This paper was selected by a process of
anonymous peer reviewing for presentation at

COMMONSENSE 2007

8th International Symposium on Logical Formalizations of Commonsense Reasoning

Part of the AAAI Spring Symposium Series, March 26-28 2007,
Stanford University, California

Further information, including follow-up notes for some of the
selected papers, can be found at:

www.ucl.ac.uk/commonsense07

Defeasible laws, parallel actions, and reasoning about resources

Sandeep Chintabathina and **Michael Gelfond** and **Richard Watson**
Texas Tech University
Department of Computer Science
Lubbock, TX, USA
{chintaba,mgelfond,rwatson}@cs.ttu.edu

Abstract

We introduce a new action language, $\mathcal{CARD}$, which allows defeasible dynamic causal laws, default fluents, concurrent and non-deterministic actions, and actions which use resources. We give syntax and semantics of the language and several simple examples of its use. Comparison with some other languages is also given though limited by space requirements.

Introduction

Representing knowledge about dynamic domains and using this knowledge for solving classical AI tasks (such as planning, diagnostics, and learning) have been at the center of research in Artificial Intelligence since the late fifties. One direction of work in this area consisted of the design of action languages - formal models of parts of natural language used for reasoning about actions and their effects. A theory in an action language (often called an action description) is used to succinctly describe the collection of all possible trajectories of a given dynamic domain. Usually this is done by defining the *transition diagram*, $T(\mathcal{A})$, of an action description $\mathcal{A}$. The states of the diagram correspond to possible physical states of the domain represented by $\mathcal{A}$. Arcs of $T(\mathcal{A})$ are labeled by actions. A transition $\langle \sigma, a, \sigma' \rangle \in T(\mathcal{A})$ if execution of action a in state σ may move the domain to state σ'. In some action languages actions are elementary (or atomic). In others an action, a, is viewed as a finite non-empty collection of elementary actions. Intuitively, execution of an action $a = \{e_1, \ldots, e_n\}$ corresponds to the simultaneous execution of every $e_i \in a$. Currently there are a substantial number of action languages used to study different features of dynamic domains. Some of them are easily comparable. For instance the action language $\mathcal{AL}$ (Turner 1997; Baral & Gelfond 2000) is simply an extension of action language $\mathcal{A}$ (Gelfond & Lifschitz 1993) by state constraints (also called static causal laws) which express causal relations between fluents[1]. Similarly $\mathcal{A}_C$ (Baral & Gelfond 1997) expands $\mathcal{A}$ by allowing concurrent actions (which are prohibited in $\mathcal{A}$). Differences between other languages are far more substantial and often rooted in the choice of basic underlying principles used in their semantics. The semantics of $\mathcal{AL}$, for instance, formalize McCarthy's *Principle of Inertia* which says that "Things tend to stay the same unless they are changed by actions" (McCarthy & Hayes 1969). This principle, suggested by McCarthy as an informal solution to the frame problem, is closely connected to the notion of *default* and the notion of *beliefs* of a rational agent. The semantics of the language $\mathcal{C}+$ (Giunchiglia *et al.* 2004) is based on a different underlying principle - *The Principle of Universal Causation*. In (Giunchiglia *et al.* 2004) this principle is informally summarized as follows: "Every fact that is caused obtains, and vice versa". Yet another basic idea is used in the semantics of language K, introduced in (Eiter *et al.* 2000). Theories of K describe "transitions among states of knowledge rather than among states of the world". The list is of course incomplete and is only used to illustrate the richness of the action language landscape. Even though such diversity of approaches can be viewed as undesirable, we view it as a good thing. It has already led to interesting partial solutions of the frame, ramification, and qualification problems, and the establishment of non-trivial connections between causality, defaults, and beliefs. We consider the further development of these languages and the study of the relationship between them to be an important research direction. New results will sharpen our understanding of various basic principles used in action language design and help in developing the methodology of the use of action languages as high level descriptions of dynamic domains needed for design and development of intelligent systems.

In this paper we introduce a new action language $\mathcal{CARD}$ which extends both $\mathcal{AL}$ and $\mathcal{A}_C$. The new language allows defeasible dynamic causal laws, default fluents, concurrent and non-deterministic actions, and actions which use resources. We give syntax and semantics of the new language and several simple examples of its use.

Syntax

Let Σ be a sorted signature consisting of sorts[2] $S_1, \ldots, S_n$ and properly typed function symbols. Strings from S_i will be referred to as *object constants* of sort S_i. We assume that

Copyright © 2007, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

[1] By fluent we mean a function whose values depend on state and may change as the results of actions.

[2] By a sort we mean a non-empty countable collection of strings in some fixed alphabet.

