ISSN 1369-1961

Department of
Computer Science

Technical Report No. 769

Proceedings of the Fourth Workshop on Practical Reasoning and Rationality

Edited by:
John Bell and
Zhisheng Huang

QUEEN MARY
AND WESTFIELD COLLEGE
UNIVERSITY OF LONDON

July 1999

Proceedings of the Fourth Workshop on

Practical Reasoning and Rationality

Edited by

John Bell and Zhisheng Huang
Queen Mary and Westfield College
University of London

31 July, 1999
Stockholm, Sweden

Held in conjunction with
the IJCAI99 Workshop Series

IJCAI'99 Workshop KRR-1
Practical Reasoning and Rationality

A comprehensive logical theory of practical reasoning and rationality will include theoretical reasoning (reasoning about what is the case, involving informational attitudes such as beliefs and knowledge), practical reasoning (reasoning about what to do, involving motivational attitudes such as desires, goals, intentions, and obligations), and reasoning about actions and their effects.

Recent and current logical work has tended to focus on particular aspects of the problem; including nonmonotonic logics, belief revision, probabilistic logics, argumentation, logics for belief, action, obligation and preference, and logics for reasoning about action and change.

This is the fourth in a series of workshops which aim to promote the integration of this work. The focus of the workshop will be on the logical formalisation of the components of practical reasoning (intentions, obligations, preferences, goals, plans and actions) and particularly on the interactions between them and the rational balance amongst them. Further information on the workshop series can be found at:

http://www.dcs.qmw.ac.uk/conferences/prr/

We are grateful for the administrative support provided by the IJCAI'99 organisers.

Organizing Committee

John Bell (Queen Mary, University of London, UK),

Andreas Herzig (Universite Paul Sabatier, France),

John Horty (University of Maryland, USA),

Zhisheng Huang (Queen Mary, University of London, UK),

Gerhard Lakemeyer (Aachen University of Technology, Germany),

John-Jules Meyer (Utrecht University, The Netherlands),

Mark Ryan (University of Birmingham, UK),

Marek Sergot (Imperial College, London, UK),

Richmond Thomason (University of Michigan, USA).

Table of Contents

Multivalued logics and Topics

Robert Demolombe
ONERA/DTIM
2 Av. E. Belin, 31055, Toulouse Cedex, France
Robert.Demolomb@cert.fr

Abstract

There are several practical problems of knowledge representation where it is more natural to talk in terms of the set of sentences related to a given topic than to make explicit all the sentences in this set.

A problematic feature of classical logic is that if we want to represent this set by a small set which is closed under logical consequence, we can generate sentences related to new topics. For example from p it is possible to infer $p \vee q$, even if q has no common topic with p.

In this paper we compare several multivalued logics in order to give a formal characterisation of the relationship between topics and sentences. We give a uniform presentation for the semantics of the four valued logic suggested by D. Lewis, for the D. Bochvar's three valued logic, and for classical two valued logic. We present theorems that allow to compare their possibilities in terms of characterisation of propositional variables that occur in senetnces.

The main result is that Bochvar's logic is more suitable than Lewis's logic to characterise sentences that are logically equivalent, and that are formed with the same propositional variables.

Keywords: Multivalued logics, Relevance, Topics.

1 Introduction

There are several practical problems where we have to define which kind of usage can be done of sets of sentences. In these situations it may be quite heavy to characterise these sets by their extensions. Another possibility is to use the concept of topic to characterise sets of sentences.

For instance, we can use topics to define the sets of sentences stored in a knowledge base that a given user is permitted to access [CD96]. For example, in a company some users may be permitted to access any sentence that is about the topic health, while others may be permitted to access any sentence about incomes.

In the context of cooperative answering, techniques have been defined to provide users with additional information which is not explicitly requested, and which corresponds to their topics of interest [CD89]. For example, if some database user is asking a query about employees' salaries, the system could infer that the topic "income" is some user's topic of interest, and it could give other facts related to income. In other contexts it may be useful to characterise all a user knows about a given topic [Lak93] in order not to give to him information he already knows. Also, in the field of computer aided teaching, the "teacher" has to inform the student with new knowledge which is about the topic of the course [Mar91].

Another example is when data stored in data bases are not all guaranteed to be correct, in the sense that some of them may not be valid (they are stored in the data base while they are not true in the world), or they may not be complete (they are true in the world, but they are not stored in the data base) [Dem97]. To characterise sets of sentences for which data base content is guaranteed to be valid or to be complete, it can be quite convenient to use topics.

Finally, solutions to the frame problem have been proposed that use the concept of topic to define sets of sentences that are dependent or independent of a given change [Her96, ndCH94]. Here, independent is taken in the sense that beliefs in these sets are persistent after performance of an action or after performance of a belief revision.

In summary, for all these kinds of problems we need some linguistic means to define how sets of sentences have to be used. The aim here is not define which sen-

tences are true in the world. That is why these characterisations are based on sentence meaning and not on sentence truth. The consequence is that links between sentences and topics do not necessarily have the same properties as links between sentences and their truth values.

For instance in classical logic if a sentence p is true we can infer that sentence $p \vee q$ is also true. However, it is not clear that from the fact that sentence p is about topic t we can infer that sentence $p \vee q$ is also about topic t. Take, for instance, for p the sentence (John loves Mary)$\wedge$(Mary loves John), and for q the sentence (John loves Mary)$\wedge\neg$(Mary loves John). We can accept that p is about the topic happiness, but $p \vee q$, which is logically equivalent to (John loves Mary), alone, is not necessarily about happiness. Moreover, if r is the sentence (Mary loves Peter), we can reject the fact that sentence $p \wedge r$, which is (John loves Mary)$\wedge$(Mary loves John)$\wedge$(Mary loves Peter) is also about happiness. That is, from $p \wedge r$ is about t we cannot necessarily infer that p is about t.

Let's consider now the definition of the structure of these links from a pragmatic point of view. It may be that if we want to characterise user's topics of interest to extend classical answers, the inference of the fact: $p \vee q$ is about t from the fact: p is about t, could be accepted. However, if we have in mind to define what information a user is permitted to know, that inference is definitly not acceptable. That means that inference rules for reasoning about these links should be selected with caution, and may depend on the kind of pragmatic problem we consider.

A problematic feature of classical logic is that it is possible to derive consequences from a set of assumptions that have no topic in common with the assumptions. This problem has been extensively investigated by researchers in the field of relevant logics [A.R75].

In [Eps90] (p. 120), Epstein defines a non standard semantics for dependent implication, and he introduces a function s that assigns to a sentence p a set of topics. In [L. 94], Fariñas del Cerro and Lugardon, use the same technique.

Demolombe and Jones, in [DJar] , have defined a logic for reasoning about topics of sentences that does not require some extra feature, in the definition of the semantics, like this function s. They introduce a predicate A(t,"p"), whose meaning is that the proposition which is represented by sentence "p" is about topic t. If we denote by Var(p) the set of propositional variables in sentence p, the basic property of their logic is that, if p is logically equivalent to q, in classical logic, and Var(p)=Var(q), then we can infer that A(t,"p") is logically equivalent, in a classical sense, to A(t,"q"). To define a sound semantics for this inference rule, they make use of Bochvar's three valued logic (see [Boc72, GG84]). Independently, for the formalisation of contexts, J.Mc Carthy and S. Buvač have also used in [BBM95, MB97] Bochvar's three valued logic to define the semantics of the predicate ist(c,p), which means that p holds in context c.

Lewis in [Lew88] (p.173), has suggested, to define relevant implication, to consider a four valued logic, where, in a given world, a sentence may be true, false, true and false, or neither true nor false. The intuitive idea was that the fourth truth value "inconsistent" might be used to characterise inconsitent sentences, in the sense of classical logic, that are formed with the same propositional variables.

The objective of this paper is to compare, in a uniform logical framework, several multivalued logics, in order to select the most appropriate one for the definition of the links between sentences and topics. We consider the classical two valued logic, the four valued logic suggested by Lewis, and the Bochvar's three valued logic.

2 Four valued logic

In this section we define structures for the four valued logic. The same kind of structures will be defined in the next sections, with additional constraints for the three and two valued logic. We investigate properties of the four valued logic, and we compare this logic with classical propositional calculus (CPC).

Definition 1: Propositional Calculus Language. Let VAR be a set of propositional variables, the associated propositional calculus language is defined as usual from VAR using the logical connectives $\neg$, for negation, and $\vee$ for disjunction.

The connectives $\wedge$ and $\rightarrow$ are defined as usual from negation and disjunction.

Definition 2: Structure. A structure is a tuple S=<W,T,F> such that:

- W is a set of worlds.
- T is a function from VAR to 2^W.
- F is a function from VAR to 2^W.

From an intuitive point of view, if v is a propositional variable, T (resp. F) assigns to v the set of worlds where v is true (resp. false). The functions T and F are extended to compound sentences by the following rules:

$$T(\neg p) = F(p)$$
$$F(\neg p) = T(p)$$

$$T(p \vee q) = (T(p) \cap (T(q) \cup F(q))) \cup$$
¿ $(T(q) \cap (T(p) \cup F(p))$
$$F(p \vee q) = F(p) \cap F(q)$$

The truth values "undefined" and "inconsistent" are defined from "true" and "false". The set of worlds where a sentence p is undefined (resp. inconsistent) is denoted by U(p) (resp. I(p)). These functions are defined by:

$$U(p) \stackrel{def}{=} W - (T(p) \cup F(p)) \quad \text{and} \quad I(p) \stackrel{def}{=} T(p) \cap F(p)$$

According to these definitions we have:

$$U(\neg p) = U(p)$$
$$I(\neg p) = I(p)$$

$$U(p \vee q) = U(p) \cup U(q)$$
$$I(p \vee q) = (I(p) \cap I(q)) \cup (I(p) \cap F(q)) \cup (I(q) \cap F(p))$$

In the case where several structures are under consideration we adopt the notation $T_S(p)$ to denote the set of worlds where p is true in the structure S. Similar notations are adopted for F(p), U(p) and I(p).

For a given propositional calculus language the set of all the possible structures for the four valued logic is denoted by Σ_4.

Definition 3: Two valued logic associated to a four valued logic. Let S=<W,T,F> be a structure in Σ_4. The associated two valued structure s is the tuple s=<W,t,f> where t and f are functions from VAR to 2^W such that:

- For a propositional variable v: t(v)=T(v).
- $t(\neg p) = W - t(p)$.
- $t(p \vee q) = t(p) \cup t(q)$.

The set of worlds, f(p), where a proposition p is false is defined by: $f(p) \stackrel{def}{=} W - t(p)$

Notation: the fact that a sentence p is a tautology of classical propositional calculus (CPC) is denoted by: $\models_{CPC} p$.

Lemma 1. If for every strtucture S in Σ_4 we have $t_S(p) = W$ then we have $\models_{CPC} p$.

Proof. The lemma is a direct consequence of the fact that Σ_4 contains all the possible assignments for t.

Lemma 2. If for every strtucture S in Σ_4 we have $t_S(p) \subseteq t_S(q)$ then we have $\models_{CPC} p \rightarrow q$.

Proof. The fact $t_S(p) \subseteq t_S(q)$ holds iff the fact $t_S(p \rightarrow q) = W$ holds.

Theorem 1. Let S=<W,T,F> be a given struture in Σ_4, we define the structure S^+ in function of S by: $W^+ = W$ and for every propositional variable v:

$$T_{S^+}(v) = T_S(v)$$
$$F_{S^+}(v) = W - T_S(v).$$

Then, for every sentence p we have: $T_{S^+}(p) = t_S(p)$ and $F_{S^+}(p) = f_S(p)$.

Proof. The proof is by induction on the complexity of sentences.

Theorem 2. If for every structure S in Σ_4 we have $T_S(p) \subseteq T_S(q)$ then for every structure S in Σ_4 we have $t_S(p) \subseteq t_S(q)$.

Proof. Proof is based on Theorem 1, and uses the technique of contraposition.

Theorem 3. For every sentence p there exists a structure S in Σ_4 such that $T_S(p) \neq \emptyset$.

Proof. The proof is by induction on the complexity of sentences.

Let us denote by Var(p) the set of propositional variables in p.

Theorem 4. For every sentence p, if for every structure S in Σ_4 we have $T_S(p) \subseteq T_S(q)$, then we have $Var(q) \subseteq Var(p)$.

Proof. By contraposition, Theorem 4 is equivalent to: $Var(q) \not\subseteq Var(p)$ implies that there exists a structure S in Σ_4 such that $T_S(p) \not\subseteq T_S(q)$.

Let v be a propositional variable such that $v \in Var(q)$ and $v \notin Var(p)$.

Let S_0 and w_0 be a structure and a world defined in the same way as in the proof of Theorem 3. The world w_1 in S_0 is defined as follows:

> For every propositional variable u different of v we have:
> $w_1 \in T_{S_0}(u)$ and $w_1 \in F_{S_0}(u)$.
> For the variable v we have: $w_1 \notin T_{S_0}(v)$ and $w_1 \notin F_{S_0}(v)$
> (that is $w_1 \in U(v)$).

From the proof of Theorem 3 we know that $w_0 \in T_{S_0}(p)$. Since v is not in p, all the propositional variables in p have the same truth value in w_0 and in w_1. Therefore we have $w_1 \in T_{S_0}(p)$. Since the variable v is in q and v is undefined in w_1, from the definition of U we have $w_1 \in U_{S_0}(q)$, then we have $w_1 \notin T_{S_0}(q)$. Therefore we have $T_{S_0}(p) \not\subseteq T_{S_0}(q)$.

Corollary 1. If for every structure S in Σ_4 we have $T_S(p) = T_S(q)$ then we have Var(p)=Var(q).

Proof. Trivial.

Theorem 5. If for every structure S in Σ_4 we have $T_S(p) \subseteq T_S(q)$ then we have: $\models_{CPC} p \rightarrow q$ and $Var(q) \subseteq$

Var(p).

Proof. This theorem is a direct consequence of Theorem 2, Lemma 2 and Theorem 4.

Theorem 5 shows that the four valued logic is powerful enough to represent the same kind of implication as dependent implication defined by Epstein in [Eps90].

Theorem 6. If for every structure S in Σ_4 we have $T_S(p) = T_S(q)$ then we have: $\models_{CPC} p \leftrightarrow q$ and $Var(q) = Var(p)$.

Proof. This is a direct consequence of Theorem 5.

Theorem 7. (W. Carnielli, [Car94]) The facts $\models_{CPC} p \leftrightarrow q$ and Var(p)=Var(q) do not imply that for every structure S in Σ_4 we have $T_S(p) = T_S(q)$.

Proof. The following example shows that Theorem 7 holds. Let p be the sentence $(a \wedge \neg a) \wedge b$, and q be the sentence $(a \wedge \neg a) \wedge \neg b$. We have $\models_{CPC} p \leftrightarrow q$ and Var(p)=Var(q).

Let S be a structure such that there exists a one to one correspondance between the set of worlds in S and the natural numbers. Let us define T and F in the following way.

T(a) = set of worlds labeled by multiples of 2.
F(a) = set of worlds labeled by multiples of 3.
T(b) = set of worlds labeled by multiples of 5.
F(b)= set of worlds labeled by numbers that are not multiples of 5.

We have $T(p) = T(a) \cap F(a) \cap T(b)$ and $T(q) = T(a) \cap F(a) \cap F(b)$. Then the world w_{30} which corresponds to the integer 30 is in T(p), but it is not in T(q). Therefore we have $T_S(p) \neq T_S(q)$. Notice that we also have $w_{12} \notin T(p)$ and $w_{12} \in T(q)$.

The negative result presented by Theorem 7 is a bit surprising. Indeed, it might seem to be intuitive that two sentences, that are logically equivalent in CPC, and that are formed with the same propositional variables, have the same extensions.

Corollary 2. The facts $\models_{CPC} p \rightarrow q$ and $Var(q) \subseteq Var(p)$ do not imply that for every structure S in Σ_4 we have $T_S(p) \subseteq T_S(q)$.

Proof. The implication would hold, it would contradict Theorem 7.

The relationships between the two valued logic and the four valued logic are not obvious. Let us consider for example the following structure S in Σ_4 where we have a world w such that for the two propositional variables a and b we have: $w \in T(a)$, $w \in F(a)$, $w \notin T(b)$ and $w \notin F(b)$.

Then we have $w \in t(a)$ and $w \in F(a)$, and this shows that we may have for some structure S and for some sentence p: $t_S(p) \cap F_S(p) \neq \emptyset$. We also have $w \notin t(\neg a)$ and $w \in T(\neg a)$, and this shows that we may have $T_S(p) \not\subseteq t_S(p)$. Finally we have $w \in t_S(\neg b)$ and $w \notin T_S(\neg b)$, and this shows that we may have $t_S(p) \not\subseteq T_S(p)$.

The following theorem shows some relationships between the two logics.

Theorem 8. For every sentence p and for all structure S in Σ_4 we have: $T_S(p) - F_S(p) \subseteq t_S(p)$ and $F_S(p) - T_S(p) \subseteq f_S(p)$.

Proof. The proof is by induction on the complexity of sentences.

The relationships between the four valued logic and the two valued logic are represented by the Figure 1.

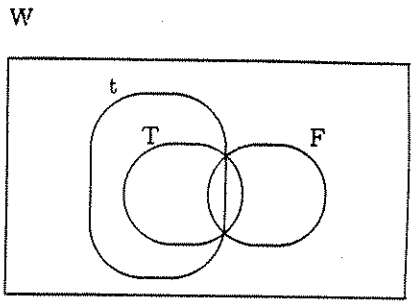


Figure 1: Relationships between the four valued logic and the two valued logic.

3 Three valued logic

In this section we consider a three valued logic. It is defined exactly in the same way as the four valued logic. The only difference is that we restrict the set of structures to those structures such that for every propositional variable v we have:

$$T_S(v) \cap F_S(v) = \emptyset$$

This set of structures is denoted by Σ_3. The definitions of functions T, F, t and f are the same as for the four valued logic.

Lemma 3. For every sentence p and for all structure S in Σ_3 we have $T_S(p) \cap F_S(p) = \emptyset$.

Proof. The proof by induction on the complexity of sentences is trivial.

Lemma 4. The fact that for every structure S in Σ_3 we have $T_S(p) \subseteq T_S(q)$ does not imply that we have $Var(q) \subseteq Var(p)$.

Proof. Consider the two sentences $p = a \wedge \neg a$ and q=b.

Lemma 5. The fact that for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ does not imply that we have $Var(q) = Var(p)$.

Proof. Consider the two sentences $p = a \wedge \neg a$ and $q = b \wedge \neg b$.

Theorem 9. For all sentence p and for every structure S in Σ_3 we have: $T_S(p) \subseteq t_S(p)$ and $F_S(p) \subseteq f_S(p)$.

Proof. The proof is by induction on the complexity of sentences.

Theorem 10. For every sentence p and for every structure S in Σ_3 we have:
$t_S(p) \subseteq T_S(p) \cup U_S(p)$ and $f_S(p) \subseteq F_S(p) \cup U_S(p)$.

Proof. The proof is by induction on the complexity of sentences.

A direct consequence of the Theorem 10 is that for every sentence p we have $t_S(p) \cap F_S(p) = \emptyset$ and $f_S(p) \cap T_S(p) = \emptyset$. Since from the Theorem 9 we have $T_S(p) \subseteq t_S(p)$ and $F_S(p) \subseteq f_S(p)$, the relationships between the three valued logic and the two valued logic are as indicated in the figure 2.

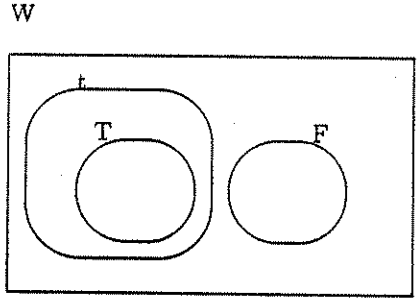


Figure 2: Relationships between the three valued logic and the two valued logic.

Theorem 11. The facts $\models_{CPC} p \rightarrow q$ and $Var(q) \subseteq Var(p)$ imply that for every structure S in Σ_3 we have $T_S(p) \subseteq T_S(q)$.

Proof. This is a consequence of Theorem 9 and Theorem 10.

Theorem 12. The facts $\models_{CPC} p \rightarrow q$ and $Var(p) \subseteq Var(q)$ imply that for every structure S in Σ_3 we have $F_S(q) \subseteq F_S(p)$.

Proof. Similar proof as for Theorem 11.

Lemma 6. The facts $\models_{CPC} p \rightarrow q$ and $Var(q) \subseteq Var(p)$ do not imply that for every structure S in Σ_3 we have $F_S(q) \subseteq F_S(p)$.

Proof. Let us consider the two sentences $p = a \wedge b$ and q=a, where a and b are propositional variables. We have $\models_{CPC} p \rightarrow q$ and $Var(q) \subseteq Var(p)$. Let S be a structure in Σ_3 and w a world of S such that that $w \in F(a)$ and $w \in U(b)$. We have $w \in F(q)$ and $w \notin F(p)$.

Theorem 13. The facts $\models_{CPC} p \leftrightarrow q$ and Var(p)=Var(q) imply that for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ and $F_S(p) = F_S(q)$.

Proof. This theorem is a direct consequence of Theorems 11 and 12.

Theorem 14. There exist sentences p and q such that for every structure S in Σ_3 we have $T_S(p) \subseteq T_S(q)$ and we do not have $Var(q) \subseteq Var(p)$.

Proof. Consider the sentences $p = a \wedge \neg a$ and q=b.

Theorem 15. If for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ and $F_S(p) = F_S(q)$, then we have Var(p)=Var(q).

