Implementing Belief Change in the Situation Calculus and an Application

Maurice Pagnucco [1] , David Rajaratnam [1] , Hannes Strass [2] , and Michael Thielscher [1]

[1] School of Computer Science and Engineering, UNSW, Australia
{morri,daver,mit}@cse.unsw.edu.au
[2] Leipzig University, Leipzig, Germany
strass@informatik.uni-leipzig.de

Abstract. Accounts of belief and knowledge in the Situation Calculus have been developed and discussed for some time yet there is no extant implementation. We develop a practical implementation of belief and belief change in the Situation Calculus based on default logic for which we have an implemented solver. After establishing the mapping with default logic we demonstrate how belief change in the Situation Calculus can be used to solve an interesting problem in robotics – reasoning with misleading information. Motivated by a challenge in the RoboCup@Home competition, we give a solution to the problem of planning robustly in cases where operators provide the robot with misleading or incorrect information.

1 Introduction

Several accounts of belief and knowledge in the Situation Calculus have been developed and discussed for some time [1–3] yet there is no extant implementation. In this paper, we show how belief and belief change in the Situation Calculus according to the account of [4, 1] can be implemented. We do so by mapping this formalisation of belief change in the Situation Calculus to an account of default logic for which we have an implemented solver [5].

As we establish this mapping into default logic, we demonstrate how belief change in the Situation Calculus can be used to solve an interesting problem in robotics – reasoning with misleading information. Motivated by a challenge in the RoboCup@Home competition, we give a solution to the problem of planning robustly in cases where operators provide the robot with misleading or incorrect information. What, for example, should a robot given the task of returning with the red cup from the kitchen table do when it arrives in the kitchen to find no red cup but instead notices a blue cup and a red plate on the table? In RoboCup@Home, the best course of action is not to return empty-handed but to attempt to salvage the situation by applying a form of commonsense preferences to return with one of the objects available. Our results pave the way for a practical and efficient solution to such problems.

The rest of the paper proceeds as follows. We first provide the technical background to understand the paper. Then we describe a formal specification of belief change in the Situation Calculus. A motivating example based on RoboCup@Home is introduced in order to demonstrate a challenging problem that can be solved using this account of belief change in the Situation Calculus. Next, we present another solution, that is based on an implementable fragment of prioritised default logic and show how belief change in the Situation Calculus can be translated into this logic. Finally, we show that the two solutions yield the same results, discuss our findings in a broader context and conclude.

2 Technical Preliminaries

2.1 Situation Calculus

The Situation Calculus provides a formal language based on that of classical first-order logic in which to describe dynamic domains [6, 7]. Three types of terms are distinguished: *situations* representing a snapshot of the world; *fluents* denoting domain properties that may change as a result of actions; and *actions* that can be performed by the reasoner. We use the predicate $Holds(f, s)$ to specify that a fluent f holds at a particular situation. As a matter of convention a short form is adopted such that for any n-ary fluent $f(x_1, \ldots, x_n)$, writing $f(x_1, \ldots, x_n, s)$ is a short form for $Holds(f(x_1, \ldots, x_n), s)$. A special function $do(a, s)$ represents the situation that results from performing action a at situation s. S_0 denotes the initial situation where no actions have taken place. For each action we need to specify preconditions $Poss(a, s)$ specifying the conditions under which action a is possible in situation s and effect axioms that specify how the value of a fluent changes when an action is performed.[3] For a more comprehensive formulation of what is required of a Situation Calculus basic action theory (BAT), the reader is referred to [7].

2.2 Iterated Belief Revision in the Situation Calculus

A request to an agent to achieve a goal affects its beliefs. For instance, when the agent is asked to collect the red cup from the kitchen table, it is reasonable for the agent to believe that there is in fact a red cup located on the kitchen table. We therefore adopt an extension to the Situation Calculus capable of representing beliefs. Several accounts exist [1–3] however we use that of Shapiro et al. [1]. It is based on the ideas of Moore and extended by Cohen and Levesque [8] who introduced knowledge into the Situation Calculus by reifying the accessibility relation in modal semantics for knowledge. Two types of actions are distinguished: *physical actions* which alter the world (and hence fluent values) when performed; and, *sensing actions* that are associated with a sensor and determine the value of a fluent (e.g., a vision system used to determine whether a red cup is on a

[3] In fact, we compile effect axioms into *successor state axioms* (SSAs) [7].

table). Sensing actions are also referred to as *knowledge producing* actions as they inform the reasoner about fluent values but do not alter the world state.