Σ contains

- Sort S_e whose elements are called *elementary actions*. We also assume some fixed enumeration $\{e_1, e_2, \ldots\}$ of S_e.
- Sort S_a consisting of strings of the form $\{e_{i_1}, \ldots, e_{i_k}\}$ where $1 \leq k$ and for every $1 \leq m < k$, $i_m < i_{m+1}$. Elements of S_a are called *actions*[3]. (For instance if $S_e = \{e_1, e_2\}$ then $S_a = \{\{e_1\}, \{e_2\}, \{e_1, e_2\}\}$. Note that $\{e_2, e_1\} \notin S_a$.)
- Usual numerical sorts and functions including sort $N = \{0, 1, 2, \ldots\}$, functions $+$, $\leq$, etc., and a Boolean sort $Bool = \{\textit{true}, \textit{false}\}$;

We also assume that function symbols of Σ are divided into two disjoint sets, called *fluents* and *statics*. The arithmetic functions belong to the latter class. *Terms* of Σ will be defined as usual. *Atoms* are expressions of the form $t_1 = t_2$ where t_1 and t_2 are properly typed terms. Atoms $t = \textit{true}$ and $t = \textit{false}$ will be often written as t and $\neg t$. Atoms containing fluents are called *fluent atoms*. Other atoms are called *static*. Function symbols of Σ with exactly one of their parameters taking values from S_a will be called *action attributes*. The set of object constants of Σ is called the *universe* of Σ and denoted by U.
An *interpretation* I of Σ is a mapping I, such that

- For any object constant c of Σ, $I(c) = c$;
- For any function symbol f of Σ with parameters from sorts $S_1, \ldots, S_k$ and values from sort S, $I(f)$ is a function f^I from $S_1 \times \ldots \times S_k$ to S;
- For a numerical sort I must coincide with standard interpretations of the corresponding function symbols.

Interpretation I is expanded to terms and atoms of Σ in the usual manner. The *truth* of atom l with respect to I ($I \models l$) is also defined as usual. I *satisfies* a collection of atoms if every atom of the collection is true in I. Note that since I is standard on numerical sorts and functions, $I(0)$ is 0, $I(3+4)$ is equal to 7, and $I(4 > 1+1)$ is *true* in any interpretation of Σ. In what follows we will often identify an interpretation I with the collection of atoms of Σ which are true in I.

Definition 1 *[Action Description of $\mathcal{CARD}$]*
An action description $\mathcal{A}$ of $\mathcal{CARD}$ consists of:

- A signature $\Sigma(\mathcal{A})$ satisfying the above assumptions.
- A logic program $\Pi(\mathcal{A})$, under answer set semantics (Gelfond & Lifschitz 1991), referred to as the static part of $\mathcal{A}$. It cannot contain fluents, and is used to define relations of the domain which hold in all its states. For simplicity we assume that $\Pi(\mathcal{A})$ has a unique answer set.
- A collection of statements of the form :

$$l_0 \text{ if } \textit{body} \quad (1)$$
$$\text{default } l_0 \text{ if } \textit{body} \quad (2)$$
$$a \text{ causes } l_0 \text{ if } \textit{body} \quad (3)$$
$$a \text{ normally causes } l_0 \text{ if } \textit{body} \quad (4)$$
$$\text{impossible } a \text{ if } \textit{body} \quad (5)$$

where a is an action, l_0 is a fluent atom of the form $t = c$ where c is an object constant, and *body* is a collection of atoms of Σ. l_0 is often referred to as the head of the corresponding statement.

Intuitively statement (1), called a *state constraint* or *static causal law*, says that every state of the domain which satisfies the body of (1) must satisfy l_0. Statement (2), referred to as a *default*, is similar to a state constraint, except that it is defeasible. Statements (3) and (4) are called *strict dynamic causal laws* and *defeasible dynamic causal laws* respectively. The former states that if a is executed in a state which satisfies the body of (3) then l_0 will be true in a resulting state. The latter says that if a is executed in a state which satisfies the body of (4) then l_0 will *normally* be true in a resulting state. Statement (5), called an *impossibility condition* for a, says that a cannot be executed in any state satisfying the body of (5). Intuitively, the effect of a concurrent action normally consists of the union of the effects of its components. In case of conflict more specific defeasible dynamic causal laws are preferred to less specific ones. These statements will be clarified by examples below and given precise meaning in the next section. In our presentation we will frequently use statements of $\mathcal{CARD}$ containing typed variables. Such a statement will be viewed as a shorthand for the set of all the ground instances of that statement which respect the typing of its variables.

Let us now consider several examples of action theories of $\mathcal{CARD}$. First notice that any actions description of $\mathcal{AL}$ is also an action description of of $\mathcal{CARD}$. In what follows we will focus on action descriptions of $\mathcal{CARD}$ not expressible in $\mathcal{AL}$.