Proof. Let us assume that for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ and $F_S(p) = F_S(q)$.

Let us assume that we have $Var(q) \not\subseteq Var(p)$, then there exists a propositional variable v such that $v \in Var(q)$ and $v \notin Var(p)$.

Let S be a structure in Σ_3 and w be a world in S. We have either $w \in t_S(p)$ or $w \in f_S(p)$. Let us assume first that we have $w \in t_S(p)$.

We define a world w' of a structure S' from w and S in the following way:

If a variable u is in Var(p) then:
if $w \in T_S(u)$ then $w' \in T_{S'}(u)$,
if $w \notin T_S(u)$ then $w' \in F_{S'}(u)$.
If a variable u is not in Var(p) then $w' \in U_{S'}(u)$.

According to this definition we have $w' \in t_{S'}(p)$, because the fact $w' \in t_{S'}(p)$ (resp. $w \in t_S(p)$) only depends on the variable u such that $w' \in T_{S'}(u)$ (resp. $w \in T_S(u)$), and for the variables u in p we have $w \in T_S(u)$ iff $w' \in T_{S'}(u)$, and we also have $w \in t_S(p)$.

From Theorem 10 we have $t_{S'}(p) \subseteq T_{S'}(p) \cup U_{S'}(p)$, then we have $w' \in T_{S'}(p)$ or $w' \in U_{S'}(p)$. From the definition of w' none of the variables in p is undefined in w' then we do not have $w' \in U_{S'}(p)$, therefore we have $w' \in T_{S'}(p)$. Since we have $T_{S'}(p) = T_{S'}(q)$, we also have $w' \in T_{S'}(q)$.

Since the variable v of q is not in p, by definition of w', we have $w' \in U_{S'}(v)$, and, by definition of U, we have $w' \in U_{S'}(q)$, which contradicts the fact $w' \in T_{S'}(q)$. Therefore we

have $Var(q) \subseteq Var(p)$.

If we assume now that we have $w \in f_S(p)$, a similar proof, based on the fact $F_{S'}(p) = F_{S'}(q)$, also allows to infer $Var(q) \subseteq Var(p)$.

Then, in both cases we have $Var(q) \subseteq Var(p)$.

Since p and q plays a similar role, we can also prove that $Var(p) \subseteq Var(q)$, and finally we have $Var(p) = Var(q)$.

Theorem 16. If for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ then we have $\models_{CPC} p \leftrightarrow q$.

Proof. We can easily see that the proofs of Theorems 1 and 2 also hold if we restric the set of structures from Σ_4 to Σ_3.

Theorem 17. For every sentence p, if for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ and $F_S(p) = F_S(q)$, then we have $\models_{CPC} p \leftrightarrow q$ and $Var(p) = Var(q)$.

Proof. This result collates the results of Theorems 15 and 16.

Theorem 18. The fact that for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ does not imply that for every structure S in Σ_3 we have $F_S(p) = F_S(q)$.

Proof. Consider the two sentences $p = a \wedge \neg a$ and $q = b \wedge \neg b$.

Theorem 19. We have $\models_{CPC} p \leftrightarrow q$ and $Var(p)=Var(q)$ iff for every structure S in Σ_3 we have $T_S(p) = T_S(q)$ and $F_S(p) = F_S(q)$.

Proof. This theorem is a direct consequence of Theorems 13 and 17.

Theorem 20. The two valued logic represented by Σ_2 has the same properties as classical propositional calculus.

Proof. We prove by induction on the complexity of sentences that for every structure S in Σ_2 we have $T_S(p) = t_S(p)$ and $F_S(p) = f_S(p)$. Moreover for every possible assignment t there exists a corresponding structure in Σ_2 such that t coincides with T.

4 Conclusion

The three logics are defined in terms of the same types of structures as defined in Definition 2. If there is no restriction on the definitions of T and F, we have the four valued logic represented by the set of structures Σ_4. If we restrict the set Σ_4 to the structures where we have $T_S(p) \cap F_S(p) = \emptyset$, we have the Bochvar's three valued logic which is represented by the set of structures Σ_3. Finally, if we restrict Σ_3 to the structures where we have $T_S(p) \cup F_S(p) = W$, we have a two valued logic represented by the set of structures Σ_2 which has the same properties as CPC, as it is shown by Theorem 20.

We have investigated how properties about the propositional variables in sentences can be represented by structures only in terms of truth values. That is, the only difference with classical propositional calculus is that we consider three or four different truth values. We have found a correspondance (Theorem 19) between properties about variables, and extensions of sentences, only in the case of the three valued logic: i.e. we have $\models_{CPC} p \leftrightarrow q$ and $Var(p) = Var(q)$ iff we have $\forall S \in \Sigma_3$ $(T_S(p) = T_S(q)$ and $F_S(q) \subseteq F_S(p))$. In the case of the four valued logic the fact $\forall S \in \Sigma_4$ $(T_S(p) = T_S(q))$ implies $\models_{CPC} p \leftrightarrow q$ and $Var(p) = Var(q)$, but the implication in the other way does not hold (Theorem 7). An open question is to found the property we have to add to $\models_{CPC} p \leftrightarrow q$ and $Var(p) = Var(q)$ in order to have the equivalence with $\forall S \in \Sigma_4$ $(T_S(p) = T_S(q))$.

These formal results show that Bochvar's three valued logic is the most adequate to define, in the semantics, links between topics and sentences.

References

[A.R75] A.R. Anderson and N.D. Belnap. *Entailment.* Princeton University Press, 1975.

[BBM95] S. Buvač, V. Buvač, and I.A. Mason. Metamathematics of contexts. *Fundamenta Informaticae*, 23(3), 1995.

[Boc72] D.A. Bochvar. Two papers on partial predicate calculus. Technical report STAN-CS-280-72, Stanford University, 1972. Translation of two papers from 1938 and 1943.

[Car94] W. Carnielli. Private communication. Technical report, 1994.

[CD89] F. Cuppens and R. Demolombe. How to recognize interesting topics to provide cooperative answering. *Information Systems*, 14(2), 1989.

[CD96] F. Cuppens and R. Demolombe. A Deontic Logic for Reasoning about Confidentiality. In *Proc. of 3rd International Workshop on Deontic Logic in Computer Science*, 1996.

[Dem97] R. Demolombe. Answering queries about validity and completeness of data: from modal logic to relational algebra. In T. Andreasen, H. Christiansen, and H. L. Larsen, editors, *Flexible Query Answering Systems.* Kluwer Academic Publishers, 1997.

[DJar] R. Demolombe and A.J.I. Jones. On sentences of the kind "sentence "p" is about topic "t": some steps toward a formal-logical analysis. In H-J. Ohlbach and U. Reyle, editor, *Logic, Language and Reasoning. Essays in Honor of Dov Gabbay.* Kluwer Academic Press, To appear.

[Eps90] R.L. Epstein. *The Semantic Foundations of Logic, Volume1: Propositional Logic.* Kluwer Academic, 1990.

[GG84] D. Gabbay and F. Guenthner. *Handbook of Philosophical Logic, Vol3.* D.Reidel Publishing Company, 1984.

[Her96] A. Herzig. The pma revisited. In L. C. Aiello and S. Shapiro, editors, *Proc. of International Conference on Knowledge Representation and Reasoning (KR'96).* Morgan Kaufmann Publishers, 1996.

[L. 94] L. Fariñas del Cerro and V. Lugardon. Sequents for dependence logic. *Logique et Analyse*, 133-134, 1994.

[Lak93] G. Lakemeyer. All they know about. In *Proc. of the 11th National Conference on Artificial Intelligence (NCAI-93)*, 1993.

[Lew88] D. K. Lewis. Relevant implication. *Theoria*, LIV(3), 1988.

[Mar91] P. Marquis. Novelty Revisited. In *Proc. of the 6th International Symposium on Methodologies for Intelligent Systems*, 1991.

[MB97] J. McCarthy and S. Buvac. Formalizing Context. In S. Buvac and L. Iwanska, editors, *AAAI-97 Symposium on Context in Knowledge Representation nad Natural Language.* AAAI Press, 1997.

[ndCH94] L. Fari nas del Cerro and A. Herzig. A conditional logic for updating in the possible models approach. In B. Nebel and L. Dreschler-Fisher, editors, *Proc. of 18th German Conference on Artificial Intelligence (KI'94).* Springer. LNAI 861, 1994.

Modeling Reasoning Agents

Maria Fasli
University of Essex
Department of Computer Science
Wivenhoe Park, Colchester CO4 3SQ
U. K.

Abstract

A great deal of research in Computer Science and Artificial Intelligence in particular is concerned with intelligent agents and the formalisation of their properties. A number of approaches have been developed in the literature based upon the intentional stance. One of these, namely the BDI-framework, views agents as having beliefs, desires and intentions. Under this description several constraints have been imposed in order to capture different types of agents. In this paper we consider a similar framework to that of a BDI-agent but furthermore the agents discussed here have the additional attitude of knowledge. Then we investigate a number of possible constraints according to the relationships between the intentional notions, which enable us to capture the properties of different types of agents.

1. Introduction

Much work in Artificial Intelligence [Cohen and Levesque, 1987; Fagin et al., 1995; Rao and Georgeff, 1991; Rao and Georgeff, 1998] has been motivated by the need to formalize theories which will ideally describe the properties of an agent and the kind of reasoning associated with intelligent behaviour. This tendency to ascribe mental qualities or attitudes to artificial agents and machines, known as the intentional stance [Dennet, 1987], is a useful and convenient means of describing complex systems, explaining and predicting their behaviour. The intentional stance describes agents or systems in terms of concepts such as knowledge, beliefs, desires and obligations. It is assumed that the agent has a mind and has goals or desires which are based upon her view of the world and the information she possesses, and she will perform actions that will lead her to the achievement of her goals (principle of rationality). The intentional notions are usually divided into two major categories: information attitudes and pro-attitudes. The information attitudes such as knowledge or beliefs are obviously related to information that the agent possesses about the world she inhabits. The difference however, between knowledge and belief is clearly a philosophical issue and it is not the subject of discussion here. Nevertheless, belief is considered to be the weaker notion of the two, and knowledge is usually associated with truth. Thus, an agent is allowed to have false beliefs without of course being aware of it, but never false knowledge. Pro-attitudes such as desires, obligations and intentions [Bratman, 1987] are responsible for the agent's actions and they enable agents to exhibit goal-motivated behaviour according to what they desire and intend to achieve.

Which exactly should be the attitudes appropriate for an agent's theory is the issue of debate in the literature. Here however, we view agents as having knowledge, beliefs, desires and intentions. Knowledge represents the true information that the agent possesses about the world. Beliefs are various pieces of information for which the agent is not absolutely sure that they hold true, nevertheless they cannot contradict her knowledge about the world. The intentions of an agent indicate her commitment to perform certain actions and achieve goals or states of the world. The desires represent states of the world or actions that the agent would prefer to achieve. Both intentions and desires seem to have a temporal aspect, in the sense that when we are referring to one's intentions or desires most of the times we usually refer to some point in the near or distant future. However, in this investigation, for the time being we are not explicitly going to involve time.

The paper continues by first discussing the basic logical machinery based on the BDI framework [Rao and Georgeff 1991; Rao and Georgeff, 1998,] with the additional concept of knowledge. Then we discuss some of the issues on the relations between the concepts of knowledge, beliefs, desires and intentions. The following subsections present constraints for modeling and capturing different types of agents based on the relationships between the intentional notions. The paper ends with a summary of our findings and a pointer to future work.

2. Formal Framework

2.1. Language and Semantics

To make our ideas precise, and to express the concepts of knowledge, belief, desires and intentions and facts about

the world we will use a propositional multi-modal language [Fagin *et al.*, 1995]. We assume that we have a number of agents 1,....,n and the world is described in terms of a non-empty set of propositions Φ. Each of the primitive propositions p,q,r,... in Φ represents a basic fact about the world. We also have the classical propositional connectives $\wedge$ and $\neg$ and the standard abbreviations for disjunction $\phi\vee\psi\equiv\neg(\neg\phi\wedge\neg\psi)$, implication $\phi\Rightarrow\psi\equiv\neg(\phi\wedge\neg\psi)$ and equivalence $\phi\Leftrightarrow\psi\equiv\neg(\phi\wedge\neg\psi)\wedge\neg(\psi\wedge\neg\phi)$. In addition we have four modal operators K,B,I,D, for expressing what an agent knows, believes, intends and desires respectively. These modal operators are indicated by a subscript denoting the agent that they refer to. Wffs in this language are:

i) Each primitive proposition p is a wff

ii) if ϕ and ψ are wffs then so are $\neg\phi$ and $\phi\wedge\psi$

iii) if ϕ is a wff then $K_i(\phi)$, $B_i(\phi)$, $I_i(\phi)$ and $D_i(\phi)$ are wffs

For the purpose of semantics we are going to use the classical possible worlds framework and Kripke structures. The basic concept behind the possible worlds is that besides the true state of affairs, the actual world, there are other possible states of affairs or worlds that the agent considers possible. More formally a Kripke structure for our multi-modal logic is a tuple $M=<W,\pi,K_i,B_i,I_i,D_i>$ where W is the set of worlds, π is a truth assignment to the primitive propositions of Φ (i.e. for each world w and each primitive proposition p, $\pi(w,p)\in\{true,false\}$), K_i is a binary relation for each agent i on W and defines which worlds the agent considers possible according to her knowledge. We write $K_i(w,w')$ and we mean that the world w' is knowledge-accessible from w, according to agent i. The accessibility relations B_i, I_i, D_i are similar to that of knowledge. A binary relation R_i in general is:

i) Serial, iff $\forall\ w\in W$, $\exists\ w'$ such that $R_i(w,w')$

ii) Reflexive, iff $\forall\ w\in W$ we have $R_i(w,w)$

iii) Symmetric, iff $\forall\ w,w'\in W$ such that $R_i(w,w')$ we have $R_i(w',w)$

iv) Transitive, iff $\forall\ w,w',w''\in W$ such that $R_i(w,w')$ and $R_i(w',w'')$ we have $R_i(w,w'')$

v) An equivalence relation, if it is reflexive, symmetric and transitive

vi) Euclidean, iff $\forall\ w,w',w''\in W$ such that $R_i(w,w')$ and $R_i(w,w'')$ we have $R_i(w',w'')$

The notion of truth for a formula in Kripke structures, is defined as follows:

i) $(M,w)\models\phi$ iff $\pi(w,\phi)$=true

ii) $(M,w)\models\phi\wedge\psi$ iff $(M,w)\models\phi$ and $(M,w)\models\psi$

iii) $(M,w)\models\neg\phi$ iff $(M,w)\not\models\phi$

iv) $(M,w)\models K_i(\phi)$ iff $(M,w')\models\phi$, $\forall\ w'\in W$ such that $K_i(w,w')$

v) $(M,w)\models B_i(\phi)$ iff $(M,w')\models\phi$, $\forall\ w'\in W$ such that $B_i(w,w')$

vi) $(M,w)\models I_i(\phi)$ iff $(M,w')\models\phi$, $\forall\ w'\in W$ such that $I_i(w,w')$

vii) $(M,w)\models D_i(\phi)$ iff $(M,w')\models\phi$, $\forall w'\in W$ such that $D_i(w,w')$

2.2. Basic Axiom Systems

The axioms that we are going to adopt are initially the K-, and D-axioms for all four notions as well as the necessitation rule. The K-axiom is the minimal system for normal modal logics [Hughes and Cresswell, 1968] and it states that if an agent knows ϕ and she knows that $\phi\Rightarrow\psi$ then she will also know ψ. The same constraint is extended so as to cover beliefs, desires, and intentions. The D-axiom expresses the consistency of knowledge, beliefs, desires and intentions and requires the accessibility relation between accessible worlds to be serial. The necessitation rule states that any valid formula is known, believed, desired and intended. Thus we have:

Knowledge

$K(\phi)\wedge K(\phi\Rightarrow\psi)\Rightarrow K(\psi)$

$K(\phi)\Rightarrow\neg K(\neg\phi)$

IF $\vdash\phi$ THEN $\vdash K(\phi)$

Belief

$B(\phi)\wedge B(\phi\Rightarrow\psi)\Rightarrow B(\psi)$

$B(\phi)\Rightarrow\neg B(\neg\phi)$

IF $\vdash\phi$ THEN $\vdash B(\phi)$

Desires

$D(\phi)\wedge D(\phi\Rightarrow\psi)\Rightarrow D(\psi)$

$D(\phi)\Rightarrow\neg D(\neg\phi)$

IF $\vdash\phi$ THEN $\vdash D(\phi)$

Intentions

$I(\phi)\wedge I(\phi\Rightarrow\psi)\Rightarrow I(\psi)$

$I(\phi)\Rightarrow\neg I(\neg\phi)$

IF $\vdash\phi$ THEN $\vdash I(\phi)$

We will name this system $K^DB^DD^DI^D$, from the initial of all four notions and the superscript indicates the characteristic axiom for each one of them. We will not impose any further axioms for desires and intentions and thus from now we are going to omit the superscript from the names of the systems. For knowledge we can also add the following:

T. $K(\phi)\Rightarrow\phi$

S4. $K(\phi)\Rightarrow K(K(\phi))$

S5. $\neg K(\phi)\Rightarrow K(\neg K(\phi))$

The T-axiom intuitively says that knowledge is true, and this is considered to be the axiom that distinguishes philosophically the notion of knowledge from that of belief. It requires the relation between knowledge-accessible worlds to be reflexive. The S4-axiom, otherwise known as the positive introspection axiom, attributes to an agent introspective capabilities about her knowledge, and therefore if she knows a proposition ϕ then she knows that she knows this proposition. The accessibility relation in order to have the S4-axiom between knowledge-accessible worlds is required to be transitive. Finally the S5-axiom, or the negative introspection axiom, states that if an agent does not know a proposition ϕ, she has knowledge that she does not know this fact. The accessibility relation in this case is required to be symmetric and transitive.

We can extend the S4- and S5-axioms so as to cover belief:

S4. $B(\phi)\Rightarrow B(B(\phi))$

S5. $\neg B(\phi)\Rightarrow B(\neg B(\phi))$

However we will not adopt the T-axiom since we allow an agent to have false beliefs. This system is known as the KD45 or "weak S5" system [Hughes and Cresswell, 1968].

Considering different combinations of axioms for knowledge and belief we can have a family of logics and each system is described by the characteristic axiom for knowledge and belief indicated in the superscript. Here we will be concerned with the $K^{S5}B^{KD45}DI$ system.

It is well known that agent formalisms based upon normal modal logics suffer from the logical omniscience problem [Fagin *et al.*, 1995]. Although this framework suffers from the same problems, some forms of logical omniscience can be alleviated, as we will see, in the form of the consequential closure principles [Rao and Georgeff, 1998].

3. Types of agents

Let us suppose that we have an agent as described in the previous sections. Although we can capture some of its reasoning the above axiomatizations do not describe any connection that may exist between the selection of intentions and desires and the agent's knowledge and beliefs about the world. In the framework that we are using desires, intentions as well as knowledge and beliefs are sets of accessible worlds and thus one possible avenue for investigating different types of agents is by imposing different kind of relationships between these sets.

In [Rao and Georgeff, 1991; Rao and Georgeff, 1998] the authors have considered such conditions in order to capture additional relationships in the BDI-framework. As they point out some of the possible set theoretic relationships between the sets of accessible worlds will be quite meaningless, and thus they examine what they call the strong realism, realism, and weak realism constraints (Figure 1).

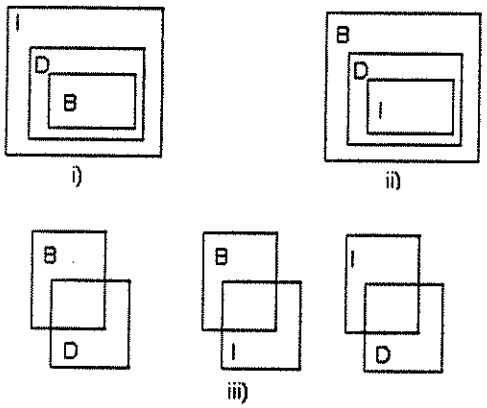


Figure 1. i)Strong Realism, ii)Realism iii)Weak Realism

According to the strong realism condition the agent under description is a very cautious agent, and only intends and desires propositions that believes to be achievable. In this case the set of belief-accessible worlds is a subset of the desire-accessible worlds and the set of desire-accessible worlds is a subset of the intention-accessible worlds. An agent based on the realism constraint on the other hand is an over-enthusiastic agent and believes that she can achieve her desires and intentions. In realism the set of intention-accessible worlds is a subset of the desire-accessible worlds which is a subset of the belief-accessible worlds. A more balanced type of agent is the one based on the weak realism constraints, where the intersection of the intention- and belief- accessible worlds is not the empty set, and the same condition applies between desire- and intention- as well as between desire- and belief-accessible worlds. In [Rao and Georgeff, 1991; Rao and Georgeff, 1998] the logic includes apart from the basic modal operators, temporal operators based on the branching temporal logics of CTL and CTL*. The notion of a world being a subworld of another is defined and some additional constraints are imposed based on the structural relationships between worlds. Since each possible world is a time tree this allows axioms over I-formulas (inevitable) and O-formulas (optional).

