Metacognition for Dropping and Reconsidering Intentions *

Darsana P. Josyula[1], **Michael L. Anderson**[2] and **Don Perlis**[1,2]
(1) Department of Computer Science
(2) Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742
{darsana, anderson, perlis}@cs.umd.edu

Abstract

In this paper, we present a meta-cognitive approach for dropping and reconsidering intentions, wherein concurrent actions and results are allowed, in the framework of the time-sensitive and contradiction-tolerant active logic.

Introduction

Intentions are commitments for future course of actions; when an agent adopts an intention to execute an action, it makes a commitment to perform that action. Therefore, intentions usually resist being dropped or reconsidered in normal situations. However, in a dynamic world wherein the environment can change any time, intentions may have to be dropped or reconsidered (Bratman 1987; 1999). For instance, an agent may create an intention to *wash the car on Saturday*. In normal situations, this will cause the agent to actually execute the action of washing the car on Saturday. But, suppose the car gets wrecked on Friday; this event has to cause the agent to drop the original intention to wash the car on Saturday. Similarly, an agent's intention to *wash the car on Saturday at 10:00 am* may have to reconsidered (and re-scheduled) if the agent has to *drive to New York at 10:00 am on Saturday*, assuming that washing the car and driving the car are not activities that can be performed concurrently.

Intention reconsideration may need to occur, not only before the action associated with an intention has been initiated (as in the examples above), but also possibly after the action has been invoked. For instance, an intention to listen to music, may cause the action of switching on the radio; but if the radio is not working, the intention remains unachieved and hence the intention needs to be reconsidered. This reconsideration may result in switching on the CD player or the tape player instead of the radio.

Selecting a good policy for dropping and reconsidering intentions is extremely important for any resource bounded agent. For, as noted in (Bratman, Israel, & Pollack 1988), reconsidering intentions too often can be too costly from a computational point of view, while not reconsidering intentions can produce incorrect results (or unwanted behaviors) in a dynamic world. Experimental studies (Kinny & Georgeff 1991; Schut & Wooldridge 2000) have shown that *cautious agents*—those that reconsider intentions at every possible opportunity—outperform *bold agents*—those that reconsider intentions only after executing the current set of intentions—in highly dynamic worlds, while the reverse is true for static worlds. Since the dynamicity of an environment can vary from time to time, it is important to have agents that are neither bold nor cautious, but adaptive—that is, agents that can adapt their commitment levels at run time.

One way of creating such an adaptive agent is to have a metacognitive process that "watches" the agent's current intentions, notes them as achievable, unachievable or achieved and drops those that have been achieved or are known to be unachievable in the future. Thus, the metacognitive process strives towards maintaining an achievable set of intentions for an agent to act on. And, the agent commits to act only on this achievable set of intentions. As a result, an agent may behave boldly or cautiously depending on the particular environment in which it is situated, and the amount of resources it has for reasoning. When there are many intentions for an agent to reason about in a limited time, it will behave boldly whereas when the agent has very few intentions to act on and it has more time in hand, it will behave cautiously.

If overlapping and concurrent actions are not allowed, then determining the set of achievable intentions primarily involves choosing intentions based on which ones have had their preconditions met. However, in many situations it is not practical to disallow overlapping and concurrent actions. For, certain effects can result only when actions are done together. A classic example is that of "clapping"; to produce a particular sound effect, the actions of moving the left hand towards the right and moving the right hand towards the left are to be done simultaneously. Although performing some actions concurrently is desirable, doing some other actions concurrently can lead to disaster. For instance, switching on two heaters in a room can warm the room quicker; however kicking with both legs can result in a fall. This is because of certain actions interfering with the preconditions of other actions as noted in (Reiter 1996). For instance, the precondition of kicking with the right leg is that one should not be kicking with the left leg at the same time. Similarly, the effects of actions can interfere with one another (Boutilier

*This research was supported in part by AFOSR and ONR.
Copyright © 2005, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

& Brafman 2001), as in pushing and pulling an object at the same time. Therefore, when overlapping and concurrent actions are allowed, implementing a metacognitive process that can choose the set of achievable intentions becomes tougher.

In this paper, we discuss how this issue is tackled within the framework of the time-sensitive active logic. The following sections examine some key features of active logic and how these features help create an agent that can make decisions about dropping and reconsidering its intentions at run time, even in situations where concurrent actions and results are allowed.

