Elaborating domain descriptions
(preliminary report)

Andreas Herzig Laurent Perrussel Ivan Varzinczak

IRIT – 118 route de Narbonne
31062 Toulouse Cedex – France

e-mail: {herzig,perrusse,ivan}@irit.fr
http://www.irit.fr/recherches/LILAC

Abstract

In this work we address the problem of *elaborating* domain descriptions (alias action theories), in particular those that are expressed in dynamic logic. We define a general method based on contraction of formulas in a version of propositional dynamic logic with an incorporated solution to the frame problem. We present the semantics of our theory change and define syntactical operators for contracting a domain description. We establish soundness and completeness of the operators w.r.t. the semantics for descriptions that satisfy a principle of modularity that we have proposed elsewhere. We also investigate an example of changing non-modular domain descriptions.

Keywords: Theory change, dynamic logic, modularity.

1 Introduction

Suppose a situation where an agent has always believed that if the light switch is up, then there is light in the room. Suppose now that someday, she observes that even if the switch is in the upper position, the light is off. In such a case, the agent must change her beliefs about the relation between the propositions “the switch is up” and “the light is on”. This example is an instance of the problem of changing propositional belief bases and is largely addressed in the literature about belief change [11] and belief update [25].

Next, let our agent believe that whenever the switch is down, after toggling it, there is light in the room. This means that if the light is off, in every state of the world that follows the execution of toggling the switch, the room is lit up. Then, during a blackout, the agent toggles the switch and surprisingly the room is still dark.

Imagine now that the agent never worried about the relation between toggling the switch and the material it is made of, in the sense that she ever believed that just toggling the switch does not break it. Nevertheless, in a stressful day, she toggles the switch and then observes that she had broken it.

Completing the wayside cross our agent experiments in discovering the world’s behavior, suppose she has believed that it is always possible to toggle the switch, provided some conditions like being close enough to it, having a free hand, the switch is not broken, etc, are satisfied. However, in a beautiful April fool’s day, the agent discovers that someone has glued the switch and, consequently, it is no longer possible to toggle it.

The last three examples illustrate situations where changing the beliefs about the behavior of the action of toggling the switch is mandatory. In the first one, toggling the switch, once believed to be deterministic, has now to be seen as nondeterministic, or alternatively to have a different outcome in a specific context (e.g. if the power station is overloaded). In the second example, toggling the switch is known to have side-effects (ramifications) one was not aware of. In the last example, the executability of the action under concern is questioned in the light of new information showing a context that was not known to preclude its execution. Carrying out modifications is what we here call *elaborating* a domain description, which has to do with the principle of *elaboration tolerance* [29].

Such cases of theory change are very important when one deals with logical descriptions of dynamic domains: it may always happen that one discovers that an action actually has a behavior that is different from that one has always believed it had.

Up to now, theory change has been studied mainly for knowledge bases in classical logics, both in terms of revision and update. Only in a few recent works it has been considered in the realm of modal logics, viz. in epistemic logic [13], and in action languages [8]. Recently, several works [35, 21] have investigated revision of beliefs about facts of the world. In our examples, this would concern e.g. the current status of the switch: the agent believes it is up, but is wrong about this and might subsequently be forced to revise his beliefs about the current state of affairs. Such belief revision operations do not modify the agent's beliefs about the action laws. In opposition to that, here we are interested exactly in such modifications. The aim of this work is to make a step toward that issue and propose a framework that deals with the contraction of action theories.

Dynamic logic, more specifically propositional dynamic logic (PDL [14]), has been extensively used in reasoning about actions in the last years [3, 4, 40, 9, 39]. It has shown to be a viable alternative to situation calculus approaches because of its simplicity and existence of proof procedures for it. In this work we investigate the elaboration of domain descriptions encoded in such a logical formalism. We show how a theory expressed in terms of static laws, effect laws and executability laws is elaborated: usually, a law has to be changed due to its generality, i.e., the law is too strong and has to be weakened. It follows that elaborating an action theory means contracting it by static, effect or executability laws, before expanding the theory with more specific laws.

The present text is organized as follows: in Section 2 we define the logical framework we use throughout this work and show how action theories are encoded. Section 3 is devoted to our semantics of theory change and Section 4 to its syntactical counterpart. In Section 5 we establish soundness and completeness of our change operators w.r.t. the semantics, where completeness is conditioned by a notion of modularity that we have proposed in previous work. We then analyse an example of correcting a non-modular theory (Section 6). Before concluding, we address related work on the field (Section 7) and discuss on how elaboration tolerant the framework here proposed is (Section 8).

2 Background

Following the tradition in the reasoning about actions community, action theories are going to be collections of statements that have the particular form: "if *context*, then *effect* after *every execution* of *action*" (effect laws); and "if *precondition*, then *action executable*" (executability laws). Statements mentioning no action at all represent laws about the world (static laws). Besides that, statements of the form "if *context*, then *effect* after *some execution* of *action*" will be used as a causal notion to solve the frame and the ramification problems.

2.1 Logical preliminaries

Let $\mathfrak{Act} = \{a_1, a_2, \ldots\}$ be the set of all *atomic action constants* of a given domain. An example of atomic action is *toggle*. To each atomic action a there is associated a modal operator $[a]$.

$\mathfrak{Prop} = \{p_1, p_2, \ldots\}$ denotes the set of all *propositional constants*, also called *fluents* or *atoms*. Examples of those are *light* ("the light is on") and *up* ("the switch is up"). The set of all literals is $\mathfrak{Lit} = \{l_1, l_2, \ldots\}$, where each l_i is either p or $\neg p$, for some $p \in \mathfrak{Prop}$. If $l = \neg p$, then we identify $\neg l$ with p.

We use small Greek letters $\varphi, \psi, \ldots$ to denote *classical formulas*. They are recursively defined in the usual way:

$$\varphi \ ::= p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \rightarrow \psi \mid \varphi \leftrightarrow \psi$$

$\mathfrak{Fml}$ is the set of all classical formulas. An example of a classical formula is $up \rightarrow light$. By $val(\varphi)$ we denote the set of valuations making φ true. We view a valuation as a maximally-consistent set of literals. For $\mathfrak{Prop} = \{light, up\}$, there are four valuations: $\{light, up\}$, $\{light, \neg up\}$, $\{\neg light, up\}$ and $\{\neg light, \neg up\}$. Given a set of formulas Σ, by $lit(\Sigma)$ we denote the set of all literals appearing in formulas of Σ.

We denote complex formulas (with modal operators) by $\Phi, \Psi, \ldots$ They are recursively defined in the following way:

$$\Phi \ ::= \varphi \mid [a]\Phi \mid \neg\Phi \mid \Phi \wedge \Psi \mid \Phi \vee \Psi \mid \Phi \rightarrow \Psi \mid \Phi \leftrightarrow \Psi$$

$\langle a \rangle$ is the dual operator of $[a]$, defined as $\langle a \rangle\Phi =_{\text{def}} \neg[a]\neg\Phi$. An example of a complex formula is $\neg up \rightarrow [toggle]up$.