When the fourth concept of knowledge is considered as part of an agent's cognitive system things get more complicated. We are confronted with a number of issues:

a) What is the connection between knowledge and belief? Is knowledge a subset of the agent's beliefs? Can we safely assume that knowledge is justified true belief?

b) What is the connection between knowledge-belief and intentions? Surely intentions are quite different from desires since intentions presuppose that the agent is committed towards their fulfillment. Is it enough for the agent to believe that her intentions are achievable or does she need something stronger like knowledge of her intentions being achievable?

c) What is now the connection between knowledge-belief and the desires of an agent? Does the agent have to know that her desires are achievable or believing that some of them are achievable is enough? Would it be possible that the agent's desires are completely decoupled from her beliefs and knowledge, and thus she can desire propositions that believes or even knows that they are not achievable?

d) If an agent intends to perform an action ϕ_1 and knows or believes that $\phi_1 \Rightarrow \phi_2$, does she always intend to do ϕ_2 as well? (consequential closure principles)

While it is not possible to provide definite answers to these questions in the following we are going to examine some possible relationships between the four sets of accessible worlds. The relationships that we examine are set theoretic and we do not examine structural relationships between worlds, since we are not using a temporal representation in this framework. However we do not provide, by any means, an exhaustive list, only some of them which we believe are quite reasonable options for modeling practical agents.

3.1 Agents with Belief-consistent Intentions

The first type of constraints that we are considering is depicted in Figure 2, and characterizes agents with belief-consistent intentions.

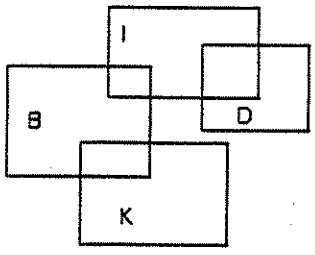


Figure 2. Agents with Belief-consistent Intentions

Hence, the intersection of the set of knowledge-accessible and belief-accessible worlds is not the empty set, and the same relationship applies between intention-accessible and belief-accessible worlds, and finally between intention-accessible and desire-accessible worlds.

It is quite obvious the desires of an agent can be decoupled from her knowledge and her beliefs although the agent is not allowed to have intentions which are inconsistent with her beliefs. In the same way she cannot have beliefs that are inconsistent with her knowledge or desires that are inconsistent with her intentions. However under these conditions she is not obliged to know that are achievable as long as she does not have contradicting beliefs.

The semantic conditions that support the above relations between the sets of accessible worlds are the following:

$\forall w \exists w' \quad K(w,w') \wedge B(w,w')$

$\forall w \exists w' \quad B(w,w') \wedge I(w,w')$

$\forall w \exists w' \quad I(w,w') \wedge D_i(w,w')$

These now entail the axioms:

$K(\phi) \Rightarrow \neg B(\neg\phi)$

$I(\phi) \Rightarrow \neg B(\neg\phi)$

$D(\phi) \Rightarrow \neg I(\neg\phi)$

Let us suppose that we have an agent called Alice. A fragment of Alice's cognitive system includes the following desires, intentions, beliefs and knowledge:

a) Alice believes that she can have a car accident

b) She knows that a Ph.D. requires a lot of hard work

c) She desires to win a beauty contest and

d) She intends to finish her Ph.D.

Now what can we say about Alice, according to the above constraints? Does she behave within the limits of what we would describe as rational behaviour? Is she a cautious, an enthusiastic or a balanced agent? On the one hand she believes that she can have a car accident, but according to the semantic conditions she may not of course know that fact for certain. Nevertheless, she might not intend to have a car accident and neither desire it. She knows that a Ph.D. requires a lot of hard work but she does not believe that a Ph.D. can be obtained without working hard. Thus her intention to obtain a Ph.D. will not come without working hard and she is aware of that. Alice's desire however to win a beauty contest does not mean that she believes that it is achievable. Alice would like to win the beauty contest but she may not believe that it is possible.

Rao and Georgeff [1998] consider the consequential closure principles. These are all formulas that are satisfiable in the context of the above semantic constraints:

C1) $I(\phi_1) \wedge B(\phi_1 \Rightarrow \phi_2) \wedge \neg I(\phi_2)$

If Alice intends to go to the dentist and believes that a visit to the dentist always results in pain, she does not have to intend to suffer pain.

C2) $I(\phi_1) \wedge D(\phi_1 \Rightarrow \phi_2) \wedge \neg I(\phi_2)$

If Alice intends to get married and she desires by getting married to have children, she may not intend to have children.

C3) $D(\phi_1) \wedge B(\phi_1 \Rightarrow \phi_2) \wedge \neg D(\phi_2)$

If Alice desires to drink alcohol and she believes that by drinking alcohol one can get drunk, she may not desire to get drunk. Another possible form of consequential closure principle is one that involves knowledge and intentions:

C4) $I(\phi_1) \wedge K(\phi_1 \Rightarrow \phi_2) \wedge \neg I(\phi_2)$

If Alice intends to go to the dentist and knows that a visit to the dentist always ends up in pain, she may not intend though, to suffer pain.

We can consider a slight variation of this type of agent where the intentions of the agent are connected with her knowledge and not her beliefs (knowledge-consistent intentions). An agent with belief-consistent intentions would be appropriate as an email agent that sorts the user's emails. In this case even if the agent intends to include emails that it believes the user will find useful, is not going to cause much trouble, apart probably from a little annoyance to the user. The variation of this type of agent with knowledge-consistent intentions, would be more appropriate for example as security protocol agents, where the agents need to choose their intentions and actions according to their knowledge about the state of affairs.

3.2 Agents with Knowledge- and Belief-consistent Intentions

This particular type of agent adopts intentions that do not contradict in any way with the information she has about the world, in the form of beliefs and in the form of knowledge. Her desires cannot be inconsistent with her intentions, although as in the previous case the desires are decoupled from her knowledge and beliefs. For instance, the agent may desire to marry her neighbour's husband, even though she knows that he is already married to another woman. This kind of agent can be captured by the set relationships depicted in Figure 3.

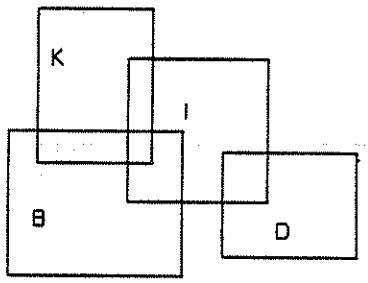


Figure 3 Agents with Knowledge- and Belief-consistent Intentions

An agent with belief and knowledge consistent intentions is more cautious as far as her selection of intentions is concerned. In belief-consistent intention agency, the agent can have intentions which are not inconsistent with her beliefs, she does not have to have conclusive information about the feasibility of her future actions or plans, for instance through previous experience etc. On the other hand if an agent has a belief- and knowledge-consistent intention, then she must have enough information in order to support her choice of that particular intention, than a simple belief. Imagine now the following scenario. Alice our agent is disabled, she cannot walk which prohibits her from playing

football and she knows that she cannot play football. However our agent believes that miracles can happen and thus that it is possible she is going to be cured in the near future, and she will be able to play football. If Alice belongs to the previous category of agents with belief-consistent intentions, her intention to play football is consistent with her beliefs, however it is not consistent with her knowledge, nevertheless an intention for playing football is allowed. Now if Alice's intentions must be belief- and knowledge-consistent her intention of playing football cannot be allowed.

The semantic conditions that support the above relations between are the following:

$\forall w \exists w'$ $K(w,w') \wedge B(w,w')$
$\forall w \exists w'$ $K(w,w') \wedge I(w,w')$
$\forall w \exists w'$ $B(w,w') \wedge I(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge D(w,w')$

And thus we have the following axioms imposed:

$K(\phi) \Rightarrow \neg B(\neg\phi)$
$I(\phi) \Rightarrow \neg K(\neg\phi)$
$I(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg I(\neg\phi)$

The consequential principles C1-C4 are satisfied in the context of an agent with knowledge- and belief-consistent intentions.

An example of an agent with knowledge and belief-consistent intentions is a football player agent. A football player agent must take under consideration both its knowledge and its beliefs when it selects its intentions and goals. Its intentions should not contradict its knowledge about the rules of the game for example and its position and assigned role on the team (defender, attacker etc.) but also its beliefs about the current state of affairs within the game.

3.3. Agents with Belief-consistent Intentions and Desires

So far the two types of agents investigated have conditions linking beliefs, intentions and knowledge but the desires of the agent are decoupled from her knowledge and beliefs about the world. In the type of agency investigated here, Figure 4, the desires of the agent are linked with her beliefs in the sense that the agent is not allowed to have desires that are inconsistent with her beliefs about the world. The same kind of restriction applies between beliefs and knowledge, beliefs and intentions and intentions and desires.

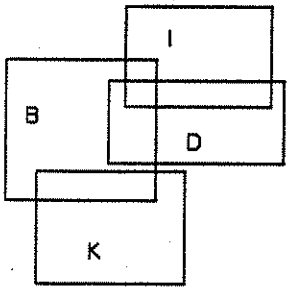


Fig. 4 Agents with Belief-consistent Intentions and Desires

Thus as in the simple belief-intention consistent agency, the agent is not allowed to have intentions that are inconsistent with her beliefs about the world and furthermore now the agent is not allowed to have desires that are inconsistent with her intentions or her beliefs.

The semantic conditions for these relationships are:

$\forall w \exists w'$ $K(w,w') \wedge B(w,w')$
$\forall w \exists w'$ $B(w,w') \wedge I(w,w')$
$\forall w \exists w'$ $B(w,w') \wedge D(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge D(w,w')$

These conditions now impose the following axioms:

$B(\phi) \Rightarrow \neg K(\neg\phi)$
$I(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg I(\neg\phi)$

Alice our agent in this case believes that she can have a car accident, but she does not know this fact for certain. She intends to obtain a Ph.D. and she does not believe that this is not achievable although, she knows it requires lots of hard work. Her desire to win a beauty contest must not be inconsistent with her beliefs and her intentions.

The consequential closure principles C1-C4 are again satisfied in this type of agency. Examples of agents of this particular type can be again email agents, search engines agents.

3.4 Inter-consistent Agents

The agent of this particular type is not allowed to have intentions that are inconsistent with her knowledge and beliefs, and desires that are inconsistent with her intentions and beliefs about the world. These relations are depicted in Figure 5.

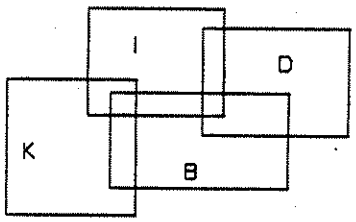


Figure 5. Inter-consistent Agents

The semantic conditions that support the above set theoretic relations between the different sets of accessible worlds are the following:

$\forall w \exists w'$ $K(w,w') \wedge B(w,w')$
$\forall w \exists w'$ $K(w,w') \wedge I(w,w')$
$\forall w \exists w'$ $B(w,w') \wedge I(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge D(w,w')$
$\forall w \exists w'$ $B(w,w') \wedge D(w,w')$

The following axioms are imposed due to the above semantic relationships between sets of accessible-worlds:

$B(\phi) \Rightarrow \neg K(\neg\phi)$
$I(\phi) \Rightarrow \neg K(\neg\phi)$
$I(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg I(\neg\phi)$

$D(\phi)\Rightarrow\neg B(\neg\phi)$

All of the consequential closure principles C1-C4 are again satisfied. Alice again intends to obtain a Ph.D. and knows that it requires a lot of work and therefore she does not believe that she can obtain a Ph.D. without doing any work. Her desire to win a beauty contest again should not be inconsistent with her beliefs but there is no apparent relation between her desires and her knowledge.

For instance a financial market agent, or a football player agent can be described by the above semantic conditions.

3.5 Agents with Belief-based Intentions

Agents who base their intentions on their beliefs are shown in Figure 6. In this type of agent the set of belief accessible worlds is a subset of the intention accessible worlds and the intersection of intention- and knowledge-accessible worlds as well as the intersection of belief- and knowledge-, and intention- and desire-accessible worlds is not the empty set:

$\forall w\forall w'\ \ B(w,w')\Rightarrow I(w,w')$
$\forall w\exists w'\ \ K(w,w')\wedge B(w,w')$
$\forall w\exists w'\ \ K(w,w')\wedge I(w,w')$
$\forall w\exists w'\ \ I(w,w')\wedge D(w,w')$

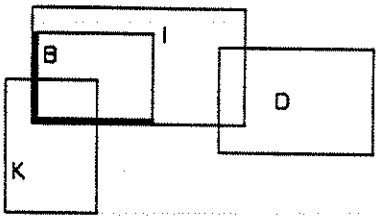


Figure 6. Agents with Belief-based Intentions

Using Alice again as our example and given the fact that she believes she can have a car accident, it is obvious that she does not know that it is impossible to have a car accident but she may not intend or desire it. Alice also knows that a Ph.D. requires a lot of hard work and she does not believe that this is not the case. Thus she intends to obtain a Ph.D. and at the same time she believes that this is achievable, she has the time and resources to achieve such a goal. Alice's desire however to win a beauty contest, only says that it does not intend to do otherwise, however she may not believe or even know for sure that such a desire is achievable. The following axioms are imposed due to the above semantic relationships between sets of accessible-worlds:

$I(\phi)\Rightarrow B(\phi)$
$K(\phi)\Rightarrow\neg B(\neg\phi)$
$I(\phi)\Rightarrow\neg K(\neg\phi)$
$D(\phi)\Rightarrow\neg I(\neg\phi)$

All of the consequential closure principles C1-C4, are again satisfied. This particular type of agent is very cautious and would be suitable for example as a security protocol agent, or a nuclear plant controller agent.

3.6 Belief-based Intentions and Belief-consistent Desires

Agents who base their intentions on their beliefs and their desires are belief-consistent, are depicted in Figure 7. In this type of agents the set of belief accessible worlds is a subset of the intention accessible worlds and the intersection of intention- and knowledge-accessible worlds as well as the intersection of belief- and knowledge-, intention- and desire-accessible and desire- and belief-accessible worlds is not the empty set. The semantic conditions that support these relations are given below:

$\forall w\forall w'\ \ B(w,w')\Rightarrow I(w,w')$
$\forall w\exists w'\ \ K(w,w')\wedge B(w,w')$
$\forall w\exists w'\ \ K(w,w')\wedge I(w,w')$
$\forall w\exists w'\ \ I(w,w')\wedge D(w,w')$
$\forall w\exists w'\ \ B(w,w')\wedge D(w,w')$

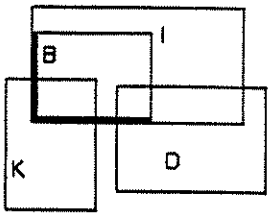


Figure 7. Agent with Belief-based Intentions and Belief-consistent Desires

Even if Alice believes she can have a car accident, she does not know that it is impossible to have a car accident. Alice also knows that a Ph.D. requires a lot of hard work and she does not believe that this is not the case. Thus she intends to obtain a Ph.D. and at the same time she believes that this is achievable. Alice's desire to win a beauty contest, implies that she does not intend to do otherwise, and she does not believe that her aim is not achievable. However her desires are decoupled from her knowledge. The following axioms are imposed due to the above semantic conditions between sets of accessible-worlds:

$I(\phi)\Rightarrow B(\phi)$
$K(\phi)\Rightarrow\neg B(\neg\phi)$
$I(\phi)\Rightarrow\neg K(\neg\phi)$
$D(\phi)\Rightarrow\neg I(\neg\phi)$
$D(\phi)\Rightarrow\neg B(\neg\phi)$

All of the consequential closure principles C1-C4 are again satisfied as in the previous cases. The kinds of applications mentioned in the previous type of agent would be appropriate here as well, since the agent under description is over-cautious with its intentions and desires.

3.7 Intention-enthusiastic Agents with Intention-consistent Desires

In this particular type of agency we have enthusiastic agents that believe that they can achieve their intentions. Hence, the set of intention-accessible worlds is a subset of the belief-accessible worlds, and on the other hand the intersection between the belief- and the knowledge-accessible

worlds as well as the intersection of the desire- and belief-accessible, and the desire and intention-accessible worlds is not the empty set. This kind of relationship is depicted in Figure 8. The semantic conditions that support these relations between the sets of accessible worlds:

$\forall w \forall w'$ $I(w,w') \Rightarrow B(w,w')$
$\forall w \exists w'$ $K(w,w') \wedge B(w,w')$
$\forall w \exists w'$ $B(w,w') \wedge D(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge D(w,w')$

The following are part of the agent's axiomatization:

$B(\phi) \Rightarrow I(\phi)$
$K(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg I(\neg\phi)$

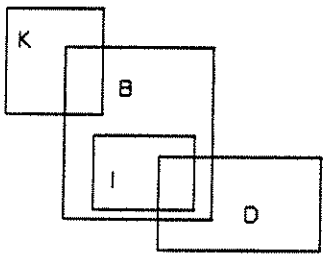


Figure 8. Intention-enthusiastic Agent with Intention-consistent Desires

In this case, Alice our agent again believes she can have a car accident, and it is obvious that she does not know that it is impossible to have such an accident. Alice also knows that a Ph.D. requires a lot of hard work and she does not believe that this is not the case. Thus she intends to obtain a Ph.D. and at the same time she does not desire not to get it. Alice's desire to win a beauty contest, implies that she does not believe it to be an impossible task nor she intends to achieve the opposite but this desire is not connected with her knowledge.

Not all of the consequential closure principles are satisfied in this type of agency. In particular the consequential closure principle C1 between intentions and beliefs is not satisfied any more due to the fact that the set of intention-accessible worlds in this case is a subset of the belief-accessible worlds, and thus whenever the agent believes something she will intend it as well. The rest of the principles C2-C4 are satisfied. An agent that acts as a user interface could be probably described by these semantic constraints.

3.8 Over-enthusiastic Agents

In this type of agency we have over enthusiastic agents that believe that they can achieve not only their intentions but their desires as well (Figure 9). Thus the sets of intention- and desire-accessible worlds are subsets of the belief-accessible worlds, and their intersection is not the empty set. The intersection between the belief- and the knowledge-accessible worlds is not the empty set in this case as well. The semantic conditions that support the theoretic relations between the different sets of accessible worlds are the following:

$\forall w \forall w'$ $I(w,w') \Rightarrow B(w,w')$
$\forall w \forall w'$ $D(w,w') \Rightarrow B(w,w')$
$\forall w \exists w'$ $K(w,w') \wedge B(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge D(w,w')$

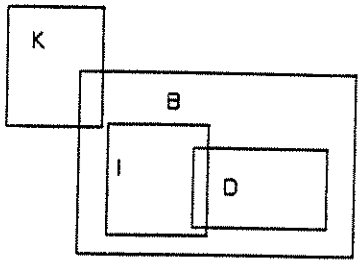


Figure 9. Over-enthusiastic Agent

Alice believes she can have a car accident, and she does not know that it is impossible to have a car accident but in this type of agency Alice will actually desire and intend to have a car accident if she believes that she is going to have one. Alice also knows that a Ph.D. requires a lot of hard work and she does not believe that this is not the case. Thus she intends to obtain a Ph.D. and at the same time she does not desire not to get it. Alice's desire to win a beauty contest, implies that she does not believe it to be an impossible task nor she intends to achieve the opposite result but this desire is not connected with her knowledge. The following axioms are imposed in this type of agent:

$B(\phi) \Rightarrow I(\phi)$
$B(\phi) \Rightarrow D(\phi)$
$K(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg I(\neg\phi)$

Not all of the consequential closure principles are satisfied in this particular type of agency. In particular the consequential closure principles between intentions and beliefs (C1) and desires and beliefs (C2) are not satisfied. This is due to the fact that the sets of intention- and desire-accessible worlds in this case are subsets of the belief-accessible worlds, and thus whenever the agent believes something she will intend and desire it as well.

3.9 Over-enthusiastic agents with Knowledge-consistent Intentions

An over enthusiastic agent is an agent that believes that she can achieve not only her intentions but her desires as well. Thus the sets of intention- and desire-accessible worlds are subsets of the belief-accessible worlds, and their intersections is not the empty set. In addition the intersection of the intention-accessible and the knowledge-accessible worlds is not the empty set in this case, which in other words means that when an agent intends to do something she must not know it is not achievable. The intersection between the belief- and the knowledge-accessible worlds is not the empty set in this case as well, Figure 10. The semantic constraints supporting these relations are as follows:

$\forall w \forall w'$ $I(w,w') \Rightarrow B(w,w')$
$\forall w \forall w'$ $D(w,w') \Rightarrow B(w,w')$
$\forall w \exists w'$ $K(w,w') \wedge B(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge D(w,w')$
$\forall w \exists w'$ $I(w,w') \wedge K(w,w')$

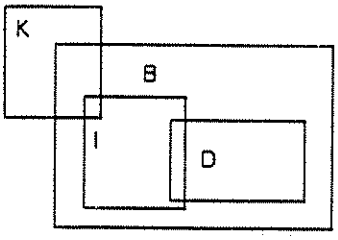


Figure 10. Over-enthusiastic agent with Knowledge-consistent Intentions

In this case Alice believes she can have a car accident, and it is obvious that she does not know that it is impossible to have a car accident but in this type of agency Alice will actually desire and intend to have a car accident if she believes it. Alice also knows that a Ph.D. requires a lot of hard work and she does not believe that this is not the case. Thus she intends to obtain a Ph.D. and at the same time she does not desire not to obtain it. Alice's desire to win a beauty contest, implies that she does not believe it to be an impossible task nor she intends to achieve its opposite but this desire is not connected with her knowledge. Thus the following axioms are part of an agent's axiomatization:
$B(\phi) \Rightarrow I(\phi)$
$B(\phi) \Rightarrow D(\phi)$
$K(\phi) \Rightarrow \neg B(\neg\phi)$
$D(\phi) \Rightarrow \neg I(\neg\phi)$
$I(\phi) \Rightarrow \neg K(\neg\phi)$
Not all of the consequential closure principles are satisfied as in the previous case. In particular the consequential closure principles between intentions and beliefs (C1), and desires and beliefs (C3) are not satisfiable any more. This is due to the fact that the sets of intention- and desire-accessible worlds in this case are subsets of the belief-accessible worlds, and thus whenever the agent believes something she will intend and desire it as well.
A possible application for this and the previous type of agent is a user interface agent.