Active Logic

Active logics (Elgot-Drapkin & Perlis 1990), are a family of formalisms that combine inference rules with a constantly evolving measure of time (a 'now') that itself can be referenced in those rules; thus the evolving knowledge base is naturally integrated into the ongoing reasoning processes. Each "step" in an active logic proof takes one active logic time-step; thus inference always moves into the future at least one step. An active logic usually has an observation function that incorporates new formulas into the logic at any step.

To achieve much of their reasoning, active logics employ a notion of "now" that is constantly updated by a "clock rule" (1), which states that from the fact that it is step t at the current step, the step number of the next step is $t + 1$.

$$\frac{t \quad : \quad \text{now}(t)}{t+1 \quad : \quad \text{now}(t+1)} \tag{1}$$

Since the notion of inference in active logics is time-dependent, only those inferences that have actually been carried out so far can affect the present state of the agent's knowledge. As a result, even if directly contradictory wffs, P and ¬P, are in the agent's KB at time t, it need not be the case that those wffs have been used by time t to derive any other wff, Q. Indeed, it may be that t is the £rst moment at which both P and ¬P occur simultaneously in the KB.

By endowing an active logic with a "con¤ict-recognition" inference rule such as that in (2), *direct* contradictions can be recognized as soon as they occur.

$$\frac{t \quad : \quad P, \neg P}{t+1 \quad : \quad \text{contra}(i, P, \neg P)} \tag{2}$$

In active logic, most formulas (exceptions include the ones related to the current time) in one step that are not directly contradicting are inherited to the next step. This is controlled by inheritance rules. One simple version of such an "inheritance rule" which also illustrates the use of £ring conditions, is shown in (3).

$$\frac{t \quad : \quad A \quad [condition : \neg A \notin KB, A \neq \text{now}(t)]}{t+1 \quad : \quad A} \tag{3}$$

In active logic, negative introspection—the ability to determine that one does *not* know something—is often encoded as the following inference rule (4), where the notation ...[B] means that B is not present.

$$\frac{t \quad : \quad ...[B]}{t+1 \quad : \quad \neg Know(t, B)} \tag{4}$$

Active logic maintains a temporal history of its reasoning process that can be used by the logic for further reasoning. The history enables the logic to determine when each formula was added or deleted in the past.

Each formula in an active logic has an associated name and the logic permits an individual formula to be referenced by its name. This quotation feature together with the history mechanism allows the logic to reason about its own past reasoning.

Finally, the logic has a special proposition, *call*, which if proven, will initiate external action that can be reasoned about and tracked through observation.

Implementation

Alma/Carne (Purang 2001) is a general purpose implementation of active logic. It has a dual role—(i) acting as the language to specify active logic based applications and (ii) providing the core reasoning engine for these applications.

In its role as a language, Alma/Carne allows applications to be speci£ed as a set of logical sentences and procedures. When the sentences are loaded into Alma and the procedures into Carne, Alma/Carne takes the role of a reasoning engine. In this role, Alma generates active logic inferences, some of which trigger procedures in Carne. These procedures can perform some computation or cause effects in the world. Alma's state is updated with the status of the procedures (e.g.. $done$, $doing$) which enables reasoning about the processes Alma triggered. Failure of a procedure, for instance, can lead to reasoning that causes retraction of earlier assumptions. Carne can also monitor the world and assert formulas about the state of the world into Alma, implementing the observation functionality of active logic. This enables Alma to react to changes in the world. Thus Alma/Carne can initiate, observe and respond to external events and non-logical processes.

The notation used in this paper to represent some of the symbols in the Alma/Carne language is given in Table 1.