Example 1 *[Concurrent Actions]*
Let action theory $\mathcal{E}_1$ consist of a signature with elementary actions e_1, e_2, Boolean fluents f, g and dynamic causal laws:

$$\{e_1, e_2\} \text{ normally causes } f \quad (1)$$
$$e_1 \text{ normally causes } g \quad (2)$$

Under the intuitive semantics of $\mathcal{CARD}$ the transition diagram $T(\mathcal{E}_1)$ contains transitions $\langle\{\neg f, \neg g\}, \{e_1\}, \{\neg f, g\}\rangle$, $\langle\{\neg f, \neg g\}, \{e_1, e_2\}, \{f, g\}\rangle$, etc. The former is determined by law (2) and the axiom of inertia applied to $\neg f$. The latter is obtained by combining the effects of laws (1) and (2).

In what follows we often omit description of the signature of an action theory $\mathcal{E}$ and assume that it consists of symbols occurring in $\mathcal{E}$.

Example 2 *[Defeasibility of Dynamic Causal Laws]*
Consider $\mathcal{E}_2$ consisting of causal laws:

$$e_1 \text{ normally causes } f \quad (1)$$
$$e_1 \text{ normally causes } \neg f \quad (2)$$

According to our intuitive semantics the laws behave similarly to contradictory defaults in answer set programming or Reiter's default theories (Reiter 1980). Neither of the laws is more specific than the other. Consequently $T(\mathcal{E}_2)$ consists of transitions $\langle\{\neg f\}, e_1, \{f\}\rangle$, $\langle\{f\}, e_1, \{f\}\rangle$ and transitions $\langle\{\neg f\}, e_1, \{\neg f\}\rangle$, $\langle\{f\}, e_1, \{\neg f\}\rangle$ obtained by applications of first and second laws respectively. Notice

[3] To simplify the presentation we often abuse notation by using e instead of $\{e\}$.

that if one of the laws were strict, then it would fire and the remaining defeasible law would be blocked. If both were strict, then no transition would exist.

Example 3 *[Concurrency and Defeasibility]*
In the previous example neither of the two contrary causal laws is more specific than the other. The situation is different for action description $\mathcal{E}_3$ consisting of laws:

$$\begin{array}{ll} \{e_1, e_2\} \text{ normally causes } f & (1) \\ e_1 \text{ normally causes } \neg f & (2) \end{array}$$

This time law (1) is more specific then law (2). Since more specific information is preferable, law (1) defeats law (2). As a result $T(\mathcal{E}_3)$ contains transitions $\langle\{\neg f\}, e_1, \{\neg f\}\rangle$, $\langle\{\neg f\}, e_2, \{\neg f\}\rangle$ and $\langle\{\neg f\}, \{e_1, e_2\}, \{f\}\rangle$, but does not contain a transition $\langle\{\neg f\}, \{e_1, e_2\}\{\neg f\}\rangle$.

Example 4 *[Defaults]*
Let $\mathcal{E}_4$ consist of statements:

$$\begin{array}{ll} e \text{ normally causes } f & (1) \\ \text{default } \neg f \text{ if } p & (2) \\ \text{default } g & (3) \end{array}$$

Intuitively, the transition diagram $T(\mathcal{E}_4)$ contains a transition $\langle\{p, \neg f, \neg g\}, e, \{p, f, g\}\rangle$. In the successor state, p becomes true by inertia; f by the dynamic causal law (1) (which, though defeasible, is stronger than default (2)); and g by default (3).

Action description $\mathcal{E}_5$ from the next example describes an action, e, which causes a fluent, f, to non-deterministically take on a value from 1 to 3.

Example 5 *[Non-determinism]*
Let $\mathcal{E}_5$ consist of statements:

$$\begin{array}{ll} e \text{ normally causes } f = 1 & (1) \\ e \text{ normally causes } f = 2 & (2) \\ e \text{ normally causes } f = 3 & (3) \end{array}$$

After the execution of e the domain can move into either state $\{f = 1\}$, $\{f = 2\}$, or $\{f = 3\}$.

The next example illustrates the use of $\mathcal{CARD}$ for representing knowledge about actions which manipulate resources.