4. Concluding Remarks

The research reported here was motivated by the need to formalise different types of agents according to the relationships between their knowledge, beliefs, desires and intentions. Needless to say, we do not claim that the list of types of agents presented here is exhaustive. There are other ways in which we can combine the sets of accessible worlds and thus get different types of agents with different constraints and axiomatizations. Our aim however was to show the variety of agents that we can produce by considering the fourth notion of knowledge in addition to those of beliefs, intentions, and desires. Nevertheless we tried to present a few of the available types of agents which we consider as most closely exhibiting a bit of practical reasoning as far as their choice of intentions and desires is concerned. Different type of agents may be appropriate for different types of applications, and we tried to give some examples of applications for most of these types of agents. These examples of applications are not exclusive for one and only one type of agent. The details and choice of the most appropriate type are left to the designers of an agent and can vary considerably.
It would be useful to consider as part of a future investigation, to add a temporal component like CTL or CTL*, since we mentioned that desires and intentions have a temporal aspect, and examine structural relationships between worlds. Furthermore the concept of common knowledge [Fagin *et al.*, 1995] can be incorporated in order to reason about group knowledge, and how this could be used by each particular type of agent in order to form intentions and plans. It is clear that much additional work will be required in order to study all possible useful types of agents and their properties but we hope that this report will stimulate further investigation.

Acknowledgements

The author would like to thank Ray Turner and the anonymous reviewers for their useful comments. The research reported here was funded by GSSF fund 189/96.

References

[Bratman, 1987] Bratman M.E. Intentions, Plans and Practical Reason. Harvard University Press, Cambridge, Massachusetts, 1987

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

[Dennet, 1987] Dennet D.C. The Intentional Stance. Cambridge, Massachusetts: The MIT Press, 1987

[Fagin *et al.*, 1995] Fagin R., Halpern J., Moses Y. and Vardi M. Reasoning About Knowledge, Cambridge, Massachusetts: The MIT Press, 1995

[Hughes and Cresswell, 1968] Hughes G.E. and Cresswell M.J. An Introduction to Modal Logic, London: Methuen, 1968

[Rao and Georgeff, 1991] Rao A. and Georgeff M. Modeling Rational Agents within a BDI-Architecture. In Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning. San Mateo, Calif.: Morgan Kaufmann Publishers, 1991

[Rao and Georgeff, 1998] Rao A. and Georgeff M. Decision Procedures for BDI Logics. In Journal of Logic and Computation. Vol:8, pp:293-343, 1998

Knowledge, actions, and tests

Andreas Herzig, Jerôme Lang, Thomas Polacsek
IRIT-CNRS, Université Paul Sabatier
118 route de Narbonne, F-31062 Toulouse Cedex 04
Tel.: (+33) 56155-6344, Fax: -8325
mailto: Andreas.Herzig@irit.fr, http://www.irit.fr/~Andreas.Herzig

Abstract

We study a modal logic of knowledge and action, focussing on epistemic tests. We view an epistemic test as an action undertaken by an agent in order to establish whether a given formula is true. Such tests increase the knowledge of agents. We propose a semantics, and associate an axiomatics and a proof procedure.

1 Introduction

Imagine a robot that wants to open a door that might be locked. If the robot is cute enough, he starts by checking whether the door is effectively locked up. Such test actions are an important form of interaction. They are central e.g. in diagnosis in order to discriminate the possible fault configurations) or in decision under uncertainty.

Tests are a one-sided form of communication: the agent acquires knowledge about the environment, while that knowledge-gathering action does not change the environment. (There are two simplifying hypotheses we make here: first, we suppose that the environment of the agent doesn't change while the test is done; second, we suppose that tests do not change the environment.)

What do we test? We can test the physical objects of the world, e.g. a battery or a computer program. Here we are rather interested in tests of facts, i.e. to check *whether* the battery is empty, or to establish *that* the battery is empty.

These two actions are different: we may suppose at least in certain domains that an agent can always check whether a given fact is true or false, while we consider that the action of establishing a fact only succeeds if the fact we try to establish is indeed the case: an agent can only establish that a battery is empty if it really is; in the opposite case we consider that the action cannot be executed. Nevertheless, in the sequel we shall see that we can intertranslate these notions.

In this paper we restrict our analysis to the propositional case. We first present the standard logics of knowledge and action (sections 2 and 3). Then we integrate these two concepts in a single logic, and we investigate axiomatization and automated theorem proving (section 4).

2 Epistemic logic

The analysis of the notions of knowledge and belief in terms of possible states of affairs has been proposed by Hintikka [Hintikka, 1962, Fagin *et al.*, 1995]. We adopt S5 as our logic of knowledge. In order to simplify the reading of the formulas we suppose w.l.o.g. that there is only one agent.

The language of epistemic logic is constructed from a set of atomic formulas FML_0, the usual logical operators of classical logic, and the modal operator $\mathcal{K}$. An example of a formula is $\neg\mathcal{K}p\wedge\neg\mathcal{K}\neg p$. It is read "the agent neither knows p nor $\neg p$ and thus expresses the agent's ignorance w.r.t. the truth of p. The formula $p\wedge\neg\mathcal{K}p$ means that the agent ignores that p, while the formula $p\wedge\mathcal{K}\neg p$ means that the agent is wrong about p. That last formula is inconsistent if we view knowledge true belief, which is what we shall do here.

The semantics of epistemic logic is in terms of possible states. A model of S5 is a triple $M=\langle W,R_{\mathcal{K}},V\rangle$ where

- W is a set of states (or possible worlds);
- $R_{\mathcal{K}}$ is an equivalence relation on W;
- V associates to each state a valuation: $V(w)\subseteq FML_0$; we often write V_w instead of $V(w)$.

We shall sometimes identify $R_{\mathcal{K}}$ with the function $R_{\mathcal{K}}: W\longrightarrow 2^W$ by stipulating $R_{\mathcal{K}}(w)=\{v: wR_{\mathcal{K}}v\}$.

Given a model $M=(W,V)$, we define as usual truth in a state $w\in W$. In particular :

- $\models_{M,w} p$ if $p\in V_w$;
- $\models_{M,w}\mathcal{K}A$ if for every state $v\in R_{\mathcal{K}}(w)$ we have $\models_{M,v} A$.

It is part of the classical results in modal logics that the set of valid formulas of S5 is axiomatized by

MP	$\frac{A\ ,\ A\rightarrow B}{B}$
N($\mathcal{K}$)	$\frac{A}{\mathcal{K}A}$
Class	The set of theorems of classical logic
K($\mathcal{K}$)	$(\mathcal{K}A\wedge\mathcal{K}(A\rightarrow C))\ \rightarrow\ \mathcal{K}C$
T($\mathcal{K}$)	$\mathcal{K}A\rightarrow A$
4($\mathcal{K}$)	$\mathcal{K}A\rightarrow\mathcal{K}\mathcal{K}A$
5($\mathcal{K}$)	$\neg\mathcal{K}A\rightarrow\mathcal{K}\neg\mathcal{K}A$

3 Dynamic logic

There exists already a well-known logic containing a test operator, viz. dynamic logic [Harel, 1984]. To the presentation in the latter we prefer that of [Goldblatt, 1992] in terms of standard models, because it is more appropriate for our purposes.

The language of propositional dynamic logic PDL is constructed from a set of atomic formulas FML_0, the classical logic operators $\rightarrow, \wedge, \vee, \neg$, a set of atomic actions ACT_0, the action operators $\lambda, \cup, ;, ?$, and the modal operator $[.]$.[1] An example of a formulas are $\neg p \wedge [p?]p$. We read the formula $[p?]q$ as "after establishing p, q is true", or "after checking that p, q is true".

$\alpha;\beta$ means "execute α and then β", and $\alpha \cup \beta$ means "choose nondeterministically between α and β, and then execute the chosen action".

We define the action $A??$ of checking *whether* A as an abbreviation of the complex action $A?\cup(\neg A)?$. This formally expresses that to check whether A is true amounts to nondeterministically choose between trying to establish that A and trying to establish that $\neg A$.

Remark Note that if we are only interested in tests of the type $A??$, formulas written using $??$ will explode exponentially if we expand the abbreviation. Therefore it is of interest to consider the other way round that $??$ is primitive. In this case we can define the formula $[A?]B$ to be an abbreviation of $[A??](A \rightarrow B)$.

Semantics is in terms of a transition system between states: a model is a triple $M = \langle W, \{R_\alpha : \alpha \in ACT\}, V\rangle$ where W is a set of states and V is a valuation as for epistemic logic, and

- each R_α is a relation between states : $R_\alpha \subseteq W \times W$ (called transition relation or accessibility relation).

As we did for epistemic logic, we shall sometimes view R_α as a function.

Given a model M as above, we define as usual the truth of complex formulas in a state, in particular :

- $\models_{M,w} [\alpha]A$ if for every state $v \in R_\alpha(w)$, $\models_{M,v} A$.

As we want the transition relations to reflect the intended meanings of complex actions, we restrict our attention to ***standard models***, which satisfy

- $R_\lambda(w) = \{w\}$
- $R_{\alpha\cup\beta} = R_\alpha \cup R_\beta$
- $R_{\alpha;\beta} = R_\alpha \circ R_\beta$
- $R_{A?}(w) =$ (if $\models_{M,w} A$ then $\{w\}$ else $\emptyset$)

The notion of validity is that of validity in the class of standard models.

We give a somewhat unusual axiomatization of the set of formulas of PDL that are valid in the class of standard models, in order to take profit of it lateron.

MP $\quad \dfrac{A\ ,\ A \rightarrow B}{B}$

[1] To simplify we have dropped the iteration operator *.

N($[\alpha]$)	$\dfrac{A}{[\alpha]A}$
Class	The set of theorems of classical logic
K($[\alpha]$)	$([\alpha]A \wedge [\alpha](A \rightarrow B)) \rightarrow [\alpha]B$
Def(λ)	$[\lambda]A \leftrightarrow A$
Def(;)	$[\alpha;\beta]A \leftrightarrow [\alpha][\beta]A$
Def($\cup$)	$[\alpha \cup \beta]A \leftrightarrow ([\alpha]A \wedge [\beta]A)$
Id(?)	$[A?]A$
Exec(?)	$A \rightarrow \neg[A?]\neg A$
Pres(?)	$B \rightarrow [A?]B$

The axioms Def(;), Def(λ) and Def($\cup$) can be viewed as formulating abbreviations of the respective action constructors. In the standard presentations of PDL, not only the former, but also the test operator ? is defined by

Def(?) $\quad [A?]B \leftrightarrow (A \rightarrow B)$

At least we obtain the same set of provable formulas:

Theorem 1 *Given the rest of the axiomatics, the axioms Id(?), Exec(?) and Pres(?) are equivalent to Def(?).*

A corollary of that result is that our axiomatization is complete w.r.t. standard PDL models.

4 An epistemic dynamic logic

It is the combination of epistemic logic and dynamic logic which will permit us to speak about tests done by agents in order to augment their knowledge. We call that logic epistemic dynamic logic EDL . In such a framework we must consider that actions are accomplished by agents. In consequence, after having established that A an agent knows that A, i.e. the formula $[A?]\mathcal{K}A$ should be valid.

But, while being a conservative extension of epistemic logic, our logic cannot be simply a conservative extension of dynamic logic. Indeed, suppose p is true, and suppose the agent ignores that p is true. Then the agent's ignorance cannot be preserved after establishing that p. Formally, this means that the instance

$$\neg\mathcal{K}p \rightarrow [p?]\neg\mathcal{K}p$$

of Pres($A?$) should not be valid.

In the sequel we shall make an important *restriction*: we shall suppose that all complex actions are constructed from tests. This hypothesis will allow us to simplify the models and the completeness proof. It will be relaxed in future work.

4.1 Language

We combine the languages of epistemic logic and dynamic logic. Actions and complex formulas are defined by mutual recursion in a way similar to dynamic logic. An example of formula is $\neg\mathcal{K}p \wedge \neg\mathcal{K}\neg p \wedge [p?]\mathcal{K}p$.

As said above, we suppose in the rest of the paper that $ACT_0 = \emptyset$.

We respectively note ACT and FML the set of actions and formulas thus defined. We say that a formula from FML is *objective* if it contains no occurrence of $\mathcal{K}$.

4.2 Semantics

Without surprise, EDL -models are combinations of S5 and PDL models: a model is a 4-tuple $M = \langle W, R_{\mathcal{K}}, \{R_\alpha : \alpha \in ACT\}, V\rangle$ where W and V are as before. $R_{\mathcal{K}}$ is an equivalence relation as for epistemic logic, and the R_α are transition relations as for dynamic logic. Moreover M must satisfy

- if $R_\alpha(w) \neq \emptyset$ then $R_{\mathcal{K}} \circ R_\alpha = R_\alpha \circ R_{\mathcal{K}}$.

This condition expresses that the agent is aware of his actions: if A is true and it is possible for him that action $A?$ results in some state v, then v is possible for him after the execution of $A?$, and vice versa.

Truth in a state $w \in W$ is defined as before. What differs is the notion of a *standard model*: as before it must satisfy that

- $R_\lambda(w) = \{w\}$
- $R_{\alpha;\beta} = R_\alpha \circ R_\beta$
- $R_{\alpha\cup\beta} = R_\alpha \cup R_\beta$

but the conditions for ? are slightly weaker than that for PDL:

- if $wR_{A?}u$ then $V_w = V_u$
- if $\models_{M,w} \neg A$ then $R_{A?}(w) = \emptyset$

The first condition expresses that a test has no effect on the physical world, while the second condition says that the action of establishing that A can only be executed if A holds.

4.3 Axiomatization

Now our extensive presentation of dynamic logic turns out to be useful. Indeed, the PDL axiom Def($A?$) is not valid, only $[A?]B \rightarrow (A \rightarrow B)$ is. The axioms Id($A?$), Exec($A?$) and Pres($A?$) allow us to fine-tune: Id($A?$) and Exec($A?$) are valid, while Pres($A?$) must be restricted.

We give the following axiomatization of EDL :

MP	$\frac{A\ ,\ A \rightarrow B}{B}$
N($\mathcal{K}$)	$\frac{A}{\mathcal{K}A}$
N($[\alpha]$)	$\frac{A}{[\alpha]A}$
Class	The set of theorems of classical logic
K($\mathcal{K}$)	$(\mathcal{K}A \wedge \mathcal{K}(A \rightarrow C)) \rightarrow \mathcal{K}C$
T($\mathcal{K}$)	$\mathcal{K}A \rightarrow A$
4($\mathcal{K}$)	$\mathcal{K}A \rightarrow \mathcal{K}\mathcal{K}A$
5($\mathcal{K}$)	$\neg\mathcal{K}A \rightarrow \mathcal{K}\neg\mathcal{K}A$
K($[\alpha]$)	$([\alpha]A \wedge [\alpha](A \rightarrow C)) \rightarrow [\alpha]C$
Def(λ)	$[\lambda]A \leftrightarrow A$
Def(;)	$[\alpha;\beta]A \leftrightarrow [\alpha][\beta]A$
Def($\cup$)	$([\alpha\cup\beta]A \leftrightarrow ([\alpha]A \wedge [\beta]A)$
Id(?)	$[A?]A$
Exec(?)	$A \rightarrow \neg[A?]\neg A$
Pres(?)	$C \rightarrow [A?]C$ if C is an objective formula
Det(?)	$\neg[A?]C \rightarrow [A?]\neg C$
Perm(?, $\mathcal{K}$)	$A \rightarrow ([A?]\mathcal{K}C \leftrightarrow \mathcal{K}[A?]C)$

The axioms Id(?), Exec(?) and Pres(?) are as before. The axiom Det(?) (which is a theorem in PDL) is added here explicitly. Perm(?, $\mathcal{K}$) relates the knowledge of the agent before and after the test.

It follows from Id(?), N($\mathcal{K}$), and Perm(?, $\mathcal{K}$) that the agent does tests consciously, i.e. $[A?]\mathcal{K}A$.

Property 1 *The following equivalences are theorems of EDL .*

1. $[A?]\mathcal{K}C \leftrightarrow (A \rightarrow \mathcal{K}[A?]C)$
2. $[A?]\neg C \leftrightarrow (A \rightarrow \neg[A?]C)$
3. $[A?](C_1 \wedge C_2) \leftrightarrow ([A?]C_1 \wedge [A?]C_2)$
4. $[A?](C_1 \vee C_2) \leftrightarrow ([A?]C_1 \vee [A?]C_2)$
5. $[A?]C \leftrightarrow (A \rightarrow C)$ *if C is an objective formula*
6. $A \leftrightarrow \neg[A?]\bot$

The proof of these equivalences is straightforward.

Note that although the formula $\mathcal{K}B \rightarrow [A?]\mathcal{K}B$ seems to be a theorem at first glance (expressing something like "knowledge is preserved under tests") this is not the case. This is due to the negative introspection axiom 5($\mathcal{K}$).

We postpone the completeness proof, and consider first of all a method of automated theorem proving for our logic.

4.4 Automated theorem proving

We reduce in this section the problem of proving theorems in EDL to that of proving theorems in the standard modal logic S5. The reduction is done by rewrite rules.

Indeed, a glance at the four first equivalences of the above property shows us that applying these equivalences from the left to the right we can 'push down' the modal operator of test through all the other connectives $\mathcal{K}, \neg, \wedge, \vee$. When $[A?]$ reaches an objective formula then we can apply $[A?]C \leftrightarrow (A \rightarrow C)$, and thus eliminate one modal operator of test from the formula. (We suppose here that we start with an operator $[A?]$ with no other $[B?]$ in its scope, and that the other action construction operators have been eliminated using axioms Def(λ), Def(;), and Def($\cup$).)

Iterating these rewrite steps we can obtain formulas without occurrences of test operators.

Theorem 2 *Let A be a formula of EDL . Then there exists a formula A' without test operators such that $A \leftrightarrow A'$ is a theorem of EDL .*

4.5 Soundness and completeness

Each of the axioms that we have given is valid, and the inference rules preserve validity. Hence our axiomatics is sound.

The above theorem gives us completeness.

Theorem 3 *Let A be a formula of EDL . A is EDL -valid iff A is a EDL -theorem.*

Proof Let A be consistent. According to the preceeding theorem there exists a formula A' without test operators such that $A \leftrightarrow A'$ is a EDL -theorem. Hence A' is consistent. Now A' is in the language of S5, and given that the axiomatics of EDL contains that of the epistemic logic S5, A' is as well consistant in S5. Via the completeness of S5 there must therefore exist a S5-model containing a state w where A' is true. Then from that model it is straightforward to extend that model to a EDL -model where A' is true in w. Finally, given that (due to soundness) the equivalences that we have used to rewrite formulas are valid, that EDL -model must also satisfy A in w.

4.6 Complexity

The fragment of EDL without nested tests has an interesting complexity. In this case our rewriting procedure is a polynomial transformation into S5. The problem of deciding whether a given S5-formula is a theorem is coNP-complete: it follows that the decision problem for the fragment of EDL without nested tests is also coNP-complete.

5 Related work

A lot of logics of knowledge and action exist. Closest to ours is the work of Gerbrandy and Groeneveld [Gerbrandy, 1997, Gerbrandy et Groeneveld, 1997, Groeneveld, 1995]. Their Dynamic Epistemic Logic has two sorts of test, the first of which is noted $?A$[2] and is the standard dynamic logic test: it "succeeds [...] when A is true, and fails otherwise". Consequently $[?A]C$ is an abbreviation of $A \rightarrow C$. The second one is noted $U\alpha$ and "corresponds to [the] agent [...] learning that program α has been executed". (We have slightly adapted notation.) This means that agents act *a priori* unconsciously and must explicitly learn about the executions of their actions. While this might be considered to be unnatural (in particular for artificial agents), it leaves more flexibility than our language e.g. to speak about agent i learning that agent j learned that A has been tested (expressible here as $U_a U_b ?A$).

$U_b ?A$ is similar to our $A?$. More precisely, our logic can be mapped into Gerbrandy's logic of [Gerbrandy, 1997]: our action $A?$ can be translated into their $?A; U?A$.

In [Gerbrandy, 1997] there is given an axiomatics, which is similar to ours. Nevertheless there are subtle differences. We have already mentioned the first one: there is a non-epistemic test $?A$ supplementing the epistemic test $U?A$.

The second main difference is that there, the logic of knowledge is K, while ours is S5. Hence there are no axioms T($\mathcal{K}$), 4($\mathcal{K}$), and 5($\mathcal{K}$). It seems to be problematic to add these axioms to the logic. This will be detailed after our next point.

The third main difference is that there, instead of axiom Exec(?) $A \rightarrow \neg[A?]\neg A$ there is an equivalence

$$A \leftrightarrow \neg[A?]\neg A$$

(axiom 5 in [Gerbrandy, 1997]). This means that an agent can always successfully learn about the execution of some action.[3] This leads to difficulties at least if we suppose that the epistemic notion under concern satisfies a consistency requirement as expressed by the modal axiom D($\mathcal{K}$) $\mathcal{K}A \rightarrow \neg\mathcal{K}\neg A$ (that is a consequence of axiom T($\mathcal{K}$)). Indeed, suppose p is an atom. Then $[U?p]\mathcal{K}p$ is derivable in their logic, as well as $\mathcal{K}\neg p \rightarrow [U?p]\mathcal{K}\neg p$. But from these two we can derive $\mathcal{K}\neg p \rightarrow [U?p]\mathcal{K}(p \wedge \neg p)$. While in our logic this means that the test action fails, in theirs the test $U?p$ always succeeds, and therefore axiom D($\mathcal{K}$) cannot be added to their logic as it stands.

