expressing time independent relations between fluents (Baral & Gelfond 1993; Kartha & Lifschitz 1994; Bornscheuer & Thielshcer 1994). We also have by now a collection of sound and often complete translations from domain descriptions in these languages into disjunctive, abductive and equational logic programs (Dung 1993; Denecker & De Schreye 1993; Holldobler & Thielsher 1993; Turner 1994). This work helped to better understand the underlying ontological principles of reasoning about actions as well as advantages and limitations of general-purpose non-monotonic formalisms. It also allowed to establish equivalence of some of the previously known theories of actions seemingly based on different intuitions, languages and logics (Kartha 1993) and stimulated work on the theory and implementation of logic programming languages (Apt \& Bezem 1991; Turner 1993; Lifschitz & Turner 1994).

The goal of this paper\(^2\) is to further expand the expressive power of \(\mathcal{A}\) and its dialects. In particular, we propose an extension of \(\mathcal{A}\) that can express actual situations, observations of the truth values of fluents in these situations (as opposed to hypothetical values of fluents expressible in \(\mathcal{A}\)), and observations of actual occurrences of actions. The corresponding entailment relation formalizes various types of common-sense reasoning about actions and their effects not modeled by the previous approaches.

We use this extension of \(\mathcal{A}\) to formalize planning in a changing environment. The following example (a simpler version of the London-Glasgow problem (McCarthy)) further explains our goal.

John has the knowledge that if he has a car then by doing the action drive-to-the-airport he will be at-the-airport. Similarly, if the action hit-car occurs then he will not have-a-car, if the action rent-a-car occurs then he will have-a-car, and if the action pack occurs he will have his suitcase packed. He knows that he has a car and his suitcase is unpacked and his goal is to bring his packed suitcase to the airport. His plan of

\(^1\)By fluents in this paper we mean propositions whose truth values depend on time.

packing the suitcase and driving to the airport is adequate to achieve his goal. He then follows his plan and starts packing his suitcase. But after finishing packing he observes his car being hit. Following the rest of his original plan will no longer achieve his goal. Instead the plan of first performing rent-a-car and then performing drive-to-the-airport would be adequate.

Our language should allow elegant representation of the above story and should have a powerful mechanism to reason about the above plans. In the next sections we discuss the syntax and semantics of such a language.

**Syntax of $L_0$**

We will start with the description of a language $L_0$ capable of expressing actual observations. We will then extend $L_0$ to express hypotheses.

The alphabet of $L_0$ consists of three disjoint nonempty sets of symbols $F, A$ and $S$, called *fluents*, *actions*, and *actual situations*. Elements of $A$ and $S$ will be denoted by (possibly indexed) letters $a$ and $s$ respectively. We will also assume that $S$ contains two special situations $s_0$ and $s_N$ called *initial* and *current* situations. The 'N' in $s_N$ corresponds to the word 'Now'.

A fluent literal is a fluent possibly preceded by $\neg$. Fluent literals will be denoted by (possibly indexed) letters $f$ and $p$ (possibly preceded by $\neg$). $\neg f$ will be equated with $\neg f$.

There are two kinds of propositions in $L_0$ called causal laws and facts.

An *effect law* is an expression of the form

$$a \text{ causes } f \text{ if } p_1, \ldots, p_n$$

where $a$ is an action, and $f, p_1, \ldots, p_n$ are fluent literals. $p_1, \ldots, p_n$ are called *preconditions* of (1). We will read this law as "$f$ is guaranteed to be true after the execution of an action $a$ in any state of the world in which $p_1 \ldots p_n$ are true". If $n = 0$, we write the effect law as

$$a \text{ causes } f \quad (1a)$$

An atomic fluent fact is an expression of the form

$$f \text{ at } s$$

where $f$ is a fluent literal and $s$ is a situation. (Unless otherwise stated by situations we will mean actual situations.) The intuitive reading of (2) is "$f$ is observed to be true in situation $s$".

An atomic occurrence fact is an expression of the form

$$\alpha \text{ occurs at } s \quad (3)$$

where $\alpha$ is a sequence of actions, and $s$ is a situation. It states that "the sequence $\alpha$ of actions was observed to have occurred in situation $s$". (We assume that actions in the sequence follow the next action in the sequence immediately).

An atomic precedence fact is an expression of the form

$$s_1 \text{ precedes } s_2 \quad (4)$$

where $s_1$ and $s_2$ are situations. It states that situation $s_1$ occurred after situation $s_2$.