Example 6 *[Moving Resources]*
Consider a simple scenario in which John, who initially has 5 apples, gives 2 apples to Bob. To reason about this scenario we introduce a signature Σ containing sorts for *names* (including John and Bob) and types of *resources* (including "apples"); *natural numbers* will be used to measure quantity of resources involved in the transaction. We use capital letters P and R for variables which range over the first two sorts respectively. Possibly indexed variables X and Y range over natural numbers; E and A range over elementary actions and actions respectively. We also assume that Σ contains a fluent *amount(R,P)* - the number of items of resource R owned by a person P. The elementary action, say e_1, mentioned in the scenario belongs to a class of actions called *move_resources*. Syntactically *move_resources* is an action attribute with Boolean values. Other attributes characterizing actions which move resources are *origin*, *destination*, *resource* and *quantity*. We also need static functions *net_incr(A,R,P)* (*net_decr(A,R,P)*) which return the net increase (decrease) in the amount of resource R owned by person P which is caused by the execution of action A, and auxiliary relations, u and v defined below.
$\mathcal{E}_6$ will consist of

- A static part including
  - attribute atoms for e_1:

$$\begin{array}{ll} move\_resource(e_1) & origin(e_1) = john \\ dest(e_1) = bob & resource(e_1) = apples \\ quantity(e_1) = 2 & \end{array}$$

  - rules for u

$$\begin{array}{l} u(E, R, P, X) \leftarrow \\ \qquad move\_resource(E),\ \ dest(E) = P, \\ \qquad resource(E) = R,\ quantity(E) = X. \\ u(E, R, P, 0) \leftarrow \text{not } ab(E, R, P). \\ ab(E, R, P) \leftarrow u(E, R, P, X), X > 0. \end{array}$$

  - rules for v (similar to rules for u except $dest(E) = P$ is replaced by $origin(E) = P$).
  - rules

$$\begin{array}{l} net\_incr\ (\{e_1, \ldots, e_k\}, R, P) = Y \leftarrow \\ \qquad u(e_1, R, P, X_1), \ldots, u(e_k, R, P, X_k), \\ \qquad Y = X_1 + \ldots + X_k. \\ net\_decr\ (\{e_1, \ldots, e_k\}, R, P) = Y \leftarrow \\ \qquad v(e_1, R, P, X_1), \ldots, v(e_k, R, P, X_k), \\ \qquad Y = X_1 + \ldots + X_k. \end{array}$$

    which define net_incr and net_decr for every action $\{e_1, \ldots, e_k\}$. The program has a single answer set[4] containing $net\_decr(e_1, apple, john) = 2$ and $net\_incr(e_1, apple, bob) = 2$.
- A set of defeasible dynamic causal laws

$$\begin{array}{lll} A \text{ normally causes } amount(R, P) = Y & \text{if} & (6) \\ \qquad amount(R, P) = Y_0, & & \\ \qquad net\_incr(A, R, P) = Y_1, & & \\ \qquad net\_decr(A, R, P) = Y_2, & & \\ \qquad Y = Y_0 + Y_1 - Y_2. & & \end{array}$$

- A set of impossibility conditions

$$\begin{array}{l} \text{impossible } A \text{ if } amount(R, P) = Y_0, \\ \qquad net\_decr(A, R, P) = Y_1, Y_0 < Y_1. \end{array}$$

Laws of form (6) compute the amount of resource R owned by P after the execution of action A. The impossibility condition simply says that one cannot give away more than one has.

Execution of action e_1 in the state
$\{amount(apples, john) = 5, amount(apples, bob) = 1\}$
moves the state to
$\{amount(apples, john) = 3, amount(apples, bob) = 3\}$.

[4] To translate $\Pi(\mathcal{E}_6)$ into a more conventional syntax, we replace any atom of the form $f(\overline{x}) = y$ by $f(\overline{x}, y)$ and add rules of the form $\neg f(\overline{X}, Y_2) \leftarrow f(\overline{X}, Y_1), Y_1 \neq Y_2$.

If $\mathcal{E}_{6a}$ is obtained from $\mathcal{E}_6$ by adding a new action:

$$move\_resource(e_2) \quad origin(e_2) = mary$$
$$dest(e_2) = bob \quad resource(e_2) = apples$$
$$quantity(e_2) = 3$$

then, assuming that $\{e_1, e_2\}$ is executable in a state σ containing $amount(apples, bob) = 1$, the resulting state will contain $amount(apples, bob) = 6$. The amounts of apples owned by John and Mary will be appropriately decreased. Notice that the instances of the defeasible dynamic causal law (6) above in which $A = \{e_1\}$ and $A = \{e_2\}$ are defeated by the dynamic causal law in which $A = \{e_1, e_2\}$.

Semantics

The semantics presented in this paper maps action descriptions of $\mathcal{CARD}$ into the corresponding transition diagrams. Let $\mathcal{A}$ be an action description with signature Σ. By $\mathcal{D}(\mathcal{A})$, $\mathcal{S}(\mathcal{A})$, and $\mathcal{DF}(\mathcal{A})$ we denote the sets of defeasible dynamic causal laws, state constraints, and defaults of $\mathcal{A}$.