Finally a more technical difference are the respective completeness proofs. While ours basically uses a reduction to a modal logic without tests, theirs is a (much longer) Henkin type proof. Nevertheless, our technique also applies to their logic, and permits thus to obtain a much simpler proof. To wittness, the K-axiom for $[U\alpha]$ together with the above equivalence $A \leftrightarrow \neg[U\alpha]\neg A$ permit to pass the modal operator $[U\alpha]$ through conjunction, disjunction, and negation, and their axiom 7 $[U\alpha]\mathcal{K}A \leftrightarrow \mathcal{K}[\alpha][U\alpha]A$ permits to pass through the epistemic operator $\mathcal{K}$. Finally their axiom 6 permits to eliminate the $[U\alpha]$ operator from formulas. Thus one can follow the same line of reasoning as in our completeness proof.

In a series of articles Segerberg has developped a logic of belief and action called Doxastic Dynamic Logic (DDL) [Segerberg, 1995, Segerberg,]. There are three types of modalities $+A$, $-A$, and $*A$ the first of which corresponds to our $A?$. He discusses axioms for $+A$ that are similar to ours, but nevertheless closer to Gerbrandy and Groeneveld's work. To witness, he also considers that tests are always executable and deterministic, i.e. he has the axiom $[+A]\neg C \leftrightarrow \neg[+A]C$ (his axiom 13), as well as a preservation axiom in terms of equivalence (his axiom 10). Therefore our above remarks also apply to this approach.

Another line of research has been developped in the AI field of reasoning about actions around the concept of knowledge gathering actions [Scherl et Levesque, 1993, Levesque, 1996, Lakemeyer et Levesque, 1998]. We here focus on the latter approach of Lakemeyer and Levesque. The logic AOL proposed there has similarities to our EDL . The main difference is that our logic does not contain the concept of only knowing. To wittness we consider an example given in their paper. "Suppose we have a robot that knows nothing about the initial state

[2] The authors consider several agents and groups of agents. We abstract from that here.

[3] This makes it also possible to write the preservation axiom Pres(?) as an equivalence.

of the environment, but that there is asensing action, reading a sonar, which tells the robot when it is getting close to a wall." Let us read the atomic formulas c and s respectively as 'the robot is close to the wall' and 'the sonar works', and let us interpret mc and ma respectively as the atomic actions of moving closer and moving away from the wall. In our language (allowing actions other than tests), what they then want to prove is

1. $[c??](\mathcal{K}c \vee \mathcal{K}\neg c)$
2. $\mathcal{K}w \rightarrow \mathcal{K}[c??](\mathcal{K}c \vee \mathcal{K}\neg c)$
3. $\mathcal{K}\neg c \rightarrow [ma](\mathcal{K}\neg c)$
4. $\mathcal{K}\neg c \rightarrow [mc](\neg\mathcal{K}c \wedge \neg\mathcal{K}\neg c)$

It is only the last formula that requires the non-monotonic only knowing notion.

6 Conclusion

We have defined a logic of knowledge and action EDL , to which we have associated an automated theorem proving procedure.

As we have noted in section 3, if we are only interested in tests of the type $A??$, formulas written using ?? will explode exponentially if we expand the abbreviation $A??$ to $A? \cup (\neg A)?$. It is nevertheless possible to give a polynomial reduction into S5 similar to that for tests of the type $A?$, which makes that complexity of theoremhood stays within co-NP.

We plan to continue that work in two directions.

First, our logic allows to reason about the evolution of knowledge by tests, but it does not allow planning of test sequences. This might be achieved in a way similar to our approach in [Castilho *et al.*, 1997b, Castilho *et al.*, to appear].

Second, our actions being restricted to tests (and their sequential and nondeterministic composition), our aim is to relax that restriction. This will probably require to move from a rewriting-based proof procedure towards a semantic tableaux procedure. Here we shall also make use of previous work [Castilho *et al.*, 1997a].

References

[Castilho *et al.*, 1997a] Marcos A. Castilho, Luis Fariñas del Cerro, Olivier Gasquet, et Andreas Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3/4):281–297, 1997.

[Castilho *et al.*, 1997b] Marcos A. Castilho, Olivier Gasquet, et Andreas Herzig. Modal tableaux for reasoning about actions and plans. In Sam Steel et Rachid Alami, editors, *European Conference on Planning (ECP'97)*, number 1348 in LNAI, pages 104–116. Springer Verlag, 1997.

[Castilho *et al.*, to appear] Marcos A. Castilho, Olivier Gasquet, et Andreas Herzig. Formalizing action and change in modal logic I: the frame problem. *Journal of Logic and Computation*, to appear.

[Fagin *et al.*, 1995] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, et Moshe Y. Vardi. *Reasoning about knowledge*. MIT Press, 1995.

[Gerbrandy et Groeneveld, 1997] Jelle Gerbrandy et Willem Groeneveld. Reasoning about information change. *J. of Logic, Language and Information*, 6(2), 1997.

[Gerbrandy, 1997] Jelle Gerbrandy. Dynamic epistemic logic. Technical report, ILLC, Amsterdam, 1997.

[Goldblatt, 1992] Robert Goldblatt. *Logics of time and computation*. Number 7 in Lecture Notes. CSLI, 1992.

[Groeneveld, 1995] Willem Groeneveld. *Logical investigations into dynamic semantics*. PhD thesis, ILLC, 1995. ILLC Dissertation Series 1995-18.

[Harel, 1984] David Harel. Dynamic logic. In Dov M. Gabbay et Franz Günthner, editors, *Handbook of Philosophical Logic*, volume II, pages 497–604. D. Reidel, Dordrecht, 1984.

[Hintikka, 1962] Jaakko K. K. Hintikka. *Knowledge and belief*. Cornell University Press, Ithaca, N.Y., 1962.

[Lakemeyer et Levesque, 1998] Gerhard Lakemeyer et Hector J. Levesque. Aol: a logic of action, sensing, knowing, and only knowing. In *Proc. 7th Int. Conf. on Knowledge Representation and Reasoning (KR'98)*, pages 316–326. Morgan Kaufmann Publishers, 1998.

[Levesque, 1996] Hector J. Levesque. What is planning in the presence of sensing. In *Proc. Nat. Conf. on AI (AAAI'96)*. AAAI Press, 1996.

[Scherl et Levesque, 1993] Richard Scherl et Hector J. Levesque. The frame problem and knowledge producing actions. In *Proc. Nat. Conf. on AI (AAAI'93)*, pages 689–695. AAAI Press, 1993.

[Segerberg,] Krister Segerberg. Two traditions in the logic of belief: bringing them together. to appear.

[Segerberg, 1995] Krister Segerberg. Belief revision from the point of view of doxastic logic. *Bulletin of the IGPL*, 3:534–553, 1995.

Formalizing Agent's Attitudes with the Polyadic π-Calculus[1]

Wenpin Jiao
Institute of Computing Technology, CAS
P.O.Box 2704-28
Beijing, 100080
P. R. China

Zhongzhi Shi
Institute of Computing Technology, CAS
P.O.Box 2704-28
Beijing, 100080
P. R. China

Abstract

To formalize mental attitudes of agents we apply a new approach different from those with logical frameworks. In this paper, we define mental attitudes as processes in a process calculus, the polyadic π-Calculus. We also give formal definitions for agents and agent-based systems. Based on those definitions, we can attain the result of interaction between the agent-based system and its environment.

1 Introduction

In varied computer systems, agents have been considered as the key computer-based components. Autonomous agents and multi-agent systems represent a new way of analyzing, designing, and implementing complex software systems [Jennings *et al.*, 1998].

However, there is no agreement on what an agent is, and every one declared that his system was based on agents though he assumed agent a different definition from others. This makes people understand agents in almost different ways since there are lack of effective means of holding agents' properties. It is necessary to describe or define agents precisely in a formal way in order to provide a unified basis for people to understand agents' properties and behaviors. By formalizing agent and its mental attitudes, we can conveniently analyze relationships among those attitudes and formally reason about agents' behaviors. In addition, we can provide not only a uniform semantics framework for agents, but also a theoretical and practical basis for designing and building agent-based software systems.

An agent is a computer system, situated in an environment, which is capable of flexible autonomous actions in order to meet its design objectives [Wooldridge and Jennings, 1995]. It is convenient to describe an agent by the intention stance. There are two important categories of attitudes to represent an agent appropriately:

- Information attitudes, such as belief, and knowledge, are related to the information that an agent has about the world it occupies.
- Pro-attitudes, such as desire, intention, obligation, commitment, and choice, etc., are those that in a way guide the agent's actions.

These two categories of attitudes are closely related. To characterize an agent, one should specify at least one information attitude and one pro-attitude for the agent.

When people tried to formalize and reason about intention notations in classical logic, they found that intention notations are referentially opaque [Wooldridge and Jennings, 1995]. So, alternative formalisms are required. There are two basic approaches to the semantic problem. One is to adopt a possible worlds semantic model [Chellas, 1980], to which there are many alternatives. The other is to use a sentential, or interpreted symbolic structures approach, in which beliefs are viewed as symbolic formulae explicitly represented in a data structure associated with an agent [Konolige, 1986].

In the possible worlds semantic model, there also associated many difficulties, for instance, the well-known logical omniscience problem. To address this problem, people tried to find alternatives to the possible worlds model.

For instance, Levesque[Levesque, 1984] proposed a solution that involves making a distinction between explicit and implicit belief. The semantics of the explicit belief operator were given in terms of a weakened possible worlds semantics, and the semantics of the implicit belief operator were given in terms of a standard possible worlds approach. However, it does not allow quantification; it does not seem to allow for nested beliefs; the notion of a situation is more mysterious than the notion of a world in possible worlds; and under certain circumstances, his proposal will make unrealistic predictions about agent's reasoning capabilities [Reichgelt, 1989].

Konolige [Konolige, 1986] proposed a deduction model to model resource bounded believers, which is a direct attempt to model the beliefs of symbolic AI systems.

Those formalisms above have focussed on just one aspect of agency. A realistic and complete agent theory, expressed in a logic, must can represent the static and dynamic aspects of agency and must define how the attributes of agency are related [Wooldridge and Jennings, 1995].

[1] Supported by National '863' Hi-Tech Project of China.

One of the best-known and most influential contributions to the area of agent theory is due to Cohen and Levesque [Cohen, 1990]. Their formalism was originally used to develop a theory of intention as a pre-requisite for a theory of speech acts.

In related work, Rao and Georgeff [Rao and Georgeff, 1991] have developed a logical framework for agent theory based on three primitive modalities: beliefs, desires, and intentions, which are based on a branching model of time. In the BDI architecture, beliefs correspond to information that the agent has about its environment; desires represent options available to the agent; and intentions represent states of affairs that the agent has chosen and has committed resources to. Researchers interested in practical reasoning architectures have developed a number of logical theories of BDI systems.

Singh[Singh, 1994] took a different approach to model agents. He developed a family of logic for representing intentions, beliefs, knowledge, know-how, and communication in a branching-time framework. However, it is too complex.

To reason about others, Shi [Shi *et al.*, 1997] proposed a knowledge representation framework called RAO to represent concepts and rules used in reasoning about knowledge of others. In the framework, a logic axiom schema was used to establish a direct relationship between speech acts and common sense, and this axiom is very like that one of situation calculus which describes the relationship between action and its effect.

However, those formalizing methods lack of a uniform semantics, which makes it hard to develop formalisms to capture the relationship between the various elements that comprise an agent's cognitive state. The questions such as which mental attitudes are the most essential ones, which attitudes can be derived from others, how those attitudes evolve, and which combination of attitudes is required to characterize an agent, are still required to be taken more attention.

In this paper, to specify agent and its mental attitudes, we adopt a new approach different from those with logical frameworks. We will formalize an agent and its attitudes with a process calculus, called Polyadic π-Calculus [Milner, 1993], which is an elementary calculus for describing and analyzing a concurrent system with evolving communication structure. In the polyadic π-calculus, a system is a collection of independent processes that communicate via channels.

In the rest part of this paper, we will first give an introduction to the polyadic π-calculus, and point out why we choose it as our formal method to formalize agents in Section 2. Then we will formally describe varied mental attitudes of agents in the polyadic π-calculus in Section 3. In Section 4, we will analyze relationships among those attitudes based on the formal descriptions, and then give formal definitions for agents. In Section 5, we give a formal definition for agent-based systems and an example to show how to attain the result of interaction between the system and its environment. Lastly, we will summary the whole paper and point out the further research direction we will go on with.

2 Preliminaries

The polyadic π-calculus [Milner, 1993] is a generalization from the (monadic) π-calculus [Milner *et al.*, 1992]. The π-calculus is a model of concurrent computation based upon the notion of naming, and it is a way of describing and analyzing systems consisting of agents which interact among each other, and whose configuration or neighborhood is continually changing. One can naturally express processes that have changing structure by using the π-calculus.

In the π-calculus, processes send or receive messages through a link between two ports opposite to each other. On a link, one can transmit variables, ordinary data values, and even link names, all of which are called names in the π-calculus. Since links can be transmitted through processes, that makes the π-calculus capable of describing dynamic structure of processes easily. After names are introduced into the π-calculus, the π-calculus can describe a set of data structures, more importantly, it can describe functional computation as the λ-calculus.

The polyadic π-calculus extends the π-calculus. In the polyadic π-calculus, messages transmitted between ports can be a name vector instead of a single name. In addition, The polyadic π-calculus uses the sort and sorting notations to guarantee the consistence of messages transmitted between ports.

2.1 The Components of the Calculus

In the polyadic π-calculus, names are the most primitive entity with no structure, and processes are built from names as follows:

1. A Summation $\Sigma_{i\in I}P_i = P_1+P_2+\ldots+P_n$. Execute one of P_i. When n=0 the sum is written as **0** and means stop.
2. A prefix form $\overline{yx}\cdot P$, $\tau\cdot P$, $y(\vec{x})\cdot P$. Output/Input the name vector x along the link y, or perform the silent action τ, and then behaves like P.
3. A composition $P_1|P_2$. P_1 and P_2 execute concurrently. The operation is commutative and associative.
4. A restriction (vy)P. Introduce a new name y with scope P (bind all free occurrences of y in P).
5. A match [x=y]P. Behave like P if the names x and y are identical, and otherwise like **0**.
6. A replication !P. Provide any number of copies of P.

2.2 The Transitional Semantics

In the following action rules, $\rightarrow$ represents a reduction procedure by which a process reduces to another process after an action such as yx, τ, $y(x)$

1. Communicating rule

In the polyadic π-calculus, computation is expressed by the following communicating rule.

COOM: $(\cdots + \overline{yx}.P) \mid (\cdots + y(\vec{z}).Q \rightarrow P \mid Q\{^{x}/_{z}\}$

This means sending vector x along channel y reduces the left-hand side to P|Q with all free occurrences of z in Q replaced by x. Where, vector x and vector z should have equal arity.

2. Parallel rule

Action between two parallel processes can be expressed by the following parallel rule.

PAR: $\dfrac{P \rightarrow P'}{P \mid Q \rightarrow P' \mid Q}$

It means if there is no communication between the two processes P and Q, their actions are interleaving.

3. Restriction Rule

RES: $\dfrac{P \rightarrow P'}{(\nu x)P \rightarrow (\nu x)P'}$

It means restriction by a name, which does not occur freely in a process, does not affect its behavior.

4. Structural Congruence Rule

STRUCT: $\dfrac{Q \equiv P \quad P \rightarrow P' \quad P' \equiv Q'}{Q \rightarrow Q'}$

It means if there are two structural congruence processes, they will act in the same way.

2.3 Simulation and Equivalence

In the polyadic π-calculus, one process may simulate or act similarly as another process; furthermore, two processes may have equivalent behaviors.

Definition 1. Strong Simulation. A binary relation S on processes is a strong simulation if it satisfies the following condition. S is a simulation if PSQ implies that

1. If $P \xrightarrow{\alpha} P'$ and α is free action,
   then for some Q', $Q \xrightarrow{\alpha} Q'$ and $P'SQ'$
2. If $P \xrightarrow{x(y)} P'$ and $y \notin n(P,Q)$,
   then for some Q', $Q \xrightarrow{x(y)} Q'$ and for all w, $P'\{w/y\}SQ'\{w/y\}$
3. If $P \xrightarrow{\bar{x}(y)} P'$ and $y \notin n(P,Q)$,
   then for some Q', $Q \xrightarrow{\bar{x}(y)} Q'$ and $P'SQ'$

This definition indicates that a process is strongly similar to another process if it can take the same actions as the later and has the same effects after taking the same actions.

If P simulates Q and Q simulates P as well, we say that P and Q are strongly bisimilar and marked as $P \dot{\sim} Q$.

Definition 2. Strong Equivalence. We say P and Q are strongly equivalent if Q and Q are strongly bisimilar and Pσ and Qσ are strongly bisimilar for all substitutions σ.

Where a substitution is a function from N to N, which is the name set of all processes in the polyadic π-calculus. A substitution σ can be represented as $\{y_i/x_i\}_{1 \le i \le n}$, for which $x_i\sigma=y_i$, $1 \le i \le n$, and otherwise $x\sigma=x$.

A process with free names likes an abstract model for concurrent computations and can be concretized by substituting those free names with new names. This definition means that two processes are the same computations if they can be concretized to be two similar processes. If P and Q are strongly equivalent, the relation between them can be marked as P~Q.

2.3 Why the π-Calculus

The powerful ability to represent concurrence, communication, composition, and dynamic structures among processes simply and flexibly in the polyadic π-calculus makes it be a good choice to formalize agent.

First, an agent has many static attributes, but it is not a static conception since it has its own actions and behaviors. In general, to describe an agent's static and dynamic properties, one may use two distinct formalizing strategies and approaches for each aspect, which is obviously unsuitable to grasp those properties and analyze relationships among them precisely and effectively. So, it has special signification to provide a uniform formal framework for an agent's static and dynamic properties. In process calculus, behaviors of agent can naturally be considered as concurrent processes, which cooperate by communication to accomplish distributed tasks; those static attributes of agent can also be regarded as processes, which provide some specified information about the agent to outside world.

Secondly, an agent is different from an object in conception [Jennings *et al.*, 1998], but an agent can be regarded as an autonomous, personified object with some mental attitudes. Behaviors in an agent are crudely concurrent, and an agent is a process-like, concurrent entity [Jennings *et al.*,1998].

Thirdly, an agent is situated in its environment and may possess intelligence to some extent. To adapt to its environment, an agent may dynamically change its own configuration or structure, which can be easily described in the polyadic π-calculus.

Fourthly, in multi-agent systems, behaviors of agents are inevitably concurrent, and the communicating counterpart of one agent may change dynamically. It may be a good alternative by using the polyadic π-calculus to describe concurrence and cooperating protocols among multi-agents.

3 Formalizing mental attitudes in the polyadic π-calculus

For convenience, we give an agent example as follows. In the following context, we will also formalize its mental attitudes in order to make our formal definitions more clearly.

"It is perfectly coherent to treat a light switch as a (very cooperative) agent with the capability of transmitting current at will, who invariably transmits current when it believes that we want it transmitted and not otherwise; flicking the switch is simply our way of communicating our desires". [Shoham, 1993]

In the example, we assume the switch agent can respond to users' intentions by turning switch on or off autonomously. From this example, we can summary that:

The agent knows that (1) if there is current being transmitted on the circuit the light will be on; otherwise, the light will be off. (2) Turning the switch off can cut off the current and turning on can make the current trans-

mitted. (3) It is capable of turning itself on/off autonomously.

The agent believes that at sometime, one may turn the light on when it is off; on the contrary, others may turn it off when it is on. After the agent perceives users' actions and understands the intentions related to those actions, the agent will react appropriately to achieve its desired goal such as turning the light on or off.

In the rest of this section, we will formalize agents' mental attitudes one by one. Before the formal definition is put forward, we will first describe each other informally.

3.1 Clock

In general, the behaviors of an agent are always related to a specified time, such as the past, the current, and the future. We define a clock process first to provide the system time for other attitude processes we will describe in the following context.

$$CLOCK(t) = !time(t) \qquad [1]$$

This process will provide the system time for others through the port *time*.

3.2 Knowledge

An agent's knowledge represents its understanding to the world, which includes information related to "what-is", such as facts, relationships, and capabilities of itself or others, and "how-to", such as actions or behaviors that the agent will adopt while it is going to achieve a goal.

In our opinions, an agent's knowledge is information about objective facts, which are unrelated to time. To define knowledge formally, we will not consider time into account. In addition, we will not distinguish knowledge related to "what-is" from that related to "how-to".

In the definition of the knowledge process, each kind of knowledge the agent possesses will correspond to a sub-process, and the whole knowledge process is composed of all of these sub-processes. In order to make an agent's knowledge be able to be referred to by other processes conveniently, we define each sub-process as the form: "Input names related to which knowledge to be referred to → the body of sub-process → output other names needed to be referred to more deeply."

The knowledge process can be defined as follows:

$$KNOWLEDGE(id) = \sum (vxy)\, knowledge_{id}(x).KBody.\bar{x}(y) \qquad [2]$$

Where, *id* is the identity of an agent.

