

Belief change in the Situation Calculus: a new proposal without plausibility levels

Robert Demolombe
ONERA Toulouse, France
Robert.Demolombe@cert.fr

Pilar Pozos Parra
Macquarie University, Australia
pilar@ics.mq.edu.au

Abstract

The Situation Calculus has been used by Scherl and Levesque to represent beliefs and belief change without modal operators, thanks to a predicate that plays the role of an accessibility relation. Their approach has been extended by Shapiro et al. to support belief revision. In this extension plausibility levels are assigned to each situation, and the believed propositions are the propositions that are true in all the most plausible accessible situations.

This solution is quite elegant from a theoretical point of view but it raises many difficulties when we have to apply it to a given application domain, due to the definition of the plausibility assignment.

In this paper is presented a new proposal that does not make use of plausibilities. The idea is to distinguish the successor state axioms that apply to the imaginary situations and those that apply to the real situations. In order to revise the agent beliefs the knowledge producing actions are included in the first type of axioms.

In this framework each agent may have a different successor state axiom for a given fluent. Then, each agent may have his subjective view of the evolution of the world. Also, agents may know or may not know that a given action has been performed. That is, the actions are not necessarily public.

1 A brief introduction to the Situation Calculus

In the very beginning the Situation Calculus was introduced by McCarthy [6]. It has been used by several authors to propose solutions to the frame problem (see [9] chapter 3 for a good overview of the literature about the frame problem and the Situation Calculus) and a simple solution has been proposed by Reiter in [8].

Since several variants¹ of the Situation Calculus have been presented in the literature, we have chosen in this paper to refer to the Situation Calculus as it is defined in Reiter’s book [9].

The Situation Calculus is a many sorted classical logic. Most of it uses first order logic, but some very limited fragments use second order logic (the second order is needed when we have to formalise the notion of transitive closure).

There are two kinds of predicates. Those whose truth values may change when actions are performed, which are called “fluents”, and those whose truth values do not change². The fluents have exactly one argument of the type situation which is the last argument.

For instance, if we consider a fragile toy, the fluent *broken*(*s*) can be used to represent the fact that in the situation *s* the toy is broken. The fluent *position*(*x*, *s*) can represent the fact that in the situation *s* the toy is in the position *x*. If the color of the toy never changes, its color is not represented by a fluent. For instance, it can be represented by the predicate *color*(*x*), which has no argument of the type situation, and whose meaning is that the toy color is *x*.

The situations are represented by terms that may be constants of the type situation, or complex terms formed with the designated binary function symbol *do*. If *a* is a term of the type action and *s* is of the type situation, then *do*(*a*, *s*) is of the type situation. From an intuitive point of view, *do*(*a*, *s*) represents the situation resulting from the performance of *a* in the situation *s*.

For example, the constant *s*₀ can be used to represent the initial situation, and, if *drop* and *repair* are of the type action, the terms: *do*(*drop*, *s*₀), *do*(*repair*, *s*₀) and *do*(*repair*, *do*(*drop*, *s*₀)), are of the type situation.

The solution to the frame problem is based on the assumption that we (the people who are modeling an application domain) know **all** the actions that can change the truth value of each fluent.

The fact that the actions *drop* and *repair* respectively cause the fluent *broken*(*s*) to be true or false is represented by the axioms:

$$\forall s \forall a (a = \textit{drop} \rightarrow \textit{broken}(\textit{do}(a, s)))$$

¹For example, in some papers the Successor State Axioms, that are presented later on, have preconditions about the feasibility of the actions, and others not.

²A similar distinction is made for function symbols.

$$\forall s \forall a (a = \text{repair} \rightarrow \neg \text{broken}(\text{do}(a, s)))$$

The fact that there is no other action that can change the truth value of $\text{broken}(s)$ is represented by the “Successor State Axiom”³:

$$\forall s \forall a (\text{broken}(\text{do}(a, s)) \leftrightarrow a = \text{drop} \vee \text{broken}(s) \wedge \neg(a = \text{repair}))$$

It is assumed that two actions that are represented by two different symbols are distinct actions. From this assumption we can infer that an action like $\text{move}(x, y)$ (whose meaning is that the toy moves from the position x to y) does not change the truth value of $\text{broken}(s)$. That is, from the previous SSA we can infer:

$$\forall s \forall x \forall y (\text{broken}(\text{do}(\text{move}(x, y), s)) \leftrightarrow \text{broken}(s))$$

The general form of a SSA for a fluent p is:

$$(S_p) \quad \forall s \forall a \forall \vec{x} (p(\vec{x}, \text{do}(a, s)) \leftrightarrow \Gamma_p^+(\vec{x}, a, s) \vee p(\vec{x}, s) \wedge \neg \Gamma_p^-(\vec{x}, a, s))$$

where $\Gamma_p^+(\vec{x}, a, s)$ and $\Gamma_p^-(\vec{x}, a, s)$ are first order formulas whose free variables are: \vec{x} ⁴, a and s .

For instance, for the fluent $\text{position}(x, s)$ we have:

$$\Gamma_{\text{position}}^+(x, a, s) = \exists y (a = \text{move}(y, x) \wedge \text{position}(y, s))$$

In [10] Scherl and Levesque have extended the Situation Calculus to represent beliefs and belief changes. One of the basic ideas is to see situations like possible worlds in Kripke models⁵. The second idea is to introduce a fluent $K(s', s)$ which has the same intuitive meaning as an accessibility relation in doxastic or epistemic logics.

To represent what an agent believes in a given situation, say s_0 , Reiter says in [9]: “we can picture this in terms of an agent inhabiting s_0 , and imagining all possible alternative situations to the one he is in”. Those imaginary situations are represented through the fluent K ⁶, that is, they are the situations s' such that: $K(s', s_0)$.