The semantics is that of multimodal logic K [31].

Definition 2.1 A PDL-*model* is a tuple $\mathscr{M} = \langle W, R \rangle$ where W is a set of maximally-consistent sets of literals, and R maps action constants a to accessibility relations $R_a \subseteq W \times W$.

Definition 2.2 Given a PDL-model $\mathscr{M} = \langle W, R \rangle$,

- $\models^{\mathscr{M}}_{w} p$ (p is true at world w of model $\mathscr{M}$) if $p \in w$;
- $\models^{\mathscr{M}}_{w} [a]\Phi$ if for every w' such that wR_aw', $\models^{\mathscr{M}}_{w'} \Phi$;

- truth conditions for the other connectives are as usual.

Definition 2.3 $\mathscr{M}$ is a model of Φ (noted $\models^{\mathscr{M}} \Phi$) if and only if for all $w \in W$, $\models^{\mathscr{M}}_{w} \Phi$. $\mathscr{M}$ is a model of a set of formulas Σ (noted $\models^{\mathscr{M}} \Sigma$) if and only if $\models^{\mathscr{M}} \Phi$ for every $\Phi \in \Sigma$. A formula Φ is a *consequence of the set of global axioms* $\{\Phi_1, \ldots, \Phi_n\}$ in the class of all PDL-models (noted $\Phi_1, \ldots, \Phi_n \models_{\mathsf{PDL}} \Phi$) if and only if for every PDL-model $\mathscr{M}$, if $\models^{\mathscr{M}} \Phi_i$ for every Φ_i, then $\models^{\mathscr{M}} \Phi$.

2.2 Describing the behavior of actions in PDL

Given a domain, we are interested in theories whose statements describe the behavior of actions. PDL allows for the representation of such statements, that we call *action laws.* Here we distinguish several types of them. The first kind of statement represents the *static laws*, which are formulas that must hold in every possible state of the world.

Definition 2.4 A *static law* is a formula $\varphi \in \mathfrak{Fml}$.

An example of a static law is $up \rightarrow light$, saying that if the switch is up, then the light is on. The set of all static laws of a domain is denoted by $\mathcal{S} \subseteq \mathfrak{Fml}$.

The second kind of action law we consider is given by the *effect laws.* These are formulas relating an action to its effects, which can be conditional.

Definition 2.5 An *effect law for action a* is of the form $\varphi \rightarrow [a]\psi$, where $\varphi, \psi \in \mathfrak{Fml}$, with φ classically consistent.

'Classically consistent' is a shorthand for 'consistent in classical propositional logic'. The consequent ψ is the effect which always obtains when action a is executed in a state where the antecedent φ holds. If a is a nondeterministic action, then the consequent ψ is typically a disjunction. The set of effect laws of a domain is denoted by $\mathcal{E}$. An example of an effect law is $\neg up \rightarrow [toggle]light$, saying that whenever the switch is down, after toggling it, the room is lit up. If ψ is inconsistent, we have a special kind of effect law that we call an *inexecutability law.* For example, $broken \rightarrow [toggle]\bot$ expresses that *toggle* cannot be executed if the switch is broken.

Finally, we also define *executability laws*, which stipulate the context where an action is guaranteed to be executable. In PDL, the operator $\langle a \rangle$ is used to express executability. $\langle a \rangle\top$ thus reads "the execution of a is possible".

Definition 2.6 An *executability law for action a* is of the form $\varphi \rightarrow \langle a \rangle\top$, where $\varphi \in \mathfrak{Fml}$ is classically consistent.

For instance, $\neg \mathit{broken} \rightarrow \langle \mathit{toggle} \rangle \top$ says that toggling can be executed whenever the switch is not broken. The set of all executability laws of a given domain is denoted by $\mathcal{X}$.

The rest of this work is devoted to the elaboration of action models and theories.

3 Models of contraction

When an action theory has to be changed, the basic operation is that of *contraction*. (In belief-base update [37, 25] it has also been called *erasure*.) In this section we define its semantics.

In general we might contract by any formula Φ. Here we focus on contraction by one of the three kinds of laws. We therefore suppose that Φ is either φ, where φ is classical, or $\varphi \rightarrow [a]\psi$, or $\varphi \rightarrow \langle a \rangle \top$.

For the case of contracting static laws we resort to existing approaches in order to change the set of static laws. In the following, we consider any belief change operator such as Forbus' update method [10], or the possible models approach [37, 38], or WSS [15] or MPMA [7].

Contraction by φ corresponds to adding new possible worlds to W. Let $\ominus$ be a contraction operator for classical logic.

Definition 3.1 Let $\langle W, R \rangle$ be a PDL-model and φ a classical formula. The set of models resulting from contracting by φ is the singleton $\langle W, R \rangle^{-}_{\varphi} = \{\langle W', R \rangle\}$ such that $W' = W \ominus \mathit{val}(\varphi)$.

Action theories being defined in terms of effect and executability laws, elaborating an action theory will mainly involve changing one of these two sets of laws. Let us consider now both these cases.

Suppose the knowledge engineer acquires new information regarding the effect of action a. Then it means that the law under consideration is probably too strong, i.e., the expected effect may not occur and thus the law has to be weakened. Consider e.g. $\neg \mathit{up} \rightarrow [\mathit{toggle}]\mathit{light}$, and suppose it has to be weakened to the more specific $(\neg \mathit{up} \wedge \neg \mathit{blackout}) \rightarrow [\mathit{toggle}]\mathit{light}$.[1] In order to carry out such a weakening, first the designer has to contract the set of effect laws and second to expand the resulting set with the weakened law.

[1] The other possibility of weakening the law, i.e., replacing it by $\neg \mathit{up} \rightarrow [\mathit{toggle}](\mathit{light} \vee \neg \mathit{light})$ looks silly. We were not able to find examples where changing the consequent could give a more intuitive result. In this sense, we prefer to always weaken a given law by strengthening its antecedent.

Contraction by $\varphi \to [a]\psi$ amounts to adding some 'counterexample' arrows from φ-worlds to $\neg\psi$-worlds. To ease such a task, we need a definition. Let $PI(\varphi)$ denote the set of prime implicates of φ.

Definition 3.2 Let $\varphi_1, \varphi_2 \in \mathfrak{Fml}$. $NewCons_{\varphi_1}(\varphi_2) = PI(\varphi_1 \wedge \varphi_2) \setminus PI(\varphi_1)$ computes the *new consequences* of φ_2 w.r.t. φ_1: the set of strongest clauses that follow from $\varphi_1 \wedge \varphi_2$, but do not follow from φ_1 alone (cf. e.g. [20]).

For example, the set of prime implicates of p_1 is just $\{p_1\}$, that of the formula $p_1 \wedge (\neg p_1 \vee p_2) \wedge (\neg p_1 \vee p_3 \vee p_4)$ is $\{p_1, p_2, p_3 \vee p_4\}$, hence we have that $NewCons_{p_1}((\neg p_1 \vee p_2) \wedge (\neg p_1 \vee p_3 \vee p_4)) = \{p_2, p_3 \vee p_4\}$.

