Goal Change

Steven Shapiro
Computer Science Inst.
University of Leipzig
Leipzig 04109
Germany
shapiro@informatik.uni-leipzig.de

Yves Lespérance
Dept. of Computer Science
York University
Toronto, ON
M3J 1P3 Canada
lesperan@cs.yorku.ca

Hector J. Levesque
Dept. of Computer Science
University of Toronto
Toronto, ON
M5S 3G4 Canada
hector@cs.toronto.edu

Abstract

Although there has been much discussion of belief change (e.g., [Gärdenfors, 1988; Spohn, 1988]), goal change has not received much attention. In this paper, we propose a method for goal change in the framework of Reiter's [2001] theory of action in the situation calculus [McCarthy and Hayes, 1969; Levesque *et al.*, 1998], and investigate its properties. We extend the framework developed by Shapiro *et al.* [1998] and Shapiro and Lespérance [2001], where goals and goal expansion were modelled, but goal contraction was not.

1 Introduction

One of the commonsense abilities people possess is the ability to predict the behaviour of others. We attribute goals and beliefs to others and assume they will act on their beliefs to achieve their goals. One strategy for endowing machines with a similar ability is to explicitly model the beliefs and goals of agents and assume they are acting to achieve their goals in accordance with their beliefs. In dynamic environments, the beliefs and goals of agents are likely to change. Although there has been much discussion of belief change (e.g., [Gärdenfors, 1988; Spohn, 1988]), goal change has not received much attention. In this paper, we propose a method for specifying and reasoning about goal change in the framework of Reiter's [2001] theory of action in the situation calculus [McCarthy and Hayes, 1969; Levesque *et al.*, 1998]. We thus inherit Reiter's solution to the frame problem, and iterated goal change is achieved seamlessly using sequences of actions. Our approach to goal contraction is simpler than most approaches to belief revision and relies on using the history of requests rather than a specification of entrenchment. This work extends the framework developed by Shapiro *et al.* [1998] and Shapiro and Lespérance [2001], where goals and goal expansion were modelled but goal contraction was not.

We model agents that adopt goals if they are requested to do so and they do not already have a conflicting goal. They maintain their goals unless they are explicitly requested to drop them by the agent who requested the goal in the first place. We first define goals using an accessibility relation, W, which picks out the situations that the agent wants to be the case. We then give a successor state axiom for W that is affected by two communicative actions: REQUEST actions, which cause agents to adopt goals, and CANCELREQUEST actions, which cause agents to drop goals that had been adopted following prior requests. We then consider some properties of our axiomatization: expansion, contraction, and persistence. Finally, we identify the restrictions on the accessibility relations that give us positive and negative introspection of goals, and show that these restrictions persist, if they are asserted of the initial situations.

The remainder of the paper is organized as follows. In Sec. 2, we give a brief review of the situation calculus, Reiter's theory of action, and Shapiro *et al.*'s [1998] framework for representing knowledge expansion in the situation calculus. In Sec. 3, we introduce our framework for goal change, and in Sec. 4, we investigate properties of this framework. In Sec. 5, we present an example, and in Sec. 6, we conclude, discuss related work, and suggest avenues for future work.

2 Representation of Action and Knowledge

The basis of our framework for goal change is an action theory [Reiter, 2001] based on the situation calculus [McCarthy and Hayes, 1969; Levesque *et al.*, 1998]. The situation calculus is a predicate calculus language for representing dynamically changing domains. A situation represents a possible state of the domain. There is a set of initial situations corresponding to the ways the agents believe the domain might be initially. The actual initial state of the domain is represented by the distinguished initial situation constant, S_0. The term $do(a, s)$ denotes the unique situation that results from the agent doing action a in situation s. Thus, the situations can be structured into a set of trees, where the root of each tree is an initial situation and the arcs are actions. The sequence of actions that produces a situation is called the *history* of the situation. $s \preceq s'$ ($s \prec s'$, resp.) means there is a (nonempty, resp.) path from situation s to situation s'. The initial situations are defined as those having an empty history: $Init(s) \stackrel{\text{def}}{=} \neg\exists a, s'.s = do(a, s')$.

Predicates and functions whose value may change from situation to situation (and whose last argument is a situation) are called *fluents*. For instance, we might use the fluent

$\textsc{InRoom}(Agt, R_1, S)$ to represent the fact that agent *Agt* is in room R_1 in situation S. The effects of actions on fluents are defined using successor state axioms [Reiter, 2001], which provide a succinct representation for both effect axioms and frame axioms [McCarthy and Hayes, 1969].

We will be quantifying over formulae, so we assume that we have an encoding of formulae as first-order terms (we can adapt De Giacomo *et al.*'s [2000] axioms for this purpose, but we omit the details here). Therefore, we will freely quantify over formulae here. To simplify notation, we ignore the details of the encoding and use formulae directly instead of the terms that represent them. We use ψ to denote a formula that can contain the situation constants *Now* and *Then*, and ϕ to denote a formula that can contain *Now* but not *Then*. We use $\phi[s]$ ($\psi[s, s']$, resp.) to denote the formula that results from substituting s for *Now* in ϕ (s for *Now* and s' for *Then*, resp.).