For example, the knowledge process to extract the solution of "*x*'s father"from fact-typed knowledge can be defined as follows:

$$Father(id, x) = (vyz)\, father_{id}(x).[x = z]\bar{x}(y)$$

It means that if the father of z is y and the port $father_{id}$ gets a value for x equal to z the process will export y as the father of x. Then, the knowledge to extract the solution of "*x*'s grandpa" can be represented as the following process:

$$Grandpa(id, x) = (vy)\overline{father_{id}}(x) \mid Father(id,x).x(y).(\overline{father_{id}}(y) \mid Father(id,y))$$

For another example, for the light switch agent, the agent's knowledge may include:

- To turn the light on/off, the current must be or not be transmitted on the circuit.

$$LIGHT(id) = (vx)\, light_{id}(x).([x = lighton]\overline{current}(transmitted) + [x = lightoff]\overline{current}(nontransmitted))$$

- To or not to transmit current on the circuit, the switch must be turned on/off.

$$CURENT(id) = (vx)current(x).([x = transmitted]\overline{switch}(switchon) + [x = nontransmitted]\overline{switch}(switchoff))$$

- The agent knows that it can turn the switch on/off autonomously.

$$SWITCH(id) = (vx)switch(x).([x = switchon]TurnSwitch\ On) + [x = switchoff]TurnSwitch\ Off))$$

Thus, the knowledge process of the switch agent can be defined formally as follows:

$$KNOWLEDGE(id) = !LIGHT(id) \mid !CURRENT(id) \mid !SWITCH(id)$$

3.3 Belief

An agent's beliefs represent that the agent accepts something as true or real, for example, both "one believes it will rain tomorrow" and "one believes that all crows in the world are black" are beliefs. Belief is different from knowledge. Beliefs are reflections of one's subjective world, that is to say, the truths of beliefs are uncertain and do not depend completely on the object one believes. On the contrary, knowledge is the reflection of the objective world and its truth is definite.

Obviously, whether an agent believes something or not is often related to a specified time, for instance, one may believe that it will rain tomorrow before he hears the weather forecast, but he may not believe again once he heard the weather forecast. We assume that the behaviors of an agent are dominated by its beliefs, that is, the agent wants to achieve a goal is because it knows that it is capable of doing so and believes the goal will be achieved eventually.

The belief process can be defined as follows:

$$BELIEF(id) = \sum (vt, Nid, s)\, Bel(Nid, t, s) \qquad [3]$$

Where, *Bel* is

$$Bel(Nid,t,s) = \sum (v\tau)time(t).[t = \tau](belief_{id}(Nid,s).\overline{Nid}(t,s)) \qquad [4]$$

It means that the agent believes another agent identified by *Nid* will be in the state s, and export s as the result through the port *Nid* at time τ, where s may be a vector.

For example, the switch agent believes that it will be on/off at some time or will receive a request for turning on/off the light.

$$Bel(id, t, s) = (v\tau)time(t).[t = \tau]subbelief(id, s)$$

Where, $subbelief(id, s) =$

$$belief_{id}(id,s).([s = lighton]\overline{id}(t,lighton) + [s = lightoff]\overline{id}(t,lightoff))$$

Thus, the belief process of the switch agent is as follows:

$$BELIEF(id) = Bel(id, t, lighton) + Bel(id, t, lightoff)$$

3.4 Goal

Each agent has a set of achievable goals. While defining the goal process of an agent, we must provide a definition which can not only point out which goals the agent may have, but also provide ways of referring to its knowledge through its goals.

The goal process is defined as follows:

$$GOAL(id) = \sum (v\,\vec{g}, \vec{k})\, goal_{id}(\vec{g}) \overline{knowledge_{id}}(\vec{k}) \quad [5]$$

Where, g represents the goal that the agent will achieve, and k is the knowledge that the agent will use to achieve the goal. It means for a goal g, the agent will access some knowledge about k.

For example, the goal process of the switch agent can be defined as follows:

$$GOAL(id) = (vs)goal_{id}(s) \cdot \overline{light_{id}}(s)$$

Where, s can be *lighton* or *lightoff*.

3.5 Desire

An agent's desires represent which goals the agent want to achieve, which are internal reflections of the agent's autonomy. As one wants to eat when he feels hungry, desires are often inner requirements instead of coming from outer stimulation, and the agent should believe that its desires could be achieved at some time.

The desire process likes an introspective process, which can be defined as follows:

$$DESIRE(id) = \sum (vt,d)\, inlook_{id}(t,d) \cdot [d = g_i](\overline{time}(t) \mid \overline{belief_{id}}(id,d) \mid \overline{goal_{id}}(d)) \quad [6]$$

Where, $inlook_{id}(t,d)$ represents the introspecting process, which is to inspect whether there occurs a desire d at time t. If the agent is desired to achieve a goal g_i, it will judge whether it believes the goal can be achieved, and then send itself a request to achieve the goal.

3.6 Intention

An agent's intentions represent the goals that the agent has decided to achieve. Intention is different from desire. An intention goes always with actions and is the goal of some behaviors, whereas one has a desire may not be committed to action. When the agent finds its environment can meet its requirement to achieve a goal, it will take this goal as its intention.

The intention process can be defined as follows:

$$INTENTION(id) = \sum (vt,d)\, outlook_{id}(t,d) \cdot [d = g_i](\overline{time}(t) \mid \overline{belief_{id}}(id,d) \mid \overline{goal_{id}}(d)) \quad [7]$$

Where, the agent perceives events or states occurring in its environment by port $outlook_{id}(t,d)$ and understands the intention contained in those events or states. If the agent want to achieve a goal g_i at time t, it will judge whether it believes the goal can be achieved, and then send itself a request to achieve the goal.

For example, when someone wants the light to be on or off, the switch agent extracts the intention from the user's request, and produces a goal corresponding to this intention. The intention process is as follows:

$$INTENTION(id) = (vt,d)\, outlook_{id}(t,d) \cdot ([d = lighton](\overline{time}(t) \mid \overline{belief_{id}}(id,d) \mid \overline{goal_{id}}(d)) + [d = lightoff](\overline{time}(t) \mid \overline{belief_{id}}(id,d) \mid \overline{goal_{id}}(d))$$

3.7 Obligation/Commitment

An agent's obligations or commitments represent some actions the agent should carry out because it is obligated to or it has decided to do so. For instance, once the switch agent responds to the request for turning the light on, it is said that the agent has been committed to make the light on, and has obligation to do so.

The obligation process is defined as follows:

$$OBLIGATION(id) = \sum (vt,r)\, obligation_{id}(t,r) \cdot [r = g_i](\overline{time}(t) \mid \overline{belief_{id}}(id,r) \mid \overline{goal_{id}}(r)) \quad [8]$$

Where, $obligation_{id}(t,r)$ represents that the agent will be committed to the request r. Once the agent responds to or receives the request, it should try to achieve the goal related to the request. If the agent believes the goal can be achieved, it will send itself a request to achieve the goal.

3.8 Decision/Choice

An agent's Decisions are actions that the agent chooses to achieve a goal. The agent can have more than one goal at a time, and it can also have more than one solution to one goal. Even though an agent can also entertain several beliefs at a time, it cannot believe a specific goal to be achievable and unachievable at the same time. That is to say, for one goal, the agent must have only one belief. To deal with its goals needs the agent to make decision to choose an appropriate goal and a good enough solution for that goal. Since it allows that there is more than one concurrent process running parallelly within an agent, the agent can choose several goals at a time. So, we only take the decision to choose a goal and give a solution into account in the definition of the decision process.

The decision process is defined as follows:

$$DECISION(id) = \sum (vx,y)\, decision_{id}(x) \cdot DBody \cdot \bar{x}(y) \quad [9]$$

That is, the agent makes decision according to the goal x it will achieve, and then goes more deeply into another goal y or a sub-goal of x that the agent must achieve in advance.

4 Relationships among mental attitudes

In the section above, we formally described two categories of attitudes, information attitudes and pro-attitudes. By analyzing those formal descriptions, we can find some relationships among them as follows.

Relation 1. The intention process of an agent is strongly equivalent to the obligation/commitment process, that is

$$INTENTION(id) \sim OBLIGATION(id) \quad [10]$$

It means that the agent has taken actions based on its intention has the same effect as it is obligated to take actions. It could be said that the obligation of an agent is indeed a kind of intentions of the agent.

Proof. For the intention process, by substituting $outlook_{id}$ with $obligation_{id}$, its definition is equal to that of the obligation process. Thus, the conclusion can be easily drawn from the definition of strong equivalence.

Relation 2. The knowledge process of an agent is strongly similar to the decision process.

Since an agent knows how to achieve its goals, it indeed knows how to make decision to take actions when it wants to achieve a goal. As we pointed out early that an agent knows information related to both "how-to" and "what-is", the decision process is only a concrete form corresponding to a part of the knowledge process.

Relation 3. The desire process is strongly equivalent to the intention process, that is

$$DESIRE(id) \sim INTENTION(id) \quad [11]$$

As described above, an agent can not only respond to its inner stimulation, but also react to the changes of states or occurrences of events in the environment. If we do not take the difference between inner and outer stimulation into consideration, we can find that the two processes will take the same actions after they perceive some kind of stimulation. However, the two processes are not completely same. From the desire process, the autonomy of an agent can be reflected, while the reactivity of an agent from the intention process.

From these relationships among mental attitudes, we can draw a conclusion that an agent can be defined using its knowledge, beliefs, goals, desires and intentions. The agent process can be formally defined as follows.

$$AGENT(id) = KNOWLEDGE(id) \mid BELIEF(id) \mid GOAL(id) \mid DESIRE(id) \mid INTNETION(id) \quad [12]$$

Once an agent is constructed by composing those mental attitude processes, the mental attitude processes will act parallelly and interwovenly. Thus, there will be some other relations among those processes

Relation 4. An agent cannot achieve its any goal if it has no enough knowledge related to that goal.

That is a goal process cannot finish successfully unless there is some knowledge process who can provide related knowledge for the goal process.

Relation 5. Before an agent takes actions to achieve its goals, it should believe first that its goals could be achieved.

This relation can be shown from the definitions of the desire and the intention processes. While the agent is take actions after it perceives some kind of stimulation, it cannot go any more until it makes sure that a belief process says that it has believed the goal can be achieved.

5 Agent-based systems

Since an agent is always situated in an environment, it will interact with its environment while it is processing its inner transactions. That is the agent should react appropriately to those events occurring in or out of itself.

From the point of view of an agent, the environment is the place that the outer events come from, and it itself is the place that the inner events come from. The environment corresponding to an agent identified with *id* can be defined as follows.

$$ENV(id) = (v\tau, c)\, env_{id} \cdot \overline{outlook_{id}}(\tau, c) \quad [13]$$

Where, env_{id} captures events occurring in the environment, and then waits for the agent's perceiving. Similarly, the process to produce inner events can be defined as follows.

$$SELF(id) = (v\tau, c)\, self_{id} \cdot \overline{inlook_{id}}(\tau, c) \quad [14]$$

Where, $self_{id}$ captures events happening in the agent. From the two definitions above, we can find that there are two kinds of stimulation that will be perceived by two mental processes, i.e. the intention process and the desire process, respectively. For these two kinds of stimulation, the environment will produce stimulation resulting in reactive actions of the agent, and the agent itself will result in autonomous behaviors.

The agent-based system composed of agent and its environment can be defined as follows.

$$SYSTEM = \sum Agent(id_i) \mid Self(id_i) \mid Env(id_i) \quad [15]$$

For example, the switch agent process is as follows.

LightSwitch = KNOWLEDGE(id) | BELIEF(id)
| GOAL(id) | DESIRE(id) | INTENTION(id)

$$= \left(\begin{array}{l} (vx)\, light_{id}(x) \cdot ([x = lighton]\overline{current}(transmitted) + \\ \qquad [x = lightoff]\overline{current}(nontransmitted)) \\ \mid (vx) current(x) \cdot ([x = transmitted]\overline{switch}(switchon) + \\ \qquad [x = nontransmitted]\overline{switch}(switchoff)) \\ \mid (vx) switch(x) \cdot ([x = switchon] TurnSwitchOn) + \\ \qquad [x = switchoff] TurnSwitchOff)) \end{array} \right)$$

$$\mid (v\tau) time(t) \cdot [t = \tau]$$

$$belief_{id}(id, s) \cdot ([s = lighton]\overline{id}(t, lighton) + [s = lightoff]\overline{id}(t, lightoff))$$

$$\mid (vs)\, goal_{id}(s) \cdot \overline{light_{id}}(s)$$

$$\mid (vt, d)\, outlook_{id}(t, d) \cdot ([d = lighton](\overline{time}(t) \mid \overline{belief_{id}}(id, d) \mid \overline{goal_{id}}(d)) + [d = lightoff](\overline{time}(t) \mid \overline{belief_{id}}(id, d) \mid \overline{goal_{id}}(d))$$

And we assume that the user, which will play the role of the environment, wishes the light were on at time τ, then the process may be

$$USER(id) = \overline{outlook_{id}}(\tau, On)$$

The reaction that the agent perceives the stimulation from the user will be as follows.

Step 1. The agent perceives the user's action and understands the intention.

$USER(id) \mid LightSwitch$

$\rightarrow KNOWLEDGE(id) \mid BELIEF(id) \mid GOAL(id)$

$\mid (vd)\, outlook_{id}(\tau, d) \cdot$

$([d = lighton](\overline{time}(\tau) \mid \overline{belief_{id}}(id, d) \mid \overline{goal_{id}}(d)) +$

$[d = lightoff](\overline{time}(\tau) \mid \overline{belief_{id}}(id, d) \mid \overline{goal_{id}}(d))$

$\rightarrow KNOWLEDGE(id) \mid \underline{BELIEF(id)}$

$\mid GOAL(id) \mid (\overline{time}(\tau) \mid \overline{belief_{id}}(id, lighton) \mid \overline{goal_{id}}(lighton)$

Step2. The agent makes sure that it believes the goal can be achieved at time τ.

$\rightarrow KNOWLEDGE(id)$

$\mid GOAL(id) \mid \overline{id}(\tau, lighton) \mid \overline{goal_{id}}(lighton)$

Step3. To achieve the goal, the agent looks for which kind of knowledge it should refer to.

$\rightarrow KNOWLEDGE(id) \mid \overline{id}(\tau, lighton) \mid \overline{light_{id}}(lighton)$

Step4. After querying its knowledge about how to achieve the goal, the agent put the result out.

$\rightarrow \overline{id}(\tau, lighton) \mid TurnSwitchOn$

It means that the agent will export a state *lighton*, and turn the switch on at time τ.

6 Summary

When one formalizes intention notations related to an agent using logical methods such as modal and temporal logics, he will come across some problems such as *opaque contexts*, *logical omniscience*, *side effect*, etc [Wooldridge and Jennings, 1995]. Researchers have to select alternative approaches. In general, they try to use more than one modal operator to formalize agents, and build their semantics on a possible worlds model. However, too many modal operators will make the formalization more complex, and make it hard to be understood and accepted by people. On the other hand, those new modal operators, such as belief, intention modal operators, have no grounded semantics, which will lead to inconsistent cognition to agents. All of those will result in difficulties to analyze, reason about, and verify the properties of agents.

To formalize mental attitudes of agent we apply a new approach different from those with logical frameworks. In this paper, we define agent and its mental attitudes as processes in a process calculus, the polyadic π-Calculus. This method can easily describe not only those static attributes of agents, but also the dynamics of agents. By defining the mental attitudes of agents as processes, we can conveniently represent and analyze the relationship among those attitudes. More importantly, we can build the agent by composing those processes together, and reason about its actions. After providing formal definitions for varieties of attitudes, we compared them. We found that the obligation is indeed a kind of intentions; and the decision is scoped with its knowledge and can be described as a kind of knowledge. So, we can say that an agent can be defined using its knowledge, beliefs, goals, desires, and intentions. Using those definitions, we also gave the formal definitions for agent and agent-based system.

By now, we have implemented a building tool for multi-agent systems called AOSDE [Shi *et al*, 1998], which contains a general purpose agent kernel for all agents. In the next stage, we will evolve this system with more powerful functions, such as designing, implementing, and testing agent-oriented software. The formalism in this paper will act as the formal basis for specification, refinement, and verification of software built in that environment. Based on this formalism, we will go more deeply to study the formal semantics of agent description language and agent communicating language at the next step.

References

[Chellas, 1980] B. Chellas. *Modal Logic: An Introduction*. Cambridge University Press: Cambridge, England, 1980.

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

[Jennings *et al.*, 1998] N. R. Jennings, K. Sycara, and M. Wooldridge. A Roadmap of Agent Research and Development, *Int. Journal of Autonomous Agents and Multi-Agent System*, 1(1), 7-38, 1998.

[Konolige, 1986] K. Konolige. *A Deduction Model of Belief*. Pitman Publishing: London and Morgan Kaufmann: San Mateo, CA, 1986.

[Levesque, 1984] H. J. Levesque. A logic of implicit and explicit belief. In *Proceedings of the Fourth National Conference on Artificial Intelligence(AAAI-84)*, pp.198-202, Austin, TX, 1984.

[Milner, 1993] R. Milner. The polyadic π-calculus: a tutorial, in *Logic and Algebra of Specification*, ed. F.L. Bauer, W. Brauer and H. Schwichttenberg, Springer Verlag, 1993, pp.203-246.

[Milner *et al.*, 1992] R. Milner, J. Parrow, and D. Walker. A Calculus of Mobile Processes, Part I, II. *Journal of Information and Computation*, Vol.100, 1992, pp.1-77.

[Reichegelt, 1989] H. Reichgelt. Logics for reasoning about knowledge and belief. *Knowledge Engineering review*, 4(2):119-139, 1989.

[Rao and Georgeff, 1991] A. S. Rao, and M. P. Georgeff. Modeling rational agents within a BDI-architecture. In Fikes, R. and Sandewall, E., editors, *Proceedings of Knowledge Representation and Reasoning(KR&R-91)*, pp.473-484. Morgan Kaufmann Publishers: San Mateo, CA, 1991.

[Shi *et al.*, 1997] Zhongzhi Shi, Qijia Tian, and Yunfeng Li. RAO Logic for Multiagent Framework, *DAIMAS'97*, St. Petersburg, Russia, 1997.

[Shoham, 1993] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51-92, 1993.

[Singh, 1994] M. P. Singh. Multiagent Systems: A Theoretical Framework for Intentions, Know-How, and Communications, *LNAI 799*, Springer-Verlag: Heidelberg, Germny, 1994.

[Wooldridge and Jennings, 1995] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115-152, 1995.

Abstract:
On the role of action logics and deontic logics in specifying protocols

Christen Krogh
SINTEF Telecom and Informatics
P.O.Box 124 Blindern
0314 Oslo, NORWAY
mailto:christen.krogh@sintef.no
http://www.informatics.sintef.no/~chk/krogh

Abstract

Some uses of deontic logics and action logics in computer science are identified and analyzed: designing agent communication languages, security protocol analysis, and enterprise modeling. Intuitions and analyses from the philosophical use of these logics are argued to be of outermost importance for employing these logics in computer science.

1 Applications of logics for norm and action

There is a distinction to be draw between pure and applied topics. Consider the Cambridge mathematician G.H.Hardy's view on the matter [Newman, 1956, page 2024]

To qualify as pure, Hardy said, a [...] topic had to be useless; if useless, it was not only pure, but beautiful. If useful - which is to say impure - it was ugly, and the more useful, the more ugly.

Deontic logic was intended to be applied to moral and legal problems, and are by definition applied and not pure, and therefore, according to Hardy, ugly. The uglier the better, some would say, and I agree. The same holds for action logics.

When explicating basic normative or praxiological notions the deontic or action logics are usually seen to be modifiable. Through conceptual analysis a deontic or action logic is tuned to a particular family of notions. Once having done this, the logics may be viewed as a more static, but perhaps more precise tool for analysis. The logics can thus be applied on several levels. First, it can be used as a philosophical tool to analyse for instance moral principles, legal dictums, or agent rationality. Second, it can be applied within various fields either properly belonging to, or being a crossbreed with, computer science. The talk will comment on the last of these, with an emphasis on how to use deontic logic and action logic in devising agent communication languages for facilitating e-commerce and enterprise modelling.

2 Shopping and fucking[1]

Shopping is an action that results in establishing a normative relationship. Fucking, as an action resulting from shopping, is usually about violating a normative relationship. The first action is a popular example from recent experiences with internet-based ecomerce that can gain much from an analysis by deontic and action logics. The second action may be just as popular but is nevertheless not as interesting from a logical point of view.

Wesley Newcomb Hohfeld's theory legal conceptions [Hohfeld, 1913] lends itself to a semiformal analysis of the rights relations between two parties entering into a contract. The formal theories building on Hohfeld's theory that was developed by Stig Kanger [Kanger, 1957] and Lars Lindahl [Lindahl, 1977] is sufficiently precise to enable semi-exhaustive analysis of possible states regarding violation or non-violation of two or more agents bound by a contract. By trivial extensions to this theory it is possible to devise a class of protocols which can be the basis of agent communication languages for semi-autonomous retailing (cf. [Krogh, 1999]). The formal framework can be developed further in order to facilitate a normative analysis of security protocols (cf. [Krogh, 1999b])

3 The Enterprise Perpective

Recently, deontic logic has been forwarded as one possible means of modeling massively distributed computer systems from an enterprise perspective [5]. The approach follows the emerging ISO standard on Open Distributed processing [6]. Within the enterprise viewpoint, the notion of community is central. A community is established

[1] The title refers to Mark Ravenhill's infamous play.

by means of objects (or agents) entering into a contract. The notion quality of service (QoS) that is commonly believed to be of importance in future ICT-systems[2] lends itself to a rather trivial reduction to such contracts. By employing the formal framework established by Kanger and Lindahl, building on Hohfeld's jurisprudential intuitions, a formal analysis of distributed information systems, the quality of service offered by (parts of) these systems, and guidelines for how to reflect such analysis in other viewpoints is possible.