³In the following “Successor State Axiom” is abbreviated by SSA.

⁴ \vec{x} is used as an abbreviation for the tuple of variables x_1, \dots, x_n , and $\forall \vec{x}$ is an abbreviation for $\forall x_1 \dots \forall x_n$.

⁵This is an intuitive analogy. In fact there are significant differences between possible worlds and situations. One of them is that situations represent histories.

⁶The fluent K is an exception in the sense that it has two arguments of the type situation: s and s' . However, it is the situation s that defines in which situation $K(s', s)$ holds like for the other fluents. In [9] Reiter says: “Following common practice in modal logic, we call K an “accessibility relation”.

In general, $Knows(\phi, s)$ ⁷ is used as a notation to represent the fact in the situation s an agent believes ϕ . We have:

$$Knows(\phi, s) \stackrel{\text{def}}{=} \forall s'(K(s', s) \rightarrow \phi[s'])$$

where ϕ is the formula obtained from $\phi(s)$ by removing the argument of type situation s in the fluents, and $\phi[s']$ is obtained by inserting s' in the place of the argument of type situation in the fluents⁸.

For example, $Knows(\neg broken, s_0)$ denotes the formula: $\forall s'(K(s', s_0) \rightarrow \neg broken(s'))$.

It is worth noting that the function symbol *do* can also be viewed as a way to represent a dynamic modal operator, like the fluent K is used to represent a doxastic modal operator. Indeed, in [1] Demolombe has shown that we have:

$$\vdash \forall s(broken(do(drop, s)) \leftrightarrow \forall s'(s' = do(drop, s) \rightarrow broken(s')))$$

This property shows that, in general, we could use the notation: $[a](\phi, s)$ to denote the formula: $\forall s'(s' = do(a, s) \rightarrow \phi[s'])$, which is logically equivalent to: $\phi(do(a, s))$. The only difference is that we have a function instead of an accessibility relation. The reason for this difference is that in the Situation Calculus actions are deterministic.

To define belief changes two questions have to be answered:

- what are the new accessible situations after performance of an action a ?
- what are the truth values of the fluents in the new accessible situations?

The answer to the first question depends on the type of action a . Two different types of actions are considered: those that are “knowledge producing actions” and those that are not. Each knowledge producing action informs the agent about the truth of a given proposition in the situation where the agent is.

For instance, we can have the action *look* that informs the agent about the truth of $broken(s)$, and another knowledge producing action to inform him about the toy position.

Each knowledge producing action informs about the truth value of a given proposition, and in the case of a knowledge producing action the new accessible situations s' are the successor of the situations where this propo-

⁷The notation $Knows(\phi, s)$ is a bit misleading because ϕ is not necessarily true in s .

⁸See [9] for a detailed definition of the notation $\phi[s']$.

situation has the same truth value as in s . In other words, the imaginary situations that are inconsistent with what has been observed are removed.

In the case of a non knowledge producing action, the new accessible situations are the successors of the situations that were accessible before to perform a . That is, if s' was accessible from s , after performance of a the corresponding new accessible situation from $do(a, s)$ is $do(a, s')$.

In general, the new accessible situations are defined by the following axiom:

$$(S_K) \quad \forall s \forall s'' \forall a (K(s'', do(a, s)) \leftrightarrow \exists s' (K(s', s) \wedge s'' = do(a, s') \wedge (\\ \neg(a = \alpha_1) \wedge \dots \wedge \neg(a = \alpha_n)) \\ \vee a = \alpha_1 \wedge (\phi_1(s) \leftrightarrow \phi_1(s')) \\ \dots \\ \vee a = \alpha_n \wedge (\phi_n(s) \leftrightarrow \phi_n(s')))))$$

where $\alpha_1, \dots, \alpha_n$ is the set of all the knowledge producing actions, and ϕ_1, \dots, ϕ_n are their corresponding propositions.

The answer to the second question is that the truth values of the fluents in the new accessible situations are determined by the SSAs in the same way as they are determined for the situation $do(a, s)$. For example, if we have $K(do(a, s'), do(a, s))$, the truth value of $broken(do(a, s'))$ is determined by the SSA of the fluent $broken$, and by the truth value of $broken(s')$. Then, for non knowledge producing actions we can say that, in some sense, the beliefs change in the same way the world does.

However, this formalisation of belief change raises a big problem in the case of revision.

Let us assume, for example, that in the situation s_0 the agent believes that the toy is not broken, which is formally represented by: $Knows(\neg broken, s_0)$, and that in s_0 the toy is broken, which is represented by: $broken(s_0)$. Then according to (S_K) if the agent performs the action $look$, all the situations s' accessible from s_0 are removed because we have $\neg broken(s')$ and $broken(s_0)$.

The consequence is that there is no situation accessible from $do(look, s_0)$, and therefore every proposition and its negation is believed in $do(look, s_0)$. In other words we have inconsistent beliefs.

To remedy this problem Shapiro et al. have proposed in [11] to assign plausibility levels to all the situations, and to define the believed propositions as the propositions which are true in all the most plausible accessible

situations.

Then, it is possible to define the accessible situations and the plausibility levels in such a way that in s_0 , in all the most plausible accessible situations s' , we have $\neg broken(s')$, and to also have accessible situations s'' , that are less plausible, and where we have $broken(s'')$.

In that case, in s_0 the agent believes that the toy is not broken, and after performance of the action *look* the set of situations accessible from $do(look, s_0)$ is not empty, because from (S_K) all the situations like s'' have a successor situation $do(look, s'')$ which is accessible from $do(look, s_0)$. Since the knowledge producing actions do not change the truth values of the fluents, in $do(look, s_0)$ the agent believes that the toy is broken, as expected.

