The Markov Assumption: Formalization and Impact

Alexander Bochman
Computer Science Department,
Holon Institute of Technology, Israel

Abstract

We provide both a semantic interpretation and logical (inferential) characterization of the Markov principle that underlies the main action theories in AI. This principle will be shown to constitute a nonmonotonic assumption that justifies the actual restrictions on action descriptions in these theories, as well as constraints on allowable queries. It will be shown also that the well-known regression principle is a consequence of the Markov assumption, and it is valid also for non-deterministic domains.

1 Introduction

Theories of action and change form one of the central subjects of AI. A wide range of such theories have been suggested in the literature, starting from low-level representations of actions in first-order languages (such as the situation calculus) and ending with higher-order action description languages, as well as representations of actions in general formalisms of propositional dynamic logic (PDL) and linear temporal logic (LTL). Moreover, one of the useful ways of providing transferability results among different theories involves designing further, still more general theories that subsume existing ones as special cases (see, e.g., [Thielscher, 2011]).

In this study we will pursue, however, a somewhat different, top-down approach, which amounts to finding common knowledge representation principles that underly these action theories. Our hope is that it is possible to single out a relatively small number of 'AI-specific' principles and assumptions in such a way that existing action theories could be viewed as implementations of the same principles in particular representation frameworks.

The starting point of this study is the fact that actual descriptions used for representing actions in all the above mentioned formalisms employ only a modest part of expressive capabilities allowed by their host logical frameworks. This is clearly true for the action descriptions formulated in the classical first-order framework, but it pertains also to formalizations of actions in logic programming, PDL and LTL. As we are going to show in this study, the informal Markov principle provides an ultimate basis for most of these restrictions, and it is 'responsible' for many important special properties of these action theories, such as the executability assumption and the regression principle.

The plan of the paper is as follows. As a basic representation framework, we will adopt a formalism of Sequential Dynamic Logic (SDL) suggested in [Bochman and Gabbay, 2012] for a general logical description of action domains. The formalism of SDL is expressively equivalent to propositional dynamic logic (PDL), but it is formulated as a plain propositional language with the usual (though generalized) propositional connectives, and without modal operators. This makes this formalism quite transparent with respect to commonsense descriptions of dynamic domains. The formalism will also allow us to provide a concise syntactic characterization of the Markov principle as a special kind of interpolation. It will be shown, in particular, that this principle allows us to explain the actual restrictions on the form of action descriptions in action theories, as well as the constraints on allowable queries. In addition, we will show that the regression principle can also be viewed as an immediate consequence of the Markov property, and that it is valid also for non-deterministic domains.

2 Sequential Dynamic Logic

In this section we will provide a brief description of sequential dynamic logic (SDL) from [Bochman and Gabbay, 2012].

The vocabulary of SDL contains propositional atoms of two kinds, *actions* $\Pi = \{\pi, \rho, \dots\}$, and *fluents*[1] $P = \{p, q, \dots\}$. The language of the logic is a set of logical formulas $\{A, B, C, \dots\}$ constructed from propositional atoms (of either kind) using three propositional connectives, *conjunction* $\wedge$, (dynamic) *negation* $\sim$ and *disjunction* $\vee$. Semantic interpretation of these formulas are given below in the framework of a relational semantics.

The following definition describes a well-known notion of a transition model (see, e.g., [van Benthem, 1996]).

Definition. A *transition model* is a triple $M = (S, \Pi \cup P, \mathcal{F})$, where S is a set of states, $\Pi \cup P$ a set of atomic propositions, and $\mathcal{F}$ a valuation function assigning each atomic proposition a binary relation on S, subject to the constraint that, for any fluent $p \in P$, $\mathcal{F}(p) \subseteq Id$, where $Id = \{(s, s) \mid s \in S\}$.

[1]We deviate here from the terminology of [Bochman and Gabbay, 2012].

If $(s,s) \in \mathcal{F}(p)$, we will say that a proposition p *holds in a state* s. More generally, we will say that a proposition A (not necessarily an atomic one) is *static* if its interpretation $\mathcal{F}(A)$ is a subset of Id. Then the above definition stipulates that any fluent atom is a static proposition.