ALFA

ALFA—Active Logic For Agents—is a reasoner based on Alma/Carne with representations for desires, intentions, expectations and achievements. In general, an agent based on

Table 1: Alma Symbols, Description and Notation used

Alma Symbol	Description	Notation
and	classical and operator	$\wedge$
or	classical or operator	$\vee$
$forall$	forall operator	$\forall$
not	negation operator	not
$\backslash+$	negation on failure	$\neg$
if	classical if operator	$\rightarrow$
fif	forward chaining if	$\leftarrowtail$
bif	backward chaining if [a]	$\rightarrowtail$
pos_int	positive introspection	pos_int
bs	breadth £rst search [b]	bs
df	deletes formula	df
$contra$	contradiction	contra
$reinstate$	reinstate formula	reinstate

[a] Triggers backward chaining, while doing a breadth-£rst search
[b] Uses bif formulas to conduct the search

ALFA can have desires to achieve sets of concurrent actions or results at different times. ALFA transforms achievable desires to intentions and achievable intentions to actions. Actions cause changes to different properties of objects and these properties need to be observed to determine whether the changes have been effective. For this, ALFA creates expectations regarding the new value of the property, and these expectations may then get transformed into desires to actually observe the value of that property. For instance, performing the action—*turning the heater on*—causes an expectation that *the water temperature increases*; this expectation causes a new desire to *observe the water temperature*.

In ALFA, an intention for θ, where θ is a set of either concurrent actions or simultaneous results, is represented as $\mathrm{intention}(Id, Type, \theta, T_1, T_2)$. Here, Id is a unique identi£er that distinguishes different intentions and $Type$ is a constant—$action$ or $result$—depending on whether θ denotes a set of actions to be performed concurrently or a set of results to be obtained simultaneously. T_1 and T_2 denote the time period in which the actions are to be performed or the results need to be obtained. The different values that T_1 and T_2 can take include all non-negative integers.

Preconditions for actions and results are represented in ALFA as $\mathrm{precondition}(\kappa, \theta')$ where θ' is either an action or a result and κ is either $\mathrm{pos\_int}(F)$ or $\neg\,\mathrm{pos\_int}(F)$, where F is any formula.

In ALFA, the effect of a set of concurrent actions α is represented as $effect(\alpha, \beta', \tau_1, \tau_2)$, where β' is either $has(Obj, Prop, Val)$ or $observation(Obj, Prop)$ depending on whether α causes a change in the property of an object or an observation of a property of an object. τ_1 and τ_2 together specify the duration for which the result can be observed; that is, β' can be observed for τ_2 steps, starting τ_1 steps after the step at which the actions in α have been invoked. $\tau_2 = 0$ indicates that the effect persists.

ALFA maintains an achievable set of intentions for the agent to act on, with the help of a metacognitive process that notes intentions as achievable or unachievable. The different components of this metacognitive process are discussed in some detail below. In the formulas discussed below, t refers to the current time step; that is, $\mathrm{now}(t)$ evaluates to true.

Achievable vs Unachievable Intentions

ALFA makes a determination about whether an intention is achievable or not, based on the following:

a. if any precondition of the action or result associated with an intention no longer holds, then that intention is unachievable.

b. intentions that interfere with each other are unachievable.

c. all other intentions are achievable.

ALFA implements condition [a] by rules (5), (6), (7) and (8), [b] by (9) and [c] by rules (10) and (11).

Unachievable Intentions If θ has not been performed or produced before step T_2, then the corresponding intention is marked as unachievable using rule (5).

$$\begin{aligned}&\mathrm{intention}(Id, Type, \theta, T_1, T_2) \wedge (T_2 < t) \wedge \\ &\mathrm{form\_to\_name}(\mathrm{intention}(Id, Type, \theta, T_1, T_2), N) \\ &\qquad \leftarrowtail \mathrm{not}(\mathrm{achievable}(N))\end{aligned} \tag{5}$$

Formula (6), speci£es that an intention for a set of actions or results cannot be achieved if the preconditions for the actions or results associated with the intention do not hold.

$$\begin{aligned}&\mathrm{intention}(Id, Type, \theta, T_1, T_2) \wedge \exists_{\theta' \in \theta} \\ &[\mathrm{precondition}(\kappa, \theta') \wedge \neg\kappa] \wedge \\ &\mathrm{form\_to\_name}(\mathrm{intention}(Id, Type, \theta, T_1, T_2), N) \\ &\qquad \leftarrowtail \mathrm{not}(\mathrm{achievable}(N))\end{aligned} \tag{6}$$

An intention to obtain a concurrent set of results β cannot be achieved if the agent does not know how to cause some $\beta' \in \beta$. This is speci£ed as rule (7) below.

