An Action Language Based on Causal Explanation: Preliminary Report

Enrico Giunchiglia
DIST — Università di Genova
16145 Genova, Italy

Vladimir Lifschitz
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712, USA

Abstract

Action languages serve for describing changes that are caused by performing actions. We define a new action language $\mathcal{C}$, based on the theory of causal explanation proposed recently by McCain and Turner, and illustrate its expressive power by applying it to a number of examples. The mathematical results presented in the paper relate $\mathcal{C}$ to the Baral—Gelfond theory of concurrent actions.

1 Introduction

Representing properties of actions has been the subject of many papers and two recent books [Sandewall, 1995], [Shanahan, 1997]. One direction of work makes use of "action languages," such as $\mathcal{A}$ [Gelfond and Lifschitz, 1993] and its dialects. An action language serves for describing the effects of actions on fluents. The meaning of a set of propositions in an action language can be represented by a "transition diagram."

In this paper we define a new action language $\mathcal{C}$, based on the theory of causal explanation proposed in [McCain and Turner, 1997] and extended in [Lifschitz, 1997a]. The main idea of this theory [Geffner, 1990] is to distinguish between the claim that a formula is true and the stronger claim that there is a *cause* for it to be true. This idea leads to a semantics for "causal rules" of the form

$$F \leftarrow G \tag{1}$$

where F and G are formulas of classical logic. This rule expresses that there is a cause for F if G is true.

The distinction between being true and being caused is used here to define the syntax and semantics of a language for representing transition diagrams. We use the new language $\mathcal{C}$ to formalize a number of examples of reasoning about action, relate $\mathcal{C}$ to causal logic, and compare it with $\mathcal{A}$ and with the extension of $\mathcal{A}$ proposed by Baral and Gelfond [1997] for describing the concurrent execution of actions.

In this preliminary report, discussion is limited to the propositional fragment of $\mathcal{C}$, in which all fluents are truth-valued, and neither fluents nor actions are allowed to have parameters. The full language is described in a forthcoming paper.

2 Language $\mathcal{C}$

2.1 Syntax and Semantics

A *propositional signature* is a set of propositional atoms. An *interpretation* of a propositional signature σ is a truth-valued function defined on σ.

Consider a propositional signature σ partitioned into the *fluent symbols* σ^{fl} and the *action symbols* σ^{act}. An *action* is an interpretation of σ^{act}. We will identify an action symbol A with the action that assigns the value $\mathtt{t}$ to A and the value $\mathtt{f}$ to all other action symbols. Such actions will be called *elementary*. An action can be viewed as a set of elementary actions—as the set of all action symbols to which it assigns $\mathtt{t}$. Intuitively, to execute an action a means to execute concurrently all elementary actions that belong to a.

There are two kinds of propositions in $\mathcal{C}$: *static laws* of the form

$$\textbf{caused } F \textbf{ if } G \tag{2}$$

and *dynamic laws* of the form

$$\textbf{caused } F \textbf{ if } G \textbf{ after } H, \tag{3}$$

where F, G, H are formulas of σ such that F and G do not contain action symbols. In a proposition of either kind, the formula F will be called its *head*.

An *action description* is a set of propositions.

Consider an action description D. A *state* is an interpretation of σ^{fl} that satisfies $G \supset F$ for every static law (2) in D. A *transition* is any triple $\langle s, a, s' \rangle$ where s, s' are states and a is an action; s is the *initial* state of the transition, and s' is its *resulting* state. A formula F is *caused* in a transition $\langle s, a, s' \rangle$ if it is

- the head of a static law (2) from D such that s' satisfies G, or
- the head of a dynamic law (3) from D such that s' satisfies G and $s \cup a$ satisfies H.

A transition $\langle s, a, s' \rangle$ is *causally explained* according to D if its resulting state s' is the only interpretation of σ^{fl} that satisfies all formulas caused in this transition.

The *transition diagram* represented by an action description D is the directed graph which has the states of D as nodes, and which includes an edge from s to s' labeled a for every transition $\langle s, a, s' \rangle$ that is causally explained according to D.

Two abbreviations are useful. A dynamic law of the form[1]

$$\textbf{caused } F \textbf{ if } \mathit{True} \textbf{ after } H$$

will be written as

$$\textbf{caused } F \textbf{ after } H.$$

Such propositions can be used to describe a direct effect of an elementary action. In this case F is the formula that is made true by executing this action (no matter what other elementary actions are performed concurrently), and H is the conjunction of the elementary action with the preconditions for its effect. Second, a dynamic law of the form

$$\textbf{caused } F \textbf{ if } F \textbf{ after } F$$

will be written as

$$\textbf{inertial } F.$$

Such propositions can be used to express the commonsense law of inertia understood as in [McCain and Turner, 1997]: if F has remained true then there is a cause for this.

[1] *True* stands for *False* $\supset$ *False*, where *False* is a 0-place connective assumed to be available in the language of propositional logic.

We will combine a group of propositions

$$\textbf{inertial } F_1, \ \textbf{inertial } F_2, \ldots$$

into

$$\textbf{inertial } F_1, F_2 \ldots.$$

2.2 Example

Let $\sigma^{fl} = \{P, Q\}$, $\sigma^{act} = \{A\}$, and let D consist of the propositions

$$\begin{array}{l} \textbf{inertial } P, \neg P, Q, \neg Q, \\ \textbf{caused } P \textbf{ after } Q \wedge A. \end{array} \tag{4}$$

The second line of (4) tells us that P is made true by the execution of A if the precondition Q is satisfied (as, for instance, in the familiar shooting example which corresponds to *Dead* as P, *Loaded* as Q, and *Shoot* as A).