Definition 3.3 Let $\langle W, R\rangle$ be a PDL-model and $\varphi \to [a]\psi$ an effect law. The models resulting from contracting by $\varphi \to [a]\psi$ is $\langle W, R\rangle^-_{\varphi\to[a]\psi} = \{\langle W, R \cup R'_a\rangle : R'_a \subseteq \{(w, w') : \models^{\langle W,R\rangle}_{w} \varphi, \models^{\langle W,R\rangle}_{w'} \neg\psi \text{ and } w' \backslash w \subseteq lit(NewCons_{\mathcal{S}}(\neg\psi))\}\}$.

In our context, $lit(NewCons_{\mathcal{S}}(\neg\psi))$ corresponds to all the ramifications that action a can produce.

Suppose now the knowledge engineer learns new information about the executability of a. This usually occurs when there are executability laws that are too strong, i.e., the condition in the theory guaranteeing the executability of a is too weak and has to be made more restrictive. Let e.g. $\langle toggle\rangle\top$ be the law to be contracted, and suppose it has to be weakened to the more specific $\neg broken \to \langle toggle\rangle\top$. To implement such a weakening, the designer has to first contract the set of executability laws and then to expand the resulting set with the weakened law.

Contraction by $\varphi \to \langle a\rangle\top$ corresponds to removing some arrows leaving worlds where φ holds. Removing such arrows has as consequence that a is no longer always executable in context φ.

Definition 3.4 Let $\langle W, R\rangle$ be a PDL-model and $\varphi \to \langle a\rangle\top$ an executability law. The set of models that result from the contraction by $\varphi \to \langle a\rangle\top$ is $\langle W, R\rangle^-_{\varphi\to\langle a\rangle\top} = \{\langle W, R'\rangle : R' = R \setminus R''_a, R''_a \subseteq \{(w, w') : wR''_a w' \text{ and } \models^{\langle W,R\rangle}_{w} \varphi\}\}$.

In the next section we make a step toward syntactical operators that reflect the semantic foundations for contraction.

4 Contracting an action theory

Having established the semantics of action theory contraction, we can turn to its syntactical counterpart. Nevertheless, before doing that we have to consider an important issue. As the reader might have expected, the logical formalism of PDL alone does not solve the frame problem. For instance,

$$\left\{ \begin{array}{c} up \to light, \neg up \to [toggle]up, \\ up \to [toggle]\neg up, \langle toggle \rangle \top \end{array} \right\} \not\models_{\mathsf{PDL}} broken \to [toggle]broken.$$

Thus, we need a consequence relation powerful enough to deal with the frame and ramification problems. This means that the deductive power of PDL has to be augmented in order to ensure that the relevant frame axioms follow from the theory. Following the logical framework developed in [3], we consider metalogical information given in the form of a dependence relation:

Definition 4.1 (Dependence relation [3]) A *dependence relation* is a binary relation $\leadsto \subseteq \mathfrak{Act} \times \mathfrak{Lit}$.

The expression $a \leadsto l$ denotes that the execution of action a *may change* the truth value of the literal l. On the other hand, $\langle a, l \rangle \notin \leadsto$ (written $a \not\leadsto l$) means that l can never be caused by a. In our example we have $toggle \leadsto light$ and $toggle \leadsto \neg light$, which means that action *toggle* may cause a change in literals *light* and $\neg light$. We do not have $toggle \leadsto \neg broken$, for toggling the switch never repairs it.

We assume $\leadsto$ is finite.

Definition 4.2 A *model of a dependence relation* $\leadsto$ is a PDL-model $\mathscr{M}$ such that $\models^{\mathscr{M}} \{\neg l \to [a]\neg l : a \not\leadsto l\}$.

Given a dependence relation $\leadsto$, the associated consequence relation in the set of models for $\leadsto$ is noted $\models_{\leadsto}$. For our example we obtain

$$\left\{ \begin{array}{c} up \to light, \neg up \to [toggle]up, \\ up \to [toggle]\neg up, \langle toggle \rangle \top \end{array} \right\} \models_{\leadsto} broken \to [toggle]broken.$$

We have $toggle \not\leadsto \neg broken$, i.e., $\neg broken$ is never caused by *toggle*. Therefore in all contexts where *broken* is true, after every execution of *toggle*, *broken* still remains true. The consequence of this independence is that the frame axiom $broken \to [toggle]broken$ is valid in the models of $\leadsto$.

Such a dependence-based approach has been shown [6] to subsume Reiter's solution to the frame problem [32] and moreover treats the ramification problem, even when actions with both indeterminate and indirect effects are involved [4, 16].

Definition 4.3 An *action theory* is a tuple of the form $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$.

In our running example, the corresponding action theory is

$$\mathcal{S} = \{up \rightarrow light\},\ \mathcal{E} = \{\neg up \rightarrow [toggle]up, up \rightarrow [toggle]\neg up\}$$
$$\mathcal{X} = \{\langle toggle \rangle \top\},\ \leadsto = \left\{ \begin{array}{c} \langle toggle, light\rangle, \langle toggle, \neg light\rangle, \\ \langle toggle, up\rangle, \langle toggle, \neg up\rangle \end{array} \right\}$$

And we have $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} \neg up \rightarrow [toggle]light$. (For parsimony's sake, we write $\mathcal{S}, \mathcal{E}, \mathcal{X}$ instead of $\mathcal{S} \cup \mathcal{E} \cup \mathcal{X}$.)

Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ be an action theory and Φ a PDL-formula. $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_\Phi$ is the action theory resulting from the contraction of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ by Φ.

Contracting a theory by a static law φ amounts to using any existing contraction operator for classical logic. Let $\ominus$ be such an operator. Moreover, based on [19], we also need to guarantee that φ does not follow from $\mathcal{E}$, $\mathcal{X}$ and $\leadsto$ (cf. Section 6). We define contraction of a domain description by a static law as follows:

Definition 4.4 $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_\varphi = \langle \mathcal{S}^-, \mathcal{E}, \mathcal{X}^-, \leadsto \rangle$, where $\mathcal{S}^- = \mathcal{S} \ominus \varphi$ and $\mathcal{X}^- = \{(\varphi_i \wedge \varphi) \rightarrow \langle a \rangle \top : \varphi_i \rightarrow \langle a \rangle \top \in \mathcal{X}\}$.

We now consider the case of contracting an action theory by an executability law. For every executability in $\mathcal{X}$, we ensure that action a is executable only in contexts where $\neg\varphi$ is the case. The following operator does the job.

Definition 4.5 $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_{\varphi \rightarrow \langle a \rangle \top} = \langle \mathcal{S}, \mathcal{E}, \mathcal{X}^-, \leadsto \rangle$, where $\mathcal{X}^- = \{(\varphi_i \wedge \neg\varphi) \rightarrow \langle a \rangle \top : \varphi_i \rightarrow \langle a \rangle \top \in \mathcal{X}\}$.

