A Logical Account of Prioritized Goals and their Dynamics

Shakil M. Khan and Yves Lespérance
Department of Computer Science and Engineering
York University, Toronto, ON, Canada
{skhan, lesperan}@cse.yorku.ca

Abstract

Most previous logical accounts of goals do not deal with prioritized goals and goal dynamics properly. Many are restricted to achievement goals. In this paper, we develop a logical account of goal change that addresses these deficiencies. In our account, we do not drop lower priority goals permanently when they become inconsistent with other goals and the agent's knowledge; rather, we make such goals inactive. We ensure that the agent's chosen goals/intentions are consistent with each other and the agent's knowledge. When the world changes, the agent recomputes her chosen goals and some inactive goals may become active again. This ensures that our agent maximizes her utility. We prove that the proposed account has desirable properties.

Introduction

There has been much work on modeling agent's mental states, beliefs, goals, and intentions, and how they interact and lead to rational decisions about action. As well, there has been a lot of work on modeling belief change. But the dynamics of motivational attitudes has received much less attention. Most formal models of goal and goal change (Cohen and Levesque 1990; Rao and Georgeff 1991; Konolige and Pollack 1993; Shapiro *et al.* 1995) assume that all goals are equally important and many only deal with achievement goals. Moreover, most of these frameworks do not guarantee that an agent's goals will properly evolve when an action/event occurs, e.g. when the agent's beliefs/knowledge changes or a goal is adopted or dropped (one exception to this is the model of prioritized goals in (Shapiro and Brewka 2007)). Dealing with these issues is important for developing effective models of rational agency. It is also important for work on BDI agent programming languages, where handling declarative goals is an active research topic.

In this paper, we present a formal model of prioritized goals and their dynamics that addresses some of these issues. In our framework, an agent can have multiple goals at different priority levels, possibly inconsistent with each other. We define intentions as the maximal set of highest priority goals that is consistent given the agent's knowledge. Our model of goals supports the specification of general temporally extended goals, not just achievement goals.

We start with a (possibly inconsistent) initial set of *prioritized goals* or desires that are totally ordered according to priority, and specify how these goals evolve when actions/events occur and the agent's knowledge changes. We define the agent's *chosen goals* or intentions in terms of this goal hierarchy. Our agents maximize their utility; they will abandon a chosen goal ϕ if an opportunity to commit to a higher priority but inconsistent with ϕ goal arises. To this end, we keep all prioritized goals in the goal base unless they are explicitly dropped. At every step, we compute an optimal set of chosen goals given the hierarchy of prioritized goals, preferring higher priority goals such that chosen goals are consistent with each other and with the agent's knowledge. Thus at any given time, some goals in the hierarchy are active, i.e. chosen, while others are inactive. Some of these inactive goals may later become active, e.g. if a higher priority active goal that is currently blocking an inactive goal becomes impossible.

Our formalization of prioritized goals ensures that the agent always tries to maximize her utility, and as such displays an idealized form of rationality. In the fifth section, we discuss how this relates to Bratman's (1987) theory of practical reasoning. We use an action theory based on the situation calculus along with our formalization of paths in the situation calculus as our base formalism.

In the next section, we outline our base framework. In the third section, we formalize *paths* in the situation calculus to support modeling goals. In the fourth section, we present our model of prioritized goals. In the fifth and sixth section, we present our formalization of goal dynamics and discuss some of its properties. Then in the last section, we summarize our results, discuss previous work in this area, and point to possible future work.

Action and Knowledge

Our base framework for modeling goal change is the situation calculus as formalized in (Reiter 2001). In this framework, a possible state of the domain is represented by a situation. There is a set of initial situations corresponding to the ways the agent believes the domain might be initially, i.e. situations in which no actions have yet occurred. Init(s) means that s is an initial situation. The actual initial state is represented by a special constant S_0. There is a distinguished binary function symbol do where $do(a, s)$ denotes the successor situation to s resulting from performing the action a. Relations (and functions) whose truth values vary from situation to situation, are called relational (functional, resp.) fluents, and are denoted by predicate (function, resp.) symbols taking a situation term as their last argument. There

is a special predicate Poss(a, s) used to state that action a is executable in situation s.

Our framework uses a theory D_{basic} that includes the following set of axioms:[1] (1) action precondition axioms, one per action a characterizing Poss(a, s), (2) successor state axioms (SSA), one per fluent, that succinctly encode both effect and frame axioms and specify exactly when the fluent changes (Reiter 2001), (3) initial state axioms describing what is true initially including the mental states of the agents, (4) unique name axioms for actions, and (5) domain-independent foundational axioms describing the structure of situations (Levesque *et al.* 1998).