The three connectives of SDL are interpreted in a transition model as follows:

- $s,t \vDash \kappa$ iff $(s,t) \in \mathcal{F}(\kappa)$, for any propositional atom κ;
- $s,t \vDash A \wedge B$ iff there exists r with $s,r \vDash A$ and $r,t \vDash B$;
- $s,t \vDash {\sim}A$ iff $s=t$ and there is no r such that $s,r \vDash A$;
- $s,t \vDash A \vee B$ iff $s,t \vDash A$ or $s,t \vDash B$;

The above definitions extend the valuation function $\mathcal{F}$ to all formulas of the language. The definition for conjunction $\wedge$ amounts to the equality $\mathcal{F}(A \wedge B) = \mathcal{F}(A) \circ \mathcal{F}(B)$, where $\circ$ is a composition of binary relations. This conjunction is non-commutative, but it is an associative connective. Note also that conjunction of static propositions collapses to the usual classical conjunction.

By the above definition, dynamic negation $\sim$ always produces a static proposition. Moreover, when restricted to static propositions, $\sim$ behaves exactly as a classical negation.

More generally, any logical combination of static propositions in this language will be a static proposition. Moreover, restricted to static propositions, the three connectives behave classically. In this sense classical logic can be viewed as a 'static fragment' of sequential dynamic logic.

In what follows by a *fluent proposition* we will mean an arbitrary formula constructed from fluent atoms only. Clearly, any fluent proposition will be static, though not vice versa.

The entailment relation of SDL is based on dynamic inference rules $a \Vdash A$, where A is a formula and a a *sequence* of formulas. An informal meaning of such rules is that a process a causes an event A. Formally, the valuation function $\mathcal{F}$ is canonically extended to *sequences* of propositions:

$$\mathcal{F}(A_1 \dots A_n) \ = \ \mathcal{F}(A_1) \circ \dots \circ \mathcal{F}(A_n).$$

Then the following definition describes when a dynamic inference rule is valid in a transition model.

Definition. A dynamic rule $a \Vdash A$ is *valid* in a transition model M if, for any states $s,t \in S$ such that $(s,t) \in \mathcal{F}(a)$, there exists a state r such that $(t,r) \in \mathcal{F}(A)$. $\Vdash_M$ will denote the set of rules that are valid in a transition model M.

A set of dynamic rules will be said to form a *dynamic consequence relation* if it coincides with $\Vdash_M$, for some transition model M. [Bochman and Gabbay, 2012] provided a sequent calculus that is complete for such dynamic consequence relations.

In what follows, we will use two derived connectives definable in the language of SDL:

- *dynamic implication* $A \to B$, defined as ${\sim}(A \wedge {\sim}B)$.

Dynamic implication satisfies the Deduction Theorem:

$$aA \Vdash B \text{ iff } a \Vdash A \to B.$$

Consequently, it can be used to transform the rules of SDL to propositional formulas. In addition, this connective behaves exactly as a classical implication on static propositions.

- *Inconsistency* $\bot$, defined as ${\sim}A \wedge A$, where A is an arbitrary proposition[2].

Semantically, $\bot$ corresponds to the empty relation $\emptyset$. Consequently, it can be viewed as a static *falsity constant*.

The language of SDL provides a very simple and transparent representation for common action descriptions. As a typical example of representation in this language, if *Shoot* is an action, while *Loaded* and *Dead* are fluents, then an action effect rule like

$$\mathit{Shoot}\ \textbf{causes}\ \mathit{Dead}\ \textbf{if}\ \mathit{Loaded}$$

is representable in SDL simply as a dynamic inference rule

$$\mathit{Loaded}\ \mathit{Shoot} \Vdash \mathit{Dead},$$

which is equivalent to each of the following rules:

$$\mathit{Loaded} \wedge \mathit{Shoot} \Vdash \mathit{Dead},$$

$$\Vdash (\mathit{Loaded} \wedge \mathit{Shoot}) \to \mathit{Dead}$$

2.1 SDL versus PDL

The main difference between the language of SDL and that of Propositional Dynamic Logic (PDL) is that PDL allows, in effect, only static (that is, state-evaluated) propositions as full-fledged propositions of the language. The action terms function in PDL only as modifiers $[\alpha]\phi$ that allow us to construct new static propositions from given ones. Nevertheless, it has been shown in [Bochman and Gabbay, 2012] that SDL is expressively equivalent to PDL. More precisely, there are polynomial translations from each of the languages to another that preserve the respective entailment relations. Thus, the following translation τ simultaneously transforms both formulas and programs of PDL to SDL-formulas.

