Enhanced Propositional Dynamic Logic for Reasoning about Concurrent Actions (extended abstract)

Giuseppe De Giacomo and Maurizio Lenzerini

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

Introduction

This paper presents a work in progress on enhanced Propositional Dynamic Logics for reasoning about actions. Propositional Dynamic Logics (PDL's) are modal logics for describing and reasoning about system dynamics in terms of properties of states and actions[1] modeled as relations between states (see (Kozen & Tiuryn 1990; Harel 1984; Parikh 1981) for surveys on PDL's, see also (Stirling 1992) for a somewhat different account). The language of PDL includes formulae built from the boolean combinations of atomic propositions that are interpreted as simple properties of states, plus the construct $\langle R \rangle \phi$, where ϕ is a formula and R is an action, whose meaning is that it is possible to perform R and terminate in a state where ϕ is true. The action R can be either an atomic action, or a complex expression denoting sequential composition, nondeterministic choice, iteration, or test.

PDL's have been originally developed in Theoretical Computer Science to reason about program schemas (Fisher & Ladner 1979), and their variants have been adopted to specify and verify properties of reactive processes (e.g., Hennessy Milner Logic (Hennessy & Milner 1985; Milner 1989), modal mu-calculus (Kozen 1983; Larsen 1990; Stirling 1992)). They are also of interest in Philosophical Logic as a formalism to capture "procedural reasoning" (see, for example, (Van Benthem & Bergstra 1993; Van Benthem, Van Eijck, & Stebletsova 1993; de Rijke.M 1992; Van Benthem 1991)).

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 (Halpern 1992), Conditional Logics (Friedman & Halpern 1994), Description Logics (Schild 1991; De Giacomo & Lenzerini 1994a; 1994c), Features Logics (Blackburn & Spaan 1993). However they have been only sparingly adopted for reasoning about actions, main exceptions being (Rosenschein 1991; Kautz 1980) (but also (Cohen & Levesque 1990)).

[1] In this work we do not distinguish between actions and events.

Propositional Dynamic Logics offer a elegant framework with a well understood semantics and precise computational characterization, that in our opinion makes them a kind of Principled Monotonic Propositional Situation Calculus extended to deal with complex actions.[2]

In this paper we propose a new Propositional Dynamic Logic that includes boolean expressions of primitive actions denoting sets of primitive actions executed concurrently, and that allows to represent interdependencies between primitive actions as specialization or disjointness. Furthermore the logic includes constructs to impose the determinism of boolean combinations of primitive actions and their inverse. We have established that such logic is decidable and its computational complexity is EXPTIME (tight bound). We show some possible use of this logic in reasoning about actions by means of examples.

The logic $\mathcal{DIFR}$

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

Action Formulae describe, by means of boolean operators, properties of *atomic actions* -i.e., actions that cannot be broken into 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, **any** denotes a special atomic action that can be thought of as "the most general atomic action", and ρ (possibly with subscript) denotes an action formula. 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 ab-

[2] In this perspective many recent results on PDL's are relevant, for example (Danecki 1984; Vardi & Wolper 1986; Passy & Tinchev 1991; De Giacomo & Lenzerini 1994b).

stract syntax for state formulae is as follows:

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

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 $climb\_stairs \Rightarrow climb$ [3] we can represent that the action *climb_stairs* is a specialization of 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 $\mathbf{if}\,\phi\,\mathbf{then}\,R_1\,\mathbf{else}\,R_2$, which is represented by $(\phi?; R_2) \vee (\neg\phi?; R_2)$, or $\mathbf{while}\,\phi\,\mathbf{do}\,R$ 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.

The construct $(\mathbf{fun}\, r)$ called *functional restriction* allows us to impose that the performance of a simple action r (i.e. of 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 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 interpreted 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 $\mathcal{S}$ to atomic propositions such that $\mathcal{V}(s)$ determines the propositions that are true in the state s. The family $\{R_R\}$ is systematically defined as follows:

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

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:

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

$M, s \models A$ 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 $\mathcal{DIFR}$? And if yes, which is its computational complexity? Note that known results in PDL's do not help directly. We have proven that this problem is indeed decidable and we have precisely characterized its computational complexity, by providing a reduction to the PDL $\mathcal{DIF}$ presented in (De Giacomo & Lenzerini 1994a).

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 constructors at all).

Using $\mathcal{DIFR}$ for reasoning about actions

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 [4]. 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 (Reiter 1991; 1992b; 1993; Lin & Reiter 1994).

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 sides are simultaneously lifted this doesn't happen (Grosse 1994). 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[5]): *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). We do not include actions to put down the table for sake of brevity.

