PDL-based framework for reasoning about actions

Giuseppe De Giacomo and Maurizio Lenzerini

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italia
{degiacomo,lenzerini}@assi.dis.uniroma1.it

Abstract. Propositional Dynamic Logics (PDL's) provide a suitable formal framework for modeling actions and reasoning about them. However, the basic language of PDL's lack several features that are important for a sophisticated treatment of actions. In this paper, we present a new logic that is obtained by enriching the basic PDL with powerful modeling constructs that allow us to represent determinism and non-determinisms, concurrency, hierarchies, mutual exclusion, backward execution, and non-execution of actions. We demonstrate, by means of examples, the expressive power of the formalism. In particular, we show that although nonmonotonicity is not generally captured by PDL's, our logic is perfectly suited for exploiting monotonic solutions to the frame problem. Finally, we establish that the proposed formalism is decidable, and that the basic reasoning problems are EXPTIME-complete.

1 Introduction

Propositional Dynamic Logics (PDL's) are modal logics for describing and reasoning about system dynamics in terms of states and actions (or events) modeled as relations between states (see [19, 15, 23] for surveys on PDL's, see also [31] for a somewhat different account). The basic language of PDL includes atomic propositions, that are interpreted as simple properties of states, plus the construct $[R]\phi$, where ϕ is a formula and R is an action, whose meaning is that all executions of the action R terminate in a state where ϕ is true. The action R can be either atomic or complex, i.e. constituted by sequential composition, nondeterministic choice, iteration, or test.

PDL's have been originally developed in Theoretical Computer Science to reason about program schemas [9], and their variants have been adopted to specify and verify properties of reactive processes (e.g., Hennessy Milner Logic [16, 22], modal mu-calculus [18, 20, 31]). In Artificial Intelligence, PDL's have been extensively used in establishing decidability and computational complexity results of many formalisms: for example they have been used in investigating Common Knowledge [14], Conditional Logics [10], Description Logics [29, 6, 7], Features Logics [1].

Though PDL's have been only sparingly adopted for reasoning about actions (main exceptions being [28, 17], but also [4]) there are two significant arguments that make them attractive.

1. Transition systems are the semantics adopted by an increasing number of proposal in reasoning about actions (see for example [3]). Transition systems are exactly the semantics underling PDL's.
2. Reiter's work on cognitive robotics [24, 26, 27, 21] has somewhat diverged the interest from nonmonotonic solutions to the frame problem, by illustrating that monotonic solutions are often very succinct. Now, while PDL's generally do not capture nonmonotonicity, they allow for exploiting the epistemological insight of the monotonic solutions to the frame problem, as shown later (see also [8]).

The general advantages PDL's offer in reasoning about actions are, on the one hand, the ability of expressing nondeterministic and complex actions, and, on the other hand, the availability of sophisticated tools for studying their computational aspects such as decidability, complexity, and reasoning algorithms.

In this paper we propose a very powerful Propositional Dynamic Logic, $\mathcal{DIFR}$, which offers an effective framework to model and reason about actions. The logic extends the previous formalisms in many ways. It allows for boolean expressions of atomic actions by which we can denote both the concurrent execution and the nonexecution of actions. It allows for expressing interdependencies between atomic actions such as specialization or mutual exclusion. It also includes constructs to impose the determinism of boolean combinations of atomic actions and their inverse. Notably, the logic is decidable and its computational complexity is EXPTIME (tight bound) as for the simplest PDL [9].

The rest of the paper is organized as follows: In Section 2 we introduce the logic $\mathcal{DIFR}$ both formally and intuitively; In Section 3 we illustrate, by means of examples, the use of $\mathcal{DIFR}$ in modeling and reasoning about actions; In Section 4 we discuss $\mathcal{DIFR}$ main features individually and we draw some conclusions.

2 The logic $\mathcal{DIFR}$

Formulae in the logic $\mathcal{DIFR}$ are of two sorts: *action formulae* and *state formulae*.

Action Formulae describe properties, by means of boolean operators, of *atomic actions* -i.e., actions that cannot be broken in sequences of smaller actions. The abstract syntax of action formulae is as follows:

$$\rho ::= P \mid \mathbf{any} \mid \rho_1 \wedge \rho_2 \mid \rho_1 \vee \rho_2 \mid \neg\rho$$

where P denotes a primitive action, and **any** denotes a special atomic action that can be thought of as "the most general atomic action". Observe that an atomic action denoted by an action formulae is composed, in general, by a set of primitive actions intended to be executed in parallel.