According to the definitions above, (4) is shorthand for

$$\begin{array}{l} \textbf{caused } P \textbf{ if } P \textbf{ after } P, \\ \textbf{caused } \neg P \textbf{ if } \neg P \textbf{ after } \neg P, \\ \textbf{caused } Q \textbf{ if } Q \textbf{ after } Q, \\ \textbf{caused } \neg Q \textbf{ if } \neg Q \textbf{ after } \neg Q, \\ \textbf{caused } P \textbf{ if } \mathit{True} \textbf{ after } Q \wedge A. \end{array} \tag{5}$$

In this action description, there are 4 states (PQ, $P\overline{Q}$, $\overline{P}Q$, $\overline{P}\,\overline{Q}$)[2] and 2 actions ($A$ and $\overline{A}$). Consequently there are $4 \times 2 \times 4 = 32$ transitions. Out of these, 8 transitions are causally explained: $\langle \overline{P}Q, A, PQ \rangle$ and the transitions of the form $\langle s, a, s \rangle$ where $s \neq \overline{P}Q$ or $a \neq A$.

Figure 1 shows the corresponding transition diagram. We can see from it that each of the actions A, $\overline{A}$ can be executed in any state in exactly one way.

To check that the transition $\langle \overline{P}Q, A, PQ \rangle$ is causally explained, note that the formulas caused in this transition are the heads Q, P of the 3rd and 5th propositions in (5), and that the resulting state PQ of the transition is the only interpretation that satisfies both heads.

As another illustration of the definition, let us verify that $\langle \overline{P}Q, \overline{A}, PQ \rangle$ is not causally explained. The only formula caused in this transition is

[2]We represent a propositional interpretation by listing the literals that are satisfied by it. $\overline{L}$ is the literal complementary to L.

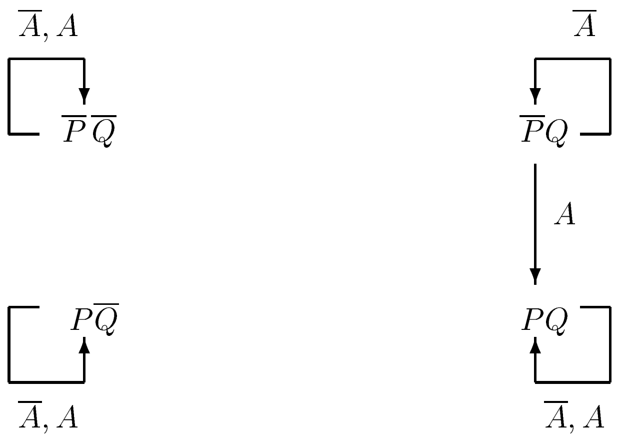

Figure 1: Transition diagram for action description (4).

the head Q of the 3rd proposition in (5). It is satisfied by more than one interpretation.

The mathematical theory presented in Section 3 makes it easier to compute the causally explained transitions in examples like this.

2.3 Relation to $\mathcal{A}$

"Effect propositions" of the language $\mathcal{A}$ from [Gelfond and Lifschitz, 1993] are essentially part of the new language $\mathcal{C}$. They correspond to dynamic laws of the form

$$\textbf{caused } L \textbf{ after } F \wedge A \tag{6}$$

where L is a literal, F a formula that does not contain action symbols, and A an action symbol. Propositions of this form will be called *$\mathcal{A}$-propositions*.

An action description D is an *$\mathcal{A}$-description* if it is the union of the set of propositions

$$\textbf{inertial } P, \neg P$$

for all fluent symbols P with a set of $\mathcal{A}$-propositions. For instance, (4) is an $\mathcal{A}$-description. Proposition 1 below shows that the transitions causally explained by an $\mathcal{A}$-description are characterized by the semantics that is given to effect propositions in the definition of $\mathcal{A}$.

To state this theorem, we need the following definitions. Let D be an $\mathcal{A}$-description. For any state s, action a and literal L of the fluent signature, we say that executing a in s *causes* L if D includes a dynamic law (6) with the head L such that $a \cup s$ satisfies $F \wedge A$. A transition $\langle s, a, s' \rangle$ is an *$\mathcal{A}$-transition* for D if s' satisfies every literal L such that

- executing a in s causes L, or
- executing a in s does not cause $\overline{L}$, and s satisfies L.

Proposition 1 *For any $\mathcal{A}$-description D, a transition is an $\mathcal{A}$-transition for D iff it is causally explained according to D.*

Note that this characterization is applicable to a transition $\langle s, a, s' \rangle$ even when a is not elementary, although such transitions are not covered by the semantics of $\mathcal{A}$.

3 Computing Causally Explained Transitions

According to [Lifschitz, 1997a], a *causal theory* is a finite set of rules (1), with some of the nonlogical constants of the underlying language designated as *explainable*. In this paper we only need the special case when the language is propositional.

The semantics of causal theories is defined in [Lifschitz, 1997a] by a translation that turns these theories into formulas of classical logic. A *model* of a causal theory T is an interpretation that satisfies the translation of T. A *theorem* of T is a formula that is entailed by the translation of T.

The formula representing T in classical logic is stronger than the conjunction of the material implications $G \supset F$ for all rules (1) of T; it contains an additional conjunctive term which makes the translation nonmonotonic (and which is similar in this sense to the minimality condition in the definition of circumscription). Details can be found in [Lifschitz, 1997a]; it is not necessary to know them to understand the computational procedure described below.

In Section 3.1 we define, for any positive integer n, a translation ct_n that turns any finite action description D into a causal theory. The models of $ct_n(D)$ correspond to "histories" of length n—to the paths of length n in the transition diagram represented by D. In particular, the models of $ct_1(D)$ correspond to the transitions causally explained according to D. Section 3.2 describes the process of literal completion, proposed in [McCain and Turner, 1997] and generalized in [Lifschitz, 1997a], which can be used to find the models of such theories.