Definition 2 *[States of $T(\mathcal{A})$]*
A state of $T(\mathcal{A})$ is an interpretation, σ, of Σ such that

- For any non-numerical static atom l, $\sigma \models l$ iff l belongs to the answer set of the static part of $\mathcal{A}$.
- σ satisfies the state constraints of $\mathcal{A}$, i.e. for every state constraint (1) of $\mathcal{A}$ either $\sigma \models l_0$ or $\sigma \not\models$ *body*.

Given a state, σ, a fluent, f, is said to be a *default fluent* in σ if there is a default, "default $f = y$ if *body*" $\in \mathcal{A}$ such that $\sigma \models$ *body*. The value, y, is called a *possible default value* of f in σ. If f is not a default fluent in σ it is called an *inertial fluent*. We define sets, σ_i and σ_d as follows:

$$\sigma_i = \{f = y \mid f = y \in \sigma \text{ and } f \text{ is an inertial fluent in } \sigma\}$$
$$\sigma_d = \{f = y \mid f = y \in \sigma, f \text{ is a default fluent in } \sigma, \text{ and } y \text{ is a possible default value of } f\}$$

We say that action b is *prohibited* in state σ if $\mathcal{A}$ contains an impossibility condition (5) such that $a \subseteq b$ and the body of (5) is satisfied by σ.

To define transitions of $T(\mathcal{A})$ we use a modification of the McCain-Turner equation from (McCain & Turner 1995). Given an action description $\mathcal{A}$, states σ and σ', and action a:

$$E(a, \sigma, \mathcal{A}) = \{l_0 \mid \exists a', l_0, \textit{body}((a' \text{ causes } l_0 \text{ if } \textit{body}) \in \mathcal{A} \text{ or } (a' \text{ normally causes } l_0 \text{ if } \textit{body}) \in \mathcal{A}) \text{ and } a' \subseteq a \text{ and } \textit{body} \subseteq \sigma\}$$

We say that $\langle \sigma, a, \sigma' \rangle$ satisfies the *extended McCain-Turner equation* for $\mathcal{A}$ if

$$\sigma' = Cn_S(E(a, \sigma, \mathcal{A}) \cup (\sigma \cap \sigma'_i) \cup \sigma'_d) \qquad (7)$$

where $Cn_S(X)$ is a minimal set of atoms containing X which satisfies all the constraints in $\mathcal{S}(\mathcal{A})$.

Action a is called *regular* with respect to $\mathcal{A}$ and σ if

1. a is not prohibited in σ;
2. there is a state σ' such that $\langle \sigma, a, \sigma' \rangle$ satisfies equation (7).

It is easy to see that both actions e_1 and $\{e_1, e_2\}$ from example 1 are regular with respect to any state and $\mathcal{E}_1$. The case is similar for action e_1 from example 3 and e from example 4. Clearly actions e_1 from examples 2, $\{e_1, e_2\}$ from example 3, and e from example 5 are not regular with respect to their respective action descriptions and any state. In example 6, so long as they are not prohibited in a state, either e_1 or e_2 alone is regular with respect to that state and $\mathcal{E}_{6a}$, however $\{e_1, e_2\}$ is never regular.

The effect of an action a, which is regular with respect to σ and $\mathcal{A}$, consists of the union of the effects of its components, i.e. $\langle \sigma, a, \sigma' \rangle \in T(\mathcal{A})$ iff it satisfies (7).

To define a transition $\langle \sigma, a, \sigma' \rangle$ for a non-regular action a we drop some defeasible statements (defeasible dynamic causal laws and defaults) of $\mathcal{A}$ responsible for this irregularity. Here are the necessary definitions.

Two defaults with the same body, and heads $f = y_1$ and $f = y_2$ such that $y_1 \neq y_2$ are called *complementary*. Given a set of defaults, X, $\psi(X)$ is the set of all sets of defaults obtained by replacing each default in X by a complementary default.

Definition 3 *[Reduct]*
Given an action description, $\mathcal{A}$, an action, a, and a state, σ, let $X \subseteq \mathcal{D}(\mathcal{A}) \cup \mathcal{DF}(\mathcal{A})$ and $T \in \psi(\mathcal{DF}(X))$ be such that

1. a is regular with respect to $(\mathcal{A} \setminus X) \cup T$ and σ.
2. there is no $X_0 \subset X$ and $T_0 \in \psi(\mathcal{DF}(X_0))$ such that a is regular with respect to $(\mathcal{A} \setminus X_0) \cup T_0$ and σ.

Action description, $R = (\mathcal{A} \setminus X) \cup T$ will be called a *reduct* of $\mathcal{A}$ with respect to a and σ.