These ideas have been formalised by introducing a designated function symbol pl whose intuitive meaning is that the plausibility level of the situation s is $pl(s)$. It is assumed that all the successors of a given situation have the same plausibility level. That is, we have: $\forall s \forall a (pl(do(a, s)) = pl(s))$.

We adopt the notation $K_{max}(s', s)$ to represent the most plausible accessible situations⁹. We have:

$$K_{max}(s', s) \stackrel{\text{def}}{=} K(s', s) \wedge \forall s'' (K(s'', s) \rightarrow pl(s') \leq pl(s''))$$

Now, we have the new definition of beliefs:

$$Knows(\phi, s) \stackrel{\text{def}}{=} \forall s' (K_{max}(s', s) \rightarrow \phi[s'])$$

However, this new belief definition raises practical problems when we have to define the plausibility function for a given application domain. For example, it may be impossible to define the pl function by extension because the number of initial accessible situations may be infinite.

That is the case, for example, when the agent believes in the initial situation that the toy is not broken, and he ignores his position, and the number of possible position is infinite.

Fortunately, we can avoid having to give an explicit extensional definition of pl by using the assumption: $Knows(broken, s_0)$.

Nevertheless, if we don't want to get inconsistent beliefs after revision we need to add the assumption: $broken(s_0) \rightarrow \exists s'' (K(s'', s_0) \wedge \neg K_{max}(s'', s_0) \wedge broken(s''))$.

Is this assumption sufficient to prove that in $do(look, s_0)$ the agent be-

⁹For practical reasons, the most plausible situations are the situations that have the lowest plausibility level.

believes that the toy is broken? The answer is not obvious.

Moreover, to express the fact that the agent ignores the toy’s position we need the assumption: $\forall x(position(x, s_0) \rightarrow \neg Knows(position(x), s_0))$.

Here again, to prevent inconsistent beliefs after the agent has learnt the toy position, we need to add the assumption: $\forall x(position(x, s_0) \rightarrow \exists s''(K(s'', s_0) \wedge \neg K_{max}(s'', s_0) \wedge position(x, s'')))$.

However, we have to add an additional constraint to prevent that the only situation s'' that satisfies: $K(s'', s_0) \wedge \neg K_{max}(s'', s_0) \wedge position(x, s'')$, was removed after performance of the action *look* (that will happen if this situation s'' is such that: $K(s'', s_0) \wedge \neg K_{max}(s'', s_0) \wedge position(x, s'') \wedge \neg broken(s'')$).

This simple example shows that it is definitely not easy to guarantee that no assumption is missing in the description of the initial situation. If in the application domain there are five, ten, or even more fluents, modeling the initial situation may be quite problematic.

To remove these difficulties, in [4] Demolombe and Pozos Parra have proposed a simple but restrictive solution to belief revision. The idea is to generalise the successor state axioms from literals to modal literals. Although in this solution the scope of the beliefs is limited to literals, Petrick and Levesque in [7] have analysed its expressive power, and they have shown that its limitations are not so strong. In [2] Demolombe and Herzig have also shown that a very similar idea has been formalised in the Dependence logic.

2 Belief revision without plausibility levels

Here a new solution is proposed to belief revision. The basic idea is to not use plausibility levels and to instead incorporate the sensing actions into the successor state axioms. Then the effect of a sensing¹⁰ action is not to remove the situations that are inconsistent with the perceived information, as Scherl and Levesque do in [10], but to change the truth values of the fluents in the imaginary successor situations according to the perceived information.

There are two other important features in this new proposal. The first one is that we may have distinct successor state axioms for real situations and for agents’ imaginary situations. The second one is that we have an

¹⁰In the following “sensing action” will be used as a shorthand for “knowledge producing action”.

explicit representation of the actions whose performance can be observed by an agent, and, if an agent has not observed the performance of an action, then this action does not change his beliefs.

2.1 From a typical example to a general framework

Before to present a general framework we analyse a particular scenario.

Scenario.

In a room there is a young baby and his mother. It is assumed that the baby observes every action he performs iff he has his eyes open. The same applies for the mother.

In the situation s_0 (see Figure 1 in Annex)¹¹ the baby holds a fragile toy in his hands. The toy is not broken and both the mother and the baby believe that the toy is not broken. The baby and the mother have their eyes open.

In the situation s_1 the baby has dropped the toy, i.e. $s_1 = do(drop(b), s_0)$ ¹². The mother believes that a fragile toy will break after it is dropped. It follows that she believes that the toy is broken. She does not need to look at the toy to hold this belief. However, the baby does not believe that dropping a fragile toy will cause it to break. So the baby in s_1 believes that the toy is not broken.

In the situation s_2 the baby has looked at the toy, i.e. $s_2 = do(look(b), s_1)$, and he observed that it is broken. Then, in s_2 he believes that the toy is broken. In s_2 the mother still believes that the toy is broken.

In the situation s_3 the baby has closed his eyes, i.e. $s_3 = do(close(b), s_2)$. The baby's beliefs and the mother's beliefs remain unchanged, except about the fact that this action has been performed.

In the situation s_4 the mother has repaired the toy, i.e. $s_4 = do(repair(m), s_3)$. Since the baby has closed his eyes he ignores that this action has been performed. Then, in s_4 he still believes that the toy is broken.

The main features of this scenario are the following ones.

- In s_4 the baby ignores that the action $repair(m)$ has been performed.

¹¹To make the figure easier to read we have respectively used the notations K_m and K_b for $K(m, -, -)$ and $K(b, -, -)$.

¹²In this section, since we may have several agents, action function symbols have an argument to make explicit the agent who is doing the action.

Then, the baby's beliefs are the same in s_4 and s_3 , i.e. the baby's imaginary situations, like s'_3 , remain unchanged after the execution of the unobserved action.