3.1 Representing Histories in Causal Logic

The translation $ct_n(D)$ of an action description D is defined as follows. Its signature σ_n consists of $n+1$ disjoint copies σ_i^{fl} $(0 \leq i \leq n)$ of the fluent signature σ^{fl} and n disjoint copies σ_i^{act} $(0 \leq i < n)$ of the action signature σ^{act}. For any formula F of the original signature σ, by F_i we denote the result of replacing every atom in F by the corresponding atom from σ_i^{fl} or from σ_i^{act}. Intuitively, the subscript i represents time. For any fluent symbol P, the atom P_i expresses that P holds at time i. For any action symbol A, the atom A_i expresses that A is among the elementary actions executed between times i and $i+1$.

The rules of $ct_n(D)$ are

$$F_i \leftarrow G_i \qquad (0 \leq i \leq n)$$

for all static laws (2) in D, and

$$F_{i+1} \leftarrow G_{i+1} \wedge H_i \qquad (0 \leq i < n)$$

for all dynamic laws (3) in D. The explainable symbols of $ct_n(D)$ are the atoms from σ_i^{fl} for all $i > 0$.

For instance, the result of applying the translation ct_1 to action description (4) is the causal theory whose rules are

$$\begin{array}{l} P_1 \leftarrow P_1 \wedge P_0, \\ \neg P_1 \leftarrow \neg P_1 \wedge \neg P_0, \\ Q_1 \leftarrow Q_1 \wedge Q_0, \\ \neg Q_1 \leftarrow \neg Q_1 \wedge \neg Q_0, \\ P_1 \leftarrow \mathit{True} \wedge Q_0 \wedge A_0 \end{array} \tag{7}$$

and whose explainable symbols are P_1 and Q_1.

Proposition 2 below shows that there is a one-to-one correspondence between the models of $ct_n(D)$ and the paths of length n in the transition diagram represented by D. To define this correspondence, we need the following notation. Let I be an interpretation of σ_n. For any $i \leq n$, the interpretation $\mathit{State}_i[I]$ of σ^{fl} is defined by the condition: for every fluent symbol P,

$$\mathit{State}_i[I](P) = I(P_i).$$

For any $i < n$, the interpretation $\mathit{Action}_i[I]$ of σ^{act} is defined by the condition: for every action symbol A,

$$\mathit{Action}_i[I](A) = I(A_i).$$

Proposition 2 *For any finite action description D and positive integer n, an interpretation I of σ_n is a model of $ct_n(D)$ iff each of the triples*

$$\langle State_i[I], Action_i[I], State_{i+1}[I]\rangle \qquad (0 \leq i < n)$$

is a transition causally explained by D.

For instance, the claim that $\langle \overline{P}Q, A, PQ\rangle$ is a transition causally explained by (4)—we verified this claim in Section 2.2—can be equivalently expressed by saying that the interpretation $\overline{P_0}Q_0A_0P_1Q_1$ is a model of causal theory (7). The problem of computing all transitions that are causally explained by (4) is equivalent to the problem of finding all models of (7).

3.2 Literal Completion

The formula of classical logic that represents the meaning of a causal theory contains, generally, bound second-order variables. In case of a propositional causal theory, these variables are propositional and can be always eliminated, although the formula can become much longer in the process.

There is a special case, however—the case of "definite" theories—when a modification of the process of completion familiar from logic programming [Clark, 1978] allows us to construct a short formula that is equivalent to the given causal theory and has no new higher-order variables.

In propositional case, a causal theory T is *definite* if the head F of every rule $F \leftarrow G$ of T is a literal or contains no explainable symbols. For instance, (7) is definite. Generally, if the head of every proposition in an action description D is a literal then $ct_n(D)$ is a definite causal theory.

Literal completion differs from Clark's completion in that the "completed definition" of any explainable symbol P consists of *two* equivalences, one "positive" and one "negative." The positive completion formula is obtained from the rules whose head is P in exactly the same way as in [Clark, 1978]. The negative completion formula is generated in a similar way from the rules whose head is $\neg P$. In addition, for every rule $F \leftarrow G$ whose head F does not contain explainable symbols, there is a corresponding completion formula, which is simply the material implication $G \supset F$. The conjunction of all completion formulas for a definite causal theory T is equivalent to the formula representing T in classical logic.

For instance, the set of completion formulas for causal theory (7) consists of the positive definition of P_1

$$P_1 \equiv (P_1 \wedge P_0) \vee (Q_0 \wedge A_0), \tag{8}$$

the negative definition of P_1

$$\neg P_1 \equiv \neg P_1 \wedge \neg P_0, \tag{9}$$

the positive definition of Q_1

$$Q_1 \equiv Q_1 \wedge Q_0 \tag{10}$$

and the negative definition of Q_1

$$\neg Q_1 \equiv \neg Q_1 \wedge \neg Q_0. \tag{11}$$

Causal theory (7) is equivalent to the conjunction of these formulas.

To simplify this conjunction, note that (9) can be rewritten as

$$P_0 \supset P_1, \tag{12}$$

and (10), (11) can be rewritten as

$$\begin{array}{l} Q_1 \supset Q_0, \\ Q_0 \supset Q_1. \end{array}$$

In the presence of (12), (8) is equivalent to

$$P_1 \equiv P_0 \vee (Q_0 \wedge A_0). \tag{13}$$

The last formula entails (12). Consequently, the conjunction of (8)–(11) is equivalent to the conjunction of (13) and

$$Q_1 \equiv Q_0. \tag{14}$$

We conclude that causal theory (7) is equivalent to the conjunction of (13) and (14). These formulas define P_1 and Q_1 in terms of P_0, Q_0 and A_0. Consequently, (7) has 8 models, corresponding to the interpretations of $\{P_0, Q_0, A_0\}$. These models represent the 8 transitions shown in Figure 1.