For instance, contracting $glued \rightarrow \langle toggle \rangle \top$ in our example would give us $\mathcal{X}^- = \{\neg glued \rightarrow \langle toggle \rangle \top\}$.

Finally, to contract a theory by $\varphi \rightarrow [a]\psi$, for every effect law in $\mathcal{E}$, we first ensure that a still has effect ψ whenever φ does not hold, second we enforce that a has no effect in context $\neg\varphi$ except on those literals that are consequences of $\neg\psi$. Combining this with the new dependence relation also linking a to literals involved by $\neg\psi$, we have that a may now produce $\neg\psi$ as outcome. In other words, the effect law has been contracted. The operator below formalizes this:

Definition 4.6 $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_{\varphi \rightarrow [a]\psi} = \langle \mathcal{S}, \mathcal{E}^-, \mathcal{X}, \leadsto^- \rangle$, with $\leadsto^- = \leadsto \cup \{\langle a, l \rangle : l \in lit(NewCons_{\mathcal{S}}(\neg\psi))\}$ and $\mathcal{E}^- = \{(\varphi_i \wedge \neg\varphi) \rightarrow [a]\psi : \varphi_i \rightarrow [a]\psi \in \mathcal{E}\} \cup \{(\neg\varphi \wedge \neg l) \rightarrow [a]\neg l : \langle a, l \rangle \in (\leadsto^- \setminus \leadsto)\}$.

For instance, contracting the law $blackout \rightarrow [toggle]light$ from our theory would give us $\mathcal{E}^- = \{(\neg up \wedge \neg blackout) \rightarrow [toggle]up, (up \wedge \neg blackout) \rightarrow [toggle]\neg up\}$.

5 Results

In this section we present the main results that follow from our framework. These require the action theory under consideration to be modular [19]. In our framework, an action theory is said to be modular if a formula of a given type entailed by the whole theory can also be derived solely from its respective module (the set of formulas of the same type) together with the static laws $\mathcal{S}$. As shown in [19], to make a domain description satisfy such a property it is enough to guarantee that there is no classical formula entailed by the theory that is not entailed by the static laws alone.

Definition 5.1 $\varphi \in \mathfrak{Fml}$ is an *implicit static law* of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ if and only if $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} \varphi$ and $\mathcal{S} \not\models \varphi$.

A theory is *modular* if it has no implicit static laws. Our concept of modularity of theories was originally defined in [19], but similar concepts have also been addressed in the literature [5, 1, 39, 26, 18]. A modularity-based approach for narrative reasoning about actions is given in [23].

To witness how implicit static laws can show up, consider the quite simple action theory below, depicting the walking turkey scenario [36]:

$$\mathcal{S} = \{walking \to alive\}, \; \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \to [shoot]\neg alive \end{array} \right\}$$

$$\mathcal{X} = \left\{ \begin{array}{c} \langle tease \rangle \top, \\ \langle shoot \rangle \top \end{array} \right\}, \; \leadsto = \left\{ \begin{array}{c} \langle shoot, \neg loaded \rangle, \langle shoot, \neg alive \rangle, \\ \langle shoot, \neg walking \rangle, \langle tease, walking \rangle \end{array} \right\}$$

With this domain description we have $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} alive$: first, $\{walking \to alive, [tease]walking\} \models_{\leadsto} [tease]alive$, second $\models_{\leadsto} \neg alive \to [tease]\neg alive$ (from the independence $tease \not\leadsto alive$), and then $\mathcal{S}, \mathcal{E} \models_{\leadsto} \neg alive \to [tease]\bot$. As long as $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} \langle tease \rangle \top$, we must have $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} alive$. As $\mathcal{S} \not\models alive$, the formula *alive* is an implicit static law of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$.

Modular theories have several advantages [17]. For example, consistency of a modular action theory can be checked by just checking consistency of $\mathcal{S}$: if $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ is modular, then $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} \bot$ if and only if $\mathcal{S} \models \bot$. Deduction of an effect of a sequence of actions $a_1; \ldots; a_n$ (prediction) does not need to take into account the effect laws for actions other than $a_1, \ldots, a_n$. This applies in particular to plan validation when deciding whether $\langle a_1; \ldots; a_n \rangle \varphi$ is the case.

Throughout this work we have used PDL. For an assessment of the modularity principle in the Situation Calculus, see [18].

Here we establish that our operators are correct w.r.t. the semantics. Our first theorem establishes that the semantical contraction of the models of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ by Φ produces models of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{\Phi}^{-}$.

Theorem 5.1 Let $\langle W, R \rangle$ be a model of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$, and let Φ be a formula that has the form of one of the three laws. For all models $\mathscr{M}$, if $\mathscr{M} \in \langle W, R \rangle_{\Phi}^{-}$, then $\mathscr{M}$ is a model of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{\Phi}^{-}$.

It remains to prove that the other way round, the models of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{\Phi}^{-}$ result from the semantical contraction of models of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ by Φ. This does not hold in general, as shown by the following example: suppose there is only one atom p and one action a, and consider the theory $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ such that $\mathcal{S} = \emptyset$, $\mathcal{E} = \{p \to [a]\bot\}$, $\mathcal{X} = \{\langle a \rangle \top\}$, and $\leadsto = \emptyset$. The only model of that action theory is $\mathscr{M} = \langle \{\{\neg p\}\}, \{(\{\neg p\}, \{\neg p\})\} \rangle$. By definition, $\mathscr{M}_{p \to \langle a \rangle \top}^{-} = \{\mathscr{M}\}$. On the other hand, $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{p \to \langle a \rangle \top}^{-} = \langle \emptyset, \{p \to [a]\bot\}, \{\neg p \to \langle a \rangle \top\}, \emptyset \rangle$. The contracted theory has *two* models: $\mathscr{M}$ and $\mathscr{M}' = \langle \{\{p\}, \{\neg p\}\}, (\{\neg p\}, \{\neg p\}) \rangle$. While $\neg p$ is valid in the contraction of the models of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$, it is invalid in the models of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{p \to \langle a \rangle \top}^{-}$.

Fortunately, we can establish a result for those action theories that are modular. The proof requires three lemmas. The first one says that for a modular theory we can restrict our attention to its 'big' models.

Lemma 5.1 Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ be modular. Then $\mathcal{S}, \mathcal{E}, \mathcal{X} \models_{\leadsto} \Phi$ if and only if $\models^{\langle W, R \rangle} \Phi$ for every model $\langle W, R \rangle$ of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ such that $W = \mathit{val}(\mathcal{S})$.

Note that the lemma does not hold for non-modular theories (because the set $\{\langle W, R \rangle : W = \mathit{val}(\mathcal{S})\}$ is empty then).

The second lemma says that modularity is preserved under contraction.

Lemma 5.2 Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ be modular, and let Φ be a formula of the form of one of the three laws. Then $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{\Phi}^{-}$ is modular.