$$\begin{aligned}&\mathrm{intention}(Id, result, \beta, T_1, T_2) \wedge \\ &\exists_{\beta' \in \beta}[\neg\,\mathrm{pos\_int}(\mathrm{effect}(\_, \beta', \_, \_))] \wedge \\ &\mathrm{form\_to\_name}(\mathrm{intention}(Id, result, \beta, T_1, T_2), N) \\ &\qquad \leftarrowtail \mathrm{not}(\mathrm{achievable}(N))\end{aligned} \tag{7}$$

Rule (8) states that an intention for a set of concurrent results β is unachievable, if there is some $\beta' \in \beta$ that cannot be achieved before T_2.

$$\begin{aligned}&\mathrm{intention}(Id, result, \beta, T_1, T_2) \wedge \\ &\exists_{\beta' \in \beta}[\forall_{\alpha}[\mathrm{effect}(\alpha, \beta', \tau_1, \tau_2) \rightarrow ((t + \tau_1 > T_2) \\ &\wedge \exists_{\alpha' \in \alpha}[\neg\,\mathrm{pos\_int}(\mathrm{doing}(\alpha', Id)) \wedge \\ &\neg\,\mathrm{pos\_int}(\mathrm{done}(\alpha', Id))])]] \wedge \\ &\mathrm{form\_to\_name}(\mathrm{intention}(Id, result, \beta, T_1, T_2), N) \\ &\qquad \leftarrowtail \quad \mathrm{not}(\mathrm{achievable}(N))\end{aligned} \tag{8}$$

Rules (7) and (8) assume that the set of results β has to be achieved on an all or none basis.

Rule (9) marks intentions that interfere either with itself or with other intentions as unachievable.

$$\begin{aligned}&\text{intention}(Id, Type, \theta, T_1, T_2) \wedge \\ &\text{form\_to\_name}(\text{intention}(Id, Type, \theta, T_1, \\ &T_2), N) \wedge \text{bs}(\text{interferes}(N, \_)) \\ &\quad \looparrowright \text{not}(\text{achievable}(N))\end{aligned} \tag{9}$$

Achievable Intentions An intention for a set of actions α is achievable, if the preconditions for each action $\alpha' \in \alpha$ hold, the time at which the actions in α are to be invoked has not elapsed and the intention does not interfere with any other intention. This is specified by rule (10).

$$\begin{aligned}&\text{intention}(Id, action, \alpha, T_1, T_2) \wedge (t < T_2) \wedge \\ &\forall_{\alpha' \in \alpha} [\forall_{\kappa} [\text{precondition}(\kappa, \alpha') \rightarrow \kappa]] \wedge \\ &\text{form\_to\_name}(\text{intention}(Id, action, \alpha, \\ &T_1, T_2), N) \wedge \neg \text{pos\_int}(\text{interferes}(N, \_)) \\ &\quad \looparrowright \text{achievable}(N)\end{aligned} \tag{10}$$

If there is an intention to achieve a set of results β, the set of actions α that can cause β is known, all the preconditions for α as well as β hold, and α can cause β before T_2, then that intention is marked as achievable using rule (11).

$$\begin{aligned}&\text{intention}(Id, result, \beta, T_1, T_2) \wedge \\ &\forall_{\beta' \in \beta} [\forall_{\kappa} [\text{precondition}(\kappa, \beta') \rightarrow \kappa] \wedge \\ &\exists_{\alpha} [\text{effect}(\alpha, \beta', \tau_1, \tau_2) \wedge (t + \tau_1 \leq T_2) \wedge \\ &\forall_{\alpha' \in \alpha} \forall_{\kappa} [\text{precondition}(\kappa, \alpha') \rightarrow \kappa]]] \wedge \\ &\text{form\_to\_name}(\text{intention}(Id, result, \beta, \\ &T_1, T_2), N) \wedge \neg \text{pos\_int}(\text{interferes}(N, \_)) \\ &\quad \looparrowright \text{achievable}(N)\end{aligned} \tag{11}$$

Interfering intentions

An intention is self-interfering if :

d. it has actions with contradictory effects or

e. it has results that are contradictory

Also, two intentions interfere with each other if any of the following conditions hold:

f. the actions associated with the intentions have contradictory effects.

g. the results associated with the intentions are contradictory.

h. an action associated with one intention has an effect that contradicts with the result associated with the other intention.

In ALFA, rules (12) and (13) implement conditions [d] and [e] respectively, while rules (14), (15) and (16) implement conditions f, g and h respectively.