The same procedure is applicable when $n > 1$. The result of applying the translation ct_n to action description (4) consists of $5n$ rules

$$\begin{array}{l} P_{i+1} \leftarrow P_{i+1} \wedge P_i, \\ \neg P_{i+1} \leftarrow \neg P_{i+1} \wedge \neg P_i, \\ Q_{i+1} \leftarrow Q_{i+1} \wedge Q_i, \\ \neg Q_{i+1} \leftarrow \neg Q_{i+1} \wedge \neg Q_i, \\ P_{i+1} \leftarrow Q_i \wedge A_i \end{array}$$

for all $i < n$, with $P_1, \dots, P_n, Q_1, \dots, Q_n$ explainable. (We dropped the trivial conjunctive term *True* in the last rule.) The literal completion of this theory is equivalent to the conjunction of $2n$ formulas

$$\begin{aligned}&P_{i+1} \equiv P_i \vee (Q_i \wedge A_i),\\&Q_{i+1} \equiv Q_i\end{aligned}$$

for all $i < n$. The models of this set of formulas represent paths of length n in the graph shown in Figure 1.

4 Example: Loading and Shooting

4.1 The Shooting Domain

The shooting scenario from [Hanks and McDermott, 1987] involves three actions: *Load*, *Wait* and *Shoot*. In $\mathcal{C}$, there is no need to introduce *Wait* as an elementary action, because it can be identified with the empty set of elementary actions. On the other hand, since there is an action in $\mathcal{C}$ that includes both *Load* and *Shoot*, we may wish to postulate that these two elementary actions cannot be executed concurrently. For any conjunction H of action symbols, we will write

$$\textbf{nonexecutable } H$$

for

$$\textbf{caused } \mathit{False} \textbf{ after } H.$$

The shooting domain is characterized by the propositions

$$\begin{aligned}&\textbf{inertial } \mathit{Loaded}, \neg \mathit{Loaded}, \mathit{Alive}, \neg \mathit{Alive},\\&\textbf{caused } \mathit{Loaded} \textbf{ after } \mathit{Load},\\&\textbf{caused } \neg \mathit{Alive} \textbf{ after } \mathit{Loaded} \wedge \mathit{Shoot},\\&\textbf{caused } \neg \mathit{Loaded} \textbf{ after } \mathit{Shoot},\\&\textbf{nonexecutable } \mathit{Load} \wedge \mathit{Shoot}.\end{aligned} \tag{15}$$

The ct_1 translation of (15) is a definite causal theory. (The head of the rule corresponding to the last line of (15) is *False* and consequently does not contain explainable symbols.) Having simplified the set of completion formulas for this translation, we get:

$$\begin{aligned}&\mathit{Loaded}_1 \equiv \mathit{Load}_0 \vee (\mathit{Loaded}_0 \wedge \neg \mathit{Shoot}_0),\\&\mathit{Alive}_1 \equiv \mathit{Alive}_0 \wedge \neg(\mathit{Loaded}_0 \wedge \mathit{Shoot}_0),\\&\neg(\mathit{Load}_0 \wedge \mathit{Shoot}_0).\end{aligned} \tag{16}$$

The first two of these formulas define Loaded_1 and Alive_1 in terms of

$$\mathit{Loaded}_0, \mathit{Alive}_0, \mathit{Load}_0, \mathit{Shoot}_0. \tag{17}$$

Consequently, the edges of the transition diagram for (15) correspond to the interpretations of (17) that satisfy the last of formulas (16). Any proper subset of the set of elementary actions $\{\mathit{Load}, \mathit{Shoot}\}$ can be executed in any state in exactly one way.

More generally, the ct_n translation of (15) is equivalent to the conjunction of the formulas

$$\begin{array}{l} \mathit{Loaded}_{i+1} \equiv \mathit{Load}_i \vee (\mathit{Loaded}_i \wedge \neg \mathit{Shoot}_i), \\ \mathit{Alive}_{i+1} \equiv \mathit{Alive}_i \wedge \neg(\mathit{Loaded}_i \wedge \mathit{Shoot}_i), \\ \neg(\mathit{Load}_i \wedge \mathit{Shoot}_i) \end{array} \tag{18}$$

for all $i < n$.

4.2 Temporal Reasoning and Planning

The "Yale Shooting Problem" [Hanks and McDermott, 1987] and the "Stanford Murder Mystery" [Baker, 1991] correspond to propositional reasoning problems involving formulas (18). In the first case, the goal is to establish that $\neg \mathit{Alive}_3$ is entailed by

$$\begin{array}{l} \mathit{Load}_0, \neg \mathit{Shoot}_0, \\ \neg \mathit{Load}_1, \neg \mathit{Shoot}_1, \\ \neg \mathit{Load}_2, \mathit{Shoot}_2 \end{array}$$

conjoined with formulas (18) for all $i < 3$. The second problem is to establish that Loaded_0 is entailed by

$$\begin{array}{l} \mathit{Alive}_0, \\ \neg \mathit{Load}_0, \neg \mathit{Shoot}_0, \\ \neg \mathit{Load}_1, \mathit{Shoot}_1, \\ \neg \mathit{Alive}_2 \end{array}$$

conjoined with formulas (18) for all $i < 2$. Both claims can be easily verified.

Consider now a planning problem in the shooting domain: initially *Alive* is true and *Loaded* is false; the goal is to make *Alive* false. To find a plan of length 2, we look for an interpretation that satisfies the formulas

$$\begin{array}{l} \mathit{Alive}_0, \neg \mathit{Loaded}_0, \\ \neg \mathit{Alive}_2 \end{array}$$

and formulas (18) for all $i < 2$. The only such interpretation assigns $\mathtt{t}$ to $Load_0$ and $Shoot_1$. In the spirit of "satisfiability planning" [Kautz and Selman, 1992], this interpretation suggests a solution to the planning problem.