To axiomatize a dynamic domain in the situation calculus, we use Reiter's [2001] action theory, which consists of (1) successor state axioms for each fluent (including K and W introduced below); (2) initial state axioms, which describe the initial state of the domain and the initial mental states of the agents; (3) precondition axioms, which specify the conditions under which the distinguished predicate $Poss(A, S)$ holds, i.e., when it is physically possible to execute an action A in situation S; (4) unique names axioms for the actions, and domain-independent foundational axioms (we adopt the ones given by Levesque *et al.* [1998] which accommodate multiple initial situations, but we do not describe them further here); and (5) the axioms to encode formulae as terms.

Moore [1985] defined a possible-worlds semantics for a logic of knowledge in the situation calculus by treating situations as possible worlds. Scherl and Levesque [2003] adapted this to Reiter's theory of action. Scherl and Levesque gave a successor state axiom for K that states how actions, including sensing actions, affect knowledge. Shapiro *et al.* [1998] adapted this axiom to handle multiple agents and INFORM actions, which we adopt here. $K(agt, s', s)$ will be used to denote that in situation s, agt thinks that situation s' might be the actual situation. Note that the order of the situation arguments is reversed from the convention in modal logic for accessibility relations. $\textsc{inform}(infr, agt, \phi)$ is the action of the agent $infr$ informing another agent, agt, that ϕ holds. Here is Shapiro *et al.*'s [1998] successor state axiom for K.[1]

Axiom 2.1

$$
\begin{aligned}
&K(agt, s'', do(a, s)) \equiv \\
&\exists s'.K(agt, s', s) \land s'' = do(a, s') \land Poss(a, s') \land \\
&\quad \forall infr, \phi.a = \textsc{inform}(infr, agt, \phi) \supset \phi[s']
\end{aligned}
$$

First note that for any action other than an INFORM action directed towards agt, the specification ensures that the only change in knowledge that occurs in moving from s to $do(a, s)$ is that it is known (by all agents) that the action a has been successfully executed. This is true because the K-alternative[2] situations to $do(a, s)$ are the situations that result from doing a in a K-alternative situation to s where a is possible to execute (denoted by $Poss(a, s')$). For the action $\textsc{inform}(infr, agt, \phi)$, the idea is that in moving from s to $do(\textsc{inform}(infr, agt, \phi), s)$, agt not only knows that the action has been executed (as above), but it also knows that ϕ holds. In this case, the K-alternative situations to $do(a, s)$ for agt are the situations that result from performing a in a K-alternative situation to s where a is possible to execute, except the ones where ϕ is false.[3] As usual, we say that an agent knows a formula ϕ in s, if ϕ holds in all situations K-accessible from s, i.e., $\mathbf{Know}(agt, \phi, s) \stackrel{\text{def}}{=} \forall s'.K(agt, s', s) \supset \phi[s']$.

Following Shapiro *et al.* [1998], we say that an agent can only execute an inform action if it knows that the content of the action is true.

Axiom 2.2

$$Poss(\textsc{inform}(infr, agt, \phi), s) \equiv \mathbf{Know}(infr, \phi, s).$$

Since we are dealing with knowledge here, not belief, we assert that K is (initially) reflexive.[4]

Axiom 2.3 $Init(s) \supset K(agt, s, s)$.

Following Scherl and Levesque, we also assert that initial situations can only be K-related to other initial situations. We use an initial state axiom for this purpose:

Axiom 2.4 $K(agt, s', s) \land Init(s) \supset Init(s')$.

This axiom and the successor state axiom for K imply that only situations with the same history can be K-related. Introspection will be discussed in Sec. 4.

3 Goal Change

Just as an agent's beliefs can change, its goals should also be able to change. For example, if an agent's owner requests it to do something, the agent's goals should change to reflect the request. We extend the framework developed by Shapiro *et al.* [1998] and Shapiro and Lespérance [2001], where goals and goal expansion were modelled but not goal contraction.

Following Cohen and Levesque [1990], we model goals using an accessibility relation over possible worlds (situations,

[1] Since we are using terms for formulae in our underlying representation, $\textsc{inform}(infr, agt, \phi) = \textsc{inform}(infr, agt, \phi')$, only if ϕ and ϕ' are the same term. It would not be difficult to modify our representation so that the equality holds only if ϕ and ϕ' are the representations of equivalent formulae. We adopt the convention that unbound variables are universally quantified in the widest scope.

[2] We say that S' *is a* K*-alternative situation to* S *for Agt* or S' *is* K*-related to* S *for Agt*, if $K(Agt, S', S)$ holds. We may drop *Agt* when it is understood from the context.

[3] Note that here all agents are aware of all actions that occur, including inform actions, so we have a broadcast model of communication. In Shapiro and Lespérance [2001], encrypted speech acts were introduced, ensuring that only the intended recipient of a message could know its contents.

[4] We do not handle belief revision here, but a treatment of belief revision in our framework was discussed in [Shapiro *et al.*, 2000].

in our case). The accessible worlds are the ones that are compatible with what the agent *wants* to be the case. Cohen and Levesque's accessibility relation for goals, G, was a subset of their accessibility relation for beliefs. This constraint precludes an agent wanting something that it believes is impossible (unless its goals are inconsistent). In our case, we define G in terms of a more primitive accessibility relation, which we call W and is intended to model what the agent wants independently of what it knows. We can then define Cohen and Levesque's goal accessibility relation, G, to be the intersection[5] of W and K.