- The mother and the baby have different beliefs about the SSA of the fluent *broken*. That is why we have $broken(s''_1)$ and $\neg broken(s'_1)$ (see Figure 1).
- In s_2 the action $look(b)$ has informed the baby about the truth value of the fluent *broken* in s_1 . Thus we have $broken(s'_2)$ because in the real situation s_1 we have $broken(s_1)$.

Formalisation.

The following notations are adopted.

$open(i)$: the agent i opens his eyes.

$close(i)$: the agent i closes his eyes.

$drop(i)$: the agent i drops the toy.

$look(i)$: the agent i looks whether the toy is broken.

$repair(i)$: the agent i repairs the toy.

$broken(s)$: in the situation s the toy is broken.

$real(s)$: s is a real situation.

It is the person who is modeling an application who defines which situations are real situations.

$observe(i, a, s)$: in the situation s , if the action a is performed, then after performance of a the agent i will be informed that a has been performed.

Note that $observe(i, a, s)$ does not mean that i has observed performance of a , but that i has the ability to observe performance of a **if** a is performed.

$K(i, s', s)$: in the situation s the agent i believes that s' is a situation where he might be in. That is, s' is compatible with what i believes in s .

$B(i, \phi(s', s), s', s)$: $\phi(s', s)$ holds in every situation s' related with i and s . In other words $\phi(s', s)$ is compatible with what i believes in s . Note that the free variables s' and s of ϕ must be the third and the fourth argument of B respectively, since s may not be a free variable of ϕ .

In $B(i, \phi(s', s), s', s)$ the situation s may be a real situation or an imaginary situation.

Notice that B is an abbreviation and not a fluent. We have:

$$B(i, \phi(s', s), s', s) \stackrel{\text{def}}{=} \forall s' (K(i, s', s) \rightarrow \phi(s', s))$$

The formula $\phi(s', s)$ may be any formula in a first order language with the only restriction being that the fluent K only appears in formulas that represent beliefs.

Evolution of imaginary situations.

If the action a is performed in the situation s , the situations that are accessible from $do(a, s)$ are the successors of the situations accessible from s if the action a has been observed by i , and they are the same situations as the situations accessible from s if the action a has not been observed by i . Then, we have:

$$(EK) \quad \forall s \forall s'' \forall a \forall i (K(i, s'', do(a, s)) \leftrightarrow \exists s' (K(i, s', s) \wedge ((observe(i, a, s) \wedge s'' = do(a, s')) \vee (\neg observe(i, a, s) \wedge s'' = s'))))$$

If we define the fluent $done(a, s)$, whose intuitive meaning is that in s the action a has just been performed, and whose SSA is: $\forall s \forall a (done(a, do(a, s)))$, then we can show from (EK) that we have: $\forall s \forall a \forall i (observe(i, a, s) \rightarrow B(i, done(a, s''), s'', do(a, s)))$. That intuitively means that after an action has been performed an agent will believe it has been performed if he observes it.

We have to define a SSA for the fluent $observe(i, a, s)$. This SSA depends on each application domain. For the scenario we have presented here we have adopted the SSA:

$$(OBS) \quad \forall s \forall a \forall a' \forall i (observe(i, a, do(a', s)) \leftrightarrow a' = open(i) \vee observe(i, a, s) \wedge \neg(a' = close(i)))$$

The intuitive meaning of (OBS) is that an agent has the capacity to observe any action provided he has opened and not closed his eyes

Since s is not restricted to real situations, the SSA (OBS) is believed by every agent¹³.

Notice that in this formalisation, for an action that leads to a revision (like the action $look(b)$ after the situation s_1) no situation accessible by K is removed (as it is the case in the solution proposed by Scherl and Levesque in [10]). The fact that an action like $look(b)$ leads to a revision is expressed in SSA_b (see SSA_b below).

The SSAs for the fluents depend on the context where we are. We may have different SSAs for the same fluent depending on the fact that we are in a real situation or in some agent's imaginary situation.

¹³As a matter of simplification we have assumed that an agent performs no action, except open eyes if he has closed his eyes.

The SSA for the real situations is:

$$(SSA_r) \quad \forall s \forall a (real(s) \rightarrow (broken(do(a, s)) \leftrightarrow \exists i (a = drop(i)) \vee broken(s) \wedge \neg(\exists i (a = repair(i)))))$$

In the case of the baby and its mother the SSA for the mother's imaginary situations is:

$$(SSA_m) \quad \forall s \forall s' \forall a (real(s) \rightarrow (K(m, s', s) \rightarrow (broken(do(a, s')) \leftrightarrow \exists i (a = drop(i)) \vee (a = look(m) \wedge broken(s)) \vee broken(s') \wedge \neg(\exists i (a = repair(i)) \vee (a = look(m) \wedge \neg broken(s)))))$$

Notice that the fluent *broken* comes to be true (resp. false) in a mother's imaginary situation $do(a, s')$ if the mother has looked at the toy (action $look(m)$) and the toy is broken (resp. not broken) in the real situation s .

We can see with the axiom (SSA_m) how the sensing action $look(m)$ has the effect of changing the truth value of the fluent *broken* in the mother's imaginary situations, while in the Scherl and Levesque approach its effect is to remove the imaginary situations that are not consistent with the sensing action.