If an action, a, is regular with respect to a state σ and an action description $\mathcal{A}$, it can easily be seen that the only reduct of $\mathcal{A}$ with respect to a and σ is $\mathcal{A}$ itself. Notice that for an arbitrary state, σ, actions from examples 2, 3, 5, and 6 are not regular: $\mathcal{E}_2$ from example 2 has two reducts, $R_1 = \{(1)\}$ and $R_2 = \{(2)\}$ with respect to e_1 and σ; $\mathcal{E}_3$ from example 3 also has two reducts, $R_1 = \{(1)\}$ and $R_2 = \{(2)\}$ with respect to $\{e_1, e_2\}$ and σ. In example 5, there are 3 reducts with respect to e and σ, each containing only one of the three rules. Finally, in example 6, if action $\{e_1, e_2\}$ is executable in σ, there are 3 reducts of $\mathcal{E}_{6a}$ with respect to $\{e_1, e_2\}$ and σ, each containing a different single instance of dynamic causal law (6) - R_1 in which $A = e_1$, R_2 where $A = e_2$, and R_3 where $A = \{e_1, e_2\}$.

As discussed in the previous section, intuitively the two reducts from $\mathcal{E}_2$ are equally good. Similarly, any of the three reducts from example 5 have equal preference. Reduct R_1 of $\mathcal{E}_3$ is preferred to R_2 because law (1) is more specific than law (2). By the same argument, R_3 is the preferred reduct of $\mathcal{E}_{6a}$. These notions are captured by the following definition.

Definition 4 *[Preferred Reduct]*

- Let R_1 and R_2 be reducts of $\mathcal{A}$ with respect to some action a and state σ. We say that R_1 is preferred to R_2 if for every $d_2 \in \mathcal{D}(R_2 \setminus R_1)$ there is $d_1 \in \mathcal{D}(R_1 \setminus R_2)$ such that d_1 is *more specific* than d_2, i.e. the action in d_2 is a proper subset of the action in d_1.

- A reduct, R, of $\mathcal{A}$ with respect to a and σ is called *preferred* if no reduct of $\mathcal{A}$ with respect to a and σ is preferred to it.

One can check that the intuitive preferences given for reducts of our examples match those given by the definition.

Definition 5 *[Transitions of $T(\mathcal{A})$]*
Let σ and σ' be states of $T(\mathcal{A})$. A transition $\langle \sigma, a, \sigma' \rangle \in T(\mathcal{A})$ iff

- a is not prohibited in σ;
- there is a preferred reduct R of $\mathcal{A}$ with respect to a and σ such that $\langle \sigma, a, \sigma' \rangle$ satisfies equation (7) for R.

In examples (1 – 6) we used informal reading of our laws to construct some transitions of the corresponding diagrams. Definition 5 can be now used to justify these constructions.

The following theorem can be useful for understanding the semantics of $\mathcal{CARD}$.

Theorem 1
Let $\mathcal{A}$ be an action description of $\mathcal{CARD}$ which does not contain strict dynamic causal laws. For any state σ and action a which is not prohibited in σ there is at least one state σ' such that $\langle \sigma, a, \sigma' \rangle \in T(\mathcal{A})$

Comparisons with other languages

In this section we compare $\mathcal{CARD}$ with several other action languages. A more complete comparison will be given in the full paper.

$\mathcal{CARD}$, $\mathcal{AL}$, and $\mathcal{A}_C$

$\mathcal{CARD}$ borrows some of its ideas from $\mathcal{AL}$ and $\mathcal{A}_C$. It extends $\mathcal{AL}$ by allowing: 1) concurrent actions in dynamic causal laws; 2) defeasible dynamic causal laws; and 3) defaults and non-boolean fluents. An action description which does not use these extensions will define the same transition diagram under the semantics of both $\mathcal{CARD}$ and $\mathcal{AL}$.

$\mathcal{A}_C$ differs from $\mathcal{CARD}$ in that $\mathcal{A}_C$ doesn't have static causal laws, defaults, or non-boolean fluents. Moreover, if *normally causes* in $\mathcal{CARD}$ is translated as *causes* in $\mathcal{A}_C$, transition diagrams of action descriptions from the examples 1 and 3 are the same under both semantics. However, under the semantics of $\mathcal{A}_C$ the transition diagram for example 2 has no transition labeled by e_1 - the dynamic causal laws behave like strict laws of $\mathcal{CARD}$ in this case.