An agent's goals are future oriented. For example, an agent might want some property to hold eventually, i.e., the agent's goal is of the form **Eventually**(ψ), for some formula ψ. We evaluate formulae such as these with respect to a path of situations rather than a single situation, and we call such formulae *path formulae*. Cohen and Levesque used infinite time-lines to evaluate such formulae, but for simplicity, we evaluate path formulae with respect to finite paths of situations which we represent by pairs of situations, $(\mathit{Now}, \mathit{Then})$, such that $\mathit{Now} \preceq \mathit{Then}$. *Now* corresponds to the "current time" on the path of situations defined by the sequence of situations in the history of *Then*. Path formulae may contain two situation constants, *Now* and *Then*. For example, $\exists r.\mathrm{INROOM}(\mathrm{JOHN}, r, \mathit{Now}) \wedge \neg \mathrm{INROOM}(\mathrm{JOHN}, r, \mathit{Then})$ could be used to denote the goal that John eventually leaves the room he is in currently.

Our W relation, like our K relation, is a relation on situations. While K only relates situations with the same histories, W can relate situations with different histories. Intuitively, $W(\mathit{Agt}, S', S)$ holds if in situation S, *Agt* considers that in S' everything that it wants to be true is actually true. For example, if the agent wants to become a millionaire in S, then in all situations W-related to S, the agent is a millionaire, but these situations can be arbitrarily far in the future.

Recall that the situations in the W accessibility relation are the ones that the agent wants to actualize independently of what it knows. Following Cohen and Levesque, we want the goals of the agent to be compatible with what it knows. The situations that the agent wants to actualize should be on a path from a situation that the agent considers possible. Therefore, the situations that will be used to determine the goals of an agent will be the W-accessible situations that are also compatible with what the agent knows, in the sense that there is K-accessible situation in their history. We will say that S' $K_{\mathit{Agt},S}$-*intersects* S'' if $K(\mathit{Agt}, S'', S)$ and $S'' \preceq S'$. We will suppress *Agt* or S if they are understood from the context. We define the goals of agt in s to be those formulae that are true in all the situations s' that are W-accessible from s and that K-intersect some situation, s'':

$$\mathbf{Goal}(agt, \psi, s) \stackrel{\text{def}}{=} \forall s', s''. W(agt, s', s) \wedge K(agt, s'', s) \wedge s'' \preceq s' \supset \psi[s'', s'].$$

Note that s'' corresponds to the "current situation" (or the current time) in the path defined by s'.

[5] As we will see below, the operation used on W and K to obtain G is not quite intersection, but a related operation.

As noted by Konolige and Pollack [1993], one has to be careful when using this definition of a goal. Suppose an agent has a conjunctive goal, $\psi \wedge \psi'$. According to this definition of **Goal**, both ψ and ψ' are also goals. Therefore, we could imagine a rational agent working to achieve one of the conjuncts as a subgoal. But it is easy to think of circumstances where achieving only one component of a conjunctive goal is undesirable. For example, I may have as a goal to be holding the bomb *and* that the bomb be defused. However, I want these to be true simultaneously and I do not have the goal to be holding the bomb if it is not also defused. As Konolige and Pollack did for their intention modality (I), we consider formulae that are an agent's "only goals", i.e., formulae that are true in all and *only* the W-paths:

$$\begin{array}{l}\mathbf{OGoal}(agt, \psi, s) \stackrel{\text{def}}{=} \\ \quad \forall s', s''. K(agt, s'', s) \wedge s'' \preceq s' \supset \\ \qquad\qquad (W(agt, s', s) \equiv \psi[s'', s']).\end{array}$$

We now give the successor state axiom for W. $W^+(agt, a, s', s)$ ($W^-(agt, a, s', s)$, resp.), which is defined below, denotes the conditions under which s' is added to (dropped from, resp.) W due to action a:

Axiom 3.1

$$\begin{array}{l} W(agt, s', do(a, s)) \equiv \\ (W^+(agt, a, s', s) \vee (W(agt, s', s) \wedge \neg W^-(agt, a, s', s))).\end{array}$$

An agent's goals are expanded when it is requested to do something by another agent. After the $\mathrm{REQUEST}(requester, agt, \psi)$ action occurs, agt should adopt the goal that ψ, unless it currently has a conflicting goal (i.e., we assume agents are maximally cooperative). Therefore, the $\mathrm{REQUEST}(requester, agt, \psi)$ action should cause agt to drop any paths in W where ψ does not hold. This action is taken into account in the definition of W^-:

$$\begin{array}{l} W^-(agt, a, s', s) \stackrel{\text{def}}{=} \\ \exists requester, \psi, s''. a = \mathrm{REQUEST}(requester, agt, \psi) \wedge \\ \quad \neg\mathbf{Goal}(agt, \neg\psi, s) \wedge K(agt, s'', s) \wedge s'' \preceq s' \wedge \\ \quad \neg\psi[do(a, s''), s'].\end{array}$$

According to this definition, s' will be dropped from W, if for some $requester$ and ψ, a is the $\mathrm{REQUEST}(requester, agt, \psi)$ action and agt does not have a conflicting goal, i.e., the goal that $\neg\psi$ in s[6], and s' K-intersects some s'' such that ψ does not hold on the path $(do(a, s''), s')$. The reason that we check whether $\neg\psi$ holds at $(do(a, s''), s')$ rather than at (s'', s') is to handle goals that are relative to the current time. If, for example, ψ states that the very next action should be to get some coffee, then we need to check whether the next action after the request is getting the coffee. If we checked $\neg\psi$ at (s'', s'), then the next action would be the REQUEST action.