$$\begin{aligned}
\tau(\kappa) &= \kappa, \text{ for any atom } \kappa \in \Pi \cup P \\
\tau(\phi \,\&\, \psi) &= \tau(\phi) \wedge \tau(\psi) \\
\tau(\neg\phi) &= {\sim}\tau(\phi) \\
\tau([\alpha]\phi) &= \tau(\alpha) \to \tau(\phi) \\
\tau(\alpha;\beta) &= \tau(\alpha) \wedge \tau(\beta) \\
\tau(\alpha \cup \beta) &= \tau(\alpha) \vee \tau(\beta) \\
\tau(?\phi) &= \tau(\phi)
\end{aligned}$$

This translation preserves the inference relation of PDL in the sense that $\phi_1, \dots, \phi_n \vdash \psi$ holds in PDL if and only if $\tau(\phi_1) \dots \tau(\phi_n) \Vdash \tau(\psi)$ holds in SDL.

In fact, purely propositional formulas of PDL correspond exactly to fluent formulas of SDL, while arbitrary PDL formulas correspond to static formulas of SDL. The latter correspondence can be described more precisely as follows. Recall that PDL formulas are constructed from fluent atoms using the classical connectives and the modal construct $[\pi]F$, which corresponds to the implication $\pi \to F$ in the SDL notation. Accordingly, let us say that a static SDL formula is of *PDL-type* if it is constructed recursively from fluent atoms using the propositional connectives and implications of the form $\pi \to \phi$, where π is an action atom, and ϕ a PDL-type formula. Then it can be shown that any static formula of SDL can be transformed into an equivalent formula of PDL-type. We will use this correspondence in what follows.

[2]Note that the order is important here, since $A \wedge {\sim}A$ may well be consistent.

3 Markov Assumption in Action Domains

It is widely held (especially in the AI literature) that a proper description of dynamic domains should conform to the following *Markov principle*: in every state of a dynamic system, both an executability and particular effects of every action should depend only on fluent propositions that hold in this state. Accordingly, a complete fluent information about a particular state should exhaustively determine all possible future developments from this state. Note that this principle does not presuppose determinism, so it is compatible with existence of nondeterministic actions. Still, it states that the range of possible outcomes of such an action in every state is fully determined by the fluents that hold in this state.

It is important to note that the Markov principle is not an intrinsic, 'ontological' property of action domains. Rather, it should be viewed as a general *knowledge representation principle*, or assumption, according to which a dynamic system ought to be described in a way that conforms to this principle. Thus, if an actual description of a dynamic system does not satisfy the Markov principle, this could be viewed as an evidence that this description simply lacks expressivity, since the fluent information it can articulate is insufficient for distinguishing states with different possibilities of actions.

3.1 Semantics: Transparent Models

Arbitrary transition models (that constitute the semantics of our language) do not satisfy, in general, the Markov property. Indeed, in such models two states satisfying the same fluent facts may occupy different places in a relational structure, and hence have different dynamic properties such as executability of actions as well as their effects. This immediately suggests, however, that a natural way of imposing the Markov principle consists in restricting transition models to models in which different states are distinguishable by their 'fluent content':

Definition.
- A transition model will be called *transparent* if for any two distinct states $s, t \in S$ there is a fluent formula F that holds in s, but does not hold in t.
- A dynamic consequence relation will be called *transparent* if it is determined by some transparent transition model.

Any state of a transparent model is uniquely determined by the set of fluent formulas that hold in it. Accordingly, transparent models can be alternatively described as quadruples $(I, P, \Pi, \mathcal{F})$, where I is a set of *propositional valuations* assigning a truth value to each fluent atom from P, while $\mathcal{F}$ assigns each action a binary relation on such valuations.

As a matter of fact, transparent models have been used explicitly or implicitly in overwhelming majority of action theories developed in AI. Thus, it is a standard model for action description languages [Gelfond and Lifschitz, 1998], logic programming (see, e.g., [Baral and Gelfond, 2005]), causal theories of action [Giunchiglia *et al.*, 2004], and even some representations of actions in the PDL framework (see, e.g., [Herzig and Varzinczak, 2007]). And though this fact is less obvious for first-order representations of actions like the situation calculus, we will see in what follows that such action theories directly embody the Markov principle.

It should be noted, however, that transparency is not a simple logical constraint on dynamic consequence relations. In particular, it cannot be expressed in terms of additional axioms or inference rules of a usual kind. This follows already from the fact that any transition model can be conservatively transformed into a transparent model simply by adding new propositional fluents that distinguish the states of the model (in the extended language). Nevertheless, restriction to transparent models may well sanction additional conclusions that are not derivable in the underlying logical formalism. The following example illustrates this.