State Formulae describe properties of states in terms of propositions and complex actions. The abstract syntax for state formulae is as follows:

$$\begin{aligned} \phi &::= A \mid \top \mid \bot \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi \mid \\ &\quad [R]\phi \mid \langle R\rangle\phi \mid (\mathbf{fun}\, r) \\ r &::= \rho \mid \rho^- \\ R &::= r \mid R_1 \vee R_2 \mid R_1; R_2 \mid R^* \mid \phi? \mid R^- \end{aligned}$$

where A denotes a primitive proposition, $\top$ denotes "true", $\bot$ denotes "false", ϕ (possibly with subscript) denotes a state formula, r denotes a *simple action* which is either an atomic action or the inverse of an atomic action (i.e, set of primitive actions or of inverse of primitive actions), and finally R (possibly with subscript) denotes a complex action composing simple actions by nondeterministic choice, sequential composition, reflexive transitive closure, test, and inverse.

Let us explain the intuitive meaning of some formulae: the action formula $P_1 \wedge P_2$ means "perform P_1 and P_2 in parallel"; $\neg P$ means "don't perform P". In general an atomic formula ρ denotes a set of primitive actions that are performed in parallel and a set that are not performed at all (note that primitive actions that are not in these sets could be performed as well -i.e., we are adopting an open semantics for action formulae).

By forcing the validity of action formulae we can represent *hierarchies* of atomic actions, for example by $climb\_stairs \Rightarrow climb$ [1] we can represent that the action $climb\_stairs$ is a specialization of the action $climb$. In the same way we can represent *mutual exclusion*, for example by $\neg(open\_window \wedge close\_window)$ we can represent that the actions $open\_window$ and $close\_window$ cannot be performed together.

From atomic actions we build complex actions by means of constructors that are intuitively interpreted as follows: $R_1 \vee R_2$ means "nondeterministically perform R_1 or perform R_2"; $R_1; R_2$ means "perform R_1 and then R_2"; R^* means "repeat R a nondeterministically chosen number of times"; $\phi?$ means "test ϕ and proceed only if true"; R^- means "perform R in reverse". By using these constructs we can build complex action such as **if** ϕ **then** R_1 **else** R_2, which is represented by $(\phi?; R_2) \vee (\neg\phi?; R_2)$, or **while** ϕ **do** R which is represented by $(\phi?; R)^*; \neg\phi?$.

Turning to state formulae: $[R]\phi$ expresses that after *every performance* of the action R the property ϕ is satisfied; $\langle R\rangle\phi$ expresses that after *some performance* of the action R the property ϕ is satisfied -i.e. the execution of R *can* lead to a state where ϕ holds (recall that actions are nondeterministic in general).

The formula $\langle R\rangle\top$ expresses the *capability* of performing R; $[R]\bot$ expresses the *inability* of performing R; $[\neg r]\bot$ expresses the *inability* of performing any atomic actions *other than* those denoted by r; $[\neg\mathbf{any}]\bot$ expresses the inability of performing any atomic actions at all; $\langle\mathbf{any}\rangle\top \wedge [\neg r]\bot$ expresses the *necessity* or *inevitability* to perform some of the atomic actions denoted by r.

[1] As usual we will use $a \Rightarrow b$ an abbreviation of $\neg a \vee b$.

The construct $(\mathbf{fun}\, r)$ called *functional restriction* allows us to impose that the performance of a simple action r (i.e., an atomic action or the inverse of an atomic action) is deterministic. Hence $[r]\phi \wedge (\mathbf{fun}\, r)$ expresses that *if* the atomic action r is performed, then it *deterministically* leads to a state where ϕ holds. Note that this does not implies that the action r can be performed. The formula $\langle r \rangle \phi \wedge (\mathbf{fun}\, r)$ expresses that atomic action r *can* be performed and deterministically leads to a state where ϕ holds.