If the agent gets a request for ψ and it already has the goal that $\neg\psi$, then it does not adopt the goal that ψ, otherwise its

[6] Note that according to the definition of **Goal**, if the agent has a goal that implies $\neg\psi$, then it also has the goal that $\neg\psi$, i.e., if **Goal**(Agt, ψ', S) holds and $\forall s, s'. \psi'[s, s'] \supset \neg\psi[s, s']$ holds, then **Goal**$(\mathit{Agt}, \neg\psi, S)$ also holds. In other words, if the agent has a conflicting goal that implies $\neg\psi$, then ψ will not be adopted as a goal.

goal state would become inconsistent and it would want everything. This is a simple way of handling goal conflicts. A more interesting method would be to give more credence to requests from certain individuals, or requests of certain types. For example, if an agent gets a request from its owner that conflicts with a previous request from someone else, it should drop the previous request and adopt its owner's request instead. We reserve a more sophisticated handling of conflicting requests for future work.

We assume that request actions are always possible to execute, although executability conditions could easily be added.

Axiom 3.2 $\mathit{Poss}(\textsc{request}(\mathit{reqr}, \mathit{agt}, \psi), s)$.

We now turn our attention to goal contraction. Suppose that the owner of an agent asks it to do ψ and later changes his mind. The owner should be able to tell the agent to stop working on ψ. We use the action $\textsc{cancelRequest}(\mathit{requester}, \mathit{agt}, \psi)$ for this purpose. This action causes agt to drop the goal that ψ. A CANCELREQUEST action can only be executed if a corresponding REQUEST action has occurred in the past.

Axiom 3.3 $\mathit{Poss}(\textsc{cancelRequest}(\mathit{reqr}, \mathit{agt}, \psi), s) \equiv$
$\mathit{do}(\textsc{request}(\mathit{reqr}, \mathit{agt}, \psi), s') \preceq s$

We handle CANCELREQUEST actions by determining what the W relation would have been if the corresponding REQUEST action had never happened. Suppose a $\textsc{cancelRequest}(\mathit{Requester}, \mathit{Agt}, \psi)$ action occurs in situation S. We restore the W relation to the way it was before the corresponding REQUEST occurred. Then, starting just after the REQUEST, we look at all the situations $\mathit{do}(A^*, S^*)$ in the history of S and remove from W any situation S' that satisfies $W^-(\mathit{Agt}, A, S', S^*)$; such situations S' should be removed to reflect the adoption of a subsequent request. We first define the predicate $\textsc{Cancels}(a, a')$, which says that action a cancels the action a'. In our case, only CANCELREQUEST actions cancel the corresponding REQUEST's:

$$\begin{array}{l}\textsc{Cancels}(a, a') \stackrel{\text{def}}{=} \\ \quad \exists \mathit{reqr}, \psi. a = \textsc{cancelRequest}(\mathit{reqr}, \mathit{agt}, \psi) \wedge \\ \qquad\qquad a' = \textsc{request}(\mathit{reqr}, \mathit{agt}, \psi).\end{array}$$

We now define W^+ which states the conditions under which s' is added to W^+ after a is executed in s:

$$\begin{array}{l}W^+(agt, a, s', s) \stackrel{\text{def}}{=} \\ \quad (\exists s_i . W(agt, s', s_i) \wedge \\ \qquad (\exists a_1 . \mathit{do}(a_1, s_i) \preceq s \wedge \textsc{Cancels}(agt, a, a_1) \wedge \\ \qquad\quad \forall a^*, s^* . \mathit{do}(a_1, s_i) \prec \mathit{do}(a^*, s^*) \preceq s \supset \\ \qquad\qquad\qquad \neg W^-(agt, a^*, s', s^*)))\end{array}$$

To help explain this definition, we will refer to Fig. 1, where a segment of a situation forest is shown. The situation S' is not W-related to S. However, S' was W-related to S_1, which is in the history of S. The next action after S_1 in the history of S was $\textsc{request}(\mathit{Requester}, \mathit{Agt}, \psi)$. We suppose S' was dropped from W after the REQUEST. If none of the actions between $\mathit{do}(\textsc{request}(\mathit{Requester}, \mathit{Agt}, \psi), S_1)$ and

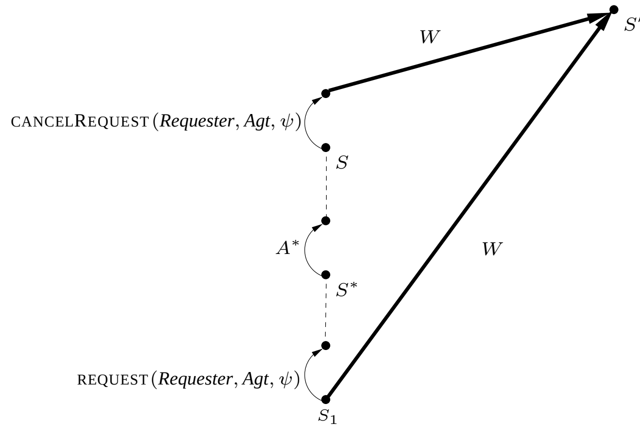

Figure 1: An example of goal contraction.