McCain and Turner [1998] define conditions under which finding a satisfying interpretation guarantees a valid plan, and report experimental results on solving large planning problems described as causal theories.

4.3 Action Preconditions

In action description (15), *Loaded* is a "fluent precondition" for the action *Shoot*: if it is not satisfied then the action is still possible to execute, although its effect on the fluent *Alive* is not guaranteed. We may wish to treat *Loaded* as an "action precondition," that is to say, to postulate that the action *Shoot* is nonexecutable when *Loaded* is false.

It is convenient to extend the notation introduced at the beginning of Section 4.1 as follows: if H is a conjunction of action symbols and F is a formula that does not contain action symbols, we will write

$$\textbf{nonexecutable } H \textbf{ if } F$$

for

$$\textbf{caused } \mathit{False} \textbf{ after } H \wedge F.$$

In the following modification of (15), *Loaded* is an action precondition for *Shoot*:

$$\begin{array}{l}
\textbf{inertial } \mathit{Loaded}, \neg \mathit{Loaded}, \mathit{Alive}, \neg \mathit{Alive}, \\
\textbf{caused } \mathit{Loaded} \textbf{ after } \mathit{Load}, \\
\textbf{caused } \neg \mathit{Alive} \textbf{ after } \mathit{Shoot}, \\
\textbf{caused } \neg \mathit{Loaded} \textbf{ after } \mathit{Shoot}, \\
\textbf{nonexecutable } \mathit{Shoot} \textbf{ if } \neg \mathit{Loaded}, \\
\textbf{nonexecutable } \mathit{Load} \wedge \mathit{Shoot}.
\end{array}$$

The ct_1 translation of this domain description is equivalent to

$$\begin{array}{l}
\mathit{Loaded}_1 \equiv \mathit{Load}_0 \vee (\mathit{Loaded}_0 \wedge \neg \mathit{Shoot}_0), \\
\mathit{Alive}_1 \equiv \mathit{Alive}_0 \wedge \neg \mathit{Shoot}_0, \\
\mathit{Shoot}_0 \supset (\mathit{Loaded}_0 \wedge \neg \mathit{Load}_0).
\end{array}$$

5 Expressive Possibilities of $\mathcal{C}$

5.1 Indirect Effects and Implicit Preconditions

So far we have not had a chance to use static laws—propositions of form (2). As observed in [Lin, 1995] and [McCain and Turner, 1995], a static law can play two roles in reasoning about action. First, postulating **caused** F **if** G may allow us to conclude that an action has an indirect effect: any action that causes G to become true will also indirectly cause F to become true. Second, we may be able to conclude that an action has an implicit precondition: an action that causes F to become false is not executable if G is true (unless it causes G to change its truth value also).

For instance, if an object is submerged in water then there is a cause for it to be wet. The action of putting a puppy in water causes it to be in water; consequently, it has an indirect effect: it will make the puppy wet. The action of drying a puppy with a towel causes it to be dry; consequently, it has an implicit precondition: it is impossible to dry a puppy with a towel when it is in water.

This example can be formalized in $\mathcal{C}$ as follows:

$$\begin{array}{l}\textbf{inertial } \mathit{InWater}, \neg \mathit{InWater}, \mathit{Wet}, \neg \mathit{Wet},\\ \textbf{caused } \mathit{InWater} \textbf{ after } \mathit{PutInWater},\\ \textbf{caused } \neg \mathit{Wet} \textbf{ after } \mathit{DryWithTowel},\\ \textbf{caused } \mathit{Wet} \textbf{ if } \mathit{InWater}.\end{array} \tag{19}$$

The states of action description (19) are the interpretations of the fluent signature $\{\mathit{InWater}, \mathit{Wet}\}$ that satisfy

$$\mathit{InWater} \supset \mathit{Wet};$$

there are 3 such interpretations. To find the causally explained transitions, we write out the completion formulas for the ct_1 translation of (19) and simplify them. The result can be written in the form

$$\begin{array}{l}\mathit{InWater}_1 \equiv \mathit{InWater}_0 \vee \mathit{PutInWater}_0,\\ \mathit{Wet}_1 \equiv (\mathit{Wet}_0 \wedge \neg \mathit{DryWithTowel}_0) \vee \mathit{PutInWater}_0,\\ \mathit{InWater}_0 \supset \mathit{Wet}_0,\\ \mathit{DryWithTowel}_0 \supset (\neg \mathit{InWater}_0 \wedge \neg \mathit{PutInWater}_0).\end{array}$$

The first two formulas characterize $\mathit{InWater}_1$ and Wet_1 in terms of the other atoms; they show that any action can be performed in at most one way. The second disjunctive term in the second line represents the indirect effect of *PutInWater* on *Wet*. The third line says that the initial state of a transition is indeed a state. The last line shows that the action *DryWithTowel* can be executed only when *InWater* is false—this is the implicit precondition mentioned above—and also that this action cannot be executed concurrently with *PutInWater*.

5.2 Nondeterminism

When Jack goes to work, he can walk there or, if his car is in his garage, he can drive. The action *GoToWork* always makes the fluent *JackAtWork* true; it will also make the fluent *CarInGarage* false if Jack chooses to drive. One of the two effects of *GoToWork* is nondeterministic.