Scherl and Levesque [9] introduced the relation $B(s', s)$ denoting that if the agent were in situation s, it would consider s' to be possible.[4] This is adopted by Shapiro et al. [1]. The successor state axiom for the B relation is given in the table below as Axiom (B1) and states that s'' is possible at the situation resulting from performing action a at situation s whenever the sensing action associated with a agrees on its value at s and s'. $SF(a, s)$ is a predicate that is true whenever the sensing action a returns the sensing value 1 at s and was introduced by Levesque [10]. The innovation of Shapiro et al. [1] is to associate a plausibility with situations. Plausibility values are introduced, in decreasing order with a value of 0 being the most plausible, for initial situations and these values remain the same for all successor situations as expressed in Axiom (B2) below. This is critical for preserving the introspection properties for belief. The plausibility values themselves are not important, only the ordering over situations that they induce. Axioms (B3) and (B4) define the situations s' that are most plausible (MP) and most plausible situations that are possible – i.e., B-related – (MPB) at s, respectively. In Axiom (B5) we define sentence ϕ to be believed in situation s ($Bel(\phi, s)$) whenever it is true at all the most plausible situations that are possible at s. Finally, Axiom (B6) specifies that any situations B-related to an initial situation are also initial situations. The distinguished predicate $Init(s)$ indicates that s is an initial situation.

B1. $B(s'', do(a, s)) \equiv \exists s'[B(s', s) \wedge s'' = do(a, s') \wedge SF(a, s') \equiv SF(a, s))]$
B2. $pl(do(a, s)) = pl(s)$
B3. $MP(s', s) \stackrel{\text{def}}{=} \forall s''.B(s'', s) \supset pl(s') \leq pl(s'')$
B4. $MPB(s', s) \stackrel{\text{def}}{=} B(s', s) \wedge MP(s', s)$
B5. $Bel(\phi, s) \stackrel{\text{def}}{=} \forall s'.MPB(s', s) \supset \phi[s']$
B6. $Init(s) \wedge B(s', s) \supset Init(s')$

3 Formalisation in the Situation Calculus

The formalisation of our approach is based on the iterated belief revision extension to the Situation Calculus. Notably, the problem is specified in terms of *primitive fluents*, *primitive actions*, *sensing actions*, an *initial state*, *precondition axioms*, and *successor state axioms*. In order to deal with the potential for defeasible information we introduce a number of restrictions to this formalism. In essence these restrictions are designed to exploit the way in which abstract logical names can be anchored to the perception of actual objects in the environment.

Objects We require a fixed set I of individual objects, to which a unique names assumption is applied. Intuitively, they identify the items that a robot is trained to recognise. We introduce the fluent $Same(x, y)$ to express that two names refer

[4] Note the order of the arguments as it differs from that commonly used in modal semantics of knowledge.

to the same real object, and allow a set of additional names $N = \{O_1, \ldots, O_n\}$, ensuring that these names only refer to existing objects in the domain:

$$\bigvee_{A \in I} \mathit{Same}(O_i, A, s), \text{ for } 1 \leq i \leq n$$

Same is required to be reflexive, symmetric and transitive and is further axiomatised using "substitutivity" axioms to enforce that identical objects agree on all fluent properties F of the domain

$$\mathit{Same}(x, y, s) \supset (F(\bar{z}, s)[z_i/x] \equiv F(\bar{z}, s)[z_i/y])$$

and the SSA $\mathit{Same}(x, y, do(a, s)) \equiv \mathit{Same}(x, y, s)$. The trivial successor state axiom of this fluent reflects the intuition that hypotheses about names referring to objects will only be affected by knowledge-producing actions.

Informing the robot Informing the robot about the operator's belief in the state of the world is formalised outside of the underlying action calculus at the meta-level and is subsequently compiled into the initial state axioms.

Let f be a fluent literal. Then $\mathit{Told}(f, S_0)$, which we abbreviate as $\mathit{Told}(f)$, represents the act of the operator informing the robot about the operator's understanding of the initial state of the world. Additionally, object references in f must consist only of the names in N, reflecting the intuition that the operator may only ever refer to objects on the basis of their properties, but not by using their names. Finally, a set of operator commands T is *consistent* provided there is no fluent f such that $\mathit{Told}(f) \in T$ and $\mathit{Told}(\neg f) \in T$.

Setting goals Requests from the user for the robot to perform a task are required to be of the form $\mathit{Goal}(\exists s.\phi(s))$ where $\phi(s)$ is a sentence expressing the goal to be achieved. As with the operator commands, all objects referenced in $\phi(s)$ must be referred to only by the names in N.

Motivating Example: Dealing with Misleading Information

The following example will be used to illustrate our approach. It represents a reasonably practical example of moderate sophistication sufficient for the space available. Moreover it is of interest as it represents an instance of goal revision which can be innovatively handled by the account of belief change in the Situation Calculus that we adopt here.

The Robocup@Home (`robocupathome.org`) competition is an international initiative to foster research into domestic robots. Effective domestic robots must be able to perform tasks in response to user commands and to behave *robustly* if the information provided is in some way erroneous. This is demonstrated in the "General Purpose Service Robot" challenge of the Robocup@Home 2010 Competition,[5] and the following scenario is based on an example from this challenge.