Example 1. Let us consider a language that contains a single fluent p and single action π. Any transparent model for this language has at most two states (= propositional valuations), and it can be verified that in any such model, if there is a trace $\pi p \pi$ from some state of a model, then performing π in this state leads to a state in which π will always be executable whenever p holds in it. Formally, this description corresponds to the following rule of SDL:

$$\sim\sim(\pi \wedge p \wedge \pi)\pi p \Vdash \pi$$

Note, however, that this rule is not valid in SDL, and it is refuted in a non-transparent model with three states s, t, t_1 such that p holds in t and t_1 only, and $\mathcal{F}(\pi) = \{(s,t), (s,t_1), (t,t_1)\}$.

3.2 Basic Action Theories and Markov Completion

A key to a better understanding of the Markov principle is provided by the following restriction on dynamic rules:

Definition. A dynamic rule will be called *basic* if it has the form $F\pi \Vdash F_1$, or the form $\Vdash F$, where π is an action atom and F, F_1 are fluent formulas. A set of basic rules will be called a *basic action theory*.

Rules of the form $F\pi \Vdash F_1$ (or formulas $F \to [\pi]F_1$ in the PDL notation) correspond to action descriptions used in practically all action theories in AI. Such rules correspond also to the so-called *Hoare rules* $\{F\}\pi\{F_1\}$ in the theory of computation (see [Harel *et al.*, 2000]). Note that these rules include *inexecutability conditions* $F\pi \Vdash \bot$ as a special case.

Rules of the form $\Vdash F$ correspond to the usual *static laws* or constraints. Note that in SDL a rule $F \Vdash F_1$ is equivalent to $\Vdash F \to F_1$, so such rules are reducible to basic static laws.

It should be noted that static laws in a basic action theory are completely independent of the action rules: any static law that is derivable from a basic action theory is derivable already from its static laws alone[3]. The situation is not symmetric, however, for action laws, since static laws can be used to derive new action laws from given ones.

Now, the following key result shows that transparent dynamic consequence relations are uniquely determined by their basic rules.

Theorem 1. *Two transparent dynamic consequence relations coincide if and only if they have the same basic rules.*

[3]This will no longer hold if we would allow non-basic rules - see, e.g., [Herzig and Varzinczak, 2007].

The above theorem states that all the inferences sanctioned by a transparent consequence relation are determined ultimately by the basic rules that belong to it. In other words, if we accept the Markov principle, then a complete description of a dynamic model can always be achieved by using only basic rules. This formal result can be viewed as an ultimate logical justification for restricting action descriptions in AI domains to basic action theories.

It is interesting to note also that for deterministic action theories, the above result can be strengthened even further.

Definition. A transition model $(S, \Pi \cup P, \mathcal{F})$ is called *deterministic* if, for any action atom π, $\mathcal{F}(\pi)$ is a partial function: if $(s,t) \in \mathcal{F}(\pi)$ and $(s,r) \in \mathcal{F}(\pi)$, then $t = r$.

We will say that a dynamic consequence relation is deterministic, if it is generated by a deterministic transition model. Then the following result can be obtained:

Theorem 2. *Any transparent deterministic consequence relation is determined by the set of its static laws and the set of definite action rules of the form $L\,\pi \Vdash l$, where l is a fluent literal and L a conjunction of fluent literals.*

The above result shows that, for describing deterministic action domains, we can safely use only definite action rules of the form $L\,\pi \Vdash l$.

Actually, the correspondence between transparent dynamic consequence relations and basic action theories is bidirectional in the sense that any basic action theory determines a unique transparent consequence relation associated with it. This consequence relation can be constructed as follows.

Let Δ be a basic action theory. Then its *canonical transparent transition model* $M^t_\Delta = (S_\Delta, P, \Pi, \mathcal{F}_\Delta)$ is defined by taking S_Δ to be the set of all maximal sets of fluent formulas that are consistent with respect to (the static laws of) Δ and stipulating that, for any action atom π, $\mathcal{F}(\pi)$ is the set of all pairs (S,T) such that, for any basic rule $F\pi \Vdash F_1$ from Δ, if $F \in S$, then $F_1 \in T$.

Lemma. *M^t_Δ is a transparent model of Δ.*

We will denote by $\Vdash^t_\Delta$ the dynamic consequence relation that is determined by the above canonical model M^t_Δ.