We will use

$$\textbf{possibly caused } F \textbf{ after } H$$

as an abbreviation for

$$\textbf{caused } F \textbf{ if } F \textbf{ after } H.$$

The example above can be formalized as follows:

$$\begin{array}{l}
\textbf{inertial } \mathit{JackAtWork}, \neg \mathit{JackAtWork}, \mathit{CarInGarage}, \neg \mathit{CarInGarage}, \\
\textbf{caused } \mathit{JackAtWork} \textbf{ after } \mathit{GoToWork}, \\
\textbf{possibly caused } \neg \mathit{CarInGarage} \textbf{ after } \mathit{CarInGarage} \wedge \mathit{GoToWork}, \\
\textbf{nonexecutable } \mathit{GoToWork} \textbf{ if } \mathit{JackAtWork}.
\end{array}$$

The ct_1 translation of this action description is equivalent to

$$\begin{array}{l}
\mathit{JackAtWork}_1 \equiv \mathit{JackAtWork}_0 \vee \mathit{GoToWork}_0, \\
\neg \mathit{GoToWork}_0 \supset (\mathit{CarInGarage}_1 \equiv \mathit{CarInGarage}_0), \\
\mathit{CarInGarage}_1 \supset \mathit{CarInGarage}_0, \\
\neg(\mathit{JackAtWork}_0 \wedge \mathit{GoToWork}_0).
\end{array}$$

The nondeterministic behavior of *CarInGarage* is described by the two formulas in the middle. After performing the action *GoToWork* in a state in which *CarInGarage* is true, this fluent can either remain true or become false. The corresponding transition diagram has two edges that are labeled *GoToWork* and begin in the state in which *JackAtWork* is false and *CarInGarage* is true.

Regarding the abbreviation **possibly caused** we can observe that

$$\textbf{inertial } F$$

(Section 2.1) is identical to

$$\textbf{possibly caused } F \textbf{ after } F.$$

5.3 Noninertial Fluents

In all examples so far, every literal in the fluent signature was postulated to be inertial. In some useful action descriptions this is not the case.

We would not assume inertia, first of all, for a fluent that is explicitly defined in terms of other fluents. Imagine, for instance, that we want to enhance the shooting domain from Section 4.1 by saying that the potential victim is in danger if he is alive and the gun is loaded. This idea can be formalized by adding *InDanger* to the fluent signature, and adding the static law

$$\textbf{caused } \mathit{InDanger} \equiv (\mathit{Alive} \wedge \mathit{Loaded}) \textbf{ if } \mathit{True}. \tag{20}$$

to action description (15). This amounts to adding the formulas

$$\mathit{InDanger}_i \equiv \mathit{Alive}_i \wedge \mathit{Loaded}_i \qquad (i = 0, 1)$$

to the formulas of classical logic representing the ct_1 translation of (15).

There is no need to assume inertia for the new fluent in this example. Actually, adding

$$\textbf{intertial } \mathit{InDanger}, \neg \mathit{InDanger}$$

to (15) along with (20) would have led to unintuitive results. (The execution of *Load* in the state in which *Loaded* is false and *Alive* is true would be nondeterministic: it would be possible for both fluents to be affected.)

Note that the head of (20) is not a literal, so that the literal completion method is not applicable to the corresponding causal theory. But this proposition can be equivalently replaced here by a pair of propositions whose head are literals:

$$\begin{array}{l}\textbf{caused } \mathit{InDanger} \textbf{ if } \mathit{Alive} \wedge \mathit{Loaded},\\ \textbf{caused } \neg\mathit{InDanger} \textbf{ if } \neg(\mathit{Alive} \wedge \mathit{Loaded}).\end{array}$$

Second, there are fluents whose values tend to change in a specific way, rather than remain unchanged. Consider, for instance, a pendulum that moves from its leftmost position to the rightmost and back, with each swing taking one unit of time. We want to describe the action of holding the pendulum steady in its current position for the duration of one unit of time.

In the following action description, *Right* is a fluent symbol, and *Hold* is an action symbol:

$$\begin{array}{l}\textbf{possibly caused } \mathit{Right} \textbf{ after } \neg \mathit{Right},\\ \textbf{possibly caused } \neg \mathit{Right} \textbf{ after } \mathit{Right},\\ \textbf{caused } \mathit{Right} \textbf{ after } \mathit{Right} \wedge \mathit{Hold},\\ \textbf{caused } \neg \mathit{Right} \textbf{ after } \neg \mathit{Right} \wedge \mathit{Hold}.\end{array} \tag{21}$$

The first two lines of (21) are similar to the inertia assumption but different: if the pendulum has changed its position then there is a cause for this. The "normal" behavior of an unloaded gun is to remain unloaded; the "normal" behavior of a pendulum in the leftmost position is to move to the rightmost position.

The completion formulas for the ct_1 translation of (21) are equivalent to

$$\mathit{Right}_1 \equiv (\mathit{Hold}_0 \equiv \mathit{Right}_0).$$

The pendulum is in the rightmost position in two cases: if it was in this position earlier and was held there, and if it was in the leftmost position earlier and was not held there.

Finally, some fluents have a "default" value that they take unless an action causes them to take a different value. A spring-loaded door is closed unless someone has just opened it; its behavior is not described by inertia. For any formula F without action symbols, let

$$\textbf{default } F$$

stand for the static law

$$\textbf{caused } F \textbf{ if } F$$

—if F holds then there is a cause for this. Several propositions of this form can be combined like inertia propositions (Section 2.1).

A spring-loaded door is characterized by the following action description:

$$\begin{array}{l}\textbf{default } \mathit{Closed},\\ \textbf{caused } \neg \mathit{Closed} \textbf{ after } \mathit{OpenDoor}.\end{array}$$

Its ct_1 translation is equivalent to

$$\mathit{Closed}_1 \equiv \neg \mathit{OpenDoor}_0.$$

The door is closed unless it has just been opened.

5.4 Interaction between Concurrent Actions

Pednault [1987] considers lifting the opposite ends of a table upon which various objects have been placed. If one end of the table has been raised, a cup on the table falls off. But if both ends are lifted simultaneously, the cup remains fixed.