[5] `http://www.robocupathome.org/documents/rulebook2010_FINAL_VERSION.pdf`

Scenario 1 *The robot is in the living room of the home. The home has a kitchen with a table in the middle. The robot is told to fetch the red cup from the kitchen table. However, there is no red cup on the kitchen table and the robot only discovers this fact once it arrives in the kitchen and looks for the cup on the table.*

We highlight two separate cases. In the base case there is only a blue cup on the table. In the extended case there is a blue cup and a red plate on the table.

While a robot cannot know the precise intentions of the human operator, it can nevertheless apply commonsense knowledge in its responses. In the first case, faced with no alternatives, it might simply fetch the blue cup. In the second case, it might assume that the user is more interested in the type of object than its colour and so would prefer the blue cup over the red plate.

For simplicity of presentation we provide a compact encoding of this scenario. In particular for binary properties we adopt only one of each binary pair, with the intuition that the negation of the given property implies that its pair must hold. For example, if an object is not a cup then it must be a plate.

Objects The Robocup@Home challenge deals with a fixed set of household objects that are determined at the start of the competition. This allows the teams time to train their vision systems to be able to detect and distinguish between these objects. In our example scenario, there are two cups, one red and one blue, and a red plate: $I = \{C_R, C_B, P_R\}$.

Primitive fluents The primitive fluents in our domain and their meanings are as follows. $\mathit{InKitchen}$: the robot is in the kitchen, $\mathit{Holding}(o)$: the robot is holding an object, $\mathit{OnTable}(o)$: the object is on the kitchen table, $\mathit{Cup}(o)$: the object is a cup, $\mathit{Red}(o)$: the object is red.

Primitive actions $\mathit{SwitchRoom}$: moving from the living room to the kitchen and vice-versa; $\mathit{PickUp}(o)$: pick up an object from the kitchen table.

Sensing The robot is trained to recognise the pre-determined set of objects I. The main sensing task is then to detect whether or not these specific objects are located on the kitchen table. This is encapsulated by the sensing action $\mathit{SenseOT}(o)$ that senses if object $o \in I$ is on the table. The $\mathit{SF}(a, s)$ predicate, introduced in the previous section, is used to axiomatise the act of sensing:

$$\mathit{SF}(\mathit{PickUp}(o), s) \equiv \mathit{true}$$
$$\mathit{SF}(\mathit{SwitchRoom}, s) \equiv \mathit{true}$$
$$\mathit{InKitchen}(s) \supset (\mathit{SF}(\mathit{SenseOT}(o), s) \equiv \mathit{OnTable}(o, s))$$

Initial state In the initial state the robot is in the living room (i.e., not in the kitchen) and is not holding anything: $\neg\mathit{InKitchen}(S_0) \wedge (\forall x)(\neg\mathit{Holding}(x, S_0))$.

Informing the robot The robot is told that there is a red cup on the table: $\mathit{Told}(\mathit{Cup}(O_1))$, $\mathit{Told}(\mathit{Red}(O_1))$, $\mathit{Told}(\mathit{OnTable}(O_1))$.

Precondition axioms The robot can only pick up an item when it is not already holding an object and the item in question is on the kitchen table: $\mathit{Poss}(\mathit{PickUp}(o), s) \equiv (\forall x)(\neg\mathit{Holding}(x, s)) \wedge \mathit{InKitchen}(s) \wedge \mathit{OnTable}(o, s)$; the robot can always switch locations: $\mathit{Poss}(\mathit{SwitchRoom}, s) \equiv \mathit{true}$.

Successor state axioms If the robot wasn't already in the kitchen then it will be as a result of switching rooms: $InKitchen(do(a,s)) \equiv (\neg InKitchen(s) \wedge a = SwitchRoom) \vee (InKitchen(s) \wedge a \neq SwitchRoom)$; an item will be on the table only if it was previously on the table and has not been picked up: $OnTable(o, do(a,s)) \equiv OnTable(o,s) \wedge a \neq PickUp(o)$; the robot will be holding an object if it picks it up or was already holding the object: $Holding(o, do(a,s)) \equiv a = PickUp(o) \vee Holding(o,s)$; object type is persistent: $Cup(o, do(a,s)) \equiv Cup(o,s)$; colour is persistent: $Red(o, do(a,s)) \equiv Red(o,s)$.

Preferences In order to use the framework for belief change in the Situation Calculus to deal with misleading information we proceed as follows. Every planning problem (i.e., request to achieve a goal) is considered a new reasoning problem.[6] The statements, $Told(f(\bar{x}))$ and $Goal(\exists s.\phi(s))$, are used to ascribe initial beliefs and a goal to achieve. They are interpreted at the meta-level and are not part of the object language. In our example scenario, the request $Told(Cup(O_1))$, $Told(Red(O_1))$, $Told(OnTable(O_1))$, $Goal(\exists s.Holding(O_1, s))$ asks the agent to collect a red cup from the table. This results in the specification of a reasoning about action problem in the Situation Calculus extended with beliefs. In particular, the request specifies what should be believed in the initial situation S_0 and as such partially restricts the plausibility relation $pl()$. However, our beliefs may be mistaken – there is no red cup on the table – and as a result we need to formulate an alternative course of action to get the best out of the situation at hand. Which alternative course of action to take is determined by preferences that are specified using a meta-level preference relation $<_C$. These preferences further restrict the plausibility of situations $pl()$.