The third one establishes the required link between the contraction operators and contraction of 'big' models.

Lemma 5.3 Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ be modular, and let Φ be a formula of the form of one of the three laws. If $\mathscr{M}' = \langle \mathit{val}(\mathcal{S}), R' \rangle$ is a model of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle_{\Phi}^{-}$, then there is a model $\mathscr{M}$ of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ such that $\mathscr{M}' \in \mathscr{M}_{\Phi}^{-}$.

Putting the three above lemmas together we get:

Theorem 5.2 Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ be modular, Φ be a formula of the form of one of the three laws, and $\langle \mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^-, \leadsto^- \rangle$ be $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_\Phi$. If it holds that $\mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^- \models_{\leadsto^-} \Psi$, then for every model $\mathscr{M}$ of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ and every $\mathscr{M}' \in \mathscr{M}^-_\Phi$ it holds that $\models^{\mathscr{M}'} \Psi$.

Our two theorems together establish correctness of the operators:

Corollary 5.1 Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ be modular, Φ be a formula of the form of one of the three laws, and $\langle \mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^-, \leadsto^- \rangle$ be $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_\Phi$. Then $\mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^- \models_{\leadsto^-} \Psi$ if and only if for every model $\mathscr{M}$ of $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ and every $\mathscr{M}' \in \mathscr{M}^-_\Phi$ it holds that $\models^{\mathscr{M}'} \Psi$.

We also give a necessary condition for the success of a contraction.

Theorem 5.3 Let Φ be an effect or an inexecutability law such that $\mathcal{S} \not\models_{\mathsf{PDL}} \Phi$, and let $\langle \mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^-, \leadsto^- \rangle$ be $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_\Phi$. If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ is modular, then $\mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^- \not\models_{\leadsto^-} \Phi$.

6 Contracting implicit static laws

There can be many reasons why a theory should be changed. Following [17, 19], here we focus on the case where it has some classical consequence φ the designer is not aware of.

If φ is taken as intuitive, then, normally, no change has to be done at all, unless we want to keep abide on the modularity principle and thus make φ explicit by adding it to $\mathcal{S}$. In the scenario example of Section 4, if the knowledge engineer's universe has immortal turkeys, then she would add the static law *alive* to $\mathcal{S}$.

The other way round, if φ is not intuitive, as long as φ is entailed by $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$, the goal is to avoid such an entailment, i.e., what we want is $\mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^- \not\models_{\leadsto^-} \varphi$, where $\langle \mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^-, \leadsto^- \rangle$ is $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle^-_\Phi$. In the mentioned scenario, the knowledge engineer considers that having immortal turkeys is not reasonable and thus decides to change the domain description to $\langle \mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^-, \leadsto^- \rangle$ so that $\mathcal{S}^-, \mathcal{E}^-, \mathcal{X}^- \not\models_{\leadsto^-}$ *alive*.

This means that action theories that are not modular need to be changed, too. Such a changing process is driven by the problematic part of the theory detected by the algorithm defined in [17] and improved in [19].

The algorithm works as follows: for each executability law $\varphi \rightarrow \langle a \rangle \top$ in the theory, construct from $\mathcal{E}$ and $\leadsto$ a set of inexecutabilities $\{\varphi_1 \rightarrow [a]\bot, \ldots, \varphi_n \rightarrow [a]\bot\}$ that potentially conflict with $\varphi \rightarrow \langle a \rangle \top$. For each i,

$1 \leq i \leq n$, if $\varphi \wedge \varphi_i$ is satisfiable w.r.t. $\mathcal{S}$, mark $\neg(\varphi \wedge \varphi_i)$ as an implicit static law. Incrementally repeat this procedure (adding all the $\neg(\varphi \wedge \varphi_i)$ that were caught to $\mathcal{S}$) until no implicit static law is obtained.

For an example of the execution of the algorithm, consider $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \leadsto \rangle$ as above. For the action *tease*, we have the executability $\langle \mathit{tease} \rangle \top$. Now, from $\mathcal{E}$, and $\leadsto$ we try to build an inexecutability for *tease*. We take $[\mathit{tease}]\mathit{walking}$ and compute then all indirect effects of *tease* w.r.t. $\mathcal{S}$. From $\mathit{walking} \rightarrow \mathit{alive}$, we get that *alive* is an indirect effect of *tease*, giving us $[\mathit{tease}]\mathit{alive}$. But $\langle \mathit{tease}, \mathit{alive} \rangle \notin \leadsto$, which means the frame axiom $\neg \mathit{alive} \rightarrow [\mathit{tease}]\neg \mathit{alive}$ holds. Together with $[\mathit{tease}]\mathit{alive}$, this gives us the inexecutability $\neg \mathit{alive} \rightarrow [\mathit{tease}]\bot$. As $\mathcal{S} \cup \{\top, \neg \mathit{alive}\}$ is satisfiable ($\top$ is the antecedent of the executability $\langle \mathit{tease} \rangle \top$), we get the implicit static law *alive*. For this example no other inexecutability for *tease* can be derived, so the computation stops.

It seems that in general implicit static laws are not intuitive. Therefore their contraction is more likely to happen than their addition.[2] In the example above, the action theory has to be contracted by *alive*.[3] In order to contract the action theory, the designer has several choices:

1) Contract the set $\mathcal{S}$. (In this case, such an operation is not enough, since *alive* is a consequence of the rest of the theory.)

2) Weaken the effect law $[\mathit{tease}]\mathit{walking}$ to $\mathit{alive} \rightarrow [\mathit{tease}]\mathit{walking}$, since the original effect law is too strong. This means that in a first stage the designer has to contract the theory and in a second one expand the effect laws with the weaker law. The designer will usually choose this option if she focuses on the preconditions of the effects of actions.

3) Weaken the executability law $\langle \mathit{tease} \rangle \top$ by rephrasing it as $\mathit{alive} \rightarrow \langle \mathit{tease} \rangle \top$: first the executability is contracted and then the weaker one is added to the resulting set of executability laws. The designer will choose this option if she focuses on preconditions for action execution.

The analysis of this example shows that the choice of what change has to be carried out is up to the knowledge engineer. Such a task can get more complicated when ramifications are involved. To witness, suppose our scenario has been formalized as follows: $\mathcal{S} = \{\mathit{walking} \rightarrow \mathit{alive}\}$, $\mathcal{E} = \{[\mathit{shoot}]\neg \mathit{alive}\}$, $\mathcal{X} = \{\langle \mathit{shoot} \rangle \top\}$, and $\leadsto = \{\langle \mathit{shoot}, \neg \mathit{alive} \rangle\}$. From this action theory we can derive the inexecutability $\mathit{walking} \rightarrow [\mathit{shoot}]\bot$ and thus the implicit static

[2] In all the examples in which we have found implicit static laws that are intuitive they are so evident that the only explanation for not having them explicitly stated is that they have been forgotten by the theory's designer.