$\mathcal{CARD}$ and $\mathcal{C}+$

There are some purely syntactic differences between $\mathcal{CARD}$ and $\mathcal{C}+$. Heads and bodies of the laws of $\mathcal{C}+$ allow arbitrary formulas while those of $\mathcal{CARD}$ are limited to atoms. Some causal laws of $\mathcal{C}+$ are simply not allowed in $\mathcal{CARD}$, etc. These differences are not essential. If needed the syntax of $\mathcal{CARD}$ can be extended to make it more compatible with $\mathcal{C}+$. Important semantic differences between $\mathcal{CARD}$ and $\mathcal{C}+$ are inherited from differences between $\mathcal{C}+$ and both $\mathcal{AL}$ and $\mathcal{A}_C$. Even though action descriptions of $\mathcal{AL}$ can, syntactically, be viewed as action descriptions of $\mathcal{C}+$ these languages are based on different underlying assumptions and can differ in meaning. For instance in $\mathcal{AL}$, a state constraint

$$f \text{ if } f \qquad (8)$$

is trivially satisfied by any state, hence, its addition to an action theory $\mathcal{A}$ of $\mathcal{AL}$ does not change $\mathcal{A}$'s meaning. This is not the case for $\mathcal{C}+$'s counterpart to (8) which is read as "if f is true then there is a cause for this" which can be interpreted as assigning default value *true* to f.

Differences between $\mathcal{A}_C$ and $\mathcal{C}+$ stem from the different interpretations of dynamic causal laws. In $\mathcal{C}+$, causal law

$$e_1 \text{ causes } f$$

implies that the effect, f, of e_1 holds after any event which involves execution of e_1, even if other actions are executed concurrently. This is different from $\mathcal{A}_C$ where this law is defeasible and can be defeated by, say,

$$\{e_1, e_2, e_3\} \text{ causes } \neg f$$

To achieve the same effect in $\mathcal{C}+$ one may write

$$\begin{gathered} e_1, \neg e_2 \text{ causes } f \\ e_1, \neg e_3 \text{ causes } f \\ e_1, e_2, e_3 \text{ causes } \neg f \end{gathered}$$

In (Giunchiglia & Lifschitz 1998) the authors show two ways in which action descriptions of $\mathcal{A}_C$ can be encoded in $\mathcal{C}+$. The precise relationship between these encodings and our approach will be investigated in the full paper.

There are several substantial differences between $\mathcal{CARD}$ and $\mathcal{C}+$ which are not inherited from older languages. One stems from a difference in the defeasible dynamic causal laws of each language. In $\mathcal{C}+$ there is no preference given to more specific laws. Let $\mathcal{E}_3'$ be an action description of $\mathcal{C}+$ obtained from $\mathcal{E}_3$ by replacing *normally causes* by *may cause*. If action $\{e_1, e_2\}$ is performed in an arbitrary state, $\mathcal{E}_3'$ yields two transitions while $\mathcal{E}_3$ only yields one. $\mathcal{CARD}$ and $\mathcal{C}+$ also differ by the means of defeating defaults incorporated in their semantics. In $\mathcal{C}+$ a default can be defeated only by another statement with the conclusion contrary to that of the default, as in example 4 or in the following:

Example 7 *[Conflicting Defaults]*
Let $\mathcal{E}_7$ consists of

e causes h	(1)	default f	(2)
default g	(3)	$\neg f$ if g	(4)

Under the semantics of $\mathcal{CARD}$ if e is executed in any state, regardless of the truth values of f, g, and h, rules 1, 3, and 4 fire and state $\{g, h, \neg f\}$ is the result. The same behavior will be exhibited by $\mathcal{C}+$.

The situation changes when defaults are defeated "indirectly", as for instance in

Example 8 *[Indirectly Conflicting Defaults]*
Let $\mathcal{E}_8$ consists of

e causes h	(1)	default f	(2)
default g	(3)	$\neg r$ if f	(4)
r if g	(5)		

This time the transition diagram of $\mathcal{E}_8$ defined by the semantics of $\mathcal{C}+$ does not have a transition labeled by e. Under the semantics of $\mathcal{CARD}$ for any σ the diagram has two such transitions with successor states $\{h, f, \neg g, \neg r\}$ and $\{h, \neg f, g, r\}$ respectively.

Another difference is related to the treatment of actions dealing with resources. The *causes* construct of $\mathcal{C}+$ is not directly applicable to describing effects of concurrent execution of such actions. To remedy this problem (Lee & Lifschitz 2003) introduces the notion of additive fluent and a new syntactic construct, *increments*. A new law,

$$e \text{ increments } c \text{ by } n \text{ if } p$$

says that the execution of action e in a state satisfying p will increment the amount of resource, c, by n. Formalization of example 6 in $\mathcal{C}+$ will look as follows:

Example 9 *[Moving Resources in $\mathcal{C}+$]*
Consider an action description of $\mathcal{C}+$[5] in which actions e_1 and e_2 are defined as in example 6, fluent $amount(R, P)$ is declared as additive, and causal laws are of the form:

$$\begin{array}{lll} E \text{ increments } & amount(R,P) & \text{by } N \text{ if} \\ & move\_resource(E), & \\ & resource(E) = R, & \\ & dest(E) = P, & \\ & quantity(E) = N. & \\ E \text{ decrements } & amount(R,P) & \text{by } N \text{ if} \\ & move\_resource(E), & \\ & resource(E) = R, & \\ & origin(E) = P, & \\ & quantity(E) = N. & \end{array}$$

The transition diagram defined by this action description differs from that defined in example 6 only by static expressions formed by net_incr and net_decr. Note however that $\mathcal{CARD}$'s formalization uses a standard construct of *causes*. Moreover, dealing with resources in $\mathcal{CARD}$ is not limited to addition.