Following (Scherl and Levesque 2003), we model knowledge using a possible worlds account adapted to the situation calculus. $K(s', s)$ is used to denote that in situation s, the agent thinks that she could be in situation s'. Using K, the knowledge of an agent is defined as:[2] $\text{Know}(\Phi, s) \stackrel{\text{def}}{=} \forall s'.\ K(s', s) \supset \Phi(s')$, i.e. the agent knows Φ in s if Φ holds in all of her K-accessible situations in s. K is constrained to be reflexive, transitive, and Euclidean in the initial situation to capture the fact that agents' knowledge is true, and that agents have positive and negative introspection. As shown in (Scherl and Levesque 2003), these constraints then continue to hold after any sequence of actions since they are preserved by the successor state axiom for K. We also assume that all actions are public, i.e. whenever an action (including exogenous events) occurs, the agent learns that it has happened. Note that, we work with knowledge rather than belief. Although much of our formalization should extend to the latter, we leave this for future work.

Paths in the Situation Calculus

To support modeling temporally extended goals, we introduce a new sort of *paths*, with (possibly sub/super-scripted) variables p ranging over paths. A path is essentially an infinite sequence of situations, where each situation along the path can be reached by performing some *executable* action in the preceding situation. We introduce a predicate OnPath(p, s), meaning that the situation s is on path p. Also, Starts(p, s) means that s is the starting situation of path p. A path p starts with s iff s is the earliest situation on p:[3]

Axiom 1

$$\text{Starts}(p, s) \equiv \text{OnPath}(p, s) \wedge \forall s'.\ \text{OnPath}(p, s') \supset s \leq s'.$$

In the standard situation calculus, paths are implicitly there, and a path can be viewed as a pair (s, F) consisting of a situation s representing the starting situation of the path, and a function F from situations to actions (called *Action Selection Functions* (ASF) or strategies in (Shapiro *et al.* 1995)), such that from the starting situation s, F defines an infinite sequence of situations by specifying an action for every situation starting from s. Thus, one way of axiomatizing paths is by making them correspond to such pairs (s, F):

Axiom 2

$$\begin{aligned}\forall p.\ \text{Starts}(p, s) \supset (\exists F.\ \text{Executable}(F, s) \\ \wedge \forall s'.\ \text{OnPath}(p, s') \equiv \text{OnPathASF}(F, s, s')),\\ \forall F, s.\ \text{Executable}(F, s) \supset \exists p.\ \text{Starts}(p, s) \\ \wedge \forall s'.\ \text{OnPathASF}(F, s, s') \equiv \text{OnPath}(p, s').\end{aligned}$$

This says that for every path there is an executable ASF that produces exactly the sequence of situations on the path from its starting situation. Also, for every executable ASF and situation, there is a path that corresponds to the sequence of situations produced by the ASF starting from that situation.

$$\text{OnPathASF}(F, s, s') \stackrel{\text{def}}{=} s \leq s' \wedge \forall a, s^*.\ s < do(a, s^*) \leq s' \supset F(s^*) = a,$$

$$\text{Executable}(F, s) \stackrel{\text{def}}{=} \forall s'.\ \text{OnPathASF}(F, s, s') \supset \text{Poss}(F(s'), s').$$

Here, OnPathASF(F, s, s') means that the situation sequence defined by (s, F) includes the situation s'. Also, the situation sequence encoded by a strategy F and a starting situation s is executable iff for all situations s' on this sequence, the action selected by F in s' is executable in s'.

We will use both state and path formulae. A state formula $\Phi(s)$ is a formula that has a free situation variable s in it, whereas a path formula $\phi(p)$ is one that has a free path variable p. State formulae are used in the context of knowledge while path formulae are used in that of goals. Note that, by incorporating infinite paths in our framework, we can evaluate goals over these and handle arbitrary temporally extended goals; thus, unlike some other situation calculus based accounts where goal formulae are evaluated w.r.t. finite paths (e.g. (Shapiro and Brewka 2007)), we can handle for example unbounded maintenance goals.

We next define some useful constructs. A state formula Φ *eventually holds* over the path p if Φ holds in some situation that is on p, i.e. $\Diamond\Phi(p) \stackrel{\text{def}}{=} \exists s'.\ \text{OnPath}(p, s') \wedge \Phi(s')$. Other Temporal Logic operators can be defined similarly, e.g. always Φ: $\Box\Phi(p)$.

An agent *knows* in s that ϕ has become *inevitable* if ϕ holds over all paths that starts with a K-accessible situation in s, i.e. $\text{KInevitable}(\phi, s) \stackrel{\text{def}}{=} \forall p.\ \text{Starts}(p, s') \wedge K(s', s) \supset \phi(p)$. An agent knows in s that ϕ is impossible if she knows that $\neg\phi$ is inevitable in s, i.e. $\text{KImpossible}(\phi, s) \stackrel{\text{def}}{=} \text{KInevitable}(\neg\phi, s)$.