[3] Here the change operation is a revision-based operation rather than an update-based operation since we mainly "fix" the theory.

law $\neg walking$. In this case we have to change the theory by contracting the frame axiom $walking \rightarrow [shoot]walking$ (which amounts to adding the missing indirect dependence $shoot \leadsto \neg walking$).

7 Related work

Following [27, 28], Eiter *et al.* [8] have investigated update of action domain descriptions. They define a version of action theory update in an action language and give complexity results showing how hard such a task can be.

Update of action descriptions in their sense is always relative to some conditions (interpreted as knowledge possibly obtained from earlier observations and that should be kept). This characterizes a constraint-based update. In the example they give, change must be carried out preserving the assumption that pushing the button of the remote control is always executable. Actually, the method is more subtle, as new effect laws are added constrained by *the addition* of viz. an executability law for the new action under concern. In the example, the constraint (executability of push) was not in the original action description and must figure in the updated theory.

They describe domains of actions in a fragment of the action language $\mathcal{C}$ [12]. However they do not specify which fragment, so it is not clear whether the claimed advantages $\mathcal{C}$ has over $\mathcal{A}$ really transfer to their framework. At one hand, their approach deals with indirect effects, but they do not talk about updating a theory by a law with a nondeterministic action. Anyway, except for concurrency, their account can be translated into ours, as shown in [3].

Eiter *et al.* consider an action theory $\mathcal{T}$ as comprising two main components: $\mathcal{T}_u$, the part of the theory that must remain unchanged, and $\mathcal{T}_m$, the part concerning the statements that are allowed to change. The crucial information to the associated solution to the frame problem is always in $\mathcal{T}_u$.

Given an action theory $\mathcal{T} = \mathcal{T}_u \cup \mathcal{T}_m$, $((\mathcal{T}_u \cup \mathcal{T}_m), \mathcal{T}', C)$ is the problem of updating $\mathcal{T}$ by $\mathcal{T}' \subseteq \mathcal{S} \cup \mathcal{E}$ warranting the result satisfies all constraints in $C \subseteq \mathcal{S} \cup \mathcal{X}$.

Even though they do not explicitly state postulates for their kind of theory update, they establish conditions for the update operator to be successful. Basically, they claim for consistency of the resulting theory; maintenance of the new knowledge and the invariable part of the description; satisfaction of the constraints in C; and minimal change.

In some examples that they develop, the illustrated "partial solution" does not satisfy C due to the existence of implicit laws (cf. Example 1, where there is an implicit inexecutability law). To achieve a solution, while keeping

C, some other laws must be dropped (in the example, the agent gives up a static law).[4]

Just to see the link between update by subsumed laws and addition of implicit static laws, we note that Proposition 1 is the same as Theorem 14 in [19]: every implicit static law in Herzig and Varzinczak's sense is trivially a subsumed law in Eiter *et al.*'s sense.

With their method we can also contract by a static and an effect law. Contraction of executabilities are not explicitly addressed, and weakening (replacing a law by a weaker one) is left as future work.

A main difference between the approach in [8] and ours is that we do not need to add new fluents at every elaboration stage: we still work on the same set of fluents, refining their behavior w.r.t. an action a. in Eiter *et al.*'s proposal an update forces changing all the variable rules appearing in the action theory by adding to each one a new update fluent. This is a constraint when elaborating action theories.

8 Elaboration tolerance

The principle of elaboration tolerance has been proposed by McCarthy [29]. Roughly, it states that the effort required to add new information to a given representation (new laws or entities) should be proportional to the complexity of the information being added, i.e., it should not require the complete reconstruction of the old theory [34].

Since then many formalisms in the reasoning about actions field claim, in a more or less tacit way, to satisfy such a principle. However, for all this time there has been a lack of good formal criteria allowing for the evaluation of theory change difficulty and, consequently, comparisons between different frameworks are carried out in a subjective way.

The proposal by Amir [2] made the first steps in formally answering what difficulty of changing a theory means by formalizing one aspect of elaboration tolerance. The basic idea is as follows: let $\mathcal{T}_0$ be the original theory and let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two equivalent (and different) theories such that each one results from $\mathcal{T}_0$ by the application of some sequence of operations (additions and/or deletions of formulas). The resulting theory whose transformation from $\mathcal{T}_0$ has the shortest length (number of operations) is taken as the most elaboration tolerant.

Nevertheless, in the referred work only addition/deletion of *axioms* is considered, i.e., changes in the logical language or contraction of consequences of

[4] This does not mean however that the updated theory will necessarily contain no implicit law.

the theory not explicitly stated in the original set of axioms are not taken into account. This means that even the formal setting given in [2] is not enough to evaluate the complexity of theory change in a broad sense. Hence the community still needs formal criteria that allow for the comparison between more complex changes carried out by frameworks like ours and Eiter *et al.*'s, for example.

Of course, how elaboration tolerant a given update/revision method is strongly depends on its underlying formalism for reasoning about actions, i.e., its logical background, the solution to the frame problem it implements, the hypothesis it relies on, etc. In what follows we discuss how the dependence-based approach here used behaves when expansion is considered. Most of the comments concerning consequences of expansion can also be stated for contraction. We do that with respect to some of the qualitative criteria given in [30]. In all that follows we suppose that the resulting theory is consistent.

Adding effect laws In the dependence-based framework, adding the new effect law $\varphi \rightarrow [a]\psi$ to the theory demands a change in the dependence module $\leadsto$. In that case, the maximum number of statements added to $\leadsto$ is $|\{l : l \in \mathit{lit}(\mathit{NewCons}_{\mathcal{S}}(\psi))\}|$ (dependences for all indirect effects have to be stated, too). This is due to the explanation closure nature of the reasoning behind dependence (for more details, see [3]). Because of this, according to Shanahan [34], explanation closure approaches are not elaboration tolerant when dealing with the ramification problem. In order to achieve that, the framework should have a mechanism behaving like circumscription that automatically deals with ramifications. This raises the question: "if we had an automatic (or even semi-automatic) procedure to do the job of generating the indirect dependences, could we say the framework is elaboration tolerant?". We think we can answer positively to such a question, and, supported by Reiter [33], we are working on a semi-automatic procedure for generating the dependence relation from a set of effect laws.

Adding executability laws Such a task demands only a change in the set $\mathcal{X}$ of executabilities, possibly introducing implicit static laws as a side effect.

Adding static laws Besides expanding the set $\mathcal{S}$, adding new (indirect) dependences may be required (see above).

Adding frame axioms If the frame axiom $\neg l \rightarrow [a]\neg l$ has to be valid in the resulting theory, expunging the dependence $a \leadsto l$ should do the job.

Adding a new action name Without loss of generality we can assume the action in question was already in the language. In that case, we expect just to add effect or executability laws for it. For the former, at most $|\mathfrak{Lit}|$ dependences will be added to $\leadsto$. (We point out nevertheless that the requirement made in [30] that the addition of an action irrelevant for a given plan in the old theory should not preclude it in the resulting theory is too strong. Indeed, it is not difficult to imagine a new action forcing an implicit static law from which an inexecutability for some action in the plan can be derived. The same holds for the item below.)