Preferences reflect the robot's commonsense knowledge. In our scenario, for example, the robot may prefer to fetch an object that is of the same type as requested but of a different colour, and most of all prefer to find an object in the room to which it was sent.

$$OnTable <_C Cup <_C Red \tag{1}$$

It is of course possible to conceive of a scenario in which the above preference for, say, non-red cups in the kitchen over red non-cups elsewhere is reversed. The operator may be a child building a colour collage and therefore assign greater importance to the colour of the object than its type.

In reality, determining the best set of preferences would be a complex task requiring the robot to combine subtleties of natural language processing with specific knowledge about the operator. Such considerations are beyond the scope of this paper, and so we just presuppose a given *commonsense preference ordering*, represented by a partial order among fluent names.

Next, we directly compile the $Told()$ statements plus an ordering like (1) into a plausibility ordering over all the initial situations. Here, the initial situations encode all possible hypotheses of what the operator might have meant by their

[6] This is not crucial to our approach but considerably simplifies the notation and formal machinery required and, in any case, is not central to the main contributions.

commands. The commonsense preference is then used to rank these hypotheses according to their plausibility. In order to relate this preference ordering to the $Told()$ statements we introduce the notation $\langle\cdot\rangle$ to extract the fluent name from a fluent literal (e.g., $\langle\neg Cup(O_1)\rangle = Cup$).

Definition 1 *Let Σ be a Situation Calculus BAT, B the axioms for iterated belief revision in the Situation Calculus, I be the set of domain objects, N be a set of additional names, T be a set of consistent operator commands and $<_C$ be a commonsense preference ordering. Then $(\Sigma \cup B, T, <_C)$ is a Situation Calculus BAT extended with belief and commonsense preferences such that:*

1. *The initial situations are created by the axioms*

$$(\exists s)\left(B(s, S_0) \wedge \bigwedge_{O \in N} Same(O, \sigma(O), s)\right) \tag{2}$$

   for all functions $\sigma : N \to I$.
2. *For every pair of initial situations s_1, s_2, we define*

$$pl(s_1) < pl(s_2)$$

   iff both
   (a) there is some $Told(f(\bar{x})) \in T$ such that $\Sigma \cup \{(2)\} \models f(\bar{x}, s_1)$ and $\Sigma \cup \{(2)\} \models \neg f(\bar{x}, s_2)$; and,
   (b) for every $Told(f(\bar{x}_1)) \in T$ such that

$$\Sigma \cup \{(2)\} \models \neg f(\bar{x}_1, s_1) \text{ and } \Sigma \cup \{(2)\} \models f(\bar{x}_1, s_2)$$

   there is a $Told(g(\bar{x}_2)) \in T$ such that $\langle g\rangle <_C \langle f\rangle$,

$$\Sigma \cup \{(2)\} \models g(\bar{x}_2, s_1) \text{ and } \Sigma \cup \{(2)\} \models \neg g(\bar{x}_2, s_2)$$

Part 1 creates all the initial situations. Intuitively, the function σ says which names are assigned to which real object; so $\sigma_1(O_1) = \sigma_1(O_3) = C_R$ means that O_1 and O_3 are considered the same as the red cup C_R. The number of axioms thus generated is polynomial in the number of domain objects, but exponential in the number of additional names.[7] This is one of the main reasons why a direct implementation of Situation Calculus with belief change would be practically infeasible for our problem at hand and why we are interested in developing a more practical implementation based on default logic.

Part 2 restricts the plausibility relation over initial situations. Initial situation s_1 is preferred to s_2 whenever s_1 assigns the value true to a fluent preferred under the preference ordering $<_C$ and s_2 assigns the value false; additionally, for all fluents f where this is the other way around (f is true in s_2 and false in s_1), there must be a preferred fluent g which holds in s_1 but not in s_2.

From this formalisation of the scenario, we can establish the fact that the robot will initially believe what it is told.

[7] Recall that the number of functions $\sigma : N \to I$ is $|I|^{|N|}$.