S also cause S' to be dropped from W, then S' is returned to W after the $\textsc{cancelRequest}(\mathit{Requester}, \mathit{Agt}, \psi)$ action is executed in S. In other words, $W^+(\mathit{Agt}, \textsc{cancelRequest}(\mathit{Requester}, \mathit{Agt}, \psi), S', S)$ holds because the following hold: $W(\mathit{Agt}, S', S_1)$, $\mathit{do}(\textsc{request}(\mathit{Requester}, \mathit{Agt}, \psi), S_1) \preceq S$, $\textsc{cancelRequest}(\mathit{Requester}, \mathit{Agt}, \psi)$ cancels $\textsc{request}(\mathit{Requester}, \mathit{Agt}, \psi))$, and for every situation $\mathit{do}(A^*, S^*)$ between $\mathit{do}(\textsc{request}(\mathit{Requester}, \mathit{Agt}, \psi), S_1)$ and S, $\neg W^-(\mathit{Agt}, A^*, S', S^*)$ holds. Note that for our definition of W^+ to work properly, we must assume that there is only one REQUEST action in the history that is cancelled by each CANCELREQUEST. This assumption will be relaxed in future work.

4 Properties

We now consider some properties of goal change. Let Σ consist of the foundational, encoding, and unique names axioms, and the axioms from Sections 2 and 3. First, we show that our theory supports goal expansion. If an agent does not have $\neg\psi$ as a goal, then a request for ψ causes it to adopt ψ as a goal.

Theorem 4.1

$$\begin{array}{l}\Sigma \models \forall agt, \psi, requester, s. \neg \mathbf{Goal}(agt, \neg\psi, s) \supset \\ \qquad \mathbf{Goal}(agt, \psi, \mathit{do}(\textsc{request}(requester, agt, \psi), s))\end{array}$$

We also show the conditions under which an agent's only goals are expanded. If an agent has the only goal that ψ, ψ' is requested of the agent, the agent knows that the request does not affect the truth value of ψ, and ψ' does not conflict with the agent's goals, then its only goals expand to include ψ'.

Theorem 4.2

$$\begin{array}{l}\Sigma \models \forall a, agt, \psi, \psi', reqr, s. \\ \quad \mathbf{OGoal}(agt, \psi, s) \wedge a = \textsc{request}(reqr, agt, \psi') \wedge \\ \quad \mathbf{Know}(agt, [\forall then. \mathit{Poss}(a, \mathit{Now}) \supset \\ \qquad\quad (\psi[\mathit{Now}, then] \equiv \psi[\mathit{do}(a, \mathit{Now}), then])], s) \wedge \\ \quad \neg \mathbf{Goal}(agt, \neg\psi', s) \supset \\ \qquad \mathbf{OGoal}(agt, \psi \wedge \psi', \mathit{do}(a, s)).\end{array}$$

Next, we examine the persistence of goals. A goal ψ persists over an action a, if a is not a cancel request, and the agent knows that if ψ holds then a does not change its value.

Theorem 4.3

$$\begin{array}{l}\Sigma \models \forall a, agt, \psi, s.\\ \quad \mathbf{Goal}(agt, \psi, s) \wedge \\ \quad (\forall reqr, \psi'.a \neq \text{CANCELREQUEST}(reqr, agt, \psi')) \wedge \\ \quad \mathbf{Know}(agt, [\forall then.Poss(a, Now) \wedge \psi[Now, then] \supset \\ \qquad\qquad \psi[do(a, Now), then]], s) \supset \\ \qquad \mathbf{Goal}(agt, \psi, do(a, s)).\end{array}$$

We have a similar result for "only goals".

Theorem 4.4

$$\begin{array}{l}\Sigma \models \forall a, agt, \psi, s.\\ \quad \mathbf{OGoal}(agt, \psi, s) \wedge \\ \quad [\forall requester, \psi'.\\ \qquad (a = \text{REQUEST}(requester, agt, \psi') \supset \\ \qquad\quad \mathbf{Goal}(agt, \neg\psi', s)) \wedge \\ \qquad a \neq \text{CANCELREQUEST}(requester, agt, \psi')] \wedge \\ \quad \mathbf{Know}(agt, [\forall then.Poss(a, Now) \supset \\ \qquad (\psi[Now, then] \equiv \psi[do(a, Now), then])], s) \supset \\ \qquad \mathbf{OGoal}(agt, \psi, do(a, s)).\end{array}$$

We now show a property about only goal contraction. Suppose that an agent only has the goal that ψ in s, and just after s, $reqr$ requests ψ' of the agent. In some future situation s'', $reqr$ cancels its request that ψ', yielding the situation s'. We assume that there was only one request of the agent for ψ' from $reqr$, and that no other REQUEST or CANCELREQUEST actions occurred between s and s'. We also assume that ψ always persists (this condition can be weakened; we assumed it to simplify the proof). Then, we can show that the agent only has the goal that ψ in s. In other words, the request for ψ' was indeed cancelled, and the other "only goals" persisted.