Adding a new fluent name In the same way, we can suppose the fluent was already in the language. Such a task amounts thus to one or more of the above expansions. There will be at most $2 \times |\mathfrak{Act}|$ new elements added to $\leadsto$.

9 Concluding remarks

In this work we have presented a general method for changing a domain description (alias action theory) given any formula we want to contract.

We have defined a semantics for theory contraction and also presented its syntactical counterpart through contraction operators. Soundness and completeness of such operators with respect to the semantics have been established (Corollary 5.1).

We have also shown that modularity is a necessary condition for a contraction to be successful (Theorem 5.3). This gives further evidence that the notion of modularity is fruitful.

We have analysed an example of contraction of a non-modular theory by an implicit static law that is unintended.

Because of forcing formulas to be explicitly stated in their respective modules (and thus possibly making them inferable in independently different ways), intuitively modularity could be seen to diminish elaboration tolerance. For instance, when contracting a classical formula φ from a non-modular theory, it seems reasonable to expect not to change the set of static laws $\mathcal{S}$, while the theory being modular surely forces changing such a module. However it is not difficult to conceive non-modular theories in which contraction of a formula φ may demand a change in $\mathcal{S}$ as well. To witness, suppose $\mathcal{S} = \{\varphi_1 \rightarrow \varphi_2\}$ in an action theory from whose dynamic part we (implicitly) infer $\neg\varphi_2$. In this case, a contraction of $\neg\varphi_1$ keeping $\neg\varphi_2$ would necessarily ask for a change in $\mathcal{S}$. We point out nevertheless that in both cases (modular and non-modular) the extra work in changing other modules stays in the mechanical level, i.e., in the machinery that carries out the modification, and does not augment in a significant way the amount of work the knowledge engineer is expected to do.

What is the status of the AGM-postulates for contraction in our framework? First, contraction of static laws satisfies all the postulates, as soon as the underlying classical contraction operation $\ominus$ satisfies all of them.

In the general case, however, our constructions do not satisfy the central postulate of preservation $\langle\mathcal{S},\mathcal{E},\mathcal{X},\leadsto\rangle^{-}_{\Phi} = \langle\mathcal{S},\mathcal{E},\mathcal{X},\leadsto\rangle$ if $\mathcal{S},\mathcal{E},\mathcal{X} \not\models_{\leadsto} \Phi$. Indeed, suppose we have a language with only one atom p, and a model $\mathscr{M}$ with two worlds $w = \{p\}$ and $w' = \{\neg p\}$ such that $wR_a w'$, $w'R_a w$, and $w'R_a w'$. Then $\models^{\mathscr{M}} p \to [a]\neg p$ and $\not\models^{\mathscr{M}} [a]\neg p$, i.e., $\mathscr{M}$ is a model of the effect law $p \to [a]\neg p$, but not of $[a]\neg p$. Now the contraction $\mathscr{M}^{-}_{[a]\neg p}$ yields the model $\mathscr{M}'$ such that $R_a = W \times W$. Then $\not\models^{\mathscr{M}'} p \to [a]\neg p$, i.e., the effect law $p \to [a]\neg p$ is not preserved. Our contraction operation thus behaves rather like an update operation.

Now let us focus on the other postulates. Since our operator has a behavior which is close to the update postulate, we focus on the following basic erasure postulates introduced in [24]. Let $Cn(\mathcal{T})$ be the set of all logical consequences of a theory $\mathcal{T}$.

KM1 $Cn(\langle\mathcal{S},\mathcal{E},\mathcal{X},\leadsto\rangle^{-}_{\Phi}) \subseteq Cn(\langle\mathcal{S},\mathcal{E},\mathcal{X},\leadsto\rangle)$

Postulate **KM1** does not always hold because it is possible to make the formula $\varphi \to [a]\bot$ valid in the resulting theory by removing elements of R_a (cf. Definition 3.4).

KM2 $\Phi \notin Cn(\langle\mathcal{S},\mathcal{E},\mathcal{X},\leadsto\rangle^{-}_{\Phi})$

Under the condition that $\langle\mathcal{S},\mathcal{E},\mathcal{X},\leadsto\rangle$ is modular, Postulate **KM2** is satisfied (cf. Theorem 5.3).

KM3 If $Cn(\langle\mathcal{S}_1,\mathcal{E}_1,\mathcal{X}_1,\leadsto\rangle) = Cn(\langle\mathcal{S}_2,\mathcal{E}_2,\mathcal{X}_2,\leadsto\rangle)$ and $\models_{\mathsf{PDL}} \Phi_1 \leftrightarrow \Phi_2$, then $Cn(\langle\mathcal{S}_1,\mathcal{E}_1,\mathcal{X}_1,\leadsto\rangle^{-}_{\Phi_2}) = Cn(\langle\mathcal{S}_2,\mathcal{E}_2,\mathcal{X}_2,\leadsto\rangle^{-}_{\Phi_1})$.

Theorem 9.1 If $\langle\mathcal{S}_1,\mathcal{E}_1,\mathcal{X}_1,\leadsto\rangle$ and $\langle\mathcal{S}_2,\mathcal{E}_2,\mathcal{X}_2,\leadsto\rangle$ are modular and the propositional contraction operator $\ominus$ satisfies Postulate **KM3**, then Postulate **KM3** is satisfied for every $\Phi_1, \Phi_2 \in \mathfrak{Fml}$.

Here we have presented the case for contraction, but our definitions can be extended to revision, too. Our results can also be generalized to the case where learning new actions or fluents is involved. This means in general that more than one simple formula should be added to the belief base and must fit together with the rest of the theory with as little side-effects as possible. We are currently defining algorithms based on our operators to achieve that.

Acknowledgments

Ivan Varzinczak has been supported by a fellowship from the government of the Federative Republic of Brazil. Grant: CAPES BEX 1389/01-7.

References

[1] E. Amir. (De)composition of situation calculus theories. In *Proc. 17th Nat. Conf. on Artificial Intelligence (AAAI'2000)*, pages 456–463, Austin, 2000. AAAI Press/MIT Press.

[2] E. Amir. Toward a formalization of elaboration tolerance: Adding and deleting axioms. In M.-A. Williams and H. Rott, editors, *Frontiers of Belief Revision*. Kluwer, 2000.

[3] M. A. Castilho, O. Gasquet, and A. Herzig. Formalizing action and change in modal logic I: the frame problem. *J. of Logic and Computation*, 9(5):701–735, 1999.

[4] M. A. Castilho, A. Herzig, and I. J. Varzinczak. It depends on the context! a decidable logic of actions and plans based on a ternary dependence relation. In S. Benferhat and E. Giunchiglia, editors, *Workshop on Non-Monotonic Reasoning (NMR'02)*, pages 343–348, Toulouse, 2002.

[5] L. Cholvy. Checking regulation consistency by using SOL-resolution. In *Proc. 7th Int. Conf. on AI and Law*, pages 73–79, Oslo, 1999.