Proposition 1. *Let* Σ *be a Situation Calculus BAT,* B *the axioms for iterated belief revision in the Situation Calculus,* T *be a set of consistent operator commands, and* $<_C$ *be a commonsense preference ordering. Then* $(\Sigma \cup B, T, <)$ *is a Situation Calculus BAT extended with belief and commonsense preferences such that for all* $Told(f(\bar{x})) \in T$ *we have* $(\Sigma \cup B, T, <_C) \models Bel(f(\bar{x}), S_0)$ *.*

Plan Execution This formalism allows the robot to change its beliefs about what it is told. In this paper we assume that the robot has determined a plan and begun its execution. We can therefore consider the robot's changing beliefs with regards to satisfying its goal of holding object O_1 by considering the situation[8]

$$do([SwitchRoom, SenseOT(C_R), SenseOT(C_B), SenseOT(P_R), PickUp(O_1)], S_0)$$

Initially the robot believes that the object O_1 refers to the red cup C_R . However when the robot arrives in the kitchen it finds that there is only a blue cup on the table. Consequently the robot changes its belief about O_1 to now refer to the blue cup C_B . This scenario is visualised by Figure 1 showing the possible situations based on the robot's beliefs and the plausibility relation.

In the extended example the robot arrives in the kitchen to find both a blue cup and red plate on the table. It therefore has a choice, which it resolves based on its preference for object type over colour (1), consequently modifying its belief about O_1 to again refer to the blue cup.

4 A Default Logic Approach

The Situation Calculus with beliefs provides an expressive formalism for tackling the problem of agents receiving erroneous information and expected to use some basic commonsense reasoning under these circumstances. Next we address the problem of turning the theory into a practical implementation. To this end we adapt a recently developed extension of action logics with default reasoning [11], which can be effectively implemented using Answer Set Programming [12]. The idea is to treat potentially erroneous information as something that is considered true by default but can always be retracted should the agent make observations to the contrary. We extend the existing approach by *prioritised* defaults that allow us to provide our robot with preferences among different ways of remedying a situation in which it has been misled.

Supernormal Defaults To begin with, we instantiate the general framework of [11] to the Situation Calculus and to a restricted form of default rules. Each operator command $Told([\neg]f(\bar{x}), s)$ is translated into a *supernormal* default of the form

$$\frac{: f(\bar{x}, s)}{f(\bar{x}, s)} \quad \text{or} \quad \frac{: \neg f(\bar{x}, s)}{\neg f(\bar{x}, s)}$$

[8] $do([a_1, \ldots, a_n], s)$ abbreviates $do(a_n, \ldots, do(a_2, do(a_1, s)) \ldots)$.

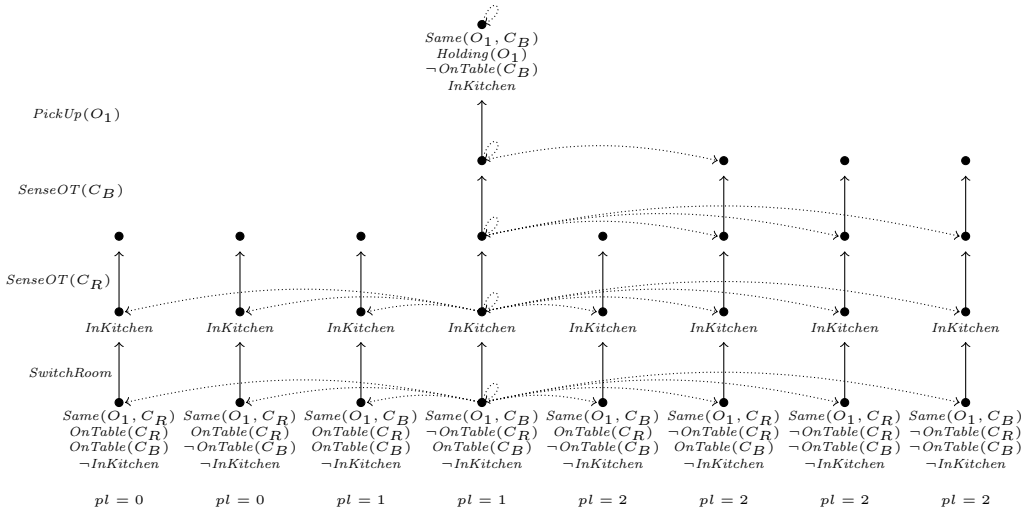

Fig. 1. The robot is told to pick up the red cup from the table, but finds only a blue cup. For succinctness, details of the red plate and the status of the persistent fluents *Cup* and *Red* are omitted. Furthermore, only the accessibility relations (dotted lines) for the actual situation (fourth from the left) are shown. The transition of situations based on actions are indicated by the solid vertical lines. Values for the plausibility relation are assigned to the initial situations based on the preferences. The initial situations in which the robot believes that it is going to pick up the red cup on the table are the most preferred ($pl = 0$). Next are those in which the robot believes that it is going to pick up the blue cup on the table ($pl = 1$). Finally, the least preferred are situations where the robot believes that the item to pick up is not on the table ($pl = 2$).

With these rules the robot will believe, by default, everything it is told. For our running example we thus obtain these three defaults about the initial situation:

$$\delta_{Cup} = \frac{: \ Cup(O_1, S_0)}{Cup(O_1, S_0)} \quad \delta_{Red} = \frac{: \ Red(O_1, S_0)}{Red(O_1, S_0)} \quad \delta_{OnTable} = \frac{: \ OnTable(O_1, S_0)}{OnTable(O_1, S_0)}$$

A *Situation Calculus default theory* is a pair (Σ, Δ) where Σ is aSituation Calculus BAT as above and Δ is a set of default rules.