Now, let $\Vdash_\Delta$ denote the least dynamic consequence relation including Δ. Clearly, $\Vdash_\Delta$ is the set of all rules that are logically derivable from Δ in SDL, and consequently $\Vdash_\Delta$ is included in $\Vdash^t_\Delta$. However, in general these two consequence relations *do not coincide*. Nevertheless, we have

Theorem 3. *$\Vdash_\Delta$ has the same basic rules as $\Vdash^t_\Delta$.*

Moreover, since any transparent consequence relation is uniquely determined by its basic rules, we obtain

Corollary. *If Δ is a basic action theory, then $\Vdash^t_\Delta$ is a unique transparent consequence relation that has the same basic rules as $\Vdash_\Delta$.*

The above results and correspondences allow us to be more precise about the exact impact of restricting the semantics of dynamic consequence relations to transparent models.

$\Vdash^t_\Delta$ contains in general more information than what is logically derivable from Δ. On the other hand, both consequence relations contain the same basic rules, so the additional information in $\Vdash^t_\Delta$ (as compared with $\Vdash_\Delta$) involves only non-basic rules. Furthermore, all these additional rules are determined ultimately only by the basic rules of Δ.

The above properties allow us to see $\Vdash^t_\Delta$ as a kind of *completion* of Δ that is based on the Markov assumption. Moreover, just as other known kinds of completions, this Markov completion is *nonmonotonic* in the sense that if Δ is extended with additional rules to another (even basic) action theory Δ', it does not necessarily hold that $\Vdash^t_\Delta$ is included in $\Vdash^t_{\Delta'}$. The following example illustrates this.

Example 2. Consider a basic action theory $\Delta = \{\pi \Vdash p\}$ in a language containing only a fluent atom p and an action π. The corresponding canonical transparent model M^t_Δ involves only two worlds (valuations) $s = \{\sim p\}$ and $t = \{p\}$, whereas $\mathcal{F}(\pi) = \{(s,t),(t,t)\}$. In this model, action π can always be executed with an effect p (that is, $\Vdash^t_\Delta \pi \wedge p$ holds). This assertion does not follow, however, logically from Δ in SDL. Moreover, if we extend Δ to a basic theory $\Delta' = \{\pi \Vdash p,\ p\,\pi \Vdash \sim p\}$, we obtain a different canonical model in which $\mathcal{F}(\pi) = \{(s,t)\}$. In this model π is already not executable in t, and consequently $\Vdash \pi \wedge p$ no longer holds.

The executability assumption. One of the most significant descriptions that *cannot* be formulated in the restricted language of basic action rules are executability claims stating that, given some conditions, a certain action is executable. Such descriptions correspond to rules of the form $F_1 \Vdash \pi \wedge F$ in SDL (and to formulas $F_1 \to \langle\pi\rangle F$ in PDL). Moreover, the above example illustrates that a canonical transparent model actually maximizes such executability claims. In this sense it embodies a well-known *executability assumption* in the action literature according to which an action is always considered executable whenever this is consistent with other facts and action rules. Accordingly, the executability assumption can be viewed as a by-product of the Markov principle.

3.3 Admissible Queries

The above results create a peculiar ambiguity about the semantic interpretation and consequences of basic action theories. On the one hand, a basic theory Δ is interpreted 'logically' by the consequence relation $\Vdash_\Delta$ - the logical closure of Δ in SDL. On the other hand, the Markov principle suggests that Δ should be interpreted by its canonical transparent model, which corresponds to a larger consequence relation $\Vdash^t_\Delta$. Still, Theorem 3 shows that the difference is inessential for basic rules. This naturally raises a question what kinds of queries are invariant with respect to these two interpretations? In order to answer this question, we have to consider a broader class of dynamic inference rules.

By *a generalized basic rule* we will mean a dynamic rule of the form

$$F_0\,\pi_1\,F_1 \ldots \pi_n\,F_n \Vdash F,$$

where F and each F_i is a fluent formula, while π_i are action atoms. Then the following result can be proved as a generalization of Theorem 3:

Theorem 4. *$\Vdash_\Delta$ has the same generalized basic rules as $\Vdash^t_\Delta$.*

By the above result, derivability of generalized basic rules does not depend on the choice between the two interpretations, so each of them can be used for their verification.