Theorem 4.5

$$\begin{array}{l}\Sigma \models \forall agt, \psi, \psi', reqr, s, s', s''.\ \mathbf{OGoal}(agt, \psi, s) \wedge \\ \quad do(\text{REQUEST}(reqr, agt, \psi'), s) \preceq s' \wedge \\ \quad s' = do(\text{CANCELREQUEST}(reqr, agt, \psi'), s'') \wedge \\ \quad [\forall s_1, s_2.do(\text{REQUEST}(reqr, agt, \psi'), s_1) \preceq s'' \wedge \\ \qquad do(\text{REQUEST}(reqr, agt, \psi'), s_2) \preceq s'' \supset \\ \qquad\quad s_1 = s_2] \wedge \\ \quad [\forall s^*, a^*.\\ \qquad do(\text{REQUEST}(reqr, agt, \psi'), s) \prec do(a^*, s^*) \wedge \\ \qquad do(a^*, s^*) \preceq s'' \supset \\ \qquad (\neg\exists reqr', \psi''.\\ \qquad\quad a^* = \text{CANCELREQUEST}(reqr', agt, \psi'') \vee \\ \qquad\quad a^* = \text{REQUEST}(reqr', agt, \psi''))] \wedge \\ \quad [\forall a, s_1, s_2.\psi(s_1, s_2) \equiv \psi(do(a, s_1), s_2)] \supset \\ \qquad \mathbf{OGoal}(agt, \psi, s').\end{array}$$

Just as agents introspect their knowledge, we want agents to be able to introspect their goals, i.e., if an agent has a goal (does not have a goal, resp.) that ψ, it should know that it has (does not have, resp.) ψ as a goal. We identify constraints that yield these properties. They are constraints on K and W. We will need the following definitions of transitivity and Euclideanness, starting in a situation s. Note that the order of the situations is reversed from the traditional definitions.

$$\begin{array}{l}Ktrans(agt, s) \stackrel{\text{def}}{=} \forall s_1, s_2.K(agt, s_1, s) \wedge K(agt, s_2, s_1) \supset \\ \qquad\qquad K(agt, s_2, s).\\ Keuc(agt, s) \stackrel{\text{def}}{=} \forall s_1, s_2.K(agt, s_1, s) \wedge K(agt, s_2, s) \supset \\ \qquad\qquad K(agt, s_2, s_1).\end{array}$$

Theorem 6 of Scherl and Levesque [2003] showed that if K is initially reflexive, and transitive or Euclidean, then the successor state axiom for K guarantees that the properties are preserved over executable sequences of actions. This result carries over here. First, we show that reflexivity is preserved. Recall that we asserted that K was initially reflexive in Axiom 2.3. Our successor state axiom for K preserves reflexivity over all executable sequences of actions.

Theorem 4.6

$$\Sigma \models \forall agt, s.Executable(s) \supset K(agt, s, s),$$

where $Executable(s) \stackrel{\text{def}}{=} \forall a, s'.do(a, s') \preceq s \supset Poss(a, s')$.

Similarly, if K is initially transitive and Euclidean, then it remains so over any executable sequence of actions.

Theorem 4.7

$$\begin{array}{l}\Sigma \models (\forall agt, s.Init(s) \supset Ktrans(agt, s) \wedge Keuc(agt, s)) \supset \\ \qquad (\forall agt, s.Executable(s) \supset Ktrans(agt, s)) \wedge \\ \qquad (\forall agt, s.Executable(s) \supset Keuc(agt, s)),\end{array}$$

For positive introspection of goals, we need a constraint similar to transitivity, but that involves both K and W. We call this constraint *cross-transitivity*:

$$\begin{array}{l}CrossTrans(agt, s) \stackrel{\text{def}}{=} \\ \quad \forall s_1, s_2, s_3.K(agt, s_1, s) \wedge K(agt, s_2, s_1) \wedge \\ \qquad W(agt, s_3, s_1) \wedge s_2 \preceq s_3 \supset \\ \qquad\quad W(agt, s_3, s).\end{array}$$

If this constraint is satisfied, and K is transitive, then the agents will have positive introspection of goals:

Theorem 4.8

$$\begin{array}{l}\Sigma \models \forall agt, s, \psi.Ktrans(agt, s) \wedge CrossTrans(agt, s) \supset \\ \qquad (\mathbf{Goal}(agt, \psi, s) \supset \\ \qquad\quad \mathbf{Know}(agt, \mathbf{Goal}(agt, \psi), s)).\end{array}$$

For negative introspection of goals, we need a constraint similar to Euclideanness, which we call *cross-Euclideanness*:

$$\begin{array}{l}CrossEuc(agt, s) \stackrel{\text{def}}{=} \\ \quad \forall s_1, s_2, s_3.K(agt, s_1, s) \wedge K(agt, s_2, s) \wedge \\ \qquad W(agt, s_3, s) \wedge s_2 \preceq s_3 \supset \\ \qquad\quad W(agt, s_3, s_1).\end{array}$$

If this constraint is satisfied, and K is Euclidean, then the agents will have negative introspection of goals:

Theorem 4.9

$$\Sigma \models \forall agt, s.\mathit{Keuc}(agt,s) \wedge \mathit{CrossEuc}(agt,s) \supset (\neg \mathbf{Goal}(agt,\psi,s) \supset \mathbf{Know}(agt, \neg \mathbf{Goal}(agt,\psi), s)).$$

CrossTrans and *CrossEuc* persist if they hold of the initial situations, and K is initially transitive and Euclidean.