Thirdly, we define what it means for a path p' to be a suffix of another path p w.r.t. a situation s:

$$\begin{aligned}\text{Suffix}(p', p, s) \stackrel{\text{def}}{=} \text{OnPath}(p, s) \wedge \text{Starts}(p', s) \\ \wedge \forall s'.\ s' \geq s \supset \text{OnPath}(p, s') \equiv \text{OnPath}(p', s').\end{aligned}$$

Fourthly, SameHist(s_1, s_2) means that the situations s_1 and s_2 share the same history of actions, but perhaps starting from different initial situations:

Axiom 3

$$\begin{aligned}&\text{SameHist}(s_1, s_2) \equiv (\text{Init}(s_1) \wedge \text{Init}(s_2)) \vee \\ &(\exists a, s_1', s_2'.\ s_1 = do(a, s_1') \wedge s_2 = do(a, s_2') \wedge \text{SameHist}(s_1', s_2')).\end{aligned}$$

Finally, we say that ϕ has become *inevitable* in s if ϕ holds over all paths that starts with a situation that has the same history as s: $\text{Inevitable}(\phi, s) \stackrel{\text{def}}{=} \forall p, s'.\ \text{Starts}(p, s') \wedge \text{SameHist}(s', s) \supset \phi(p)$.

[1] We will be quantifying over formulae, and thus assume D_{basic} includes axioms for encoding of formulae as first order terms, as in (Shapiro *et al.* 2007).

[2] Φ is a state formula that can contain a situation variable, *now*, in the place of situation terms. We often suppress *now* when the intent is clear from the context.

[3] In the following, $s < s'$ means that s' can be reached from s by performing a sequence of executable actions. $s \leq s'$ is an abbreviation for $s < s' \vee s = s'$.

Prioritized Goals

Most work on formalizing goals only deals with static goal semantics and not their dynamics. In this section, we formalize goals or desires with different priorities which we call *prioritized goals* (p-goals, henceforth). These p-goals are not required to be mutually consistent and need not be actively pursued by the agent. In terms of these, we define the consistent set of *chosen goals* or intentions (c-goals, henceforth) that the agent is committed to. In the next section, we formalize goal dynamics by providing a SSA for p-goals. The agent's c-goals are automatically updated when her p-goals change.

Not all of the agent's goals are equally important to her. Thus, it is useful to support a priority ordering over goals. This information can be used to decide which of the agent's c-goals should no longer be actively pursued in case they become mutually inconsistent. We specify each p-goal by its own accessibility relation/fluent G. A path p is G-accessible at priority level n in situation s (denoted by $G(p, n, s)$) if all the goals of the agent at level n are satisfied over this path and if it starts with a situation that has the same action history as s. The latter requirement ensures that the agent's p-goal-accessible paths reflect the actions that have been performed so far. A smaller n represents higher priority, and the highest priority level is 0. Thus here we assume that the set of p-goals are totally ordered according to priority. We say that an agent has the p-goal that ϕ at level n in situation s iff ϕ holds over all paths that are G-accessible at n in s:

$$\text{PGoal}(\phi, n, s) \stackrel{\text{def}}{=} \forall p.\ G(p, n, s) \supset \phi(p).$$

To be able to refer to all the p-goals of the agent at some given priority level, we also define *only p-goals*.

$$\text{OPGoal}(\phi, n, s) \stackrel{\text{def}}{=} \text{PGoal}(\phi, n, s) \wedge \forall p.\ \phi(p) \supset G(p, n, s).$$

An agent has the only p-goal that ϕ at level n in situation s iff ϕ is a p-goal at n in s, and any path over which ϕ holds is G-accessible at n in s.

A domain theory for our framework D includes the axioms of a theory D_{basic} as in the previous section, the axiomatization of paths i.e. axioms 1-3, domain dependent initial goal axioms (see below), the domain independent axioms 4-7 and the definitions that appear in this section and the next. We allow the agent to have infinitely many goals. We expect the modeler to include some specification of what paths are G accessible at the various levels initially. We call these axioms *initial goal axioms*. In many cases, the user will want to specify a finite set of initial p-goals. This can be done by providing a set of axioms as in the example below. But in general, an agent can have a countably infinite set of p-goals, e.g. an agent that has the p-goal at level n to know what the n-th prime number is for all n. The agent's set of p-goals can even be specified incompletely, e.g. the theory might not specify what the p-goals at some level are initially.