Conclusion and Future Work

In this paper we introduced a new action language $\mathcal{CARD}$ which allows representation and reasoning with defeasible dynamic causal laws, default fluents, concurrent and nondeterministic actions, and actions which use resources. The language substantially expands the expressive power of its predecessors, $\mathcal{A}_C$ and $\mathcal{AL}$. In some respects it is even more powerful than $\mathcal{C}+$. In the full paper[6] we discuss the methodology of using $\mathcal{CARD}$ for knowledge representation and illustrate this methodology using several larger examples. Our current research plans include careful investigation of the relationship between $\mathcal{CARD}$ and other languages such as $\mathcal{C}+$ and the languages from (Shoham 1990), (Erdem & Gabaldon 2006), and (Zhang 2003). We also plan to expand $\mathcal{CARD}$ by continuous processes, similar to that in (Chintabathina, Gelfond, & Watson 2005) and to investigate mapping of action descriptions of $\mathcal{CARD}$ into CR-Prolog (Balduccini & Gelfond 2003) - an extension of Answer Set Prolog capable of reasoning with complex contradictions between defaults. Long term plans include the development and implementation of a modular version of $\mathcal{CARD}$.

[5]We consider a version of $\mathcal{C}+$ which allows description of actions and their attributes by atomic formulas.

[6]The full paper will be posted at www.krlab.cs.ttu.edu/papers upon it's completion.

Acknowledgments

This work was partially supported by ARDA grant ASU06-C-0143 and NASA grant NASA-NNG05GP48G. The authors wish to thank Vladimir Lifschitz, Alfredo Gabaldon, and Joohyung Lee for useful discussions on this subject.

References

Balduccini, M., and Gelfond, M. 2003. Logic programs with consistency-restoring rules. In *Proc. of AAAI-03 Spring Symposium Series*, 9–18.

Baral, C., and Gelfond, M. 1997. Reasoning about effects of concurrent actions. *Journal of Logic Programming* 31(1-3):85–117.

Baral, C., and Gelfond, M. 2000. Reasoning agents in dynamic domains. In Minker, J., ed., *Logic-Based Artificial Intelligence*, 257–279. Kluwer Academic Publishers.

Chintabathina, S.; Gelfond, M.; and Watson, R. 2005. Modeling hybrid domains using process description language. In *Proc. of ASP-05*, 303–317.

Eiter, T.; Faber, W.; Leone, N.; Pfeifer, G.; and Polleres, A. 2000. Planning under incomplete knowledge. *Lecture Notes in Computer Science* 1861:807–821.

Erdem, E., and Gabaldon, A. 2006. Representing action domains with numeric-valued fluents. In *Proc. of JELIA-06*, 151–163.

Gelfond, M., and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Computing* 9(3/4):365–386.

Gelfond, M., and Lifschitz, V. 1993. Representing action and change by logic programs. *Journal of Logic Programming* 17:301–321.

Giunchiglia, E., and Lifschitz, V. 1998. An action language based on causal explanation: Preliminary report. In *AAAI/IAAI*, 623–630.

Giunchiglia, E.; Lee, J.; Lifschitz, V.; McCain, N.; and Turner, H. 2004. Nonmonotonic causal theories. *Artificial Intelligence* 153:49–104.

Lee, J., and Lifschitz, V. 2003. Describing additive fluents in action language C+. In *Proc. of IJCAI-03*, 1079–1084.

McCain, N., and Turner, H. 1995. A causal theory of ramifications and qualifications. In Mellish, C., ed., *Proc. of IJCAI-95*, 1978–1984. Morgan Kaufmann.

McCarthy, J., and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence* 4:463–502.

Reiter, R. 1980. A logic for default reasoning. *Artificial Intelligence* 13:81–132.

Shoham, Y. 1990. Nonmonotonic reasoning and causation. *Cognitive Science* 14(2):213–252.

Turner, H. 1997. Representing actions in logic programs and default theories: A situation calculus approach. *Journal of Logic Programming* 31(1-3):245–298.

Zhang, Y. 2003. Handling defeasibilities in action domains. *TPLP* 3(3):329–376.