Theorem 4.10

$$\Sigma \models [\forall s'.\mathit{Init}(s') \supset (\mathit{Ktrans}(agt,s') \wedge \mathit{Keuc}(agt,s') \wedge \mathit{CrossTrans}(agt,s') \wedge \mathit{CrossEuc}(agt,s'))] \supset \forall agt, s.\mathit{Executable}(s) \supset \mathit{CrossTrans}(agt,s) \wedge \mathit{CrossEuc}(agt,s).$$

It follows that positive and negative goal introspection persist, if these constraints hold initially.

5 Meeting Scheduler Example

To illustrate how such a formalization of goal change is useful in specifying multiagent applications, we now summarize the meeting scheduler example from Shapiro [2005] that is based on the one given by Shapiro *et al.* [1998], but was modified to include goal contraction using CANCELREQUEST.

We specify the behavior of agents with the notation of the programming language ConGolog [De Giacomo *et al.*, 2000], the concurrent version of Golog [Levesque *et al.*, 1997]. Here are the ConGolog constructs we will use:

a,	primitive action
ϕ?,	wait for a condition
$\delta_1; \delta_2$,	sequence
$\delta_1 \mid \delta_2$,	nondeterministic choice of programs
if ϕ **then** δ_1 {**else** δ_2} **endIf**,	conditional
for $x \in L$ **do** δ **endFor**,	for loop
while ϕ **do** δ **endWhile**	while loop

In the above, a denotes a situation calculus action; δ, δ_1, and δ_2 stand for programs; and L is a finite list.

In the meeting scheduler example [Shapiro *et al.*, 1998; Shapiro and Lespérance, 2001; Shapiro, 2005], there are meeting organizer agents, which are trying to schedule meetings with personal agents, which manage the schedules of their (human) owners. To schedule a meeting, an organizer agent requests of each of the personal agents of the participants in the meeting to adopt the goal that its owner attend the meeting during a given time period. If a personal agent does not have any goals that conflict with its owner attending the meeting (i.e., it has not previously scheduled a conflicting meeting), it adopts the goal that its owner attend this meeting and informs the meeting organizer that it has adopted this goal, i.e., that it accepts the meeting request. Otherwise, the personal agent informs the meeting organizer that it has not adopted the goal that its owner attend the meeting, i.e., that it declines the meeting request.

The procedure that defines the behaviour of meeting organizer agents (organizeMeeting) is given in Fig. 2.[7] Their task is to organize a meeting for a given period and set of participants on behalf of the chair of the meeting. They do this by asking the personal agent of each participant to have their owner meet during the given period. The participants' personal agent will then reply whether they have declined the meeting. The behaviour of the personal agents is specified by the MANAGESCHEDULE procedure which is not shown here, due to space constraints. In that procedure, a personal agent responds to a meeting request by adopting the goal that its owner be at the requested meeting, unless it has already scheduled a conflicting meeting, and then informs the requester whether it has adopted the requested goal. The meeting organizer waits until it knows whether one of the participants' agent declined the meeting. This will happen when either someone has declined the meeting or everyone has agreed to it. If the organizer discovers that someone declined the meeting, it cancels its request for a meeting with the personal agents of all the participants in the meeting. The personal agents that agreed to the meeting will then drop the goal that their owners be at the requested meeting.

6 Conclusions and Future Work

In this paper, we defined the goals and "only goals" of an agent, and examined properties of these definitions: expansion, contraction, and persistence. We identified the restrictions on K and W that give us introspection of goals, and showed that these restrictions persist, if they hold initially. Note that our approach is quite different and simpler than most existing approaches to belief change. These require a specification of the plausibilities of alternate situations. Our approach avoids this and instead relies on the history to undo previous goal expansions. However, we do not handle arbitrary goal contractions, only cancellations of previous requests. Note there are some related approaches in the belief change literature [Booth *et al.*, 2005; Roorda *et al.*, 2003]. This raises the question of using belief change frameworks to handle goal change, or applying our goal change approach to belief change. It would be interesting to investigate whether there are essential differences between belief revision and goal revision that warrant different solutions, or whether the same solutions can be applied to both problems. Also, it would be interesting to identify postulates for goal change and examine how they differ from belief change postulates [Gärdenfors, 1988]. Our handling of the cancelling of requests needs further work and analysis. First of all, it does not properly handle nested requests for the same goal. For the successor state axiom to work properly for goal contraction, we must assume that there is only one REQUEST

[7] Note that for us (unlike De Giacomo *et al.*) procedures are simply program definitions and cannot be recursive. In the definition of this procedure, PAG(p) is a function that maps an owner to its personal agent, MEMBER($p, parts$) holds if p is a member of the list $parts$, and $\mathbf{KWhether}(agt,\phi,s) \stackrel{\text{def}}{=} \mathbf{Know}(agt,\phi,s) \vee \mathbf{Know}(agt,\neg\phi,s)$.