The SSA for the baby's imaginary situations is:

$$(SSA_b) \quad \forall s \forall s' \forall a (real(s) \rightarrow (K(b, s', s) \rightarrow (broken(do(a, s')) \leftrightarrow (a = look(b) \wedge broken(s)) \vee broken(s') \wedge \neg(\exists i (a = repair(i)) \vee (a = look(b) \wedge \neg broken(s)))))$$

If we adopt the notations:

$$ssa(b, s', s) \stackrel{\text{def}}{=} broken(do(a, s')) \leftrightarrow (a = look(b) \wedge broken(s)) \vee broken(s') \wedge \neg(\exists i (a = repair(i)) \vee (a = look(b) \wedge \neg broken(s)))$$

$$ssa(m, s', s) \stackrel{\text{def}}{=} broken(do(a, s')) \leftrightarrow \exists i (a = drop(i)) \vee (a = look(m) \wedge broken(s)) \vee broken(s') \wedge \neg(\exists i (a = repair(i)) \vee (a = look(m) \wedge \neg broken(s)))$$

the axioms SSA_b and SSA_m can be reformulated as:

$$(SSA_b) \quad \forall s (real(s) \rightarrow B(b, ssa(b, s', s), s', s))$$

$$(SSA_m) \quad \forall s (real(s) \rightarrow B(m, ssa(m, s', s), s', s))$$

It is worth noting that, depending on the application, it can be assumed that an agent is informed about the SSAs believed by the other agents.

For example, it may be assumed that the baby believes that his mother believes, like him, that the toy is not broken if it is dropped, and also that the mother believes that her baby believes that the toy is not broken if it is dropped, even if she does not hold that belief.

That can be formally represented by the following axioms.

$$(SSA_{bm}) \quad \forall s(real(s) \rightarrow B(b, B(m, ssa(b, s'', s'), s'', s'), s', s))$$

$$(SSA_{mb}) \quad \forall s(real(s) \rightarrow B(m, B(b, ssa(b, s'', s'), s'', s'), s', s))$$

Successor State Axioms in general.

In general, to define the “real” evolution of fluents, for each fluent p we have the SSA:

$$(SSA_p) \quad \forall s \forall a \forall \vec{x}(real(s) \rightarrow (p(\vec{x}, do(a, s)) \leftrightarrow \Gamma_p^+(\vec{x}, a, s) \vee p(\vec{x}, s) \wedge \neg \Gamma_p^-(\vec{x}, a, s)))$$

To define the “subjective” evolution of fluents, for an agent i and a fluent p we have the SSA:

$$(SSA_{i,p}) \quad \forall s \forall s' \forall a \forall \vec{x}(real(s) \rightarrow (K(i, s', s) \rightarrow (p(\vec{x}, do(a, s')) \leftrightarrow \Gamma_{i,p}^+(i, \vec{x}, a, s') \vee (a = sense_p(i) \wedge p(\vec{x}, s)) \vee p(\vec{x}, s') \wedge \neg(\Gamma_{i,p}^-(i, \vec{x}, a, s') \vee (a = sense_p(i) \wedge \neg p(\vec{x}, s))))))$$

It is assumed that the successors of the real situations are real situations. Then, we have:

$$(SSA_R) \quad \forall s \forall a(real(do(a, s)) \leftrightarrow real(s))$$

2.2 Progression and regression

For reasoning about actions and beliefs we define a Basic Action and Belief Theory T which is composed of the following axioms.

1. Σ : the foundational axioms for situations (see [9]).
2. T_{SS} : a set of successor state axioms of the form¹⁴ (SSA_p) and $(SSA_{i,p})$.
3. T_K : the successor state axiom (EK) for the fluent $K(i, s', s)$.
4. T_O : the successor state axiom for the fluent $observe(i, a, s)$ which depends on the application domain and has a form similar to (OBS) .
5. T_R : the successor state axiom (SSA_R) .
6. T_{AP} : a set of precondition axioms (see [9]).
7. T_{UNA} : a set of unique name axioms for actions (see [9]).

¹⁴We omit for simplicity the axioms for functional fluents.

8. T_0 : a set of first order sentences defining the initial situation s_0 .

Progression.

The progression problem is to infer from the initial situation s_0 consequences about properties in a further situation s_i . So, to solve the progression problem for $\phi(s_i)$ we have to prove that:

$$\vdash T \rightarrow \phi(s_i)$$

We use the example of the scenario to show how to derive consequences about the next situation from the current situation.

The initial situation T_0 is represented by the following formulas.

- (1) $real(s_0)$, (2) $\neg broken(s_0)$, (3) $\forall a observe(m, a, s_0)$, (4) $\forall a observe(b, a, s_0)$,
 (5) $B(m, \neg broken(s'), s', s_0)$, (6) $B(b, \neg broken(s'), s', s_0)$.

We show in the annex how we can derive in s_1 :

$$B(b, \neg broken(s''), s'', s_1) \text{ and } B(m, broken(s''), s'', s_1)$$

and how we can derive in s_2 :

$$B(b, broken(s''), s'', s_2).$$

Regression.

The regression problem is to find a transformation R_T that transforms a formula $\phi(do(a, s))$ about $do(a, s)$ into a formula $\psi(s)$ about s (i.e. $R_T(\phi(do(a, s))) = \psi(s)$), and such that:

$$\vdash T \rightarrow (\phi(s_i) \leftrightarrow R_T^*(\phi(s_i)))$$

where s_i is a ground term and R_T^* represents an iterated application of R_T until the result of R_T application remains unchanged (in that case $R_T^*(\phi(s_i))$ is a sentence about s_0) (see [9]).

We say that R_T is **sound** if we have:

$$\vdash T \rightarrow R_T^*(\phi(s_i)) \Rightarrow \vdash T \rightarrow \phi(s_i)$$

We say that R_T is **complete** if we have:

$$\vdash T \rightarrow \phi(s_i) \Rightarrow \vdash T \rightarrow R_T^*(\phi(s_i))$$

In this paper we have not formally solved the regression problem. Nevertheless, we shall present some guidelines to solve it. As a matter of simplification we shall use the notation R instead of R_T when the theory T can be made implicit.

If the SSAs are the same for every agent's imaginary situations and for real situations, we can easily adapt the regression technique defined by Scherl and Levesque in [10].