As a matter of fact, generalized basic rules subsume practically all kinds of queries that are used in action theories. Actually, most of these theories use only a special kind of such rules, namely rules of the form $F_0\pi_1 \dots \pi_n \Vdash F_1$. However, though this restriction is reasonable for deterministic action domains, it is obviously insufficient for capturing properties of non-deterministic actions. This is because in the non-deterministic case there may be multiple ways of executing a sequence of actions $\pi_1 \dots \pi_n$, and they can be distinguished by putting fluent 'tests' between these actions.

4 Inferential Markov Property

Now we are going to describe a syntactic, inferential counterpart of the Markov principle.

Definition. A dynamic consequence relation will be said to satisfy the *Markov property* if, for any propositions A, B, if $A \Vdash B$ holds, then there is a fluent proposition F such that $A \Vdash F$ and $F \Vdash B$.

According to the above description, the Markov property is a particular kind of interpolation property requiring that all inferences of a dynamic consequence relation are mediated by fluent propositions. We will denote by $\mathrm{Cf}(A)$ the set of fluent consequences of a proposition A:

$$\mathrm{Cf}(A) = \{F \mid F \text{ is a fluent proposition } \& \ A \Vdash F\}.$$

The next result summarizes a number of important conditions that are equivalent to the above Markov property:

Theorem 5. *A dynamic consequence relation has the Markov property if and only if it satisfies one of the following conditions:*

1. *If S is a maximal consistent set of fluents, and ϕ a static proposition, then either $S \Vdash \phi$, or $S \Vdash \sim\phi$;*
2. *$\mathrm{Cf}(\phi) \Vdash \phi$, for any static proposition ϕ;*
3. *$\mathrm{Cf}(\pi \to F) \Vdash \pi \to F$, for any atomic action π and any fluent formula F.*

Condition 2 says, in effect, that for any static proposition ϕ there is an equivalent fluent proposition F, that is $\Vdash \phi \leftrightarrow F$. Similarly, Condition 3 asserts an existence of a fluent equivalent for any implication $\pi \to F$.

Remark. In the language of PDL, Condition 2 amounts to the requirement that for any PDL formula there exists an equivalent propositional formula. Similarly, Condition 3 requires existence of propositional equivalents for any modal formula of the form $[\pi]F$.

Turning to semantic descriptions, we have

Theorem 6. *If a dynamic consequence relation satisfies the Markov property, then it is transparent.*

Finally, the next result shows that in the finite case any transparent model determines a Markov consequence relation.

Theorem 7. *If M is a transparent transition model in a finite language, then $\Vdash_M$ satisfies the Markov property.*

Thus, in the finite case there is a one-to-one correspondence between the syntactic and semantic descriptions of the Markov principle. Accordingly, we obtain the following

Corollary. *A finite dynamic consequence relation is transparent if and only if it has the Markov property.*

5 Markov Property and Regression

Now we are going to show that Markov property implies the well-known regression principle for action domains.

Condition 3 of Theorem 5 says, in effect, that for any action atom π and any fluent formula F there is a fluent formula F_0 that is equivalent to $\pi \to F$. This implies, in particular, that for any action π there is a fluent formula F_π that is equivalent to $\pi \to \bot$. The formula F_π determines precise conditions for inexecutability of action π, so $\sim F_\pi$ provides us with exact fluent preconditions for *executability* of π[4], while the conjunction $F_0 \wedge \sim F_\pi$ determines the conditions under which an action π is executable and produces effect F.

The equivalence of conditions (2) and (3) of Theorem 5 implies that existence of fluent equivalents for any implication $\pi \to F$ implies existence of such fluent equivalents for any static formula. In SDL, such a fluent equivalent of an arbitrary static formula can be obtained as follows.

Recall that any static formula of SDL can be transformed into an equivalent formula of PDL-type. Now, a fluent equivalent of any PDL-type formula ϕ can be found using the following simple *regression method*:

- *Recursively replace any sub-formula of ϕ of the form $\pi \to F$ with a corresponding fluent equivalent.*

The above algorithm always terminates and produces a fluent formula that is equivalent to ϕ. This regression algorithm can be seen as an abstract generalization of the regression method in the situation calculus (see [Reiter, 1991; 2001]). Moreover, it extends the latter to general non-deterministic domains that satisfy the Markov property.

The above regression algorithm can be used for verifying arbitrary queries concerning a Markov consequence relation. To see this, note that a question whether a rule $A \Vdash B$ holds for a Markov consequence relation is reducible to the question whether the corresponding static proposition $A \to B$ is derivable which, by the above algorithm, is reducible to a validity of a certain (classical) fluent formula.