An intention for a set of actions α interferes with itself, if there are subsets $\bar{\alpha}_1$ and $\bar{\alpha}_2$ for α such that β_1 is the effect of $\bar{\alpha}_1$ and β_2 is the effect of $\bar{\alpha}_2$, and these effects (β_1 and β_2) oppose each other and their durations overlap. This is given by formula (12).

$$\begin{aligned}&\text{interferes}(N, N) \leftarrowtail \\ &\quad \text{intention}(Id, action, \alpha, T_1, T_2) \wedge \text{effect}(\bar{\alpha}_1, \beta_1, \\ &\quad \tau_{1_1}, \tau_{2_1}) \wedge (\bar{\alpha}_1 \subseteq \alpha) \wedge \text{effect}(\bar{\alpha}_2, \beta_2, \tau_{1_2}, \tau_{2_2}) \wedge \\ &\quad (\bar{\alpha}_2 \subseteq \alpha) \wedge \text{bs}(\text{contra\_effects}(\beta_1, \beta_2)) \wedge \\ &\quad \text{bs}(\text{overlaps}(T_1 + \tau_{1_1}, T_2 + \tau_{1_1} + \tau_{2_1}, \\ &\quad T_1 + \tau_{1_2}, T_2 + \tau_{1_2} + \tau_{2_2}) \wedge \\ &\quad \text{form\_to\_name}(\text{intention}(Id, action, \alpha, T_1, T_2), N)\end{aligned} \tag{12}$$

Formula (13) specifies that an intention for a set of results β interferes with itself, if $\beta_1 \in \beta$ opposes $\beta_2 \in \beta$.

$$\begin{aligned}&\text{interferes}(N, N) \leftarrowtail \\ &\quad \text{intention}(Id, result, \beta, T_1, T_2) \wedge \\ &\quad \exists_{\beta_1 \in \beta} \exists_{\beta_2 \in \beta} [\text{bs}(\text{contra\_effects}(\beta_1, \beta_2))] \wedge \\ &\quad \text{form\_to\_name}(\text{intention}(Id, result, \beta, T_1, T_2), N)\end{aligned} \tag{13}$$

An intention for a set of actions α_1 interferes with another intention for a set of actions α_2 if there are subsets $\bar{\alpha}_1$ and $\bar{\alpha}_2$ for sets α_1 and α_2 respectively, such that β_1 is the effect of $\bar{\alpha}_1$ and β_2 is the effect of $\bar{\alpha}_2$, and these effects (β_1 and β_2) oppose each other and their durations overlap. This is given by formula (14).

$$\begin{aligned}&\text{interferes}(N_1, N_2) \leftarrowtail \\ &\quad \text{intention}(Id_1, action, \alpha_1, T_{1_1}, T_{2_1}) \wedge \text{effect}(\bar{\alpha}_1, \\ &\quad \beta_1, \tau_{1_1}, \tau_{2_1}) \wedge (\bar{\alpha}_1 \subseteq \alpha_1) \wedge \text{intention}(Id_2, \\ &\quad action, \alpha_2, T_{1_2}, T_{2_2}) \wedge \text{effect}(\bar{\alpha}_2, \beta_2, \tau_{1_2}, \tau_{2_2}) \wedge \\ &\quad (\bar{\alpha}_2 \subseteq \alpha_2) \wedge \text{bs}(\text{overlaps}(T_{1_1} + \tau_{1_1}, \\ &\quad T_{2_1} + \tau_{1_1} + \tau_{2_1}, T_{1_2} + \tau_{1_2}, T_{2_2} + \tau_{1_2} + \tau_{2_2}) \wedge \\ &\quad \text{bs}(\text{contra\_effects}(\beta_1, \beta_2)) \wedge \\ &\quad \text{form\_to\_name}(\text{intention}(Id_1, action, \alpha_1, T_{1_1}, \\ &\quad T_{2_1}), N_1) \wedge \text{form\_to\_name}(\text{intention}(Id_2, \\ &\quad action, \alpha_2, T_{1_2}, T_{2_2}), N_2)\end{aligned} \tag{14}$$

Formula (15) specifies that an intention for a set of results $\bar{\beta}_1$ interferes with another intention to achieve a set of results $\bar{\beta}_2$, if the time periods for the intentions overlap and $\beta_1 \in \bar{\beta}_1$ opposes $\beta_2 \in \bar{\beta}_2$.