$$\text{INFORMWHETHER}(agt, agt', \phi) \stackrel{\text{def}}{=} [\mathbf{Know}(agt, \phi)?; \text{INFORM}(agt, agt', \phi)] \mid$$
$$[\mathbf{Know}(agt, \neg\phi)?; \text{INFORM}(agt, agt', \neg\phi)] \mid [\neg\mathbf{KWhether}(agt, \phi)?; \text{INFORM}(agt, agt', \neg\mathbf{KWhether}(agt, \phi))]$$

$$\text{SomeoneDeclined}(per, chair, parts, s) \stackrel{\text{def}}{=}$$
$$\exists p.\text{MEMBER}(p, parts) \wedge \neg\mathbf{Goal}(\text{PAG}(p), \mathbf{During}(per, \text{ATMEETING}(p, chair)), s)$$

$$\text{organizeMeeting}(oa, chair, parts, per) \stackrel{\text{def}}{=}$$

for $p \in parts$ **do**
  $\text{REQUEST}(oa, \text{PAG}(p), \mathbf{During}(per, \text{ATMEETING}(p, chair)))$ **endFor**;
$\mathbf{KWhether}(oa, \text{SomeoneDeclined}(per, chair, parts))?$;
if $\mathbf{Know}(oa, \text{SomeoneDeclined}(per, chair, parts))$ **then**
  for $p \in parts$ **do**
    $\text{cancelRequest}(oa, \text{PAG}(p), \mathbf{During}(per, \text{ATMEETING}(p, chair)))$ **endFor** **endIf**;
$\text{INFORMWHETHER}(oa, chair, \text{SomeoneDeclined}(per, chair, parts))$

Figure 2: Procedure run by the meeting organizer agents.

action in the history that is cancelled by each CANCELREQUEST. This would not be too difficult to remedy, but it leads us to another difficulty. The definition of W^+ is already quite complex, which makes proving properties about it somewhat awkward. Since the definition refers to a sequence of situations, the proofs will often involve inductions. We would like to investigate the possibility of simplifying the definition.

References

[Booth *et al.*, 2005] R. Booth, S. Chopra, A. Ghose, and T. Meyer. Belief liberation (and retraction). *Studia Logica*, 79:47–72, 2005.

[Cohen and Levesque, 1990] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42:213–261, 1990.

[De Giacomo *et al.*, 2000] G. De Giacomo, Y. Lespérance, and H. J. Levesque. ConGolog, a concurrent programming language based on the situation calculus. *Artificial Intelligence*, 121(1–2):109–169, 2000.

[Gärdenfors, 1988] P. Gärdenfors. *Knowledge in Flux*. The MIT Press: Cambridge, MA, 1988.

[Konolige and Pollack, 1993] K. Konolige and M. E. Pollack. A representationalist theory of intention. In *Proc. IJCAI-93*, pages 390–395, 1993.

[Levesque *et al.*, 1997] H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, and R. B. Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31:59–84, 1997.

[Levesque *et al.*, 1998] H. J. Levesque, F. Pirri, and R. Reiter. Foundations for a calculus of situations. *Electronic Transactions of AI (ETAI)*, 2(3–4):159–178, 1998.

[McCarthy and Hayes, 1969] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*. Edinburgh University Press, 1969.

[Moore, 1985] R. C. Moore. A formal theory of knowledge and action. In J. R. Hobbs and R. C. Moore, editors, *Formal Theories of the Common Sense World*, pages 319–358. Ablex Publishing, Norwood, NJ, 1985.

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

[Roorda *et al.*, 2003] J.-W. Roorda, W. van der Hoek, and J.-J. Meyer. Iterated belief change in multi-agent systems. *Logic Journal of the IGPL*, 11(2):223–246, 2003.

[Scherl and Levesque, 2003] R. B. Scherl and H. J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144(1–2):1–39, 2003.

[Shapiro and Lespérance, 2001] S. Shapiro and Y. Lespérance. Modeling multiagent systems with the cognitive agents specification language — a feature interaction resolution application. In C. Castelfranchi and Y. Lespérance, editors, *Proc. ATAL-2000*, pages 244–259. Springer-Verlag, Berlin, 2001.

[Shapiro *et al.*, 1998] S. Shapiro, Y. Lespérance, and H. J. Levesque. Specifying communicative multi-agent systems. In W. Wobcke, M. Pagnucco, and C. Zhang, editors, *Agents and Multi-Agent Systems — Formalisms, Methodologies, and Applications*, volume 1441 of *LNAI*, pages 1–14. Springer-Verlag, Berlin, 1998.

[Shapiro *et al.*, 2000] S. Shapiro, M. Pagnucco, Y. Lespérance, and H. J. Levesque. Iterated belief change in the situation calculus. In A. G. Cohn, F. Giunchiglia, and B. Selman, editors, *Proc. KR-2000*, pages 527–538, 2000.

[Shapiro, 2005] S. Shapiro. PhD thesis. To appear, 2005.

[Spohn, 1988] W. Spohn. Ordinal conditional functions: A dynamic theory of epistemic states. In W. Harper and B. Skyrms, editors, *Causation in Decision, Belief Change, and Statistics*, pages 105–134. Kluwer Academic Publishers, 1988.