We use the following as a running example. We have an agent who initially has the following three p-goals: $\phi_0 = \Box\text{BeRich}$, $\phi_1 = \Diamond\text{GetPhD}$, and $\phi_2 = \Box\text{BeHappy}$ at level 0, 1, and 2, respectively. This domain can be specified using the following two initial goal axioms:

(a) $\text{Init}(s) \supset ((G(p, 0, s) \equiv \text{Starts}(p, s') \wedge \text{Init}(s') \wedge \phi_0(p))$
$\wedge\ ((G(p, 1, s) \equiv \text{Starts}(p, s') \wedge \text{Init}(s') \wedge \phi_1(p))$
$\wedge\ (G(p, 2, s) \equiv \text{Starts}(p, s') \wedge \text{Init}(s') \wedge \phi_2(p))),$

(b) $\forall n, p, s.\ \text{Init}(s) \wedge n \geq 3 \supset$
$(G(p, n, s) \equiv \text{Starts}(p, s') \wedge \text{Init}(s')).$

(a) specifies the p-goals ϕ_0, ϕ_1, ϕ_2 (from highest to lowest priority) of the agent in the initial situations, and makes $G(p, n, s)$ true for every path p that starts with an initial situation and over which ϕ_n holds, for $n = 0, 1, 2$; each of them defines a set of initial goal paths for a given priority level, and must be consistent. (b) makes $G(p, n, s)$ true for every path p that starts with an initial situation for $n \geq 3$. Thus at levels $n \geq 3$, the agent has the trivial p-goal that she be in an initial situation. Assume that while initially the agent knows that all of her p-goals are individually achievable, she knows that her p-goal $\Diamond\text{GetPhD}$ is inconsistent with her highest priority p-goal $\Box\text{BeRich}$ as well as with her p-goal $\Box\text{BeHappy}$, while the latter are consistent with each other. Thus in our example, we have $\text{OPGoal}(\phi_i(p) \wedge \text{Starts}(p, s) \wedge \text{Init}(s), i, S_0)$, for $i = 0, 1, 2$. Also, for any $n \geq 3$, we have $\text{OPGoal}(\text{Starts}(p, s) \wedge \text{Init}(s), n, S_0)$.

While p-goals or desires are allowed to be known to be impossible to achieve, an agent's c-goals or intentions must be realistic. Not all of the G-accessible paths are realistic in the sense that they start with a K-accessible situation. To filter these out, we define *realistic* p-goal accessible paths:

$$G_R(p, n, s) \stackrel{\text{def}}{=} G(p, n, s) \wedge \text{Starts}(p, s') \wedge K(s', s),$$

Thus G_R prunes out the paths from G that are known to be impossible, and since we define c-goals in terms of realistic p-goals, this ensures that c-goals are realistic. We say that an agent has the *realistic p-goal* that ϕ at level n in situation s iff ϕ holds over all paths that are G_R-accessible at n in s:

$$\text{RPGoal}(\phi, n, s) \stackrel{\text{def}}{=} \forall p.\ G_R(p, n, s) \supset \phi(p).$$

Using realistic p-goals, we next define c-goals. The idea of how we calculate c-goal-accessible paths is as follows: the set of G_R-accessibility relations represents a set of prioritized temporal propositions that are candidates for the agent's c-goals. Given G_R, in each situation we want to compute the agent's c-goals such that it is the *maximal consistent* set of higher priority realistic p-goals. We do this iteratively starting with the set of all realistic paths (i.e. paths that starts with a K-accessible situation). At each iteration we compute the intersection of this set with the next highest priority set of G_R-accessible paths. If the intersection is not empty, we thus obtain a new chosen set of paths at level i. We call a p-goal chosen by this process an *active* p-goal. If on the other hand the intersection is empty, then it must be the case that the p-goal represented by this level is either in conflict with another active higher priority p-goal/a combination of two or more active higher priority p-goals, or is known to be impossible. In that case, that p-goal is ignored (i.e. marked as inactive), and the chosen set of paths at level i is the same as at level $i - 1$. Axiom 4 computes this intersection:[4]

Axiom 4 $G_\cap(p, n, s) \equiv$
if $(n = 0)$ **then**
if $\exists p'.\ G_R(p', n, s)$ **then** $G_R(p, n, s)$
else $\text{Starts}(p, s') \wedge K(s', s)$

[4] **if** ϕ **then** δ_1 **else** δ_2 is an abbreviation for $(\phi \supset \delta_1) \wedge (\neg\phi \supset \delta_2)$.

else

if $\exists p'.(G_R(p', n-1, s) \wedge G_\cap(p', n-1, s))$

then $(G_R(p, n-1, s) \wedge G_\cap(p, n-1, s))$

else $G_\cap(p, n-1, s)$.

Using this, we define what it means for an agent to have a c-goal at some level n:

$$\text{CGoal}(\phi, n, s) \stackrel{\text{def}}{=} \forall p.\ G_\cap(p, n, s) \supset \phi(p),$$

i.e. an agent has the c-goal at level n that ϕ if ϕ holds over all paths that are in the prioritized intersection of the set of G_R-accessible paths up to level n.

We define c-goals in terms of c-goals at level n:

$$\text{CGoal}(\phi, s) \stackrel{\text{def}}{=} \forall n.\ \text{CGoal}(\phi, n, s),$$

i.e., the agent has the c-goal that ϕ if for any level n, ϕ is a c-goal at n.