If we want, however, to transform the above regression algorithm into a working procedure for verifying inferences from action theories, we need a method of producing fluent equivalents for the 'effect' implications $\pi \to F$ from the actual domain descriptions given in the form of basic action theories. In the finite case, this can be done as follows.

For a finite language, the canonical transparent model M^t_Δ of a basic action theory Δ is also finite, and each state s of the model is uniquely determined by the conjunction F_s of the fluent literals that hold in this state. Then it can be verified that the following formula gives a fluent equivalent of $\pi \to F$:

$$G^\pi_F = \bigvee \{F_s \mid F \in t, \text{ for any } t \text{ such that } s\,\pi\,t\}.$$

A proof-theoretic counterpart of the above semantic procedure is as follows. Let $\Delta = \Delta_s \cup \Delta_a$ be a finite basic action

[4]Thus, it corresponds to a fluent equivalent of the predicate $Poss(\pi)$ in Reiter's situation calculus.

theory, where Δ_s are static laws, while Δ_a are action laws. $\wedge(\Delta_a)$ will denote the set of all rules of the form

$$(C_1 \wedge \cdots \wedge C_n)\,\pi \Vdash D_1 \wedge \cdots \wedge D_n$$

such that $C_i\,\pi \Vdash D_i$ belongs to Δ_a for every $i \leq n$. Clearly, any such rule is a logical consequence of Δ. Now, a fluent equivalent of $\pi \to F$ can be constructed as follows:

$$G^{\pi}_{F} = \bigvee \{C \mid C\pi \Vdash D \in \wedge(\Delta_a) \;\&\; D, \Delta_s \vDash F\}.$$

Lemma. *G^{π}_{F} is a fluent equivalent of $\pi \to F$ wrt Δ.*

Actually, the task of constructing fluent equivalents for implications $\pi \to F$ is partly simplified by the fact that $\pi \to (F_1 \wedge F_2)$ is equivalent to $(\pi \to F_1) \wedge (\pi \to F_2)$, so if G_1 and G_2 are fluent equivalents of $\pi \to F_1$ and $\pi \to F_2$, respectively, then $G_1 \wedge G_2$ is a fluent equivalent of $\pi \to F_1 \wedge F_2$. Consequently, it is sufficient to find fluent equivalents for all implications of the form $\pi \to C$, where C is a fluent clause.

A further simplification can be obtained for *deterministic* consequence relations. As has been established earlier, action rules for such consequence relations can be restricted to fluent literals. Moreover, for such consequence relations we have that $\pi \to (F_1 \vee F_2)$ is equivalent to $(\pi \to F_1) \vee (\pi \to F_2)$, so the task reduces to finding fluent equivalents for implications $\pi \to l$, where l is a fluent literal. In addition, for such consequence relations it holds also that

$$\pi \to \sim F \text{ is equivalent to } (\pi \to \bot) \vee \sim(\pi \to F).$$

Consequently the task of determining fluent equivalents for any static proposition is reduced in this case to finding fluent equivalents for implications $\pi \to p$, where p is either a fluent atom, or a falsity constant.

Finally, it turns out that the above constructions are directly related to Reiter's *basic action theories* (BATs) in the situation calculus (see [Reiter, 2001]), defined in terms of static laws, action precondition axioms and successor state axioms. Indeed, *action precondition* axioms are precisely assertions that determine fluent equivalents for implications $\pi \to \bot$, while the *successor state* axioms are nothing other than descriptions of fluent equivalents for implications $\pi \to p$, for each action π and every fluent atom p. Thus, as was rightly noted in [Lin, 2008], Reiter's basic action theories provide in this respect a precise realization of the Markov assumption in the setting of deterministic action theories.

6 Summary and Perspectives

It has been shown in this study that many of the actual restrictions and specific properties of action theories in AI can be obtained as consequences of the general Markov principle for action domains. Moreover, we have shown that the Markov principle constitutes a *nonmonotonic assumption* that allows us to 'complete' information determined by the basic action rules and static laws. This assumption embodies, in particular, the executability assumption, and provides an ultimate logical basis for the regression method.

All the above constructions and results did not take into account, however, the famous frame problem in reasoning about actions. This omission was intentional, since the main aim of this study amounted to showing that many important features and methods of current action theories stem ultimately from basic knowledge representation principles that are independent of, or complementary to, the frame problem. Still, an account of the frame problem is essential for an adequate description of action theories in AI.