Propositional Dynamic Logics are subsets of Second Order Logic, or, more precisely, of First Order Logic plus Fixpoints. Typical properties that are not first order definable are: $\langle R^* \rangle \phi$, which expresses the *capability* for performing R *until* ϕ holds, and is equivalent to the least fixpoint of the operator $\lambda X.(\phi \vee \langle R \rangle X)$; $[R^*]\phi$, which expresses that ϕ holds in any state reachable from the current one by performing R any number of times, and is equivalent to the greatest fixpoint of the operator $\lambda X.(\phi \wedge [R]X)$. Interesting special cases of the last formula are: $[\mathbf{any}^*]\phi$, which expresses that ϕ holds *from now on* -i.e., no matter how the world evolves from the current state ϕ will be true; and $[(\mathbf{any} \vee \mathbf{any}^-)^*]\phi$, which expresses that ϕ holds in the whole connected component containing the current state (the state in which the formula holds).

The formal semantics of $\mathcal{DIFR}$ is based on the notion of *Kripke structure* or *transition system*, which is defined as a triple $M = (\mathcal{S}, \{\mathcal{R}_R\}, \mathcal{V})$, where $\mathcal{S}$ denotes a set of states, $\{R_R\}$ is a family of binary relations over $\mathcal{S}$, such that each action R is given a meaning through R_R, and $\mathcal{V}$ is a mapping from atomic propositions to subsets of $\mathcal{S}$ such that $\mathcal{V}(A)$ determines the states where the proposition A is true. The family $\{R_R\}$ is systematically defined as follows:

$$
\begin{array}{l}
\mathcal{R}_{\mathbf{any}} \subseteq \mathcal{S} \times \mathcal{S}, \\
\mathcal{R}_P \subseteq \mathcal{R}_{\mathbf{any}}, \\
\mathcal{R}_{\rho_1 \wedge \rho_2} = \mathcal{R}_{\rho_1} \cap \mathcal{R}_{\rho_2}, \\
\mathcal{R}_{\rho_1 \vee \rho_2} = \mathcal{R}_{\rho_1} \cup \mathcal{R}_{\rho_2}, \\
\mathcal{R}_{\neg \rho} = \mathcal{R}_{\mathbf{any}} - \mathcal{R}_\rho, \\
\mathcal{R}_{\rho^-} = \{(s_1, s_2) \in \mathcal{S} \times \mathcal{S} \mid (s_2, s_1) \in \mathcal{R}_\rho\}, \\
\mathcal{R}_r = \mathcal{R}_\rho \quad \text{if } r = \rho, \\
\mathcal{R}_r = \mathcal{R}_{\rho^-} \quad \text{if } r = \rho^-, \\
\mathcal{R}_{R_1 \vee R_2} = \mathcal{R}_{R_1} \cup \mathcal{R}_{R_2}, \\
\mathcal{R}_{R_1 ; R_2} = \mathcal{R}_{R_1} \circ \mathcal{R}_{R_2} \qquad \text{(seq. comp. of } \mathcal{R}_{R_1} \text{ and } \mathcal{R}_{R_2}\text{)}, \\
\mathcal{R}_{R^*} = (\mathcal{R}_R)^* \qquad \text{(refl. trans. closure of } \mathcal{R}_R\text{)}, \\
\mathcal{R}_{R^-} = \{(s_1, s_2) \in \mathcal{S} \times \mathcal{S} \mid (s_2, s_1) \in \mathcal{R}_R\}, \\
\mathcal{R}_{\phi?} = \{(s, s) \in \mathcal{S} \times \mathcal{S} \mid M, s \models \phi\}.
\end{array}
$$

Note that actions (even primitive actions) are nondeterministic in general.

The conditions for a state formula ϕ to hold at a state s of a structure M, written $M, s \models \phi$, are:

$M, s \models A$ iff $s \in \mathcal{V}(A)$
$M, s \models \top$ *always*,
$M, s \models \bot$ *never*,
$M, s \models \phi_1 \wedge \phi_2$ iff $M, s \models \phi_1$ and $M, s \models \phi_2$,
$M, s \models \phi_1 \vee \phi_2$ iff $M, s \models \phi_1$ or $M, s \models \phi_2$,
$M, s \models \neg\phi$ iff $M, s \not\models \phi$,
$M, s \models \langle R\rangle\phi$ iff $\exists s'.(s, s') \in \mathcal{R}_R$ and $M, s' \models \phi$,
$M, s \models [R]\phi$ iff $\forall s'.(s, s') \in \mathcal{R}_R$ implies $M, s' \models \phi$,
$M, s \models (\mathbf{fun}\, r)$ iff exists *at most one* $s'.(s, s') \in \mathcal{R}_r$.