Priorities In a *prioritised default theory* [13], the default rules are partially ordered by $\prec$, where $\delta_1 \prec \delta_2$ means that the application of default δ_1 is preferred over the application of δ_2. For our purpose, we can map a given commonsense preference ordering among fluent names directly into a partial ordering among the defaults from above. For example, with the ordering given by (1) we obtain $\delta_{OnTable} \prec \delta_{Cup} \prec \delta_{Red}$. A *prioritised Situation Calculus default theory* is a triple $(\Sigma, \Delta, \prec)$ where (Σ, Δ) is as above and $\prec$ is a partial ordering on Δ.

Extensions Reasoning with default theories is based on the concept of so-called *extensions*, which can be seen as a way of assuming as many defaults as possible without creating inconsistencies [14, 13].

Definition 2 *Consider a prioritised Situation Calculus default theory* $(\Sigma, \Delta, \prec)$. *Let* E *be a set of formulas and define* $E_0 := Th(\Sigma)$ *and, for* $i \geq 0$,

$$E_{i+1} := Th(E_i \cup \{\gamma \mid \frac{:\gamma}{\gamma} \in \Delta,\ \neg\gamma \notin E\})$$

Then E *is an* extension *of* $(\Sigma, \Delta, \prec)$ *iff* $E = \bigcup_{i \geq 0} E_i$.
Let a partial ordering be defined as $E_1 \lessdot E_2$ *iff both*

(a) there is $\frac{:\gamma}{\gamma}$ *in* Δ *such that* $\gamma \in E_1$ *but* $\gamma \notin E_2$*; and,*
(b) for every $\frac{:\gamma_1}{\gamma_1}$ *such that* $\gamma_1 \notin E_1$ *but* $\gamma_1 \in E_2$ *there is* $\frac{:\gamma_2}{\gamma_2} \prec \frac{:\gamma_1}{\gamma_1}$ *in* Δ *such that* $\gamma_2 \in E_1$ *but* $\gamma_2 \notin E_2$.

Extension E *is a* preferred extension *of* $(\Sigma, \Delta, \prec)$ *iff there is no* E' *such that* $E' \lessdot E$. *Entailment* $(\Sigma, \Delta, \prec) \mid\!\approx \phi$ *is defined as* ϕ *being true in all preferred extensions.*

In our running example, when initially the robot has no information to the contrary it can consistently apply all defaults, resulting in a unique preferred extension that entails $Cup(O_1, S_0) \wedge Red(O_1, S_0) \wedge OnTable(O_1, S_0)$. Based on these default conclusions the Situation Calculus axioms entail the same plans for a given goal as those for the Situation Calculus extended with belief and commonsense preferences. But suppose that the robot enters the kitchen and observes what is indicated in Figure 1, that is,

$$\begin{array}{l} Same(O_1, C_R, S) \vee Same(O_1, C_B, S) \\ Cup(C_R, S) \wedge Red(C_R, S) \\ Cup(C_B, S) \wedge \neg Red(C_B, S) \\ \neg OnTable(C_R, S) \wedge OnTable(C_B, S) \end{array}$$

where S is the situation after $SwitchRoom$ followed by $SenseOT(C_R)$ and $SenseOT(C_B)$. Disregarding priorities for now, there are two extensions, characterised by

$$\begin{array}{ll} \{Same(O_1, C_R, S), \neg OnTable(O_1, S)\} & \subseteq E_1 \\ \{Same(O_1, C_B, S), \neg Red(O_1, S)\} & \subseteq E_2 \end{array}$$

However, given the priorities from above, only E_2 is a preferred extension, triggering the robot to pick up the blue cup.

In the second case of the scenario, the robot further senses that there is also a red plate on the table. In this case there will be a third extension E_3 such that $\{Same(O_1, P_B, S), \neg Cup(O_1, S)\} \subseteq E_3$. However, as with the first case, E_2 is still the only preferred extension and therefore the robot selects the blue cup.

Implementation Answer Set Programming (ASP) [12] is well-suited for efficiently implementing nonmonotonic reasoning formalisms. Extended logic programs can be seen as special kinds of default theories [15] and this correspondence can be used to transform a default theory into an answer set program. Entailment of a formula by the default theory can then be determined by querying the answer set program. This transformation technique has been developed in [5]. In the following, we outline this technique (steps 2 to 4) and extend it to

cover preferences (step 1). This allows the transformation of a sufficiently restricted prioritised Situation Calculus default theory $(\Sigma, \Delta, \prec)$ into an answer set program $P_{\Sigma,\Delta,\prec}$.