$$\begin{aligned}&\text{interferes}(N_1, N_2) \leftarrowtail \\ &\quad \text{intention}(Id_1, result, \bar{\beta}_1, T_{1_1}, T_{2_1}) \wedge \\ &\quad \text{intention}(Id_2, result, \bar{\beta}_2, T_{1_2}, T_{2_2}) \wedge \\ &\quad \text{bs}(\text{overlaps}(T_{1_1}, T_{2_1}, T_{1_2}, T_{2_2}) \wedge \\ &\quad \exists_{\beta_1 \in \bar{\beta}_1} \exists_{\beta_2 \in \bar{\beta}_2} [\text{bs}(\text{contra\_effects}(\beta_1, \beta_2))] \wedge \\ &\quad \text{form\_to\_name}(\text{intention}(Id_1, result, \bar{\beta}_1, T_{1_1}, \\ &T_{2_1}), N_1) \wedge \text{form\_to\_name}(\text{intention}(Id_2, \\ &\quad result, \bar{\beta}_2, T_{1_2}, T_{2_2}), N_2)\end{aligned} \tag{15}$$

An intention for a set of actions α interferes with another intention for a set of results β if some effect β_1 that α produces opposes some result β_2 in β and the duration of β_1 overlaps with the duration of β_2. This is given by rule (16).

$$\begin{aligned}&\text{interferes}(N_1, N_2) \hookleftarrow \\ &\quad \text{intention}(Id_1, action, \alpha, T_{1_1}, T_{2_1}) \wedge \\ &\quad \text{effect}(\bar{\alpha}, \beta_1, \tau_{1_1}, \tau_{2_1}) \wedge (\bar{\alpha} \subseteq \alpha) \wedge \\ &\quad \text{intention}(Id_2, result, \beta, T_{1_2}, T_{2_2}) \wedge \\ &\quad \exists_{\beta_2 \in \beta}[\text{bs}(\text{contra\_effects}(\beta_1, \beta_2))] \wedge \\ &\quad \text{bs}(\text{overlaps}(T_{1_1} + \tau_{1_1}, T_{2_1} + \tau_{1_1} + \tau_{2_1}, T_{1_2}, T_{2_2})) \wedge \\ &\quad \text{form\_to\_name}(\text{intention}(Id_1, action, \alpha, T_{1_1}, \\ &\quad T_{2_1}), N_1) \wedge \text{form\_to\_name}(\text{intention}(Id_2, \\ &\quad result, \beta, T_{1_2}, T_{2_2}), N_2)\end{aligned} \tag{16}$$

Contradictory Effects

Two effects or results are contradictory if they oppose each other and their durations overlap. The conditions for two time periods to overlap is specified by formula (17).

$$\begin{aligned}&\text{overlaps}(T_1, T_2, T_3, T_4) \hookleftarrow \\ &\quad \neg(T_1 < T_2 \wedge T_3 \leq T_4 \wedge T_2 < T_3) \wedge \\ &\quad \neg(T_1 \leq T_2 \wedge T_3 < T_4 \wedge T_4 < T_1)\end{aligned} \tag{17}$$

To determine whether two results or effects oppose each other, ALFA makes use of rule (18).

$$\begin{aligned}&\text{contra\_effects}(\beta_1, \beta_2) \hookleftarrow \\ &\quad \beta_1 = \text{has}(Obj, Prop, Val_1) \wedge \beta_2 = \text{has}(Obj, \\ &\quad Prop, Val_2) \wedge \text{bs}(\text{oppose}(Obj, Prop, Val_1, Val_2))\end{aligned} \tag{18}$$

Examples of axioms that represent opposing results (effects) are shown in (19).

$$\text{oppose}(\_, \_, increase, decrease) \tag{19a}$$

$$\text{oppose}(\_, \_, move\_left, move\_right) \tag{19b}$$

Achieved Intentions

Once Carne asserts that an action α has been completed, the intention that caused α is noted as achieved using (20).

$$\begin{aligned}&\text{intention}(Id, action, \alpha, T_1, T_2) \wedge (t < T_2) \wedge \\ &\forall_{\alpha' \in \alpha}[done(\alpha', Id)] \wedge \\ &\text{form\_to\_name}(\text{intention}(Id, action, \alpha, T_1, T_2), N) \\ &\quad \looparrowright \text{achieved}(N)\end{aligned} \tag{20}$$