If the SSAs are different, the SSA that has to be applied to regress fluents depends on the "context" of the situation where we are. For example, to regress $broken(do(a, s'))$ we have to know whether s' is a mother's imaginary situation or a baby's imaginary situation. That depends on the fact that we have $K(m, s', s)$ or $K(b, s', s)$. This shows that the regression of the fluents depends on the context.

The context of a situation is a tuple defined as follows. If we have $real(s)$, the context of s is $\langle r \rangle$. That means that s is a real situation. If we have $real(s) \wedge K(m, s', s)$, the context of s' is $\langle r, m \rangle$. That means that s' is a mother's imaginary situation. If we have $real(s) \wedge K(m, s', s) \wedge K(b, s'', s')$, the context of s'' is $\langle r, m, b \rangle$, and so on.

The regression transformation R is defined for a given context $\langle c \rangle$.

As usual we have: $R_{\langle c \rangle}(\phi_1 \vee \phi_2) = R_{\langle c \rangle}(\phi_1) \vee R_{\langle c \rangle}(\phi_2)$, $R_{\langle c \rangle}(\neg \phi_1) = \neg R_{\langle c \rangle}(\phi_1)$ and $R_{\langle c \rangle}(\forall v \phi_1) = \forall v R_{\langle c \rangle}(\phi_1)$.

If p is a fluent, we have: $R_{\langle c \rangle}(p(do(a, s))) = R_{\langle c \rangle}(\Phi_p(a, s))$ if for the context $\langle c \rangle$ we have the SSA: $p(do(a, s)) \leftrightarrow \Phi_p(a, s)$.

The fact that we have the SSA: $p(do(a, s)) \leftrightarrow \Phi_p(a, s)$ in the context $\langle c \rangle$ means that if $\langle c \rangle = \langle r \rangle$ we have: $\forall s \forall a (real(s) \rightarrow (p(do(a, s')) \leftrightarrow \Phi_p(a, s)))$, and if $\langle c \rangle = \langle r, m \rangle$ we have: $\forall s (real(s) \rightarrow B(m, p(do(a, s)) \leftrightarrow \Phi_p(a, s', s), s', s))$, and so on.

For the regression of a belief we have:

$$\begin{aligned} R_{\langle c \rangle}(B(i, \phi(s''), do(a, s)), s'', do(a, s)) = \\ \forall s' (R_{\langle c \rangle}(K(i, s', s) \wedge observe(i, a, s)) \rightarrow R_{\langle c, i \rangle}(\phi(do(a, s'), do(a, s)))) \wedge \\ \forall s' (R_{\langle c \rangle}(K(i, s', s) \wedge \neg observe(i, a, s)) \rightarrow R_{\langle c, i \rangle}(\phi(s', do(a, s)))) \end{aligned}$$

That gives the general idea of the regression definition but we have to investigate its definition in more details. In particular for the definition of $R_{\langle c, i \rangle}(\phi(do(a, s'), do(a, s)))$ we have to distinguish the subformulas whose unique free variable is $do(a, s')$, because these formulas are regressed with the context $\langle c, i \rangle$, and the subformulas whose unique free variable is $do(a, s)$, because they are regressed with the context $\langle c \rangle$.

For instance from SSA_b we have :

$$\forall s' (real(s_1) \wedge K(b, s', s_1) \rightarrow (broken(do(look(b), s')) \leftrightarrow broken(s_1)))$$

Then, if we have $real(s_1) \wedge K(b, s', s_1)$, the context of $do(look(b), s')$ is \langle

$r, b >$ but we have: $R_{\langle r, b \rangle}(\text{broken}(\text{do}(\text{look}(b), s'))) = R_{\langle r \rangle}(\text{broken}(s_1))$, because the context of s_1 is $\langle r \rangle$.

The derivation for the regression of (1') $B(m, \neg \text{broken}(s''), s'', s_4)$ in the annex allows to understand why the definition of the regression function $R_{\langle c \rangle}$ is sound.

Another complication comes from the fact that, **if** we are in some context $\langle c \rangle$, **then** we have some SSA for a given fluent p .

For example, from (SSA_m) we know that if s' is in the context $\langle r, m \rangle$ we have to prove, by **hypothesis generation** (see [3, 5]), that s' is in the context $\langle r, m \rangle$. For example, in the annex from (9') we can infer that it is sufficient to prove (10') to prove (1'), but we cannot infer that this is necessary. Then, the regression technique presented here is sound but not complete.

3 Conclusion

We have presented a general framework for belief revision in the Situation Calculus that does not require the definition of a plausibility distribution.

The key idea is to have for each agent i SSAs for beliefs of the form:

$$\begin{aligned} & (SSA_{i,p}) \quad \forall s \forall s' \forall a \forall \vec{x} (\text{real}(s) \rightarrow (K(i, s', s) \rightarrow \\ & (p(\vec{x}, \text{do}(a, s')) \leftrightarrow \Gamma_{i,p}^+(i, \vec{x}, a, s') \vee (a = \text{sense}_p(i) \wedge p(\vec{x}, s)) \vee p(\vec{x}, s') \wedge \\ & \neg(\Gamma_{i,p}^-(i, \vec{x}, a, s') \vee (a = \text{sense}_p(i) \wedge \neg p(\vec{x}, s)))))) \end{aligned}$$

where $\text{sense}_p(i)$ is a sensing action that informs i about the truth of $p(s)$.

These axioms define how the truth values of the fluents change in imaginary situations when a sensing action is performed, and we do not have to remove the imaginary situations that are inconsistent with the observations as Scherl and Levesque do in [10].

There is no restriction about sentences that are believed, and we have seen with (SSA_{bm}) and (SSA_{mb}) how these axioms can be extended to nested beliefs.