In our example, the agent's realistic p-goals are □BeRich, ◇GetPhD, and □BeHappy in order of priority. The $G_\cap$-accessible paths at level 0 in S_0 are the ones that start with a K-accessible situation and where □BeRich holds. The $G_\cap$-accessible paths at level 1 in S_0 are the same as at level 0, since there are no K-accessible paths over which both ◇GetPhD and □BeRich hold. Finally, the $G_\cap$-accessible paths at level 2 in S_0 are those that start with a K-accessible situation and over which □BeRich ∧ □BeHappy holds. Also, it can be shown that initially our example agent has the c-goals that □BeRich and □BeHappy, but not ◇GetPhD.

Note that by our definition of c-goals, the agent can have a c-goal that ϕ in situation s for various reasons: 1) ϕ is known to be inevitable in s; 2) ϕ is an active p-goal at some level n in s; 3) ϕ is a consequence of two or more active p-goals at different levels in s. To be able to refer to c-goals for which the agent has a primitive motivation, i.e. c-goals that result from a single active p-goal at some priority level n, in contrast to those that hold as a consequence of two or more active p-goals at different priority levels, we define *primary* c-goals:

$$\text{PrimCGoal}(\phi, s) \stackrel{\text{def}}{=}$$
$$\exists n.\ \text{PGoal}(\phi, n, s) \wedge \exists p.\ G(p, n, s) \wedge G_\cap(p, n, s).$$

That is, an agent has the primary c-goal that ϕ in situation s, if ϕ is a p-goal at some level n in s, and if there is a G-accessible path p at n in s that is also in the prioritized intersection of G_R-accessible paths upto n in s. The last two conjucts are required to ensure that n is an active level. Thus if an agent has a primary c-goal that ϕ, then she also has the c-goal that ϕ, but not necessarily vice-versa. It can be shown that initially our example agent has the primary c-goals that □BeRich and □BeHappy, but not their conjunction. To some extent, this shows that primary c-goals are not closed under logical consequence.

Goal Dynamics

An agent's goals change when her knowledge changes as a result of the occurrence of an action (including exogenous events), or when she adopts or drops a goal. We formalize this by specifying how p-goals change. C-goals are then computed using (realistic) p-goals in every new situation as above.

We introduce two actions for adopting and dropping a p-goal, $adopt(\phi, n)$ and $drop(\phi)$. The action precondition axioms for these are as follows:

Axiom 5 $\text{Poss}(adopt(\phi, n), s) \equiv \neg\exists n'.\ \text{PGoal}(\phi, n', s)$,
$\text{Poss}(drop(\phi), s) \equiv \exists n.\ \text{PGoal}(\phi, n, s)$.

That is, an agent can adopt (drop) the p-goal that ϕ at level n, if she does not (does) already have ϕ as her p-goal at some level.

In the following, we specify the dynamics of p-goals by giving the SSA for G and discuss each case, one at a time:

Axiom 6 (SSA for G) $G(p, n, do(a, s)) \equiv$

$\forall \phi, m.\ (a \neq adopt(\phi, m) \wedge a \neq drop(\phi) \wedge \text{Progressed}(p, n, a, s))$

$\vee\ \exists \phi, m.\ (a = adopt(\phi, m) \wedge \text{Adopted}(p, n, m, a, s, \phi))$

$\vee\ \exists \phi.\ (a = drop(\phi) \wedge \text{Dropped}(p, n, a, s, \phi))$.

The overall idea of the SSA for G is as follows. First of all, to handle the occurrence of a non-adopt/drop (i.e. regular) action a, we progress all G-accessible paths to reflect the fact that this action has just happened; this is done using the $\text{Progressed}(p, n, a, s)$ construct, which replaces each G-accessible path p' with starting situation s', by its suffix p provided that it starts with $do(a, s')$:

$$\text{Progressed}(p, n, a, s) \stackrel{\text{def}}{=}$$
$$\exists p'.\ G(p', n, s) \wedge \text{Starts}(p', s') \wedge \text{Suffix}(p, p', do(a, s')).$$

Any path over which the next action performed is not a is eliminated from the respective G accessibility level.

Secondly, to handle adoption of a p-goal ϕ at level m, we add a new proposition containing the p-goal to the agent's goal hierarchy at m. The G-accessible paths at all levels above m are progressed as above. The G-accessible paths at level m are the ones that share the same history with $do(a, s)$ and over which ϕ holds. The G-accessible paths at all levels below m are the ones that can be obtained by progressing the level immediately above it. Thus the agent acquires the p-goal that ϕ at level m, and all the p-goals with priority m or less in s are pushed down one level in the hierarchy.

$$\text{Adopted}(p, n, m, a, s, \phi) \stackrel{\text{def}}{=}$$
if $(n < m)$ **then** $\text{Progressed}(p, n, a, s)$
else if $(n = m)$ **then** $\exists s'.\ \text{Starts}(p, s')$
$\wedge\ \text{SameHist}(s', do(a, s)) \wedge \phi(p)$
else $\text{Progressed}(p, n-1, a, s)$.