References

[Hohfeld, 1913] W. Wesley Newcomb Hohfeld, Fundamental Legal Conceptions as Applied in Judicial Reasoning and other Legal Essays. *Yale Law journal*, 1913.

[Kanger, 1957] Stig Kanger, New Foundations for Ethical theory. Technical Report, Stockholm University, Stockholm 1957.

[Krogh, 1999] Christen Krogh and Henning Herrestad, Hohfeld in Cyberspace and other applications of normative reasoning in agent technology, *Artificial Intelligence and Law*, 7, pages 81-96, 1999.

[Krogh, 1999b] Christen Krogh and Andrew J.I. Jones, Protocol Breaches and Violation Flaws, in P. McNamara and H. Pracken (eds), *Norms, Logics and Information Systems*, IOS press, Amsterdam, 1999.

[Lindahl, 1977] Lars Lindahl, *Position and Change*, D.Reidel Publ. Comp., Dordrecht - Holland, 1977.

[Linington et al, 1998] Peter Linington and Zoran Milosevic and Kerry Raymond, Policies in communities: Extending the ODP enterprise viewpoint, IEEE 1998. 0-7803-5060-X/98/.

[Newman, 1956] R. James Newman. Commentary on G.H.Hardy. In R. James Newman, editor, *The world of mathematics*, volume I-IV, pages 2024-2026. Simon and Schuster, New York, 1956.

[RM-ODP, 1999] Information Technology - Open Distributed Processing - Reference Model - Enterprise viewpoint. ITU-T Recommendation X.911, ISO/IEC 15414, Working Draft 3/99.

[2] Information and Communication Technology.

Practical Reasoning and Plan Execution with Active Logic *

K. Purang, Darsana Purushothaman, David Traum, Carl Andersen, Don Perlis
Computer Science Department, University of Maryland
College Park, MD 20742 USA
phone: +1 (301) 405-1139 fax: +1 (301) 405-6707
{kpurang,darsana,traum,cfa,perlis}@cs.umd.edu

Abstract

We present an approach toward design of a rational agent, integrating aspects of theoretical reasoning, practical reasoning, and reasoning about and executing plans. The approach uses Active Logic, which combines reactivity and logical inference, taking resource bounds into account, and providing mechanisms for handling contradiction. We augment this logic with a formalization of practical reasoning and plan execution, which also makes uses of contradiction handling abilities to cope with plan failure. We conclude with a description of a preliminary implementation and plans for embedding that within a dialogue system.

1 Introduction

In this paper, we present an approach toward design of a rational agent, integrating aspects of theoretical reasoning, practical reasoning, and reasoning about and executing plans. The approach, based on Active Logic [Elgot-Drapkin and Perlis, 1990], couples a view of belief as resulting explicitly from inference (or observation), with a resource-bounded approach to inference. Thus not all consequences of an agent's beliefs will be believed (currently), and doing the inference necessary to establish these consequences as beliefs will take time, during which other changes to the world may happen. Also key to this approach is an ability to handle contradictory beliefs in a robust manner. The inference procedure is set up so that contradictions in beliefs will have only limited (and recoverable) effects on the inferability of other beliefs. Noticing contradictions drives much of the further inference, including both theoretical and practical reasoning.

We model the components of practical reasoning in a fairly intuitive, commonsense way rather than attempting a comprehensive account of the tricky issues involved in such notions as knowledge, intentions and obligations. Beliefs are represented directly as a sequence of sets of propositions (one set per time point), and also using an introspection operator. Part of the beliefs includes a theory of action, including plan recipes with pre- and post-conditions and linear decompositions including sub-actions and subgoal states. Practical reasoning is accomplished using modalities *Goal* (an end state), *Adopt* (marking the current state of execution of a plan), and *Expect* (marking the anticipated results of an adopted plan). A key feature of the approach is a natural integration of inference, (normal) plan execution, detection of plan failure, and re-planning and acting.

In the next section we highlight some of the main features of active logic. We then describe, in Section 3, an initial formalization for reasoning about action and practical reasoning within active logic. In section 4, we present initial efforts at implementing an agent using an architecture that gives active logic sensors and effectors to interact with the world (the electronic world). We conclude with some future directions, using this agent as the basis for a natural language dialogue system.

2 Active Logic

Active logics were developed as a means of combining the best of two worlds – inference and reactivity – without giving up much of either. This requires a special evolving-during-inference model of time.

A key example is deadline-coupled reasoning. An approaching deadline must be factored into one's reasoning, even seen as an evolving part of that reasoning, rather than as a separate concern outside the reasoning process. Thus the remaining time (deadline – current_time) shrinks steadily as one attempts to find a solution to the problem.

The formal changes required for such a logic are, in some respects, quite modest. The language can be that of a first-order logic, perhaps augmented with names for expressions to facilitate meta-reasoning. The principal change is that inference rules become time-sensitive. The most obvious case is that of reasoning about time itself, as in the rule

*This research was supported in part by the National Science Foundation (IIS-9724937)

```
i:        Now(i)
          ----------
i+1:      Now(i+1)
```

The above indicates that from the belief (at time i) that the current time is in fact i, one concludes that it *now* is the later time $i+1$. That is, time does not stand still as one reasons.

For instance, suppose you are driving en route to the airport and planning details of your route as you go. You wonder whether to take the more direct but more heavily traveled road, or another. There are many facts to consider (time of day, day of week, radio traffic and weather reports) and many implications to ferret out (the radio is not broadcasting any traffic news, but it may be due to lack of such news or to their obsession with announcing a baseball game, etc). You quickly realize that your flight will be gone before you can figure out ramifications to all these subtleties. So you decide to stop worrying about the best of all possible routes, and instead content yourself with any one that seems likely to work.

Using active-logic inference rules such as that above, deadline-coupled reasoning has been formalized and applied to planning problems (see [Nirkhe *et al.*, 1997] where time of plan-enactment is crucial.

Technically, an active logic consists of a first-order language, a set of time-sensitive inference rules, and an observation-function that specifies an environment in which the logic "runs". Thus an active logic is not pure formalism but is a hybrid of formal system and embedded inference engine, where the formal behavior is tied to the environment via the observations and the internal monitoring of time-passage (see [Elgot-Drapkin and Perlis, 1990] for a detailed description).

In the above example, the reactivity is not to external events but rather to the universal event of time-passage vis-a-vis one's own reasoning. One can conceptualize this externally in terms of looking at a clock but this is not necessary or particularly helpful. On the other hand, external events are often quite important, as we discuss later.

Active logics are able to react to incoming information (including dialogue utterances by a collaborative partner) while reasoning is ongoing, blending new inputs into its inferences without having to start up a new theorem-proving effort. Thus, any helpful communications of a partner (or user) – whether as new initiatives, or in response to system requests – can be fully integrated with the system's evolving reasoning. Similarly, external observations of actions or events can be made during the reasoning process and also factored into that process.

Thus the notion of theorem for active logics is a bit different from that of more traditional logics, in several respects:

1. **Time sensitivity.** Theorems come and go; that is, a wff once proved remains proved but only in the sense of its being a historical fact that was once proved. That historical fact is recorded for potential use, but the wff itself need not continue to be available for use in future inferences; it might not even be reprovable, if the "axioms" (belief) set has changed sufficiently. As a trivial example, suppose $Now(noon) \rightarrow Lunchtime$ is an axiom. At time t=noon, Now(noon) will be inferred from the rule given earlier, and Lunchtime will be inferred a step later. But then Now(noon+1) is inferred, and Lunchtime is no longer inferable since its premise Now(noon) is no longer in the belief set. Lunchtime will remain in the belief set until it is no longer "inherited"; the rules for inheritance are themselves inference rules. One such involves contradiction; see next item.

2. **Contradictions.** If a direct contradiction (P and $\neg P$) occurs in the belief set at time t, that fact is noted at time t+1 by means of the inference rule

```
t:        P , ~P
          ----------
t+1:      Contra(i+1,P, ~P)
```

Also, as a consequence of the contradiction having been noted, neither of these instances of P or $\neg P$ will be used in future inferences. Thus the logic is partially shielded from using particularly blatant contradictands. P and $\neg P$ remain theorems in the sense of having been proved, but are not available for further inference. For instance, if Lunchtime is contradicted by $\neg Lunchtime$, neither of these is inherited to the next time step; but $\neg Lunchtime$ may well be reproven and thus in a sense "wins". This can occur as a result of a further axiom, such as $\neg Now(noon) \rightarrow \neg Lunchtime$. We can also provide contradiction-handling axioms that are domain independent or domain dependent for specific domains. These can use information about the domain but also information about the knowledge base and proofs to arbitrate between the contradictands.

Much more subtle effects can occur from this feature. To return to the airport example: you are driving en route to the airport and planning details of your route as you go. Then your car gets a flat tire. Rather than complete your original plans, it is time to make major revisions, in fact even to change your goals for the time being.

Truth maintenance systems [Doyle, 1979] also tolerate contradictions and resolve them typically using justification information. This happens in a separate process which runs while the reasoning engine is waiting. We do not think

that this will not work in general since the reasoning needed to resolve the contradiction will depend on the very information that generated it. Reasoning and the resolution of contradictions have to take place in the same reasoning process.

3. **Defaults.** Defaults can be given a straightforward representation in an evolving-time framework:

```
t:          ~Known(~P), Q
            ---------------------
t+1:              P
```

Here from the facts that Q, and that $\neg P$ is not a belief at time t, P is inferred. This avoids the decidability issues of traditional default mechanisms, since only a linear lookup in the belief set for time t is needed to tell that $\neg P$ is not there (and that Q is there). This does not in itself deal with problems arising from interacting defaults. However, since such cases tend to involve contradictory conclusions, these then can be treated as any other contradictands.

4. **Observations.** In active logic the flat tire in the previous example can be represented in terms of *observations*. And the reasoning simply goes on with this new information. There is no executive subsystem that turns off the route planner midstream and starts up a new planning action. Rather there is a single stream of reasoning, which can monitor itself by looking backwards at one moment to see what it has been doing in the past, including the very recent past. If the previous few steps in some way conflict with new information, then the next few steps can be devoted to sorting out enough of the apparent mismatch to allow a decision as to how to proceed. All of this is carried out in the same inferential process as the original planning, without the need for level upon level of meta-reasoners. This is not to say that there is no metareasoning here, but rather that it is "in-line" metareasoning, all at one level. The advantages of this are (i) simplicity of design, (ii) no infinite regress, and (iii) no reasoning time at higher levels unaccounted for at lower levels.

   A potential disadvantage is the possibility of vicious self-reference. This matter is a topic of current investigation. However, another major advantage of such time-sensitive in-line metareasoning is that inconsistency in one's beliefs need not cause serious problems in general. The reason is largely that given above: a conflict in the reasoner's beliefs can be noted by the reasoner as a new belief, and the latter can lead to a decision to encapsulate the conflicting beliefs so that they do no harm. Now this cannot be a fully general process, since identifying contradictions is at best semi-decidable. However, deeply hidden contradictions usually do little harm; and so we have concentrated on inference rules for "direct" contradictions, that is, belief pairs that surface in the form P and $\neg P$; see [Miller, 1993] for details including a theorem providing a rather general case in which such in-line metareasoning can cope with direct contradictions.

5. **Evolving state representation.** Another feature that comes directly out of the time-coupled nature of active logics is their ability to represent the evolving status of reasoning and actions. The representation of actions can avail itself of up-to-date time information. Thus an action A can be marked as Planned, Underway, and Done; and the logic can pass from one to another of these as actions are put into execution. Thus active logics not only reason about plans, but can make and execute them while keeping track of this changing state.

   It is easy to write an inference rule that updates at each time step whether a particular plan is currently being started, is already underway, or is completed. More detail than this, such as how long the plan execution has been going on, is also readily inferred. This is important for various purposes, such as:

   (i) avoiding re-initiation of a plan already underway

   (ii) assessing whether one is spending too much time on a goal

   (iii) distinguishing between various instances of a plan, one underway, another finished, perhaps a third and fourth on the to-do list. This is useful for repetitive activities, such as transporting objects one by one, or keeping track of how many times one is performing a certain action (for instance in dialogue, where one may repeat a request a few times for emphasis or as a reminder, but not indefinitely, without a kind of breakdown of coherence [Suchman, 1987]).

Active logics can be seen either as formalisms per se, or as inference engines that implement formalisms. This double-role aspect is not accidental: it is inherent to the conception of an active logic that it have a behavior, ie, the notion of theoremhood depends directly on two things that are not part of traditional logics: (i) what is in the current evolving belief set, and (ii) what the current evolving time is.

The traditional markers of a logic are its syntax and its semantics. Active logics have both of these: the syntax is (usually) that of FOL; and the semantics can also be that of FOL with a few addenda such as that Now(x) has the meaning that the current evolving time is x. (There are also alternative

semantics available.) What is missing is a soundness and completeness theorem, and for good reason: active logics are not intended to be sound or complete but rather to reflect the step-by-step process of reasoning of a real agent. Thus many true assertions will not be proven, and many things proven are not true. In fact, active logics are designed with inconsistent belief sets in mind; and these of course can never be true.

It is best to avoid a mere terminological squabble over the word "logic". However, in many important senses, active logics are formal specifications of notions of theoremhood appropriate to the study of real agents. If we are concerned about agents and their reasoning, rather than about an agent-independent notion of truth, then we should not expect or want a tight coupling between what is proven (or provable) and what is true. Agents can only do what they have the resources to do, and whatever logic an agent uses must therefore also have that property. Thus to the extent that logic is the study of reasoning, active logics are the study of reasoning as an active process.

Active Logic provides the theoretical reasoning component of our framework. However it also has many convenient features for practical reasoning, particularly the time-situatedness and contradiction handling facilities. This provides a natural mechanism for plan reasoning and acting, as well as failure detection and re-planning. In the next section, we describe a preliminary formalization of reasoning about action and plan execution, using Active Logic.

3 Practical Reasoning and Plan Execution

Plan execution architectures (for instance CIRCA [Goldman *et al.*, 1997], ESL [Gat, 1996], PRS [Myers, 1997], RAPS [Firby, 1995]) are generally not based primarily on logic. However, we think that active logics are well suited to serve as plan execution architectures: failures of plans or of actions can be handled naturally as contradictions; the changing state of the world can be represented as time-situated changing beliefs of the agent; the reasoner can use logic to perform arbitrary reasoning. Active logics can therefore provide a uniform platform for reasoning and plan execution.

Using active logics as a plan execution architecture requires one to define representations for plans, goals and actions, to add axioms to describe the plan execution process, and to augment the contradiction handler to take care of the special cases of contradictions caused by plan execution.

We have begun developing a plan execution architecture in active logic. In this section we sketch our preliminary system and present some examples that illustrate it. We are still at the early stages of development, so we do not take complex plans (beyond a sequence of sub-actions) or situations into account yet.

3.1 Representations

The notation we use is as follows: predicates and functions are capitalized, variables are not, Greek letters are used for expression variables; $Know$ is a positive introspection predicate; the formulas we present are assumed to be universally quantified unless otherwise noted. We allow quantification over formulas that may be seen as being implicitly quoted. Lists are represented prolog style with [ ], and we use | to denote concatenation of lists. Now is a unary predicate true of the current time step.

Plan recipes are represented as $Plan(name, pre, post, steps)$ where *name* is the name of the plan, *pre* is a formula describing the preconditions for the execution of the plan, *post* is a formula describing the formulas that hold at the successful execution of the plan, *steps* is a temporally ordered list of steps that constitute the plan. These steps can be either primitive actions that can be executed, or sub-goals, requiring a new plan to be adopted and executed. The plans we currently consider are simple plans with only sequencing of plans or actions allowed.

Actions are represented in a similar way as $Action(name, pre, post, act)$ where *name* is the name of the action, *pre* is a formula describing the preconditions for the execution of the action, *post* is a formula describing the formulas that hold at the successful execution of the action, *act* is the procedure that is to be executed to implement the action.

Exogenous actions are represented by $Action(name, pre, post, Nul)$. For instance, if the agent is in a train and it depends on the train getting to X, this is represented as $Action(GetTo, InTrain(Train1, Time1), At(X, Time2), Nul)$

Goals are represented as $Goal(\phi)$ where ϕ is a formula that is to be made to hold. $Goal(\phi)$ holds only when ϕ has not been accomplished and no plan has been adopted to achieve ϕ.This goal can be a maintenance goal if ϕ quantifies over time. For example, we would represent keeping the cat fed as $Goal(\forall t\ Fed(Cat, t))$.

Plans that have been adopted and are being executed are represented by $Adopt(name, done, rest, goal)$ where *name* is the name of the plan, *done* is a list of those steps executed, *rest* is the list of the remaining steps, and *goal* is the goal.

3.2 Plan execution axioms

We adopt a plan for execution if its postcondition(ψ) implies the goal(ϕ), the preconditions(θ), are met and the goal is not already true:

$$Goal(\phi) \wedge \theta \wedge Plan(n, \theta, \psi, s) \wedge (\psi \rightarrow \phi) \wedge \neg Know(\phi) \rightarrow \neg Goal(\phi) \wedge Adopt(n, [\,], s, \phi)$$

Note the assertion of $\neg Goal(\phi)$ here. This represents that ϕ is no longer a goal that needs to be processed– $Adopt(N, [\,], S, \phi)$ indicates that ϕ is being worked on. The assertion of $\neg Goal(\phi)$ will give rise to a contradiction. This will be resolved by preferring the later formula, in this case $\neg Goal(\phi)$ (see below for more on contradiction resolution). We also require that all adopted plans for the same goal be the same:

$$Adopt(n, i, f, \phi) \wedge Adopt(m, i', f', \phi) \rightarrow n = m$$

If there are two different adopted plans for the same goal, a contradiction will be generated since $\neg n = m$. At this point, one can choose which plan to pursue.

If the precondition of the plan is not known to hold, we make it a goal:

$$Goal(\phi) \wedge \neg Known(\phi) \wedge \neg Known(\theta) \wedge Plan(n, \theta, \psi, steps) \wedge (\psi \rightarrow \phi) \rightarrow Goal(\theta)$$

Here, ϕ is still a goal so that whenever the preconditions are made true, the main plan will be started.

We now consider executing the plan. If the next step of the plan is an action, we wait until the previous step of the plan is completed and verify that the preconditions of that step hold before executing the action. $Done(act)$ is asserted in the knowledge base once action act is completed by the procedure execution module. $Do(act)$ causes act to be performed by the agent. H returns the head of a list, $Last$ returns the last element of the list and T returns the tail.

$$(Now(t) \wedge Adopt(n, i, r, \theta) \wedge Done(Last(i)) \wedge H(r) = Action(a, \phi, \psi, act) \wedge \phi) \rightarrow (Do(act) \wedge Adopt(n, i | H(r), T(r)) \wedge Expect(\exists t_1\ t < t_1 \wedge \psi(t_1)) \wedge \neg Adopt(n, i, r, \theta))$$

We assert that we expect that the postconditions will hold sometime in the future. When the action is actually done, we will have confirmation of that by the postconditions being asserted in the knowledge base. If something happens that makes this impossible (for example, if the action fails), the agent will know that it has a problem.

If the next thing on the plan is a goal, we try to plan for it:

$$(Now(t) \wedge Adopt(n, i, r, \theta) \wedge Done(Last(i)) \wedge H(r) = Goal(\phi)) \rightarrow (Goal(\phi) \wedge Adopt(n, i | H(r), T(r), \theta) \wedge \neg Adopt(n, i, r, \theta) \wedge Expect(\exists t_1\ t < t_1 \wedge \phi(t_1)))$$

If the preconditions do not hold, we make them a goal.

$$(Adopt(n, i, r, \theta) \wedge Done(Last(i)) \wedge \neg Known(\phi) \wedge Hd(r) = Action(a, \phi, \psi, act)) \rightarrow Goal(\phi)$$

If there is nothing left in the plan, we stop.

$$Adopt(n, i, r, \theta) \wedge r = [\,] \rightarrow \neg Adopt(n, i, r, \theta)$$

This causes a direct contradiction that is resolved by retracting both contradictands.

In the case that we are at the very beginning of a plan, we know that the empty action is always done:

$$Done(Nul)$$

Goal and *Expect* have some properties of modalities, and the usual rules apply, including $Expect(\phi \wedge \psi) \leftrightarrow (Expect(\phi) \wedge Expect(\psi))$, and the Barcan formula (see for instance [Hughes and Creswell, 1996]), so that we get $Expect(\forall x\ \phi(x)) \leftrightarrow \forall x\ Expect(\phi(x))$. [1]

3.3 Contradictions

Some events in the execution of plans depend on the agent noticing contradictions and reacting appropriately. As mentioned above, contradictions in active logic are automatically flagged when both P and $\neg P$ are derived. For plan execution, we also flag contradictions for $Expect(\phi)$ and $\neg\phi$: if the agent expects something to become true and the negation of it is found to be true, there is something wrong with the plan.

The contradictions are processed by a set of axioms that constitute the contradiction handler. These axioms depend on domain information as well as meta-information such as the derivation of the contradictands, their source and the time at which they were first asserted. This information is not explicitly represented in the knowledge base as formulas in the current implementation of active logic, but is instead represented in data structures associated with the formulas. Access functions allow the axioms to reason with these.

Some of the strategies for resolving contradictions between ϕ and $\neg\phi$ are: 1. if ϕ is of the form $Goal(\psi)$, then we reinstate the later one; 2. if ϕ is of the form $Expect(\psi)$ and $\neg\phi$ is $\neg\psi$ and results from an observation, then reinstate the goal that led to the expectation and remove the expectation. The rationale behind these will be made clearer below.