A structure M is a model of an action formula ρ if $\mathcal{R}_\rho = \mathcal{R}_{\mathbf{any}}$. A structure M is a model of a state formula ϕ if for all s in M, $M, s \models \phi$. Let Γ be a finite set of both state and action formulae, a structure is a model of Γ if is a model of every formula in Γ. A set of formulae Γ *logically implies* a (state or action) formula ψ, written

$$\Gamma \models \psi$$

if all the models of Γ are models of ψ as well.

A crucial question to be answered is: Is logical implication decidable in $\mathcal{DIFR}$? If yes, which is its computational complexity? Note that known results in PDL's do not help directly. It is possible to prove (see [5]) that this problem is indeed decidable and its computational complexity can be precisely characterized, by providing a reduction to the PDL $\mathcal{DIF}$ presented in [6].

Theorem 1. *Logical implication for $\mathcal{DIFR}$ is an EXPTIME-complete problem.*

Observe that logical implication is already EXPTIME-complete for the basic modal logic $\mathcal{K}$ (which corresponds to a Propositional Dynamic Logic including just one primitive action, no functional restrictions, and no action constructor at all).

3 Examples

Below we show the power of $\mathcal{DIFR}$ in modeling a dynamically changing world by means of two examples. We remark that those examples do not aim at providing the definitive $\mathcal{DIFR}$-based formalizations of the scenarios they describe, nor they exhaust the possibility of using $\mathcal{DIFR}$ in representing and reasoning about actions[2]. They are intended to give a taste of what can be done with such a logic. In the examples we refer to situation calculus as it is presented in [24, 26, 27, 21].

Example: Lifting both sides of a table A vase is on top of a table, and if just one side is lifted then it slides down and falls on the floor. However if both

[2] In addition these examples do not make use of many features of the logic such as axioms on atomic actions.

sides are simultaneously lifted this doesn't happen [12]. We formalize the scenario as follows. We consider the following primitive propositions (corresponding to "propositional" fluents in situation calculus): $vase\_on\_table$, $down\_left\_side$, $down\_right\_side$; and the following primitive actions (corresponding to actions in situation calculus[3]): $vase\_slides\_down$, $lift\_left$, $lift\_right$. The intended meaning of these propositions and actions is the natural one (sometimes we use initials as abbreviations):

As usual actions have *preconditions* which are conditions that must be satisfied in order to be able to perform the action[4].

$$\begin{array}{l}\langle lift\_left\rangle\top \equiv down\_left\_side\\ \langle lift\_right\rangle\top \equiv down\_right\_side\\ \langle vsd\rangle\top \equiv (vot \land ((dls \land \neg drs) \lor (\neg dls \land drs)))\end{array}$$

Actually the if part of the last axioms must be strengthened: If the vase is on the table and one of the side of the table is not on the floor, then it is *inevitable* (not just possible) that the vase slides towards the floor. This can be enforced by:

$$(vot \land ((dls \land \neg drs) \lor (\neg dls \land drs))) \Rightarrow \langle \mathbf{any}\rangle\top \land [\neg vsd]\bot.$$

We also need to specify when the actions $lift\_left$ and $lift\_right$ can be performed simultaneously. With the next axiom we assert that they can be performed simultaneously simply if they both can be performed individually:

$$\langle lift\_left \land lift\_right\rangle\top \equiv \langle lift\_left\rangle\top \land \langle lift\_right\rangle\top.$$

Actions have *effects* if they can be performed[5]:

$$\begin{array}{l}[lift\_left]\neg down\_left\_side\\ [lift\_right]\neg down\_right\_side\\ [vase\_slides\_down]\neg vase\_on\_table\end{array}$$