Finally, to handle the dropping of a p-goal ϕ, we replace the propositions that imply the dropped goal in the agent's goal hierarchy by the trivial proposition that the history of actions in the current situation has occurred. Thus, in addition to progressing all G-accessible paths as above, we add back all paths that share the same history with $do(a, s)$ to the existing G-accessibility levels where the agent has the p-goal that ϕ.

$$\text{Dropped}(p, n, a, s, \phi) \stackrel{\text{def}}{=} \textbf{if } \text{PGoal}(\phi, n, s)$$
then $\exists s'.\ \text{Starts}(p, s') \wedge \text{SameHist}(s', do(a, s))$
else $\text{Progressed}(p, n, a, s)$.

Returning to our example, recall that our agent has the c-goals/active p-goals in S_0 that □BeRich and □BeHappy, but not ◇GetPhD, since the latter is inconsistent with her higher priority p-goal □BeRich. Assume that, after the ac-

tion *goBankrupt* happens in S_0, the p-goal □BeRich becomes impossible. Then in $S_1 = do(goBankrupt, S_0)$, the agent has the c-goal that ◇GetPhD, but not □BeRich nor □BeHappy; □BeRich is excluded from the set of c-goals since it has become impossible to achieve (i.e. unrealistic). Also, since her higher priority p-goal ◇GetPhD is inconsistent with her p-goal □BeHappy, the agent will make □BeHappy inactive.

Note that, while it might be reasonable to drop a p-goal (e.g. ◇GetPhD) that is in conflict with another higher priority active p-goal (e.g. □BeRich), in our framework we keep such p-goals around. The reason for this is that although □BeRich is currently inconsistent with ◇GetPhD, the agent might later learn that □BeRich has become impossible to bring about (e.g. after *goBankrupt* occurs), and then might want to pursue ◇GetPhD. Thus, it is useful to keep these inactive p-goals since this allows the agent to maximize her utility (that of her chosen goals) by taking advantage of such opportunities. As mentioned earlier, c-goals are our analogue to intentions. Recall that Bratman's (1987) model of intentions limits the agent's practical reasoning – agents do not always optimize their utility and don't always reconsider all available options in order to allocate their reasoning effort wisely. In contrast to this, our c-goals are defined in terms of the p-goals, and at every step, we ensure that the agent's c-goals maximize her utility so that these are the set of highest priority goals that are consistent given the agent's knowledge. Thus, our notion of c-goals is not as persistent as Bratman's notion of intention. For instance as mentioned above, after the action *goBankrupt* happens in S_0, the agent will lose the c-goal that □BeHappy, although she did not drop it and it did not become impossible or achieved. In this sense, our model is that of an idealized agent. There is a tradeoff between optimizing the agent's chosen set of prioritized goals and being committed to chosen goals. In our framework, chosen goals behave like intentions with an automatic filter-override mechanism (Bratman 1987) that forces the agent to drop her chosen goals when opportunities to commit to other higher priority goals arise. In the future, it would be interesting to develop a logical model that captures the pragmatics of intention reconsideration by supporting control over it.

Properties

We now show that our formalization has some desirable properties. Some of these (e.g. Proposition 3a) are analogues of the AGM postulates; others (e.g. adopting logically equivalent goals has the same result, etc.) were left out for space reasons. First we show that c-goals are consistent:

Prop. 1 (Consistency) $D \models \forall s.\ \neg\mathrm{CGoal}(\mathrm{False}, s)$.

Thus, the agent cannot have both ϕ and $\neg\phi$ c-goals in a situation s. Even if all of the agent's p-goals become known to be impossible, the set of c-goal-accessible paths will be precisely those that starts with a K-accessible situation, and thus the agent will only choose the propositions that are known to be inevitable.

We also have the property of realism (Cohen and Levesque 1990), i.e. if an agent knows that something has become inevitable, then she has this as a c-goal:

Prop. 2 (Realism) $D \models \forall s.\ \mathrm{KInevitable}(\phi, s) \supset \mathrm{CGoal}(\phi, s)$.

Note that this is not necessarily true for p-goals and primary c-goals – an agent may know that something has become inevitable and not have it as her p-goal/primary c-goal, which is intuitive. While the property of realism is often criticized, one should view these inevitable goals as something that hold in the worlds that the agent intends to bring about, rather than something that the agent is actively pursuing.

A consequence of Proposition 1 and 2 is that an agent does not have a c-goal that is known to be impossible, i.e. $D \models \mathrm{CGoal}(\phi, s) \supset \neg\mathrm{KImpossible}(\phi, s)$.