Step 1. We transform the *prioritised* Situation Calculus default theory $(\Sigma, \Delta, \prec)$ into a Situation Calculus default theory $(\Sigma^{\prec}, \Delta^{\prec})$ where the preferences have been encoded at the object-level [16]. This is done by explicitly keeping track of default δ's meta-level applicability $\mathsf{ok}(\delta)$ and whether it was applied ($\mathsf{ap}(\delta)$) or blocked ($\mathsf{bl}(\delta)$). For example, δ_{Cup} and δ_{Red} are transformed into

$$\frac{\mathsf{ok}(\delta_{Cup}) : Holds(Cup(O_1), S_0)}{Holds(Cup(O_1), S_0) \wedge \mathsf{ap}(\delta_{Cup})} \qquad \frac{\mathsf{ok}(\delta_{Cup}) \wedge \neg Holds(Cup(O_1), S_0) :}{\mathsf{bl}(\delta_{Cup})}$$

$$\frac{\mathsf{ok}(\delta_{Red}) : Holds(Red(O_1), S_0)}{Holds(Red(O_1), S_0) \wedge \mathsf{ap}(\delta_{Red})} \qquad \frac{\mathsf{ok}(\delta_{Red}) \wedge \neg Holds(Red(O_1), S_0) :}{\mathsf{bl}(\delta_{Red})}$$

The preference between the defaults is enforced by statements like $(\mathsf{ap}(\delta_{Cup}) \vee \mathsf{bl}(\delta_{Cup})) \supset \mathsf{ok}(\delta_{Red})$, effectively saying that δ_{Red} can only be applied once it is clear whether the more preferred default δ_{Cup} has been "processed".

Step 2. We instantiate the defaults from $\Delta^{\prec}$ and the axioms from $\Sigma^{\prec}$ for the given Situation Calculus signature. This yields a propositional default theory.

Step 3. We rewrite the ground instantiation of $\Sigma^{\prec}$ into a set $P_{\Sigma^{\prec}}$ of extended logic program rules.

Step 4. We map $\Delta^{\prec}$ into a set of logic program rules. A default of the form $\frac{p:q}{r_1 \wedge r_2}$ becomes $\mathtt{r_i} \leftarrow \mathtt{p},\ \mathtt{not}\ {-}\mathtt{q}$ for $i = 1, 2$; a rule $\frac{p \wedge q:}{r}$ is turned into $\mathtt{r} \leftarrow \mathtt{p},\ \mathtt{q}$. Here $\mathtt{not}$ is the usual nonmonotonic negation of normal logic programs; ${-}\mathtt{q}$ is a new predicate symbol standing for the (classical) negation of $\mathtt{q}$ [15]. The resulting rules together with $P_{\Sigma^{\prec}}$ now form the corresponding answer set program $P_{\Sigma,\Delta,\prec}$ of the initial prioritised Situation Calculus default theory $(\Sigma, \Delta, \prec)$.

5 Equivalence of the Two Approaches

We are now in a position to state the central result of this paper, which says that our prioritised Situation Calculus default theories are suitable approximations of the Situation Calculus extended with belief and commonsense preferences. Unfortunately, lack of space prevents us from giving a rigorously formal account. Generally speaking, the latter is more expressive for two reasons. First, it allows to infer meta-statements about beliefs, as in the formula $Bel(Bel(Red(O_1), S_0), do(SwitchRoom, S_0))$. Second, all possible situations are ranked according to $pl()$, thus allowing to draw conclusions about their relative ordering, whereas in prioritised default logics the non-preferred extensions are not considered for entailment. However, neither of these two features is relevant for the problem at hand, and we can prove the following.

Theorem 1. *Let Σ be a Situation Calculus BAT, B the axioms for iterated belief revision in the Situation Calculus, T a set of consistent operator commands, $<_C$ a commonsense preference ordering, $\Delta, \prec$ a set of default rules and an ordering as explained above, $a_1, \ldots, a_n$ a sequence of actions,*

and $SF_n := \{[\neg]SF(a_1, S_0), \ldots, [\neg]SF(a_n, do([a_1, \ldots, a_{n-1}], S_0))\}$ *a set of literals describing a particular sequence of sensing results. Then for any objective formula* ϕ *(that is, any formula* ϕ *without* Bel*) we find*

$(\Sigma \cup B \cup SF_n, T, <_C) \models Bel(\phi, do([a_1, \ldots, a_n], S_0))$
iff $(\Sigma \cup SF_n, \Delta, \prec) \mid\!\approx Holds(\phi, do([a_1, \ldots, a_n], S_0))$

Proof (sketch): By induction on the number of actions n. If $n = 0$, by Proposition 1 the robot believes all operator commands; in a similar way it can be shown that there is a unique preferred extension which entails the exact same statements about S_0 that are true in all most plausible initial situations. For the induction step, if a_{n+1} is a physical action the claim follows from the fact that both axiomatisations share the same basic action theory. If a_{n+1} is a sensing action, then any possible situation in $do([a_1, \ldots, a_n], S_0)$ that contradicts $[\neg]SF(a_{n+1}, do([a_1, \ldots, a_n, a_{n+1}], S_0)$ is no longer possible in $do([a_1, \ldots, a_n, a_{n+1}], S_0)$; likewise, any extension of $(\Sigma \cup SF_n, \Delta, \prec)$ that contradicts this sensing literal is no longer an extension of $(\Sigma \cup SF_{n+1}, \Delta, \prec)$. The claim follows from the structural equivalence of the construction of the plausibility ordering in Def. 1 (Item 2) and of preferred extensions in Def. 2. □