Formalizing (a modification of) this example in the situation calculus is discussed in Section 8.3.2 of [Gelfond *et al.*, 1991]. Baral and Gelfond [1997] expressed Pednault's example in their action language $\mathcal{A}_C$ (see Section 6.1 below). Turner [1996] formalized it using a static law, and our representation of this example in $\mathcal{C}$ is based on the same idea.

We describe the positions of the two ends of the table by the fluents *Up1* and *Up2*. By default, these fluents are false, that is, normally the two sides are not in their up position. These fluents can be made true by performing the actions *Raise1* and *Raise2*. The fluent *OnTable* indicates that the cup is on the table. This last fluent is inertial.

Our formalization of the lifting example consists of the following propositions:

$$\begin{array}{l}
\textbf{inertial}\ \mathit{OnTable}, \neg \mathit{OnTable},\\
\textbf{default}\ \neg \mathit{Up1}, \neg \mathit{Up2},\\
\textbf{caused}\ \mathit{Up1}\ \textbf{after}\ \mathit{Raise1},\\
\textbf{caused}\ \mathit{Up2}\ \textbf{after}\ \mathit{Raise2},\\
\textbf{caused}\ \neg \mathit{OnTable}\ \textbf{if}\ \neg(\mathit{Up1} \equiv \mathit{Up2}).
\end{array}$$

The ct_1 translation of this action description is equivalent to

$$\begin{array}{l}
\mathit{Up1}_1 \equiv \mathit{Raise1}_0,\\
\mathit{Up2}_1 \equiv \mathit{Raise2}_0,\\
\mathit{OnTable}_1 \equiv \mathit{OnTable}_0 \land (\mathit{Raise1}_0 \equiv \mathit{Raise2}_0),\\
\mathit{OnTable}_0 \supset (\mathit{Up1}_0 \equiv \mathit{Up2}_0).
\end{array}$$

6 Embedding $\mathcal{A}_C$ into $\mathcal{C}$

In this section we analyze the relationship between the treatment of concurrency in two action languages—$\mathcal{A}_C$ from [Baral and Gelfond, 1997] and the new language $\mathcal{C}$—by embedding the former into the latter.

6.1 The Language $\mathcal{A}_C$

The main idea of $\mathcal{A}_C$ is that the effects of a set of elementary actions are "inherited" from its subsets, and that this inheritance is defeasible.

To specify an action description in the language $\mathcal{A}_C$, just as in case of $\mathcal{C}$, we first select a set of fluent symbols σ^{fl} and a set of action symbols σ^{act}. An *action* is a subset of σ^{act}; this definition is essentially the same as the one given in Section 2.1.

An *action description* in the language $\mathcal{A}_C$ is a set of propositions of the form

$$a \textbf{ causes } L \textbf{ after } F, \tag{22}$$

where a is an action, L a literal of the fluent signature, and F a formula of the fluent signature. We will abbreviate (22) as $\langle a, L, F\rangle$.

The semantics of $\mathcal{A}_C$ is defined as follows. Consider an action description D in this language. A *state* is an interpretation of the fluent signature. For any action a, state s and literal L of the fluent signature, we say that executing a in s *causes* L if there exists a proposition $\langle a', L, F\rangle$ in D such that

- $a' \subseteq a$,
- s satisfies F, and
- there is no proposition $\langle a'', \overline{L}, G\rangle$ in D such that

  $$a' \subset a'' \subseteq a$$

  and s satisfies G.

Thus a proposition (22) of the Baral—Gelfond language tells us what the effect of the supersets of a might be; in case of conflict, the "more specific" proposition applies. Finally, we say that a state s' is the *result* of executing a in s if s' satisfies every literal L such that

- executing a in s causes L, or
- executing a in s does not cause $\overline{L}$, and s satisfies L.

It is clear that, for any a and s, there can be at most one such resulting state s'.

Consider, for instance, an $\mathcal{A}_C$ description of the form

$$\begin{array}{l} \{A_1\} \textbf{ causes } L \textbf{ after } F_1, \\ \{A_1, A_2\} \textbf{ causes } \overline{L} \textbf{ after } F_2. \end{array} \tag{23}$$

If a does not include A_1 then the execution of a in any state s has no effect on s. If a includes A_1 but not A_2, and s satisfies F_1, then the resulting state s' satisfies L. If a includes both A_1 and A_2, and s satisfies F_1, then s' satisfies $\overline{L}$ or L depending on whether or not s satisfies F_2.

The characterization of $\mathcal{A}_C$ above differs from [Baral and Gelfond, 1997] in a number of details. Baral and Gelfond require that F in (22) be a conjunction of literals, and write **if** instead of **after**. Their language includes "v-propositions" that we do not discuss here; our version is the "action description component" [Lifschitz, 1997b] of theirs.

6.2 First Translation from $\mathcal{A}_C$ into $\mathcal{C}$

Consider a finite action description D in the language $\mathcal{A}_C$. We will define the corresponding action description $c_1(D)$ in $\mathcal{C}$. To simplify notation, we will identify an action a with the conjunction of the action symbols that belong to it. The description $c_1(D)$ consists of the dynamic laws

$$\textbf{caused } L \textbf{ after } (a \wedge F) \wedge \bigwedge_{\substack{a',G: \\ \langle a',\overline{L},G\rangle \in D,\ a \subset a'}} \neg(a' \wedge G) \tag{24}$$

for all propositions (22) in D, and of the inertia propositions

$$\textbf{inertial } P, \neg P \tag{25}$$

for all fluent symbols P.