The axiom (EK) takes into account the fact that the actions are not necessarily public, and the fluent $\text{done}(a, s)$ allows an agent to reason about the fact that an action has been performed.

We have shown with the example how we can solve the projection problem either by progression or by regression. In the case of regression the

general definition is more complex and requires more work.

The framework can be easily simplified if it assumed that every action is public (in formal terms we have: $\forall s \forall a \forall i \text{ observe}(i, a, s)$), or when the SSAs are independent of the context. In that case the definition of regression should be simpler.

In this approach the sensing actions provide information about the truth of a fluent and not about the truth of a general formula, like in [10]. Is this a strong limitation? We do not think so, because the information that is directly delivered by a sensor is an atomic information which is interpreted in the context of a theory of the application domain as a formula.

For example, a thermometer does not tell us that the temperature is in-between 9 and 11 degrees. The thermometer just tells us that the position of the liquid is at level 10, and, if we know that the accuracy of the thermometer is plus or minus one degree, we infer that the temperature is in-between 9 and 11 degrees. In the same way a sonar does not tell us that there is an obstacle. In some circumstances the sonar sends a signal, and from this signal it can be inferred that there exists an obstacle.

In conclusion, the general idea of the proposed solution is to incorporate into the successor state axioms the overall information we need.

References

- [1] R. Demolombe. Belief change: from Situation Calculus to Modal Logic. In G. Brewka and P. Peppas, editor, *Proc. of the Workshop on Non-monotonic Reasoning, Action and Change*, 2003.
- [2] R. Demolombe and A. Herzig. Obligation change in Dependence Logic and Situation Calculus. In A. Lomuscio and D. Nute, editors, *Proceedings of the 7th International Workshop on Deontic Logic in Computer Science*. Springer, LNAI 3065, 2004.
- [3] R. Demolombe and L. Fariñas del Cerro. An Inference Rule for Hypothesis Generation. In *Proc. of International Joint Conference on Artificial Intelligence*, Sydney, 1991.
- [4] R. Demolombe and M. P. Pozos-Parra. A simple and tractable extension of situation calculus to epistemic logic. In Z. W. Ras and S. Ohsuga, editors, *Proc. of 12th International Symposium ISMIS 2000*. Springer. LNAI 1932, 2000.

- [5] K. Inoue. Consequence-Finding Based on Ordered Linear Resolution. In *Proc. of International Joint Conference on Artificial Intelligence*, Sydney, 1991.
- [6] J. McCarthy. Situations, actions and causal laws. In M. Minski, editor, *Semantic Information Processing*. The MIT press, 1968.
- [7] R. Petrick and H. Levesque. Knowledge equivalence in combined action theories. In *Proceeding 8th International Conference on Principles of Knowledge Representation and Reasoning*, 2002.
- [8] 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, 1991.
- [9] R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.
- [10] R. Scherl and H. Levesque. The Frame Problem and Knowledge Producing Actions. In *Proc. of the National Conference of Artificial Intelligence*. AAAI Press, 1993.
- [11] S. Shapiro, M. Pagnuco, Y. Lespérance, and H. Levesque. Iterated belief change in the situation calculus. In *Proc. of the 7th Conference on Principles on Knowledge Representation and Reasoning (KR2000)*. Morgan Kaufman Publishers, 2000.

Annex

Progression

We have the following derivation.

- (7) $real(s_1)$ from (1) and SSA_R
- (8) $broken(s_1)$ from (7) and SSA_r
- (9) $\forall s''(K(b, s'', s_1) \leftrightarrow \exists s'(K(b, s', s_0) \wedge s'' = do(drop(b), s')))$ from (4) and (EK)
- (10) $\forall s''(K(b, s'', s_1) \rightarrow \exists s'(K(b, s', s_0) \wedge s'' = do(drop(b), s')))$ from (9)
- (11) $\forall s'(K(b, s', s_0) \rightarrow (broken(do(drop(b), s')) \leftrightarrow broken(s')))$ from (1) and SSA_b
- (12) $\forall s'(K(b, s', s_0) \rightarrow \neg broken(s'))$ from (6)
- (13) $\forall s'(K(b, s', s_0) \rightarrow \neg broken(do(drop(b), s')))$ from (11) and (12)
- (14) $\forall s''\forall s'((K(b, s', s_0) \wedge s'' = do(drop(b), s')) \rightarrow \neg broken(s''))$ from (13) and Equality axioms
- (15) $\forall s''(\exists s'(K(b, s', s_0) \wedge s'' = do(drop(b), s')) \rightarrow \neg broken(s''))$ from (14)
- (16) $\forall s''(K(b, s'', s_1) \rightarrow \neg broken(s''))$ from (10) and (15)
- (17) $B(b, \neg broken(s''), s'', s_1)$ from (16)

From SSA_m we can derive in a similar way $B(m, broken(s''), s'', s_1)$.

Now we show how to progress from s_1 to s_2 .