As usual we need to cope with the frame problem. We do it by adopting a monotonic solution as in [13, 30, 24]. We enforce the following *frame axioms* saying that if the vase is on the table then all atomic actions not including $vase\_slides\_down$ leave the vase on the table; if the vase is not on the table then no atomic action will change its position; etc.:

$$\begin{array}{l}vase\_on\_table \Rightarrow [\neg vase\_slides\_down]vase\_on\_table\\ down\_left\_side \Rightarrow [\neg lift\_left]down\_left\_side\\ down\_right\_side \Rightarrow [\neg lift\_right]down\_right\_side\\ \\ \neg vase\_on\_table \Rightarrow [\mathbf{any}]\neg vase\_on\_table\\ \neg down\_left\_side \Rightarrow [\mathbf{any}]\neg down\_left\_side\\ \neg down\_right\_side \Rightarrow [\mathbf{any}]\neg down\_right\_side\end{array}$$

[3] Note that (contrary to what is usually assumed in situation calculus) actions are not necessarily deterministic in $\mathcal{DIFR}$.

[4] Note that $\langle r\rangle\top$ have the same role as $Poss(a, s)$ in Reiter's situation calculus.

[5] Note that state formulae of the form $[a]\phi$ have the same role as $Poss(a, s) \Rightarrow \phi(do(a, s))$ which is a common formula configuration in Reiter's situation calculus [24, 27].

We adopted the last three axioms for sake of brevity.

Let us call Γ the set of the axioms above and let the starting situation be described by

$$S \doteq vase\_on\_table \wedge down\_left\_side \wedge down\_right\_side.$$

Then we can make the following two inferences. On the one hand:

$$\Gamma \models S \Rightarrow [ll \wedge lr][vase\_slides\_down]\bot$$

that is if the vase is on the table and both the sides of the table are on the floor, then lifting the two sides concurrently does not make the vase falling. On the other hand:

$$\Gamma \models S \Rightarrow [ll \wedge \neg lr][lr]\neg vase\_on\_table$$

that is if the vase is on the table and both the sides of the table are on the floor, then lifting first the left side without lifting the right side and then the right side, has as a result that the vase is fallen. Notice that the above inferences don't say anything about the possibility of performing the actions described, however this possibility is guaranteed by $\Gamma \models S \Rightarrow \langle lift\_left \wedge lift\_right\rangle\top$ and $\Gamma \models S \Rightarrow \langle (lift\_left \wedge \neg lift\_right); lift\_right)\rangle\top$ respectively.

Example: Making the heating operative We want to make our (gas) heating operative. To do so we need to strike a match, to turn its gas handle on and to ignite the security flame spot. To strike a match we need to concurrently press the match against the match box and rub it until it fires.

We make the following intuitive assumption: the past is *backward linear* that is from any state there is only one accessible (immediately) previous state. This can be easily imposed by means of the following axiom:

$$(\mathbf{fun}\,\mathbf{any}^{-}).$$

We assume the following preconditions and effects of actions.

Preconditions:

$$\begin{array}{l} \langle turn\_on\_gas\rangle\top \equiv \neg gas\_open \\ \langle turn\_off\_gas\rangle\top \equiv gas\_open \\ \langle ignite\_flame\_spot\rangle\top \equiv match\_lit \\ \langle press\rangle\top \Rightarrow \neg match\_lit \\ \langle rub\rangle\top \Rightarrow \neg match\_lit \\ \langle \mathbf{while}\ \neg match\_lit\ \mathbf{do}\ (press \wedge rub)\rangle\top \end{array}$$

Effects:

$$\begin{array}{l} match\_lit \wedge gas\_open \Rightarrow \\ \qquad [ignite\_flame\_spot]heating\_operative \\ [turn\_on\_gas]gas\_open \\ [turn\_off\_gas]\neg gas\_open \end{array}$$

In this example we model frame axioms more systematically starting from *explanation closure axioms* [30] in line with [24, 27]. There are two main difficulty in following this approach in PDL: the first is that, as in any modal logic, we can directly refer to just one state the "current one", the second is that we cannot quantify on atomic actions. In $\mathcal{DIFR}$ we can overcome these difficulties. By assuming ($\mathbf{fun}\,\mathbf{any}^-$) from the current state we can univocally refer back to *the* previous state through the action $\mathbf{any}^-$. On the other hand by using the action **any** we can simulate the universal quantification on atomic actions. Hence we proceed as follows from the current state we make a step forward and then we model the various condition backward. This leads to the following frame axioms:

$$
\begin{array}{l}
[\mathbf{any}]\neg gas\_open \Rightarrow \\
\qquad \langle \mathbf{any}^- \rangle \neg gas\_open \vee \langle turn\_off\_gas^- \rangle \top \\
[\mathbf{any}]\neg match\_lit \Rightarrow \\
\qquad \langle \mathbf{any}^- \rangle \neg match\_lit \\
[\mathbf{any}]\neg heating\_operative \Rightarrow \\
\qquad \langle \mathbf{any}^- \rangle \neg heating\_operative
\end{array}
$$

$$
\begin{array}{l}
[\mathbf{any}] gas\_open \Rightarrow \\
\qquad \langle \mathbf{any}^- \rangle gas\_open \vee \langle turn\_on\_gas^- \rangle \top \\
[\mathbf{any}] match\_lit \Rightarrow \\
\qquad \langle \mathbf{any}^- \rangle match\_lit \vee \langle (press \wedge rub)^- \rangle \top \\
[\mathbf{any}] heating\_operative \Rightarrow \\
\qquad \langle \mathbf{any}^- \rangle heating\_operative \vee \\
\qquad \langle ignite\_flame\_spot^- \rangle gas\_open
\end{array}
$$

For example the last axiom says: "consider any successor state (such a state has exactly one previous state which is the current state), if the heating is operative in such a state then either it was operative in the previous state or the action *ignite_flame_spot* was just performed and the gas was open in the previous state".[6]

Let us call Γ the set of all these axioms, and let the starting situation be described by

$$S \doteq \neg open\_gas \wedge \neg match\_lit \wedge \neg heating\_operative$$

[6] The frame axioms can be proved to be equivalent to the following ones (respecting the order):

$$
\begin{array}{l}
gas\_open \Rightarrow [\neg turn\_off\_gas] gas\_open \\
match\_lit \Rightarrow [\mathbf{any}] match\_lit \\
heating\_operative \Rightarrow [\mathbf{any}] heating\_operative
\end{array}
$$

$$
\begin{array}{l}
\neg gas\_open \Rightarrow [\neg turn\_on\_gas] \neg gas\_open \\
\neg match\_lit \Rightarrow [\neg (press \wedge rub)] \neg match\_lit \\
\neg heating\_operative \Rightarrow \\
\qquad [\neg ignite\_flame\_spot \vee \\
\qquad\quad \neg gas\_open?; \mathbf{any}] \neg heating\_operative.
\end{array}
$$

The last axiom says: "if the heating is not operative then every performance of an atomic action not including *ignite_flame_spot* and every performance of any action starting from a state in which the gas is not open, leads to a state where the heating is still not operative".

The first inference we are interested in is the following:

$$\Gamma \models S \Rightarrow \langle \mathbf{any}^* \rangle heating\_operative$$

i.e., there is a sequence of action (a plan) starting from a situation described by S resulting in making the heating operative. Assuming all primitive actions to be deterministic, inferences of the form

$$\Gamma \models S \Rightarrow \langle \mathbf{any}^* \rangle G$$

are the typical starting point in planning synthesis [11]: if the answer is yes then from the proof we can generate a working plan to achieve the goal G starting from an initial situation described by S. The dual of the above inference

$$\Gamma \models S \Rightarrow [\mathbf{any}^*] \neg G$$

is of interest as well: it establishes that there are no plan at all achieving a given goal G starting from a situation described be S.

Next inference says that the complex action "strike a match turn on the gas, ignite the control flame spot" results in making the heating operative:

$$\begin{aligned}\Gamma \models \langle & \mathbf{while}\, \neg match\_lit\ \mathbf{do}\ (press \wedge push);\\ & turn\_on\_gas;\\ & ignite\_flame\_spot\\ \rangle & heating\_operative\end{aligned}$$

Note that the similar action "turn on the gas, strike a match, ignite the control flame spot" is not guaranteed to make the heating operative.

$$\begin{aligned}\Gamma \not\models \langle & turn\_on\_gas;\\ & \mathbf{while}\, \neg match\_lit\ \mathbf{do}\ (press \wedge push);\\ & ignite\_flame\_spot\\ \rangle & heating\_operative\end{aligned}$$

The reason why above the complex action may fail is because the gas could be turned off while we are trying to strike the match.