As usual actions have *preconditions* which are conditions that must be satisfied in order to be able to perform the action[6].

$$\langle lift\_left \rangle \top \equiv down\_left\_side$$
$$\langle lift\_right \rangle \top \equiv down\_right\_side$$
$$\langle vsd \rangle \top \equiv (vot \wedge ((dls \wedge \neg drs) \vee (\neg dls \wedge drs))).$$

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:

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

We need also 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 when they both can be performed:

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

Actions have *effects* if they can be performed[7]:

$$[lift\_left] \neg down\_left\_side$$
$$[lift\_right] \neg down\_right\_side$$
$$[vase\_slides\_down] \neg vase\_on\_table.$$

As usual we need to cope with the frame problem. We do it by adopting a monotonic solution as in (Haas 1987; Schubert 1990; Reiter 1991). We enforce the following *frame axioms* saying that if the vase in 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

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

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

[6] State formulae of the form $\langle a \rangle \top$ have the same role as $Poss(a, s)$ in Reiter's situation calculus.

[7] 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 (Reiter 1991; 1993).

change its position; etc.:

$vase\_on\_table \Rightarrow [\neg vase\_slides\_down]vase\_on\_table$
$down\_left\_side \Rightarrow [\neg lift\_left]down\_left\_side$
$down\_right\_side \Rightarrow [\neg lift\_right]down\_right\_side$

$\neg vase\_on\_table \Rightarrow [\mathbf{any}]\neg vase\_on\_table$
$\neg down\_left\_side \Rightarrow [\mathbf{any}]\neg down\_left\_side$
$\neg down\_right\_side \Rightarrow [\mathbf{any}]\neg down\_right\_side.$

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 $\Gamma \models S \Rightarrow \langle (lift\_left \wedge lift\_right); lift\_right)\rangle\top$, 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:

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

We assume the following preconditions and effects of actions.

Preconditions:

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

Effects:

$match\_lit \wedge gas\_open \Rightarrow$
$\quad [ignite\_flame\_spot]heating\_operative$
$[turn\_on\_gas]gas\_open$
$[turn\_off\_gas]\neg gas\_open.$

In this example we model frame axioms more systematically starting from *explanation closure axioms* (Schubert 1990) in line with (Reiter 1991; 1993). There are two main difficulty in following this approach in PDL: the first is that, as in any standard modal logic, we can directly refer to just one state, the "current one"; the second is that we cannot quantify on atomic actions. In $\mathcal{DLFR}$ we can overcome these difficulties. By assuming $(\mathbf{fun\ any}^-)$ 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:

$[\mathbf{any}](\neg gas\_open \Rightarrow$
$\quad \langle \mathbf{any}^-\rangle\neg gas\_open \vee \langle turn\_off\_gas^-\rangle\top)$
$[\mathbf{any}](\neg match\_lit \Rightarrow$
$\quad \langle \mathbf{any}^-\rangle\neg match\_lit)$
$[\mathbf{any}](\neg heating\_operative \Rightarrow$
$\quad \langle \mathbf{any}^-\rangle\neg heating\_operative)$

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

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".[8]

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

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

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

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

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

The last axiom says: "if the heating is not operative then both 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:

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

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 (Green 1969): 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:

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

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:

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

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. (Reiter 1992b)). A typical projection problem as the form: Does G hold in a state reachable from initial situation, described as 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:

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

Inferences as the one above are answers to "historical queries" (Reiter 1992b; 1992a) -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, ϕ' is true in some state, or does it implies that the action a as been executed? These questions can be answered by checking the inferences [9]:

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

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

References

Blackburn, P., and Spaan, E. 1993. A modal perspective on computational complexity of attribute value grammar. *Journal of Logic, Language and Computation* 2:129–169.

Cohen, P., and Levesque, H. 1990. Intention is choice with communication. *Artificial Intelligence* 42:213–261.

Danecki, R. 1984. Nondeterministic propositional dynamic logic with intersection is decidable. In *Proceedings of the 5th Symposium on Computational Theory*, number 208 in Lecture Notes in Computer Science, 34–53. Springer-Verlag.

De Giacomo, G., and Lenzerini, M. 1994a. Boosting the correspondence between description logics and propositional dynamic logics. In *Proceedings of the 12th National Conference on Artificial Intelligence*, 205–212.

De Giacomo, G., and Lenzerini, M. 1994b. Converse, local determinism, and graded nondeterminism in propositional dynamic logics. Technical Report 11-94, Dipartimento di Informatica e Sistemistica, Universitá di Roma "La Sapienza".

De Giacomo, G., and Lenzerini, M. 1994c. Description logics with inverse roles, functional restrictions, and n-ary relations. In *Proceedings of the 4th European Workshop on Logics in Artificial Intelligence*, volume 838 of *Lecture Notes in Artificial Intelligence*, 332–346. Springer-Verlag.