Propositions of the type (1) express general knowledge about effects of actions and hence 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 a *domain description* of $L_0$. The sets of laws and facts of a domain description $D$ will be denoted by $D_l$ and $D_r$ respectively. We will only consider domain descriptions whose propositions do not contain the situation constant $s_N$.

To see how the domain descriptions of $L_0$ can be used to represent knowledge about actions let us consider the following example:

**Example 1** Suppose that we are given a series of observations about "Fred":

(a) when the water pistol was squirted *Fred* was seen to be alive and dry,

(b) in a later moment a shot was fired at *Fred*.

Suppose also that it is generally known that

(c) squirt makes *Fred* wet,

(d) shooting makes *Fred* dead

The above information can be represented by a domain description $D_1$ consisting of the following propositions:

- $(p_1)$ alive at $s_0$
- $(p_2)$ dry at $s_0$
- $(p_3)$ squirt occurs at $s_0$
- $(p_4)$ shoot occurs at $s_1$
- $(p_5)$ squirt causes $\neg$ dry
- $(p_6)$ shoot causes $\neg$ alive

To complete the description of $D_1$ we need to define its language. For simplicity we assume that this language contains only the fluents and actions explicitly mentioned in the propositions of $D_1$. Unless stated otherwise the same assumption will be made in other examples throughout this paper.

Domain descriptions in $L_0$ are used in conjunction with the following informal assumptions which clarify the description's meaning:

(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 causal laws.

(d) No actions occur except those needed to explain the facts in the domain description.
Actions do not overlap or happen simultaneously. These assumptions give an intuitive understanding of domain descriptions of \( \mathcal{L}_0 \).

Consider for instance domain description \( D_1 \) from Example 1. It is easy to see that \( D_1 \) together with assumption (d) implies that squirt is the only action which occurs between \( s_0 \) and \( s_1 \) and that shoot is the only action which occurs between \( s_1 \) and \( s_N \). Using \( D_1 \) with the assumptions (a) - (e) we can conclude at the moment \( s_0 \) Fred is wet but alive while at the moment \( s_N \) (i.e., at the end of the story) he is wet and dead. Our goal in this paper is to build a mathematical model which will help us to better understand and eventually mechanize these types of arguments. As the first step we suggest a semantics of domain descriptions of \( \mathcal{L}_0 \) which precisely specify the sets of acceptable conclusions which can be reached from such descriptions and assumptions (a)-(e).

### Semantics of \( \mathcal{L}_0 \)

In this section we introduce a semantics of a domain description in \( \mathcal{L}_0 \). We start with defining causal models of \( D \) and proceed by explaining when facts are true in these models.

A state is a set of fluent names. A causal interpretation is a partial function \( \Psi \) from sequences of actions to states such that:

1. Empty sequence \( \epsilon \) belongs to the domain of \( \Psi \) and
2. \( \Psi \) is prefix-closed \(^4\).

\( \Psi(\epsilon) \) is called the initial state of \( \Psi \). The partial function \( \Psi \) serves as interpretations of the laws of \( D \). If \( a \) belongs to the domain of \( \Psi \) we say that \( a \) is possible in the initial state of \( \Psi \).

Given a fluent \( f \) and a state \( \sigma \), we say that \( f \) holds in \( \sigma \) if \( f \) is true in \( \sigma \) and \( \neg f \) holds in \( \sigma \) if \( f \) is false in \( \sigma \). The truth of a propositional formula \( C \) with respect to \( \sigma \) is defined as usual.

To better understand the role \( \Psi \) plays in interpreting domain descriptions let us first use it to define models of descriptions consisting entirely of effect 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)-(e).

A fluent \( f \) is an (immediate) effect of (executing) \( \sigma \) in \( D \) whose preconditions hold in \( \sigma \). Let

\[
E^+_a(\sigma) = \{ f : f \text{ is an effect of } a \text{ in } \sigma \},
\]

\[
E^-_a(\sigma) = \{ f : \neg f \text{ is an effect of } a \text{ in } \sigma \}
\]

\( \text{Res}(a, \sigma) = \sigma \cup E^+_a(\sigma) \setminus E^-_a(\sigma) \).

The following definition captures the meaning of effect laws of \( D \).