For instance, the translation of (23) into $\mathcal{C}$ includes

$$\begin{array}{l} \textbf{caused } L \textbf{ after } A_1 \wedge F_1 \wedge \neg(A_1 \wedge A_2 \wedge F_2), \\ \textbf{caused } \overline{L} \textbf{ after } A_1 \wedge A_2 \wedge F_2 \end{array}$$

and the inertia propositions.

The following theorem shows that the translation c_1 preserves the meaning of action descriptions.

Proposition 3 *For any finite action description D in the language $\mathcal{A}_C$, a state s' is the result of executing an action a in a state s iff $\langle s, a, s'\rangle$ is a transition causally explained by $c_1(D)$.*

6.3 Second Translation from $\mathcal{A}_C$ into $\mathcal{C}$

The translation c_1 is applicable to finite descriptions only: without this restriction, the set of conjunctive terms in (24) can be infinite. Furthermore,

c_1 is not "elaboration tolerant," in the sense that adding a proposition to D requires, generally, that some of the propositions in $c_1(D)$ be modified. We therefore define a second translation c_2 in which these defects are corrected.

The fluent signature of $c_2(D)$ includes, along with every fluent symbol P of the description D, the atoms P^a for all actions a such that D includes a proposition (22) with $|L| = P$.[3] Intuitively, P^a means that the given action description "allows" a to modify P.

The translation $c_2(D)$ consists of the following propositions:

- for each proposition (22) in D, the dynamic law

$$\textbf{caused } L \textbf{ if } |L|^a \textbf{ after } a \wedge F$$

  and, for all propositions $\langle a', \overline{L}, G\rangle$ in D such that $a' \subset a$, the dynamic law

$$\textbf{caused } \neg|L|^{a'} \textbf{ after } a \wedge G;$$

- for each of the additional fluent symbols P^a, the static law

$$\textbf{default } P^a;$$

- the inertia propositions (25) for each fluent symbol P from D.

For example, the new translation of (23), with $|L|^{\{A_1\}}$ and $|L|^{\{A_1,A_2\}}$ abbreviated as P_1 and P_2, consists of the propositions

$$\begin{array}{l}
\textbf{caused } L \textbf{ if } P_1 \textbf{ after } A_1 \wedge F_1, \\
\textbf{caused } \neg P_1 \textbf{ after } A_1 \wedge A_2 \wedge F_2, \\
\textbf{caused } \overline{L} \textbf{ if } P_2 \textbf{ after } A_1 \wedge A_2 \wedge F_2, \\
\textbf{caused } P_i \textbf{ if } P_i \qquad (i = 1, 2)
\end{array}$$

and the inertia propositions (25) for all fluent symbols P from (23).

Proposition 4 *For any action description D in the language $\mathcal{A}_C$, a state s' is the result of executing an action a in a state s iff there exists a transition $\langle s_1, a, s_1'\rangle$ causally explained by $c_2(D)$ such that s and s' are the results of restricting s_1 and, respectively, s_1' to the fluent signature of D.*

[3] By $|L|$ we denote the atom contained in the literal L.

Acknowledgements

We are grateful to Norman McCain and Hudson Turner for useful discussions on the subject of this paper, and to the anonymous referees for criticisms and suggestions. This work was partially supported by National Science Foundation under grant IRI-9306751.

References

[Baker, 1991] Andrew Baker. Nonmonotonic reasoning in the framework of situation calculus. *Artificial Intelligence*, 49:5–23, 1991.

[Baral and Gelfond, 1997] Chitta Baral and Michael Gelfond. Reasoning about effects of concurrent actions. *Journal of Logic Programming*, 31, 1997.

[Clark, 1978] Keith Clark. Negation as failure. In Herve Gallaire and Jack Minker, editors, *Logic and Data Bases*, pages 293–322. Plenum Press, New York, 1978.

[Geffner, 1990] Hector Geffner. Causal theories for nonmonotonic reasoning. In *Proc. AAAI-90*, pages 524–530, 1990.

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

[Gelfond *et al.*, 1991] Michael Gelfond, Vladimir Lifschitz, and Arkady Rabinov. What are the limitations of the situation calculus? In Robert Boyer, editor, *Automated Reasoning: Essays in Honor of Woody Bledsoe*, pages 167–179. Kluwer, 1991.

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

[Kautz and Selman, 1992] Henry Kautz and Bart Selman. Planning as satisfiability. In *Proc. ECAI-92*, pages 359–363, 1992.

[Lifschitz, 1997a] Vladimir Lifschitz. On the logic of causal explanation. *Artificial Intelligence*, 96:451–465, 1997.

[Lifschitz, 1997b] Vladimir Lifschitz. Two components of an action language. *Annals of Mathematics and Artificial Intelligence*, 1997. To appear.

[Lin, 1995] Fangzhen Lin. Embracing causality in specifying the indirect effects of actions. In *Proc. IJCAI-95*, pages 1985–1991, 1995.

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

[McCain and Turner, 1997] Norman McCain and Hudson Turner. Causal theories of action and change. In *Proc. AAAI-97*, pages 460–465, 1997.

[McCain and Turner, 1998] Norman McCain and Hudson Turner. Fast satisfiability planning with causal theories. Unpublished draft, available at `http://www.cs.utexas.edu/users/vl/mccain98.ps` , 1998.

[Pednault, 1987] Edwin Pednault. Formulating multi-agent, dynamic world problems in the classical planning framework. In Michael Georgeff and Amy Lansky, editors, *Reasoning about Actions and Plans*, pages 47–82. Morgan Kaufmann, San Mateo, CA, 1987.

[Sandewall, 1995] Erik Sandewall. *Features and Fluents*, volume 1. Oxford University Press, 1995.

[Shanahan, 1997] Murray Shanahan. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia.* MIT Press, 1997.

[Turner, 1996] Hudson Turner. Splitting a default theory. In *Proc. AAAI-96*, pages 645–651, 1996.