An intention to change the property $Prop$ of an object Obj to value Val is noted as achieved if there is an observation which shows that property $Prop$ indeed has the value Val. Similarly, an achievable intention to observe the value of property $Prop$ of object Obj is marked as achieved if there is an observation that provides the value Val for property $Prop$. Both these scenarios are addressed by rule (21)

$$\begin{aligned}&\text{intention}(Id, result, \beta, T_1, T_2) \wedge \forall_{\beta' \in \beta}[(\beta' = \\ &has(Obj, Prop, Val) \vee \beta' = observation(Obj, \\ &Prop)) \wedge \text{pos\_int}(observed(Obj, Prop, Val, \tau)) \wedge \\ &(T_1 \leq \tau) \wedge (T_2 \geq \tau)] \wedge \\ &\text{form\_to\_name}(\text{intention}(Id, result, \beta, T_1, T_2), N) \\ &\quad \looparrowright \text{achieved}(N)\end{aligned} \tag{21}$$

Contradictory Predicates

Whenever, there is a contradiction (e.g., achievable vs unachievable or intention vs not an intention) ALFA rules in favor of the knowledge that it attains latter using the contradiction handling rules (22) and (23).

$$\begin{aligned}&\text{name\_to\_time}(N_1, T_1) \wedge \text{name\_to\_time}(N_2, T_2) \wedge \\ &(T_1 \geq T_2) \wedge (\text{contra}(N_1, N_2, \_) \vee \text{contra}(N_2, N_1, \_)) \\ &\quad \looparrowright \text{reinstate}(N_1)\end{aligned} \tag{22}$$

$$\begin{aligned}&\text{name\_to\_time}(N_1, T_1) \wedge \text{name\_to\_time}(N_2, T_2) \wedge \\ &(T_1 < T_2) \wedge (\text{contra}(N_1, N_2, \_) \vee contra(N_2, N_1, \_)) \\ &\quad \looparrowright \text{reinstate}(N_2)\end{aligned} \tag{23}$$

Reconsidering Intentions

In a dynamic world, intentions can change from being unachievable to achievable or vice-versa. For instance, if a particular intention is not achievable because of the lack of knowledge about how to achieve a result, then the addition of such knowledge will make the intention achievable. The contradiction handling rules (22) and (23) can aid this transformation. Since the agent acts on those intentions that it presumes achievable, intentions that get transformed from unachievable to achievable by the metacognitive process get automatically acted upon by the cognitive process.

If an intention for a set of actions is unachievable and if the agent had originally created this intention for achieving a set of results, then the metacognitive process tries to create an achievable intention to attain the same results. We omit the specific rules that implement this scenario for lack of space.

Dropping Intentions

ALFA maintains an intention as long as it holds (i.e., the intention is true but not its negation), it is considered achievable and it has not been achieved so far. Thus, an agent based on ALFA can behave like an *open-minded agent* (Rao & Georgeff 1991; 1992). Rules (24), (25) and (26) implement this behavior.

In ALFA, intentions that have been achieved are dropped by (24).

$$\begin{aligned}&\text{achieved}(N) \\ &\quad \looparrowright \text{df}(N)\end{aligned} \tag{24}$$

Intentions that are unachievable in future are dropped by (25).

$$\text{not}(\text{achievable}(N)) \wedge \text{name\_to\_formula}(N, \text{intention}(Id, \theta, T_1, T_2)) \wedge t > T_2 \quad (25)$$
$$\looparrowright \text{df}(N)$$

ALFA, drops intentions that are no longer goals for the agent using formula (26).

$$\text{not}(\text{intention}(Id, \theta, T_1, T_2)) \wedge \text{form\_to\_name}(\text{intention}(Id, \theta, T_1, T_2), N) \quad (26)$$
$$\looparrowright \text{df}(N)$$

Conclusion

The importance of meta-level control of deliberation for resource-bounded agents situated in dynamic domains has been discussed in (Kinny & Georgeff 1991; Wooldridge & Parsons 1999). The meta-level decision theoretic approach (Schut & Wooldridge 2001; Wooldridge & Parsons 1999; Parsons *et al.* 2000; Schut, Wooldridge, & Parsons 2004) allows an agent, to choose its policy for intention reconsideration at run-time, provided the frequency at which the environment changes is known before hand. However, in a dynamic world, this frequency may not be static and hence cannot be pre-determined. Besides, it is not clear how the approach extends to an agent that can have intentions to perform concurrent actions as well as achieve concurrent results.