**Definition 1** A causal interpretation \( \Psi \) satisfies effect laws of \( D \) if for any sequence \( \alpha \circ a \) from the language of \( D \)

\[
\Psi(\alpha \circ a) = \text{Res}(a, \Psi(\alpha)) \text{ if } E^+_a(\Psi(\alpha)) \cap E^-_a(\Psi(\alpha)) = \emptyset
\]

and undefined otherwise.

We say that \( \Psi \) is a causal model of \( D \) if it satisfies all the effect laws of \( D \).

Let \( D \) be an arbitrary domain description and let a causal interpretation \( \Psi \) 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 \( S \). To do that we consider a mapping \( \Sigma \) from \( S \) to sequences of actions from the language of \( D \). This mapping will be called a situation assignment of \( S \) if it satisfies the following properties:

1. \( \Sigma(s_0) = \epsilon \), and
2. for every \( s_i \in S \), \( \Sigma(s_i) \) is a prefix of \( \Sigma(s_N) \).

**Definition 2** An interpretation \( M \) of \( \mathcal{L}_0 \) is a pair \((\Psi, \Sigma)\), where \( \Psi \) is a causal model of \( D \), \( \Sigma \) is a situation assignment of \( S \) and \( \Sigma(s_N) \in \text{domain of } \Psi \).

\( \Sigma(s_N) \) will be called the actual path of \( M \).

Now we can define truth of facts of \( D \) w.r.t. an interpretation \( M \). Facts which are not true in \( M \) will be called false in \( M \).

**Definition 3** For any interpretation \( M = (\Psi, \Sigma) \).

1. \( (f \text{ at } s) \) is true in \( M \) (or satisfied by \( M \)) if \( f \) is true in \( \Psi(\Sigma(s)) \).
2. \( (a \text{ occurs at } s) \) is true in \( M \) if \( \Sigma(s) \circ a \) is a prefix of the actual path of \( M \).
3. \( (s_1 \text{ precedes } s_2) \) is true in \( M \) if \( \Sigma(s_1) \) is a proper prefix of \( \Sigma(s_2) \).
4. Truth of non-atomic facts in \( M \) is defined as usual.

A set of facts is true in interpretation \( M \) if all its members are true in \( M \).

To complete the definition of the model we need only to formalize the assumption (d). This is done by imposing a minimality condition on the situation assignments of \( S \) which leads to the following

**Definition 4** An interpretation \( M = (\Psi, \Sigma) \) will be called a model of a domain description \( D \) in \( \mathcal{L}_0 \) if the following conditions are satisfied:

\( ^3 \)In the extended version of the paper we will allow simultaneous actions. We exclude it here for simplicity.

\( ^4 \)By prefix closed we mean that for any sequence of actions \( a \) and action \( a \), if \( a \circ a \) is in the domain of \( \Psi \) then so is \( a \). (Recall that \( \circ \) denotes concatenation, and \( a \circ a \) means the sequence of actions where \( a \) follows \( a \) ).
(1) Ψ is a causal model of D,
(2) facts of D are true in M, and
(3) there is no other interpretation N = (Ψ, Σ') such that N satisfies the conditions (1) and (2) and Σ' (s_N) is a subsequence5 of Σ (s_N).

The following proposition shows that for a model (Ψ, Σ) of a domain description in L₀, Ψ is completely determined by its initial state.

Proposition 1 Let M₁ = (Ψ₁, Σ₁) and M₂ = (Ψ₂, Σ₂) be two models of a domain description D in language L₀. If Ψ₁ (f) = Ψ₂ (f) then Ψ₁ = Ψ₂.

Corollary 1 Let D be a domain description in language L₀. If for all models of D, Ψ (f) is uniquely defined then Ψ is also uniquely defined.

A domain description D is said to be consistent if it has a model.

Definition 5 A domain description D entails a fact p (written as D ⊨ p) iff p is true in all models of D.

Definition 6 A domain description D is said to define a unique actual path if for any two situations s₁ and s₂ that are explicitly mentioned in D, D ⊨ s₁ precedes s₂ or D ⊨ s₂ precedes s₁.

Lemma 1 Let D be a domain description that defines a unique actual path, and the only atomic fluent facts which occur in propositions of D are of the form f at s₀. Then situation assignments of all models of D coincide on s_N and on all the situations explicitly mentioned in D.

Examples

In this section we illustrate by way of examples how domain descriptions are used to represent information and how the above notion of entailment captures informal arguments based on the information from these descriptions and the informal assumptions (a) - (e). We start with Example 1 from Section .