6 Conclusions

We developed an effective implementation of a well established approach to belief change in the Situation Calculus [4, 1]. This was achieved by mapping a problem instance expressed using this particular approach to belief change in the situation calculus into a default logic theory for which an ASP based implementation exists [11]. We illustrated our approach using an example inspired by the RoboCup@Home rulebook. This example innovatively solves the problem of how a reasoner faced with an unachievable goal should nevertheless do its best to salvage the situation by relying on its preferences.

It is important to observe that while our example scenario encodes a user request to fetch a single item, the formalism allows for more complex cases, such as conjunctive and disjunctive goals. However care must be taken when formulating requests. For example, a disjunctive request to fetch a fork or a spoon should be encoded as a request to fetch one of two distinct objects, a spoon object or a fork object. The alternative, and less intuitive, encoding would be to fetch a single object for which the operator is unsure if it is a fork or a spoon. This latter encoding is not possible due to restrictions on the *Told* statements.

We formalised our solution using an extension of the Situation Calculus to handle beliefs and mapped this solution into a solvable default logic theory. An alternative approach to tackling the example we presented would have been to consider goal revision [17]. However note that proposals like this one modify goals at the explicit request of an agent and do not consider that the goals themselves may be unachievable. In our approach, the goal cannot be achieved and we argue that this is more accurately dealt with by reasoning about the robot's beliefs (i.e., expectations about the world).

In related work, Lee and Palla [18] implement the situation calculus in ASP. However, adding the belief axioms of our paper to their approach would entail

explicitly representing all possible alternative situations (since their plausibilities matter). Our approach avoids this technical problem by using default logic where only preferred extensions are considered for entailment.

Acknowledgements This research was supported under Australian Research Council's (ARC) *Discovery Projects* funding scheme (project DP 120102144). The fourth author is the recipient of an ARC Future Fellowship (FT 0991348) and is also affiliated with the University of Western Sydney.

References

1. Shapiro, S., Pagnucco, M., Lespérance, Y., Levesque, H.: Iterated belief change in the situation calculus. AIJ **175**(1) (2011) 165–192
2. Demolombe, R., Pozos-Parra, M.P.: A simple and tractable extension of situation calculus to epistemic logic. In: ISIS-00. Volume LNAI 1932. (2000) 515–524
3. Demolombe, R., Pozos-Parra, M.P.: Belief change in the situation calculus: A new proposal without plausibility levels. In: Proc. of the Workshop on Belief Revision and Dynamic Logic at ESSLLI. (2005)
4. Shapiro, S., Pagnucco, M., Lespérance, Y., Levesque, H.: Iterated belief change in the situation calculus. In: KR. (2000) 527–538
5. Strass, H.: The draculasp system: Default reasoning about actions and change using logic and answer set programming. In: NMR. (2012)
6. McCarthy, J.: Situations, actions and causal laws. Stanford AI Project Memo 2 (1963)
7. Reiter, R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. The MIT Press (2001)
8. Cohen, P., Levesque, H.: Rational interaction as the basis for communication. In Cohen, P., Morgan, J., Pollack, M., eds.: Intentions in Communication. MIT Press (1990) 221–256
9. Scherl, R., Levesque, H.: Knowledge, action, and the frame problem. AIJ **144**(1–2) (2003) 1–39
10. Levesque, H.: What is planning in the presence of sensing? In: AAAI. (1996) 1139–1146
11. Baumann, R., Brewka, G., Strass, H., Thielscher, M., Zaslawski, V.: State defaults and ramifications in the unifying action calculus. In: KR. (2010) 435–444
12. Gelfond, M.: Answer Sets. In: Handbook of KR. (2008) 285–316
13. Brewka, G.: Adding priorities and specificity to default logic. In: JELIA-94. Volume 838 of LNAI. (1994) 247–260
14. Reiter, R.: A logic for default reasoning. AIJ **13** (1980) 81–132
15. Gelfond, M., Lifschitz, V.: Classical Negation in Logic Programs and Disjunctive Databases. New Gen. Comp. **9** (1991) 365–385
16. Delgrande, J., Schaub, T.: Expressing preferences in default logic. AIJ **123**(1–2) (2000) 41–87
17. Shapiro, S., Lespérance, Y., Levesque, H.: Goal change. In: IJCAI-05. (2005) 582–588
18. Lee, J., Palla, R.: Situation Calculus as Answer Set Programming. In: Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI-10). (July 2010) 309–314