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

[7] P. Doherty, W. Łukaszewicz, and E. Madalinska-Bugaj. The PMA and relativizing change for action update. In A. G. Cohn, L. Schubert, and S. C. Shapiro, editors, *Proc. 6th Int. Conf. on Knowledge Representation and Reasoning (KR'98)*, pages 258–269, Trento, 1998. Morgan Kaufmann Publishers.

[8] T. Eiter, E. Erdem, M. Fink, and J. Senko. Updating action domain descriptions. In Kaelbling and Saffiotti [22], pages 418–423.

[9] N. Y. Foo and D. Zhang. Dealing with the ramification problem in the extended propositional dynamic logic. In F. Wolter, Wansing. H., M. de Rijke, and M. Zakharyaschev, editors, *Advances in Modal Logic*, volume 3, pages 173–191. World Scientific, 2002.

[10] K. D. Forbus. Introducing actions into qualitative simulation. In N. S. Sridharan, editor, *Proc. 11th Int. Joint Conf. on Artificial Intelligence (IJCAI'89)*, pages 1273–1278, Detroit, 1989. Morgan Kaufmann Publishers.

[11] P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States.* MIT Press, 1988.

[12] M. Gelfond and V. Lifschitz. Action languages. *Electronic Transactions on Artificial Intelligence*, 2(3–4):193–210, 1998.

[13] S. O. Hansson. *A Textbook of Belief Dynamics: Theory Change and Database Updating.* Kluwer Academic Publishers, 1999.

[14] D. Harel. Dynamic logic. In D. M. Gabbay and F. Günthner, editors, *Handbook of Philosophical Logic*, volume II, pages 497–604. D. Reidel, Dordrecht, 1984.

[15] A. Herzig and O. Rifi. Propositional belief base update and minimal change. *Artificial Intelligence*, 115(1):107–138, 1999.

[16] A. Herzig and I. Varzinczak. An assessment of actions with indeterminate and indirect effects in some causal approaches. Technical Report 2004–08–R, Institut de recherche en informatique de Toulouse (IRIT), Université Paul Sabatier, May 2004. `http://www.irit.fr/ACTIVITES/LILaC/`.

[17] A. Herzig and I. Varzinczak. Domain descriptions should be modular. In R. López de Mántaras and L. Saitta, editors, *Proc. 16th Eur. Conf. on Artificial Intelligence (ECAI'04)*, pages 348–352, Valencia, 2004. IOS Press.

[18] A. Herzig and I. Varzinczak. Cohesion, coupling and the meta-theory of actions. In Kaelbling and Saffiotti [22], pages 442–447.

[19] A. Herzig and I. Varzinczak. On the modularity of theories. In R. Schmidt, I. Pratt-Hartmann, M. Reynolds, and H. Wansing, editors, *Advances in Modal Logic*, volume 5, pages 93–109. King's College Publications, 2005. Selected papers of AiML 2004 (also available at `http://www.aiml.net/volumes/volume5`).

[20] K. Inoue. Linear resolution for consequence finding. *Artificial Intelligence*, 56(2–3):301–353, 1992.

[21] Y. Jin and M. Thielscher. Iterated belief revision, revised. In Kaelbling and Saffiotti [22], pages 478–483.

[22] L. Kaelbling and A. Saffiotti, editors. *Proc. 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05)*, Edinburgh, 2005. Morgan Kaufmann Publishers.

[23] A. Kakas, L. Michael, and R. Miller. $\mathcal{M}$odular-$\mathcal{E}$: an elaboration tolerant approach to the ramification and qualification problems - preliminary report. Proc. of the 7th Int. Symp. on Logical Formalizations of Commonsense Reasoning. Corfu, Greece, 2005.

[24] H. Katsuno and A. O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3):263–294, 1991.

[25] H. Katsuno and A. O. Mendelzon. On the difference between updating a knowledge base and revising it. In P. Gärdenfors, editor, *Belief revision*, pages 183–203. Cambridge University Press, 1992.

[26] J. Lang, F. Lin, and P. Marquis. Causal theories of action – a computational core. In V. Sorge, S. Colton, M. Fisher, and J. Gow, editors, *Proc. 18th Int. Joint Conf. on Artificial Intelligence (IJCAI'03)*, pages 1073–1078, Acapulco, 2003. Morgan Kaufmann Publishers.

[27] R. Li and L.M. Pereira. What is believed is what is explained. In H. Shrobe and T. Senator, editors, *Proc. 13th Nat. Conf. on Artificial Intelligence (AAAI'96)*, pages 550–555, Portland, 1996. AAAI Press/MIT Press.

[28] P. Liberatore. A framework for belief update. In *Proc. 7th Eur. Workshop on Logics in AI (JELIA 2000)*, pages 361–375, 2000.

[29] J. McCarthy. *Mathematical logic in artificial intelligence.* Daedalus, 1988.

[30] J. McCarthy. Elaboration tolerance. In *Proc. Common Sense'98*, London, 1998. Available at `http://www-formal.stanford.edu/jmc/elaboration.html`.

[31] S. Popkorn. *First Steps in Modal Logic.* Cambridge University Press, 1994.

[32] 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 359–380. Academic Press, San Diego, 1991.

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

[34] M. Shanahan. *Solving the frame problem: a mathematical investigation of the common sense law of inertia.* MIT Press, Cambridge, MA, 1997.

[35] S. Shapiro, M. Pagnucco, Y. Lespérance, and H. J. Levesque. Iterated belief change in the situation calculus. In T. Cohn, F. Giunchiglia, and B. Selman, editors, *Proc. 7th Int. Conf. on Knowledge Representation and Reasoning (KR'2000)*, pages 527–538, Breckenridge, 2000. Morgan Kaufmann Publishers.

[36] M. Thielscher. Computing ramifications by postprocessing. In C. Mellish, editor, *Proc. 14th Int. Joint Conf. on Artificial Intelligence (IJCAI'95)*, pages 1994–2000, Montreal, 1995. Morgan Kaufmann Publishers.

[37] M.-A. Winslett. Reasoning about action using a possible models approach. In R. G. Smith and T. M. Mitchell, editors, *Proc. 7th Nat. Conf. on Artificial Intelligence (AAAI'88)*, pages 89–93, St. Paul, 1988. Morgan Kaufmann Publishers.

[38] M.-A. Winslett. Updating logical databases. In D. M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 4, pages 133–174. Oxford University Press, 1995.

[39] D. Zhang, S. Chopra, and N. Y. Foo. Consistency of action descriptions. In *PRICAI'02, Topics in Artificial Intelligence.* Springer-Verlag, 2002.

[40] D. Zhang and N. Y. Foo. EPDL: A logic for causal reasoning. In B. Nebel, editor, *Proc. 17th Int. Joint Conf. on Artificial Intelligence (IJCAI'01)*, pages 131–138, Seattle, 2001. Morgan Kaufmann Publishers.