[9] Observe that ϕ' (a) could be true (executed) before the starting of α in the formulation above. If we want to avoid this, we may assume that the initial situation does not have a past, which can be done by including in S the state formula $[\mathbf{any}]\bot$. Another solution is to add a new proposition *starting_state* to S and impose the axiom

$$\mathit{starting\_state} \Rightarrow [(\mathbf{any}; \mathbf{any}^*) \vee (\mathbf{any}^-; (\mathbf{any}^-)^*)] \neg \mathit{starting\_state}$$

saying that in every state reachable from one in which *starting_state* holds, *starting_state* does not hold (this does not influence logical implications not involving the proposition *starting_state*, since $\mathcal{DLFR}$ satisfies the tree model property). In this way we can test for the starting state in going backward along α.

de Rijke.M. 1992. A system of dynamic modal logic. Technical Report LP-92-08, Institute for Logic, Language and Computation, University of Amsterdam.

Fisher, N. J., and Ladner, R. E. 1979. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences* 18:194–211.

Friedman, N., and Halpern, J. 1994. On the complexity of conditional logics. In *Proc. of the 4th Int. Conf. on Principles of Knowledge Representation and Reasoning.*

Green, C. 1969. Theorem proving by resolution as basis for question-answering systems. In *Machine Intelligence*, volume 4. American Elsevier. 183–205.

Grosse, G. 1994. Propositional state event logic. In *Proceedings of the 4th European Workshop on Logics in Artificial Intelligence*, volume 838 of *Lecture Notes in Artificial Intelligence*, 316–331. Springer-Verlag.

Haas, A. 1987. The case for domain-specific frame axioms. In *Proc. of the Workshop on the Frame Problem.* Morgan Kaufmann Publishers. 343–348.

Halpern, J.Y.and Moses, Y. 1992. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence* 54:319–379.

Harel, D. 1984. Dynamic logic. In Gabbay, D. M., and Guenthner, F., eds., *Handbook of Philosophical Logic.* Oxford: D. Reidel Publishing Company. 497–603.

Hennessy, M., and Milner, R. 1985. Algebraic laws for nondetrminism and concurrency. *Journal of Association for Computing Machinary* 32:137–162.

Kautz, H. 1980. A first order dynamic logic for planning. Master's thesis, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada.

Kozen, D., and Tiuryn, J. 1990. Logics of programs. In van Leeuwen, J., ed., *Handbook of Theoretical Computer Science.* Elsevier Science Publishers. 790–840.

Kozen, D. 1983. Results on the propositional mu-calculus. *Theoretical Computer Science* 27:333–355.

Larsen, K. J. 1990. Proof systems for satisfiability in Hennessy-Milner logic with recursion. *Theoretical Computer Science* 72:265–288.

Lin, F., and Reiter, R. 1994. State constraints revisited. *Journal of Logic and Computation, Special Issue on Action and Processes.* To appear.

Milner, M. 1989. *Communication and Concurrency.* Prentice-Hall.

Parikh, R. 1981. Propositional dynamic logic of programs: A survey. In *Proceedings of the 1st Workshop on Logic of Programs*, number 125 in Lecture Notes in Computer Science, 102–144. Springer-Verlag.

Passy, S., and Tinchev, T. 1991. An essay in combinatory dynamic logic. *Information and Computation* 93:263–332.

Reiter, R. 1991. 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.* Academic Press. 359–380.

Reiter, R. 1992a. Formalizing database evolution in the situation calculus. In *Proc. Int. Conf. on Fifth Genearation Computer Systems*, 600–609.

Reiter, R. 1992b. 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*, 198–203.

Reiter, R. 1993. Proving properties of states in the situation calculus. *Artificial Intelligence* 64:337–351.

Rosenschein, S. 1991. Plan synthesis: a logical approach. In *Proc. of the 8th Int. Joint Conf. on Artificial Intelligence.*

Schild, K. 1991. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence.*

Schubert, L. 1990. 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.* Kluwer Academic Press. 23–67.

Stirling, C. 1992. Modal and temporal logic. In Abramsky, S.; Gabbay, D. M.; and Maibaum, T. S. E., eds., *Handbook of Logic in Computer Science.* Oxford: Clarendon Press. 477–563.

Van Benthem, J., and Bergstra, J. 1993. Logic of transition systems. Technical report, Institute for Logic, Language and Computation, University of Amsterdam.

Van Benthem, J.; Van Eijck, J.; and Stebletsova, V. 1993. Modal logic, transition systems and processes. Technical Report CS-R9321, CWI.

Van Benthem, J. 1991. *Language in Action: Categories, Lambdas and Dynamic Logic*, volume 130 of *Studies in Logic.* Elsevier.

Vardi, M. Y., and Wolper, P. 1986. Automata-theoretic techniques for modal logics of programs. *Journal of Computer and System Sciences* 32:183–221.