We next discuss some properties of the framework w.r.t. goal change. Proposition 3 says that (a) an agent acquires the p-goal that ϕ at level n after she adopts it at n, and (b) that she acquires the primary c-goal that ϕ after she adopts it at some level n in s, provided that she does not have the c-goal in s that $\neg\phi$ next.

Prop. 3 (Adoption) (a) $D \models \mathrm{PGoal}(\phi, n, do(adopt(\phi, n), s))$, (b) $D \models \neg\mathrm{CGoal}(\neg\exists p'.\ \mathrm{Starts}(p, s') \wedge \mathrm{Suffix}(p', p, do(adopt(\phi, n), s')) \wedge \phi(p'), s) \supset \mathrm{PrimCGoal}(\phi, do(adopt(\phi, n), s))$.

Also, after dropping the p-goal that ϕ at n in s, an agent does not have the p-goal (and thus the primary c-goal) that the progression of ϕ at n, i.e. ϕ', provided that ϕ' is not inevitable in $do(drop(\phi), s)$.

Prop. 4 (Drop) $D \models \mathrm{PGoal}(\phi, n, s) \wedge [(\forall p, p'.\ \mathrm{Starts}(p, s') \wedge \mathrm{SameHist}(s', s) \wedge \mathrm{Suffix}(p', p, do(drop(\phi), s'))) \supset (\phi(p) \equiv \phi'(p'))] \wedge \neg\mathrm{Inevitable}(\phi', do(drop(\phi), s)) \supset \neg\mathrm{PGoal}(\phi', n, do(drop(\phi), s))$.

Note that, this does not hold for CGoal, as ϕ could still be a consequence of her remaining primary c-goals.

The next few properties concern the persistence of these motivational attitudes. First we have a persistence property for achievement realistic p-goals:

Prop. 5 (Persistence of Achievement RPGoals) $D \models \mathrm{RPGoal}(\Diamond\Phi, n, s) \wedge \mathrm{Know}(\neg\Phi, s) \wedge \forall\psi.\ a \neq drop(\psi) \supset \exists n'.\ \mathrm{RPGoal}(\Diamond\Phi, n', do(a, s))$.

This says that if an agent has a realistic p-goal that $\Diamond\Phi$ in s, then she will retain this realistic p-goal after some action a has been performed in s, provided that she knows that Φ has not yet been achieved, and a is not the action of dropping a p-goal. Note that, we do not need to ensure that $\Diamond\Phi$ is consistent with higher priority active p-goals, since the SSA for G does not automatically drop such incompatible p-goals from the goal hierarchy. Also, the level n where Φ is a p-goal may change, e.g. if the action performed is an adopt action with priority higher than or equal to n.

For achievement chosen goals we have the following:

Prop. 6 (Persistence of Achievement Chosen Goals) $D \models \mathrm{OPGoal}(\Diamond\Phi \wedge \exists s'.\ \mathrm{Starts}(s') \wedge \mathrm{SameHist}(s'), n, s) \wedge \mathrm{CGoal}(\Diamond\Phi, s) \wedge \mathrm{Know}(\neg\Phi, s) \wedge \forall\psi.\ a \neq drop(\psi) \wedge \forall\psi, m.\ \neg(a = adopt(\psi, m) \wedge m \leq n) \wedge \neg\mathrm{CGoal}(\neg\Diamond\Phi, n-1, do(a, s)) \supset \mathrm{CGoal}(\Diamond\Phi, n, do(a, s))$.

Thus, in situation s, if an agent has the only p-goal at level n that $\Diamond\Phi$ and that the correct history of actions in s has been performed, and if $\Diamond\Phi$ is also a chosen goal in s (and thus she has the primary c-goal that $\Diamond\Phi$), then she will retain the c-goal that $\Diamond\Phi$ at level n after some action a has been performed in s, provided that: she knows that Φ has not yet been achieved, that a is not the action of dropping a p-goal, that a is not the action of adopting a p-goal at some higher priority level than n or at n, and that at level $n-1$ the agent does not have the c-goal that $\neg\Diamond\Phi$, i.e. $\Diamond\Phi$ is consistent with higher priority c-goals.

Note that, this property also follows if we replace the consequent with $\text{CGoal}(\Diamond\Phi, do(a, s))$, and thus it deals with the persistence of c-goals. Note however that, it does not hold if we replace the OPGoal in the antecedent with PGoal; the reason for this is that the agent might have a p-goal at level n in s that ϕ and the c-goal in s that ϕ, but not have ϕ as a primary c-goal in s, e.g. n might be an inactive level because another p-goal at n has become impossible, and ϕ could be a c-goal in s because it is a consequence of two other primary c-goals. Thus even if $\neg\phi$ is not a c-goal after a has been performed in s, there is no guarantee that the level n will be active in $do(a, s)$ or that all the active p-goals that contributed to ϕ in s are still active.