In this paper, we have presented a more general approach for dropping and reconsidering intentions, wherein concurrent actions and results are allowed, based on the time-sensitive active logic. Our approach uses a metacognitive process, that dynamically marks intentions as achievable, unachievable or achieved, drops futile or achieved intentions and creates alternative intentions for currently unachievable intentions when possible. Since, this process runs concurrently with the cognitive activities of the agent, the amount of resources available to it, depends on real-time conditions. Therefore, when time is limited, the metacognitive process might not mark an intention as unachievable on time; consequently, the resulting agent may actually try to act on that intention. Thus, at times the agent may act too boldly while at other times it may act too cautiously, but the idea is that, it adapts its commitment level based on the conditions and resources available at run-time.

References

Boutilier, C., and Brafman, R. I. 2001. Partial-order planning with concurrent interacting actions. *Journal of Arti£cial Intelligence Research* 14:105–136.

Bratman, M.; Israel, D.; and Pollack, M. 1988. Plans and Resource-bounded Practical Reasoning. *Computational Intelligence* 4(4):349–355.

Bratman, M. E. 1987. *Intention, Plans and Practical Reason*. Massachusetts, USA: Harvard University Press.

Bratman, M. E. 1999. *Faces of Intention: Selected Essays on Intention and Agency*. Cambridge, UK: Cambridge University Press. 93–161.

Elgot-Drapkin, J., and Perlis, D. 1990. Reasoning Situated in Time I: Basic Concepts. *Journal of Experimental and Theoretical Arti£cial Intelligence* 2(1):75–98.

Kinny, D. N., and Georgeff, M. P. 1991. Commitment and effectiveness of situated agents. In *Proceedings of the Twelfth International Joint Conference on Arti£cial Intelligence (IJCAI-91)*, 82–88.

Parsons, S.; Pettersson, O.; Saf£otti, A.; and Wooldridge, M. 2000. Intention reconsideration in theory and practice. In Horn, W., ed., *Proceedings of the Fourteenth European Conference on Arti£cial Intelligence (ECAI-2000)*. John Wiley.

Purang, K. 2001. Alma/Carne: Implementation of a Time-situated Meta-reasoner. In *Proceedings of the Thirteenth International Conference on Tools with Arti£cial Intelligence (ICTAI-01)*, 103–110.

Rao, A. S., and Georgeff, M. P. 1991. Modeling Rational Agents within a BDI-Architecture. In Allen, J.; Fikes, R.; and Sandewall, E., eds., *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR-91)*, 473–484. Morgan Kaufmann.

Rao, A. S., and Georgeff, M. P. 1992. An Abstract Architecture for Rational Agents. In Rich, C.; Swartout, W.; and Nebel, B., eds., *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR-92)*, 439–449. Morgan Kaufmann.

Reiter, R. 1996. Natural Actions, Concurrency and Continuous Time in the Situation Calculus. In *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR -96)*, 2–13.

Schut, M. C., and Wooldridge, M. 2000. Intention reconsideration in complex environments. In Sierra, S.; Gini, M.; and Rosenschein, J., eds., *Proceedings of the Fourth International Conference on Autonomous Agents (AGENTS - 00)*, 209–216. ACM Press.

Schut, M., and Wooldridge, M. 2001. Principles of intention reconsideration. In Müller, J. P.; Andre, E.; Sen, S.; and Frasson, C., eds., *Proceedings of the Fifth International Conference on Autonomous Agents - (AGENTS-01)*, 340–347. ACM Press.

Schut, M. C.; Wooldridge, M. J.; and Parsons, S. 2004. The theory and practice of intention reconsideration. *Journal of Experimental and Theoretical Arti£cial Intelligence* 16(4):261–293.

Wooldridge, M., and Parsons, S. 1999. Intention reconsideration reconsidered. In Müller, J. P.; Singh, M. P.; and Rao, A. S., eds., *LNAI - Proceedings of the 5th International Workshop on Intelligent Agents V : Agent Theories, Architectures, and Languages (ATAL-98)*, volume 1555, 63–80. Springer-Verlag.