The problem of checking inferences as the two above is known as *projection problem* (see e.g. [26]). A typical projection problem as the form: Does G hold in the state reachable from initial situation described by S by executing the (complex) action α? This corresponds to checking the inference below:

$$\Gamma \models S \Rightarrow \langle \alpha \rangle G.$$

We have seen that executing the complex action "turn on the gas, strike a match, ignite the control flame spot" may fail to make the heating operative. If this is the case, the following inference tells us that the gas has been turned off before striking the match succeeded:

$$\begin{aligned}\Gamma \models \langle & turn\_on\_gas;\\ & \mathbf{while}\, \neg match\_lit\ \mathbf{do}\ (press \wedge push);\\ & ignite\_flame\_spot\\ \rangle & (\neg heating\_operative \Rightarrow\\ & \qquad \langle (\mathbf{any}^-; \mathbf{any}^-)^*; turn\_off\_gas \rangle \top)\end{aligned}$$

Inferences as the one above are answers to "historical queries" [26, 25] -i.e., queries of the form: if from the initial state described by S we execute the complex action α getting ϕ, then does this implies that before the termination of α, a given formula ϕ' is true in some state, or that a given action a has been executed? These questions can be answered by checking the inferences[7]:

$$\Gamma \models S \Rightarrow \langle\alpha\rangle(\phi \Rightarrow \langle(\mathbf{any}^-)^*\rangle\phi')$$

$$\Gamma \models S \Rightarrow \langle\alpha\rangle(\phi \Rightarrow \langle(\mathbf{any}^-)^*; a^-\rangle\top).$$

4 Discussion

It is our opinion that Propositional Dynamic Logics offer a elegant framework with a well understood semantics and precise computational characterization, that makes them a kind of Principled Monotonic Propositional Situation Calculus extended to deal with complex actions.

According to this perspective, $\mathcal{DIFR}$ has been designed to address issues that are important in modeling actions but are not satisfactorily dealt with by traditional PDL's. Here, we briefly discuss the most relevant features of $\mathcal{DIFR}$ in modeling actions.

• The ability of specifying the performance of different atomic actions concurrently. This characteristic, illustrated in the examples of Section 3, is one of the most original aspects of $\mathcal{DIFR}$. Indeed, the attention to reason about concurrent actions has emerged only recently. $\mathcal{DIFR}$ takes into account concurrency of actions that cannot be interrupted (atomic actions in our terminology). Obviously further work has to be done for capturing more general forms of concurrency. In this context, we argue that it is relevant for the AI community to look at the vast computer science literature on modeling concurrent processes.

• The ability of specifying the "non-execution" of atomic actions. This feature called for a careful definition of the notion of "non-executing an action". In our approach, this notion has been formulated by interpreting it as "the execution of some action other than a given one". Observe that it is essentially this feature that allows us to provide a compact representation of the frame axioms, as illustrated in the examples above.

• The ability of structuring atomic actions. In particular, $\mathcal{DIFR}$ allow the designer to organize actions in hierarchies, where actions are related by means of two basic mechanisms: one for stating that an action is a specialization of another one, and the other for representing mutual exclusion between actions.

[7] Observe that ϕ' (a) could be true (executed) before the starting of α in the formulation above. To avoid this we need to distinguish the initial state, for example by assuming that the initial situation does not have a past, which can be done by including in S the state formula $[\mathbf{any}^-]\bot$.

• The ability of distinguishing deterministic and nondeterministic atomic actions. Note that in $\mathcal{DIFR}$ the determinism or nondeterminism of an atomic action may be modeled on a state-to-state basis. This ability provides the designer with more expressive power with respect to the case where actions are assumed to be always deterministic. Indeed in this last case there is no distinction between nondeterminism and incomplete knowledge about the situation resulting from executing an action (see for example [2]).

• The ability of expressing properties of both future and past states. In particular, the usual linearity of the past can be asserted. This ability makes our logic capable to reason about not only projection problems but also historical queries. Some examples of these have been provided in Section 3.

The result on the computational properties of $\mathcal{DIFR}$ shows that the logic is decidable, which means that reasoning procedure that are sound, complete, and terminating are available. Space limitations prevented us to elaborate more on this issue, the interested reader is referred to [5] for a deep investigation.