Discussion and Future Work

While in our account chosen goals are closed under logical consequence, primary c-goals are not. Thus, our formalization of primary c-goals is related to the non-normal modal formalizations of intentions found in the literature (Konolige and Pollack 1993), and as such it does not suffer from the side-effect problem (Cohen and Levesque 1990).

Our framework can be extended to model subgoal adoption and the dependencies between goals and the subgoals and plans adopted to achieve them. The later is important since subgoals and plans adopted to bring about a goal should be dropped when the parent goal becomes impossible, is achieved, or is dropped. One way of handling this is to ensure that the adoption of a subgoal ψ w.r.t. a parent goal ϕ adds a new p-goal that contains *both this subgoal and this parent goal*, i.e. $\psi \wedge \phi$. This ensures that when the parent goal is dropped, the subgoal is also dropped, since when we drop the parent goal ϕ, we drop all the p-goals at all G-accessibility levels that imply ϕ including $\psi \wedge \phi$.

Also, since we are using the situation calculus, we can easily represent procedural goals/plans, e.g. the goal to do a_1 and then a_2 can be written as: $\text{PGoal}(\text{Starts}(p, s_1) \wedge \text{OnPath}(p, s) \wedge s = do(a_2, do(a_1, s_1)), 0, S_0)$. Golog (Reiter 2001) can be used to represent complex plans/programs. So we can model the adoption of plans as subgoals.

Recently, there have been a few proposals that deal with goal change. Shapiro *et al.* (2007) present a situation calculus based framework where an agent adopts a goal when she is requested to do so, and remains committed to this goal unless the requester cancels this request; a goal is retained even if the agent learns that it has become impossible, and in this case the agent's goals become inconsistent. Shapiro and Brewka (2007) modify this framework to ensure that goals are dropped when they are believed to be impossible or when they are achieved. Their account is similar to ours in the sense that they also assume a priority ordering over the set of (in their case, requested) goals, and in every situation they compute chosen goals by computing a maximal consistent goal set that is also compatible with the agent's beliefs. However, their model has some unintuitive properties: the agent's chosen set of goals in $do(a, s)$ may be quite different from her goals in s, although a did not make any of her goals in s impossible or inconsistent with higher priority goals, because inconsistencies between goals at the same priority level are resolved differently (this can happen because goals are only partially ordered). Also, we provide a more expressive formalization of prioritized goals – we model goals using infinite paths, and thus can model many types of goals that they cannot. Finally they model prioritized goals by treating the agent's p-goals as an arbitrary set of temporal formulas, and then defining the set of c-goals as a subset of the p-goals. But our possible world semantics has some advantages over this: it clearly defines when goals are consistent with each other and with what is known. One can easily specify how goals change when an action a occurs, e.g. the goal to do a next and then do b becomes the goal to do b next, the goal that $\Diamond\Phi \vee \Diamond\Psi$ becomes the goal that $\Diamond\Psi$ if a makes achieving Φ impossible, etc.

Most approaches to agent programming languages with declarative goals are not based on a formal theory of agency, and to the best of our knowledge none deals with temporally extended goals or maintain the consistency of (chosen) goals.

One limitation of our account is that one could argue that our agent wastes resources trying to optimize her c-goals at every step. In the future, we would like to develop an account where the agent is strongly committed to her chosen goals, and where the filter override mechanism is only triggered under specific conditions.

References

M. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press, Cambridge, 1987.

P. Cohen and H. Levesque. Intention is Choice with Commitment. *Artificial Intelligence*, 42(2–3):213–361, 1990.

K. Konolige and M. Pollack. A Representationalist Theory of Intention. In *Thirteenth Intl. J. Conf. on Artificial Intelligence (IJCAI-93)*, pp. 390–395, Chambéry, France, 1993.

H. Levesque, F. Pirri, and R. Reiter. Foundations for a Calculus of Situations. *Electronic Transactions of AI (ETAI)*, 2(3–4):159–178, 1998.

A. Rao and M. Georgeff. Modeling Rational Agents with a BDI-Architecture. In R. Fikes and E. Sandewall, editors, *Second Intl. Conf. on Principles of Knowledge Rep. and Reasoning*, pp. 473–484, 1991.

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

R. Scherl and H. Levesque. Knowledge, Action, and the Frame Problem. *Artificial Intelligence*, 144(1–2), 2003.

S. Shapiro and G. Brewka. Dynamic Interactions Between Goals and Beliefs. In *Twentieth Intl. J. Conf. on Artificial Intelligence (IJCAI-07)*, pp. 2625–2630, India, 2007.

S. Shapiro, Y. Lespérance, and H. Levesque. Goals and Rational Action in the Situation Calculus - A Preliminary Report. In *Working Notes of the AAAI Fall Symp. on Rational Agency*, pp. 117–122, 1995.

S. Shapiro, Y. Lespérance, and H. Levesque. Goal Change in the Situation Calculus. *J. of Logic and Computation*, 17(5):983–1018, 2007.