Generally speaking, the frame problem is a by-product of an additional, and also highly reasonable *inertia assumption*, according to which the world changes only as a result of actions, while each particular action directly affects only a relatively small number of fluents. Hence each time a large set of fluent literals should retain their value in such a change. Now, in a descriptive setting of action theories, this means that any action description should involve (explicitly or implicitly) a large number of inertia, or frame, rules of the form $l\,\pi \Vdash l$, where l is a fluent literal. Accordingly, the frame problem amounts to the problem how such inertia rules (which are a special kind of basic rules) can be efficiently represented and handled.

Markov and inertia assumptions are completely independent knowledge representation principles, but they jointly determine the ultimate form and associated logical formalisms of action theories in AI. In some simple cases (specifically, in the deterministic setting without static laws), the impact of the inertia assumption can be directly 'compiled' into a fluent equivalent of an implication $\pi \to p$. In fact, this is the essence of Reiter's 'simple' solution to the frame problem[5]. Unfortunately, the task becomes much more complex in the presence of static laws, mainly because it immediately invokes both the qualification and ramification problems. These problems have been discussed in [Ginsberg and Smith, 1988; Lin and Reiter, 1994] and in many subsequent studies. A predominant way of resolving these problems that emerged from these studies amounted to augmenting domain descriptions with directional static *causal* laws for fluents, combined with an appropriate nonmonotonic reasoning mechanism for dealing with such laws. Such a combined description has been implemented, for instance, in causal theories of actions (see [Giunchiglia *et al.*, 2004]).

The formalism of SDL, served as a logical basis of this study, naturally suggests, however, a more uniform approach to the problem of incorporating the inertia assumption. Note first that SDL can be viewed as a direct dynamic generalization of the (static) classical propositional calculus, a generalization that is adequate for a logical description of dynamic domains. On the other hand, the inertia assumption can also be viewed as a natural dynamic counterpart of the well-known (static) Closed World Assumption that constitutes the basis of logic programming. This immediately suggests that a prospective generalization of SDL that would incorporate the inertia assumption should take the form of a dynamic generalization of the current (static) Logic Programming. Such a generalization, if feasible, would provide a uniform knowledge representation formalism for reasoning and computation in action domains. This is a subject of an ongoing study.

[5]The corresponding regression principle in PDL has been described in [Demolombe *et al.*, 2003].

References

[Baral and Gelfond, 2005] C. Baral and M. Gelfond. Logic programming and reasoning about actions. In D. Gabbay M. Fisher and L. Vila, editors, *Handbook of Temporal Reasoning in Artificial Intelligence*, volume 1 of *Foundations of Artificial Intelligence*, pages 389 – 426. Elsevier, 2005.

[Bochman and Gabbay, 2012] A. Bochman and D. Gabbay. Sequential dynamic logic. *Journal of Logic, Language and Information*, 21:279–298, 2012.

[Demolombe *et al.*, 2003] R. Demolombe, A. Herzig, and I. J. Varzinczak. Regression in modal logic. *Journal of Applied Non-Classical Logics*, 13(2):165–185, 2003.

[Gelfond and Lifschitz, 1998] M. Gelfond and V. Lifschitz. Action languages. *Electronic Transactions on Artificial Intelligence*, 3:195–210, 1998.

[Ginsberg and Smith, 1988] M. L. Ginsberg and D. E. Smith. Reasoning about action II: The qualification problem. *Artificial Intelligence*, 35(3):311–342, 1988.

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

[Harel *et al.*, 2000] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.

[Herzig and Varzinczak, 2007] A. Herzig and I. Varzinczak. Metatheory of actions: Beyond consistency. *Artificial Intelligence*, 171(16-17):951 – 984, 2007.

[Lin and Reiter, 1994] F. Lin and R. Reiter. State constraints revisited. *J. Log. Comput.*, 4(5):655–678, 1994.

[Lin, 2008] F. Lin. Situation calculus. In F. van Harmelen, V. Lifschitz, and B. Porter, editors, *Handbook of Knowledge Representation*, pages 649–669. Elsevier, 2008.

[Reiter, 1991] R. Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In V. Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 318–420. Academic Press, 1991.

[Reiter, 2001] R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamic Systems*. MIT Press, 2001.

[Thielscher, 2011] M. Thielscher. A unifying action calculus. *Artificial Intelligence*, 175(1):120 – 141, 2011.

[van Benthem, 1996] J. van Benthem. *Exploring Logical Dynamics*. CSLI Publ., 1996.