References

1. P. Blackburn and E. Spaan. A modal perspective on computational complexity of attribute value grammar. *Journal of Logic, Language and Computation*, 2:129–169, 1993.
2. C. Boutilier and N. Friedman. Nondeterministic actions and the frame problem. In [3], 39–44, 1995.
3. C. Boutilier, M. Goldszmidt, T. Dean, S. Hanks, D. Heckerman, and R. Reiter, editors. *Working notes of the AAAI 1995 Spring Symposium on Extending Theories of Action: Formal and Practical Applications*, Stanford, CA, USA, 1995.
4. P. Cohen and H. Levesque. Intention is choice with communication. *Artificial Intelligence*, 42:213–261, 1990.
5. G. De Giacomo. *Decidability of Class-Based Knowledge Representation Formalisms and their Application to Medical Terminology Servers.* PhD thesis, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", 1995.
6. G. De Giacomo and M. Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proceedings of the 12th National Conference on Artificial Intelligence*, pages 205–212, 1994.
7. G. De Giacomo and M. Lenzerini. Description logics with inverse roles, functional restrictions, and n-ary relations. In *Proceedings of the 4th European Workshop on Logics in Artificial Intelligence*, LNAI 838, pages 332–346. Springer-Verlag, 1994.
8. G. De Giacomo and M. Lenzerini. Enhanced propositional dynamic logic for reasoning about concurrent actions (extended abstract). In [3], pages 62–67, 1995.
9. N. J. Fisher and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18:194–211, 1979.
10. N. Friedman and J. Halpern. On the complexity of conditional logics. In *Proc. of the 4th Int. Conf. on Principles of Knowledge Representation and Reasoning*, 1994.
11. C. Green. Theorem proving by resolution as basis for question-answering systems. In *Machine Intelligence*, volume 4, pages 183–205. American Elsevier, 1969.

12. G. Grosse. Propositional state event logic. In *Proceedings of the 4th European Workshop on Logics in Artificial Intelligence*, LNAI 838, pages 316–331. Springer-Verlag, 1994.
13. A. Haas. The case for domain-specific frame axioms. In *Proc. of the Workshop on the Frame Problem*, pages 343–348. Morgan Kaufmann Publishers, 1987.
14. J. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379, 1992.
15. D. Harel. Dynamic logic. In D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, pages 497–603. D. Reidel Publishing Company, Oxford, 1984.
16. M. Hennessy and R. Milner. Algebraic laws for nondetrminism and concurrency. *Journal of Association for Computing Machinery*, 32:137–162, 1985.
17. H. Kautz. A first order dynamic logic for planning. Master's thesis, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, 1980.
18. D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–355, 1983.
19. D. Kozen and J. Tiuryn. Logics of programs. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 790–840. Elsevier Science Publishers, 1990.
20. K. J. Larsen. Proof systems for satisfiability in Hennessy-Milner logic with recursion. *Theoretical Computer Science*, 72:265–288, 1990.
21. F. Lin and R. Reiter. State constraints revisited. *Journal of Logic and Computation, Special Issue on Action and Processes*, 4(5):655–678, 1994.
22. M. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.
23. R. Parikh. Propositional dynamic logic of programs: A survey. In *Proceedings of the 1st Workshop on Logic of Programs*, LNCS 125, pages 102–144. Springer-Verlag, 1981.
24. R. Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 359–380. Academic Press, 1991.
25. R. Reiter. Formalizing database evolution in the situation calculus. In *Proc. Int. Conf. on Fifth Generation Computer Systems*, pages 600–609, 1992.
26. R. Reiter. The projection problem in the situation calculus: a soundness and completeness result, with an application to database updates. In *Proc. First Int. Conf. on AI Planning Systems*, pages 198–203, 1992.
27. R. Reiter. Proving properties of states in the situation calculus. *Artificial Intelligence*, 64:337–351, 1993.
28. S. Rosenschein. Plan synthesis: a logical approach. In *Proc. of the 8th Int. Joint Conf. on Artificial Intelligence*, 1981.
29. K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence*, 1991.
30. L. Schubert. Monotonic solution of the frame problem in the situation calculus: an efficient method for worlds with fully specified actions. In *Knowledge representation and Defeasible Reasoning*, pages 23–67. Kluwer Academic Press, 1990.
31. C. Stirling. Modal and temporal logic. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, pages 477–563. Clarendon Press, Oxford, 1992.

This article was processed using the LaTeX macro package with LLNCS style