Proposition 2 Consider the domain description D₁ from Example 1. We have

D₁ ⊨ ((¬dry ∧ ¬alive) at s₀), and
D₁ ⊨ ((¬dry ∧ alive) at s₁).

Example 2 (Reasoning by cases) Let us consider a modification of Example 1 where there is a precondition of being loaded for the shoot action to be deadly and where there are two guns at least one of which is initially loaded.

\[
\begin{align*}
(1) & \text{alive at } s₀, \\
(2) & \text{loaded₁ at } s₀ \lor \text{loaded₂ at } s₀, \\
(3) & \text{[shoot₁, shoot₂] occurs at } s₀, \\
(4) & \text{shoot₁ causes ¬alive if loaded₁,} \\
(5) & \text{shoot₂ causes ¬alive if loaded₂}.
\end{align*}
\]

D₂

Proposition 3 D₂ ⊨ ¬alive at sₙ.

Example 3 [Explaining observations] Let us now consider a modification of Example 1 where instead of (b) "In a later moment a shot was fired at Fred", we have (b') In a later moment Fred was observed to be dead and where we assume that our domain contains unit actions a₁, ..., aₙ different from squirt and shoot. The resultant story can be represented by a domain description D₃ consisting of the propositions (p1) - (p₄) and (p₆) - (p₇) of D₁ and the following proposition.

(p₅') ¬alive at s₁.

Proposition 4 D₃ ⊨ [squirt, shoot] occurs at s₀.

Domain descriptions language and hypothetical reasoning

Even though domain descriptions of L₀ can express types of knowledge and reasoning not easily expressible in other variants of A, they lack the ability of the latter to do hypothetical reasoning. Even the simple original version of A allows propositions of the form

\[ f \text{ after } a₁, ..., aₙ \]

read as "Assuming that the sequence of actions \([a₁, ..., aₙ]\) occurs starting at the initial situation, fluent f would be true in the resulting situation", which are used to query domain descriptions about possible outcomes of actions. In this section we introduce propositions of the form

\[ f \text{ after } [a₁, ..., aₙ] \text{ at } s \]

called hypotheses which slightly generalizes (5). Hypotheses are read as "Assuming that the sequence of actions \([a₁, ..., aₙ]\) occur starting at the situation s fluent f would be true in the resulting situation".

If s in (6) is sₙ then we simply write

\[ f \text{ after } [a₁, ..., aₙ] \text{ at } s \]

If n in (7) is 0, then we simply write

\[ f \text{ currently } \]

The language L₀ when augmented with propositions of the form (6) is referred to as L₁. Even though L₁ extends L₀ by allowing propositions of the form (6), domain descriptions in L₁ are in the language of L₀, i.e., hypotheses are not part of a domain description in L₁. Only laws and facts are part of a domain description. But, hypotheses can be entailed from a domain description in L₁. We now define this entailment.

Let D be a domain description and M = (Ψ, Σ) be an interpretation of D. We say that a hypothesis (6) is true in interpretation M if f is true in Ψ(Σ(s) o [a₁, ..., aₙ]).

A set H of hypotheses is true in M if every hypothesis from H is true in M.
Definition 7 Let D be a domain description and H be a hypothesis in L1. We say $D \models H$ iff H is true in all models of D.

Let $H_1$ and $H_2$ be two sets of hypotheses. We say that the premise $H_1$ entails conclusion $H_2$ in D if $H_2$ is true in every model of D in which $H_1$ is true. We will denote this by $H_1 \models_D H_2$.

Proposition 5 $\emptyset \models_D H$ iff $D \models H$.

A set of hypotheses H is inconsistent w.r.t. a domain description D if no model of D satisfies H.

It is important to notice that the entailment relation ($\models_D$) defined by a domain description D is monotonic - addition of new hypothesis to the set of hypotheses $H_1$ can only decrease the set of models of D satisfying it and hence can only increase the set of conclusions. Non-monotonicity occurs only when new information about the real world (i.e. new laws or new facts) are added to a reasoner's knowledge.

Example 4 Consider the following domain description $D_4$ consisting of (p1) and (p2):

(p1) shoot causes ~alive if loaded
(p2) load causes loaded

Suppose that, given the domain description $D_4$, a reasoner would like to know if Fred would be dead after shooting under the assumption that initially the gun is loaded. Notice that both statements are hypothetical and therefore are naturally represented as follows:

$H_1 = \{\text{loaded after } [] \text{ at } s_0\}$
$H_2 = \neg\text{alive after } [\text{shoot}]\text{ at } s_0$

The question can be formulated as $H_1 \models_D H_2$? The answer is obviously yes.