- (18) $\forall a observe(b, a, s_1)$ from (4) and (OBS)
- (19) $\forall s''(K(b, s'', s_2) \leftrightarrow \exists s'(K(b, s', s_1) \wedge s'' = do(look(b), s')))$ from (18) and (EK)
- (20) $\forall s''(K(b, s'', s_2) \rightarrow \exists s'(K(b, s', s_1) \wedge s'' = do(look(b), s')))$ from (19)
- (21) $\forall s'(K(b, s', s_1) \rightarrow broken(do(look(b), s')))$ from (7), (8) and (SSA_b)
- (22) $\forall s''(K(b, s'', s_2) \rightarrow broken(s''))$ from (20) and (21) (like (16) is derived from (10) and (12))
- (23) $B(b, broken(s''), s'', s_2)$ from (22)

Regression

- (1') $B(m, \neg broken(s''), s'', s_4)$
- (1') \Leftrightarrow (2') $\forall s''(K(m, s'', s_4) \rightarrow \neg broken(s''))$ by definition of B
- (2') \Leftrightarrow (3') $\forall s''(K(m, s'', do(repair(m), s_3)) \rightarrow \neg broken(s''))$ by definition of s_4
- (4') $\forall s''(K(m, s'', do(repair(m), s_3)) \leftrightarrow \exists s'(K(m, s', s_3) \wedge ((observe(m, repair(m), s_3) \wedge s'' = do(repair(m), s')) \vee (\neg observe(m, repair(m), s_3) \wedge s'' = s'))))$ from (EK)
- (3') \Leftrightarrow (5') $\forall s''(\exists s'(K(m, s', s_3) \wedge ((observe(m, repair(m), s_3) \wedge s'' = do(repair(m), s')) \vee (\neg observe(m, repair(m), s_3) \wedge s'' = s'))))$

$s'' = s')$) $\rightarrow \neg broken(s'')$) from (4')

(5') \Leftrightarrow (6') $\forall s'' \forall s'$ (

$(K(m, s', s_3) \wedge observe(m, repair(m), s_3) \wedge s'' = do(repair(m), s') \rightarrow \neg broken(s'')) \wedge$

$(K(m, s', s_3) \wedge \neg observe(m, repair(m), s_3) \wedge s'' = s' \rightarrow \neg broken(s''))$)

(6') \Leftrightarrow (7') $\forall s' (K(m, s', s_3) \wedge observe(m, repair(m), s_3) \rightarrow \neg broken(do(repair(m), s')) \wedge$

$\forall s' (K(m, s', s_3) \wedge \neg observe(m, repair(m), s_3) \rightarrow \neg broken(s'))$) from Equality

axioms

(7') \Leftrightarrow (8') $(observe(m, repair(m), s_3) \rightarrow \forall s' (K(m, s', s_3) \rightarrow \neg broken(do(repair(m), s')))) \wedge$

$(\neg observe(m, repair(m), s_3) \rightarrow \forall s' (K(m, s', s_3) \wedge \rightarrow \neg broken(s')))$

(9') $real(s_3) \rightarrow \forall s' (K(m, s', s_3) \rightarrow \neg broken(do(repair(m), s')))$ from (SSA_m)

(8') \Leftrightarrow (10') $(observe(m, repair(m), s_3) \rightarrow real(s_3)) \wedge$

$(\neg observe(m, repair(m), s_3) \rightarrow B(m, \neg broken(s'), s', s_3))$ from (9')

This shows that the formula (1'), which is “about” s_4 , is a consequence of the regressed formula (10'), which is “about” s_3 .

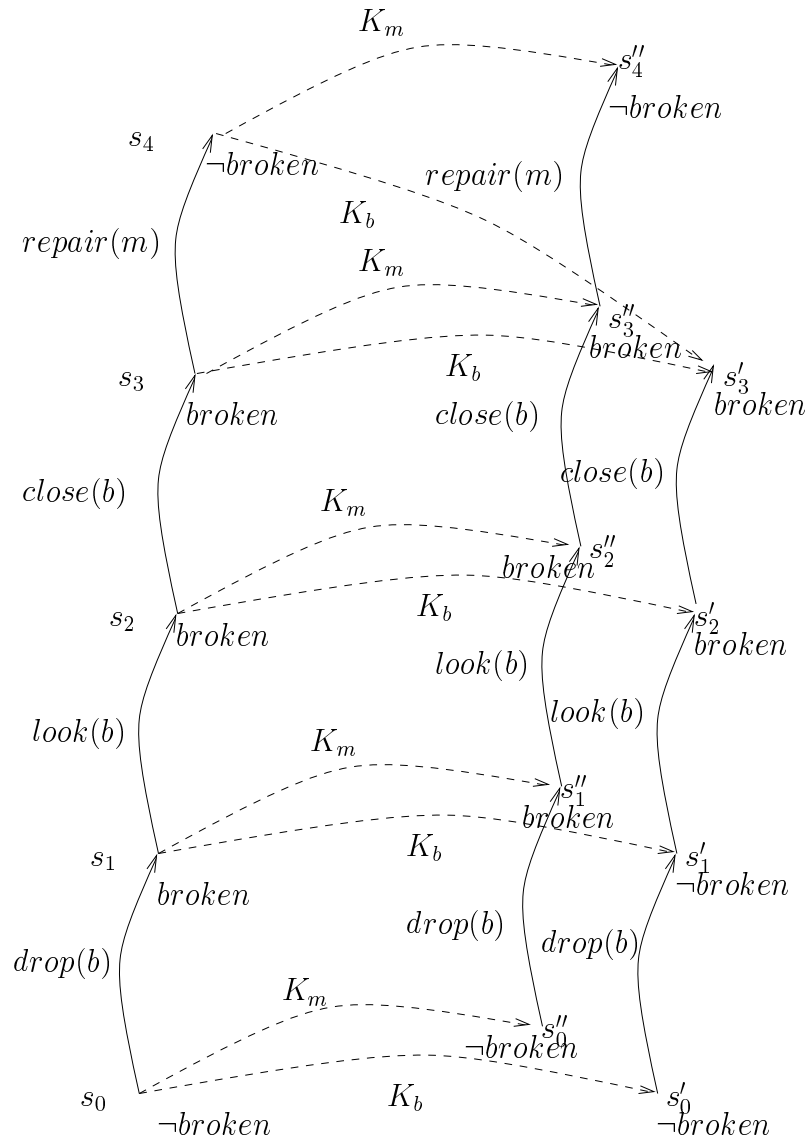


Figure 1: Mother and baby beliefs' evolution.