Planning in a dynamic environment

Example 5 Consider the story about John from the introduction. It has actions pack, drive, rent, and hit and fluents home, at_airport, has_car, and packed. The effects of the actions together with some initial conditions are described by the domain description

\[
D_5 = \{(\text{packed } \land \text{at_airport}) \ \text{after} \ \alpha_0, 6 \}
\]

where $\alpha_0 = [\text{pack, drive}]$. Theoretically, such an $\alpha_0$ can be found by generating sequences of actions and testing them using the entailment relation of D. More sophisticated methods of course are needed for practical planning but we will not discuss them in this paper.

Satisfied with the plan John packs his suitcase. Execution of this action is reported by expanding $D_5$ by (f4) pack occurs_at s1 (f5) s0 precedes s1

We denote the resulting description by $D_6$. All he needs to do now is to execute $\alpha_1 = [\text{drive}]$. Suppose however, that John observes that his car being hit by a truck, i.e. $D_7$ is obtained from $D_6$ by adding the statements, (f6) hit occurs_at s2 and (f7) s1 precedes s2

It is easy to see that $D_7 \models (\text{packed } \land \text{at_airport})$ after $\alpha_1$ It is again easy to check that $D_7 \models (\text{packed } \land \text{at_airport})$ after $\alpha_2$ where $\alpha_2 = [\text{rent, drive}]$. John goes on to execute $\alpha_2$ (this time without unpleasant interruptions). 

As evident from the above example, the ability to express the current situation, record facts and do hypothetical reasoning makes $L_1$ appropriate for use in designing intelligent agents capable of planning in the changing environment. More formally,

Definition 8 Let D be a domain description in $L_1$ and G be a set of fluent literals. A sequence $\alpha$ of actions is a plan for achieving a goal G from the current situation if $D \models f \ \text{after} \ \alpha$ for every fluent literal $f \in G$.

Proposition 6 Let D be a domain description with the unique actual path and let $s_k$ be a situation in D such that there does not exist a situation $s$ in D such that $D \models s_k \ \text{precedes} \ s$. Then for any sequence $\alpha = a_0 \beta$ of actions and any fluent $f$, $D \models (f \ \text{after} \ \alpha)$ iff $D \cup \{(a \ \text{occurs_at} \ s_k)\} \models (f \ \text{after} \ \beta)$.

Conclusions

We proposed the extensions $L_0$ and $L_1$ of the action description language A able to express actual situations, observations of the truth values of fluents in these situations, and observations of actual occurrences of actions. Entailment relation in this language allows modeling of various types of hypothetical reasoning. This feature, together with the ability to denote current actual situation, allows to reason about the design and correctness of plans in the changing environment. In the full paper (accessible via http://cs.utep.edu/chitta/chitta.html) we present...
provenly correct implementation of limited forms of reasoning in $L_1$ based on translation of domain descriptions of $L_1$ into logic programs.

The work in this paper can be extended in several directions. In particular, $L_1$ can be easily generalized to deal with partially defined actions, to allow concurrent and non-deterministic actions (Baral & Gelfond 1993), and global constraints (Kartha & Lifschitz 1994). Another promising direction of research is to construct planners based on $L_1$ (see full paper for more on this), particularly using extensions of logic programming and situation calculus.

Our work is obviously a continuation of the approach of formalizing actions suggested in (Gelfond & Lifschitz 1992) which is deeply rooted in situation calculus (McCarthy & Hayes 1969; Gelfond, Lifschitz, & Rabinov 1991). Our formalization, especially in its logic programming form, can be viewed as a combination of situation calculus with another prominent approach to formalizing actions - event calculus of (Kowalski & Sergot 1986). To the best of our knowledge the first paper combining the two in one formalism is (Pinto & Reiter 1993). Ideologically, their approach is similar to ours. In (Pinto & Reiter 1993) the situation calculus presented as a theory of classical logic (with some second order features) which plays the role of our action description language. Our approach seem to allow more forms of incompleteness in the representation of the domain but the investigation of the precise relationship is the subject for future work.

References


7See the October 94 issue of Journal of Logic and Computation for some very recent related works, particularly the one by Miller and Shanahan.