3.4 The domain

The domain we use to illustrate this system is part of the Washington area metro system. We assume that our agent is at College Park (CP) and wants to get to Union Station (US). The only train line that passes through CP is the green line. Since part of the green line is still under construction, there is no direct train from CP to US: one has to take the green train from CP to Fort Totten (FT) and there change to a red train that goes from FT to US. However, during rush hour, the green train bypasses FT altogether and goes to US. Therefore we have two plans to get to US from CP: one for rush hour, and one for non rush hour.

The examples we present are first a simple case of the agent getting on the train at CP during rush

[1] We intend to explore the relation between our use of modality in these cases and the uses of modality for agency as in [Belnap and Perloff, 1988; Horty and Belnap, 1995], for example.

hour and getting off at US. The second example we consider is the case of the agent thinking it is rush hour (by default), getting on the train at CP and expecting the train to go up to US. However, it is not rush hour and the train gets to FT and stops there. The agent observes this and that leads to a contradiction. This causes the agent to abandon the original plan and to form a new plan to get from FT to US.

Plans

If the agent is at p at time t, $At(p,t)$, and there is a direct train m that goes from p to q, $DirectTrain(p,q,m)$, and that train is at p at time t, $TrainAt(m,p,t)$, then the following plan will result in the agent not being at p but at q at some later time.

$$\begin{aligned} Plan(P1 \quad & ,(At(p,t) \wedge DirectTrain(p,q,m) \wedge \\ & \quad TrainAt(m,p,t)), \\ & (\exists t_1 \; t_1 > t \wedge \neg At(p,t_1) \wedge At(q,t_1)), \\ & [\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3]) \end{aligned}$$

Here, $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ stand for actions $A1$, $A2$, and $A3$ that we present below. We use $\mathcal{A}_1$ and so on for the convenience of not having to write the actions here. These are not part of the language.

Another plan is for the case that there is no direct train between the source and the destination:

$$\begin{aligned} Plan(P2, \quad & (\exists m_1 m_2 \; At(x,t) \wedge DirectTrain(x,z,m_1) \\ & \wedge \neg DirectTrain(x,y,m_0) \wedge \\ & \quad DirectTrain(z,y,m_2)), \\ & (\exists t_1 \; t_1 > t \wedge \neg At(x,t_1) \wedge At(y,t_1)), \\ & [Goal(\exists t_2 \; t_2 > t \wedge At(z,t_2)), \\ & \quad Goal(\exists t_3 \; t_3 > t \wedge At(y,t_3))]) \end{aligned}$$

Primitive Actions

The primitive actions used in the plans are as follows.

If we are at the station at the same time as the train is, we can get on the train and we will no longer be considered to be at the station.

$$\begin{aligned} Action(A1, \quad & (At(x,t_0) \wedge TrainAt(m,x,t_0)), \\ & (InTrain(m,t_0+1) \wedge \neg At(x,t_0+1)), \\ & GetOnTrain(m,x,t_0)) \end{aligned}$$

A2 is an exogenous event: if the agent is in the train at station X and there is a direct connection to station Y, then, at some later time, the train and the agent will end up in Y.

$$\begin{aligned} Action(A2, \quad & (InTrain(m,t_0) \wedge TrainAt(m,x,t_0) \wedge \\ & \quad DirectTrain(x,y,m)), \\ & (\exists t_2, \; t_2 > t_0 \wedge TrainAt(m,y,t_2) \wedge \\ & \quad InTrain(m,t_2)), \\ & Nul) \end{aligned}$$

The third action is to get off the train: if we are in the train and it is at a station, we can get off the train and we will be at the station.

$$\begin{aligned} Action(A3, \quad & (InTrain(t_0) \wedge TrainAt(m,y,t_0)), \\ & (\neg InTrain(t_0+1) \wedge At(y,t_0+1), \\ & GetOffTrain(m,y,t_0)) \end{aligned}$$

Domain Information

During rush hour, there is a direct train from College Park to Union Station:

$$Now(t) \wedge RushHour(t) \rightarrow DirectTrain(CP,US,Green)$$

It is usually not rush hour:

$$Now(t) \wedge \neg Know(\neg RushHour(t)) \rightarrow \neg RushHour(t)$$

When it is not rush hour, there is a green train from CP to FT and a red train from FT to US:

$$Now(t) \wedge \neg RushHour(t) \rightarrow DirectTrain(CP,FT,Green)$$

$$Now(t) \wedge \neg RushHour(t) \rightarrow DirectTrain(FT,US,Red)$$

If the train reaches a terminal station, we have to get off:

$$InTrain(m,t) \wedge TrainTerminus(m,x,t) \rightarrow Do(GetOffTrain(m,x,t))$$

This is an instance of an action being done depending directly on the state of the agent and not being part of a plan. When getting off the train succeeds, the following: $At(x,t)$ and $\neg InTrain(m,t)$ are added to the knowledge base.

When a train reaches the terminus, it goes nowhere else:

$$TrainTerminus(m,x,t_0) \wedge \neg x = y \wedge t_1 > t_0 \rightarrow \neg TrainAt(m,y,t_1)$$

3.5 Example 1

We present axioms for the first example: The goal is to get to Union Station, the agent is at College Park at time 0, and it is rush-hour and the train is at College Park.

$$Goal(At(US,T)) \wedge Now(0) \wedge At(CP,0) \wedge RushHour(0) \wedge TrainAt(Green,CP,0)$$

We do not show the details of the plan execution, but highlight some of the aspects. At time 1, plan $P1$ is adopted and at the next step, the action $GetOnTrain(Green,CP,0)$ is executed. This succeeds and adds to the knowledge base the following: $Done(GetOnTrain(Green,CP,0),3) \wedge InTrain(Green,3), \neg At(CP,3)$. Since $Done(GetOnTrain(Green,CP,0),3)$ is a precondition for the next action, we can now execute it. However, the next action is a Nul action, so we can only wait until the preconditions for the action after are satisfied and we assert the expectations at this point: $Expect(\exists t_1 \; t_1 > 3 \wedge TrainAt(Green,US,t_1) \wedge InTrain(m,t_1))$. Later, say at time 10, the train does get to US, and these expectations are observed to be true. The agent then executes the last step of the plan, which is to get off the train, and that results in asserting $At(US,11)$ in the knowledge base.

3.6 Example 2

In this case too, the agent is at College Park and thinks it is rush hour and gets on the train just as before. Once it does get on the train, we expect $Expect(\exists t_1\ t_1 > 3 \wedge TrainAt(Green, US, t_1) \wedge InTrain(m, t_1))$. From this we can derive that $Expect(\exists t_1 TrainAt(Green, US, t_1))$.

However when the agent gets to action A2, the train reaches the terminus at FT. The agent observes $TrainTerminus(Green, FT, 10)$ (assume the time is 10). This leads to the agent getting off the train $Do(GetOffTrain(Green, FT, 10))$ which results in $At(FT, 10)$. The agent also concludes that this train is not getting anywhere: $\forall t, p\ \neg FT = p \rightarrow \neg TrainAt(Green, p, t)$. In particular, this train is not getting to Union Station: $\forall t\ \neg TrainAt(Green, US, t)$. This however contradicts the expectation that the green train will indeed get to Union Station: $Expect(\exists t_2\ TrainAt(Green, US, t_2))$.

The plan has failed and since the agent is executing the plan, it cannot back up to a preceding state– it has to try to accomplish its goal from its current state. A contradiction is generated and the handling of the contradiction results in reinstating the original goal $Goal(At(US, t))$ and the removal of the expectations. Now the agent is at FT and has the goal of getting to US and knows there is a red train that goes there directly, so it can get to Union Station using the same procedure as in the previous example.

4 Alma/Carne: An Active Logic Agent

Our concern is not just with "theoretical" practical reasoning, but with using this reasoning about rationality in a "practical" way, as a specification of an artificial agent. We have thus been constructing a test-bed system both for testing the ideas above and for attempting to apply the general approach for practical problems such as human-computer natural language dialogue. Alma/Carne is an implementation of active logic that includes a facility for representing and using procedural knowledge. This gives the active logic the ability to interact in arbitrary ways with the environment and to execute procedures the details of which are of no interest inferentially. Alma and Carne are separate processes with Alma the reasoner and Carne the action execution module. This gives us a clear separation between the procedural and the declarative parts of the model of the agent while requiring declarative knowledge about the procedures to be explicitly stated.

4.1 Alma

Alma implements active logic and is the repository for declarative knowledge in the agent. All inferences and all decisions to act are done in Alma, controlled by domain axioms and active logic rules of inference. Alma has a few features that enhance the efficiency of the logic including: 1. applying the inference rules to new formulas only; 2. allowing the programmer to specify in what sorts of proofs each formula is to be used (forward or backward or both); 3. allowing the programmer to specify policies that determine which inferences to actually do at each step.

The problem of controlling the logic is a crucial one, which will get worse as the agent is used in more realistic settings and these features are just the start of our attempt to address this problem.

Alma also has the capability to interact with Carne, in particular, using Carne to "execute" basic actions. We describe that following a description of Carne.

4.2 Carne

Carne contains the procedural knowledge of the reasoner. It allows the programmer to specify programs in Prolog that fall into the following main categories:

- Programs triggered by Alma to effect a change in the environment.
- Programs that are responsive to events in the environment and that automatically update Alma's knowledge base with observations.
- Programs that do computations on behalf of Alma.

These give Alma the ability to effectively interact with the world and to offload resource intensive computations to a separate process. A simple interface is used to link Alma and Carne.

4.3 The Alma/Carne interface

On the Alma side, there are special purpose rules of inference and predicates. These predicates can be used in axioms to initiate programs in Carne, and to reason about the status of the programs.

$call(\phi, Id)$ If a formula of the form $call(\phi, Id)$ is derived in Alma, an inference rule comes into play that sends a message to the Carne process for it to execute program ϕ (which, of course, has to be known to Carne). The rule also results in the assertion $doing(\phi, Id)$ in the knowledge base. The Id is a unique identifier used to distinguish between multiple invocations of the same program with the same arguments. An alma rule to perform an action of a plan would be to call a program whenever a $Do(Act)$ proposition of the appropriate type is inferred.

$doing(\phi, Id)$ This asserts that Carne is in the process of executing ϕ.

$done(\phi, Id)$ Once the program has completed successfully in Carne, a message is sent to Alma that results in the assertion of $done(\phi, Id)$ in the

knowledge base and the deletion of $doing(\phi, Id)$ (although that remains in the Alma history).

$error(\phi, Id)$ In case the program fails to execute in Carne, $error(\phi, Id)$ is added to and $doing(\phi, Id)$ is deleted from the Alma database.

These predicates track the status of the programs in Alma and enables decisions to be made about actions as described in the previous section.

On the Carne side, a Prolog predicate af (add formula) is provided to the Carne programs that allows them to assert formulas to the Alma knowledge base. This facility is independent of the above status predicates and is used to assert the results of computations and to include input from the environment, into Alma in the appropriate form. Similarly, df (delete formula) can be used by Carne programs to remove formulas from the Alma knowledge base.

5 Current and Future Work: A Conversationally Adequate Dialog Agent

Using the Alma/Carne implementation, we are designing and implementing a natural-language-dialogue and commonsense-reasoning engine that has a heavy emphasis on metareasoning [Traum and Andersen, 1999]. The hypothesis we wish to test is that metareasoning is essential to flexible discourse and cognition, in which (miscommunication and other) errors must be detected and repaired during the same episode of reasoning (see [Perlis *et al.*, 1998]). An agent capable of doing this will have to reason with and represent: (1) ongoing time; (2) history; (3) linguistic objects; (4) meanings; (5) contradictions.

The architecture we have designed involves traditional modules (e.g., speech-processor, parser, dialogue manager, problem solver, output/action manager), but organized in terms of logical and non-logical behaviors. Thus our logic engine, Alma, receives and sends communications from the rest of the system (via Carne – whose only job is to facilitate such internal messages, see Figure 1). As has been suggested often before (e.g., [Rieger, 1974]) we view dialogue as simply one special kind of problem-solving.

One major ongoing application of active logics is that of building a "conversationally adequate" dialogue agent. Conversationally adequate agents should be able to engage in "free-ranging" conversation: successfully exchanging information with another agent over the course of a conversation covering any arbitrary topic. Such an agent will have the ability to learn in McCarthy's sense of advice-taking, via conversation [McCarthy, 1958]. We hypothesize that the ability to use meta-reasoning (coupled with other crucial skills like learning) to correct errors is an ability that, once sufficiently sophisticated, allows

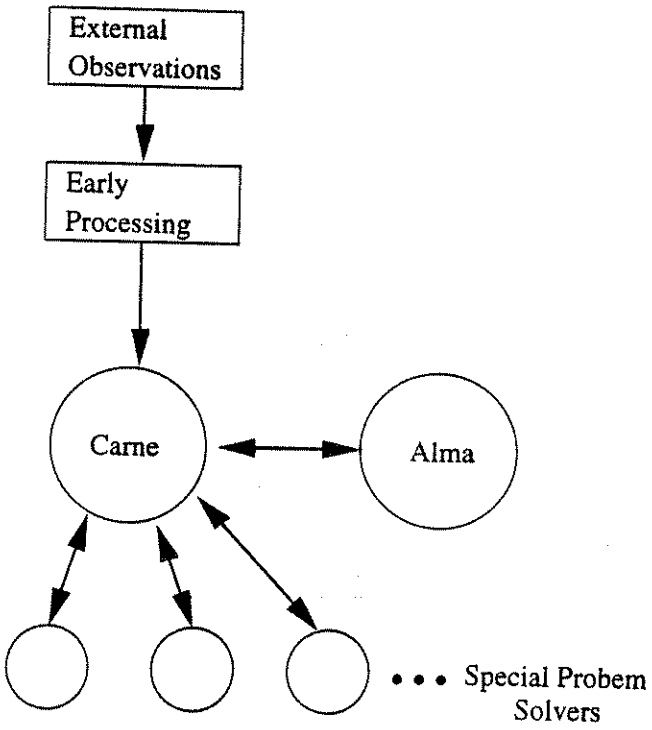


Figure 1: The conversational agent architecture

agents to engage in free-ranging conversation.

Preliminary work on applying active logics to problems in language processing has been done [Gurney *et al.*, 1997; Perlis *et al.*, 1996], and we have proposed an abstract view of how we would build such a conversationally adequate agent [Perlis *et al.*, 1998]. We view metareasoning to be a crucial part of that type of agent and believe that active logics are well suited for that. We are currently investigation use of the plan execution framework presented above in addressing dialog performance.

References

[Belnap and Perloff, 1988] N. D. Belnap and M. Perloff. Seeing to it that: a canonical form for agentives. *Theoria*, 54:175–199, 1988.

[Doyle, 1979] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12(3):231–272, 1979.

[Elgot-Drapkin and Perlis, 1990] J. Elgot-Drapkin and D. Perlis. Reasoning situated in time I: Basic concepts. *Journal of Experimental and Theoretical Artificial Intelligence*, 2(1):75–98, 1990.

[Firby, 1995] R. J. Firby. The RAP language manual. Animate Agent Project Working Note AAP-6, University of Chicago, 1995.

[Gat, 1996] E. Gat. ESL: A language for supporting robust plan execution in embedded autonomous agents. In *Procs. of the AAAI Fall Symposium on Plan Execution*. AAAI Press, 1996.

[Goldman *et al.*, 1997] R. P. Goldman, D. J. Musliner, M. S. Boddy, and K. D. Krebsbach. The circa model of planning and execution. In *Working Notes of the AAAI Workshop on Robots, Softbots, Immobots: Theories of Action, Planning and Control*, 1997.

[Gurney *et al.*, 1997] J. Gurney, D. Perlis, and K. Purang. Interpreting presuppositions using active logic: From contexts to utterances. *Computational Intelligence*, 1997.

[Horty and Belnap, 1995] J. F. Horty and N. D. Belnap. The deliberative stit: A study of action, omission, ability, and obligation. *Journal of Philosophical Logic*, 24(6):583–644, 1995.

[Hughes and Creswell, 1996] G. E. Hughes and M. J. Creswell. *A new introduction to modal logic.* Routledge, 1996.

[McCarthy, 1958] J. McCarthy. Programs with common sense. In *Proceedings of the Symposium on the Mechanization of Thought Processes*, Teddington, England, 1958. National Physical Laboratory.

[Miller, 1993] M. Miller. *A View of One's Past and Other Aspects of Reasoned Change in Belief.* PhD thesis, Department of Computer Science, University of Maryland, College Park, Maryland, 1993.

[Myers, 1997] K. L. Myers. *User Guide for the Procedural Reasoning System.* SRI International,, Menlo Park, CA, 1997.

[Nirkhe *et al.*, 1997] M. Nirkhe, S. Kraus, M. Miller, and D. Perlis. How to (plan to) meet a deadline between *now* and *then*. *Journal of logic computation*, 7(1):109–156, 1997.

[Perlis *et al.*, 1996] D. Perlis, J. Gurney, and K. Purang. Active logic applied to cancellation of Gricean implicature. In *Working notes, AAAI 96 Spring Symposium on Computational Implicature.* AAAI, 1996.

[Perlis *et al.*, 1998] D. Perlis, K. Purang, and C. Andersen. Conversational adequacy: Mistakes are the essense. *International Journal of Human Computer Studies*, pages 553–575, 1998.

[Rieger, 1974] C. Rieger. *Conceptual Memory: A Theory and Computer Program for Processing the Meaning Content of Natural-Language Utterances.* PhD thesis, Department of Computer Science, Stanford University, Palo Alto, California, 1974.

[Suchman, 1987] Lucy A. Suchman. *Plans and Situated Actions.* Cambridge University Press, 1987.

[Traum and Andersen, 1999] D. Traum and C. Andersen. Representations of dialogue state for domain and task independent meta-dialogue. In *Proceedings of the IJCAI99 workshop: Knowledge And Reasoning in Practical Dialogue Systems*, pages 113–120, 1999.

Intuitive Reasoning with Pseudo-Intuitionistic Semantics

Cedric Thienot
LAFORIA, Department of Computer Science, University P. & M. Curie, Paris 75252, France
email : cedric.thienot@lip6.fr
telephone number : 33 (0)1 44 27 89 90
fax number : 33 (0)1 44 27 88 12

1.Summary

Different solutions are provided to represent uncertainty. This article explores uncertainty in a quite unusual way. It develops a theory combining Belief Functions [9] and pseudo-intuitionistic logic [10]. Until today, the use of Belief functions in Kripke-like semantics has not been exploited. Moreover, pseudo-intuitionistic logic, by weakening the deduction mechanism, offers an ideal framework to introduce uncertain reasoning.
The main feature of pseudo-intuitionistic logic is to differentiate known facts from deduced facts or in other terms axioms from conjectural theorems. This logic expresses the idea that a deduced fact is always less true than a confirmed fact.
The evaluation of the belief function will make deduced facts be confirmed. Truth will not come from syntactic operations but from semantic evaluations. Therefore, syntax leads to correctness and semantics to truth.

This article will present succinctly the pseudo-intuitionistic logic and the main result exploiting the application of belief functions to Kripke-like semantics. This article is mainly axed on an example which aims to introduce new possibilities for knowledge representation.

2. Short Presentation of Pseudo-Intuitionistic Logic

One of the interest of intuitionistic logic is that it could be interpreted in terms of proof instead of Boolean truth values. Intuitively, φ is true iff we have a proof of φ. The most immediate consequence is that the law of excluded middle doesn't hold anymore. The pseudo-intuitionistic logic is then an extension of the intuitionistic logic weakening the modus ponens or deduction :

$$\frac{\varphi \qquad \varphi \rightarrow \psi}{\mathsf{T} \rightarrow \psi}$$, where T denotes true.

Actually, in terms of proof, modus ponens means that "there exists a proof of ψ which can be derived from a proof of φ and a proof of $\varphi \rightarrow \psi$". In this framework, the existence of a proof $(\mathsf{T} \rightarrow \psi)$ is not synonymous with obtaining it (ψ). Therefore, as the syntactic deduction doesn't give truth but validates a reasoning. We will extend the semantic valuation.

2.1 *Definition of Kripke-Like Semantics of Pseudo-Intuitionistic Logic*

Let W be a set of worlds, and R be a transitive, left-surjective and binary relation on W, which is called accessibility relation. We note R^* the closure by reflexivity of R.
A *model* based on W is a structure $M = (W, R, i)$ where i is a W-valuation :

$$i : \varphi \; \alpha \; i(\varphi)$$

We defines the sentence: "In the model M, φ is true at state w", denoted by $M \models \varphi$, by :

$M \models w\ \pi$ iff $w \in i(\pi)$, π atomic
$M \models w\ \varphi \wedge \psi$ iff $M \models w\ \varphi$ and $M \models w\ \psi$
$M \models w\ \varphi \vee \psi$ iff $M \models w\ \varphi$ or $M \models w\ \psi$
$M \models w\ \neg\varphi$ iff, for all v such as wRv, not $M \models w\ \varphi$
$M \models w\ \varphi \rightarrow \psi$ iff, for all v such as $wR\ v$, if $M \models \varphi$ then, for all z such that vRz, $M \models z\ \psi$.

It is said that φ is true in M iff $M \models w\ \varphi$ for all $w \in M$ and that φ is valid on W iff φ is true for any model based on W.

We call *hereditary set* of w, noted w , the set of all the worlds v of W such as $wR=v$:

$w+ = \{v : v \in W \text{ and } wR+v\}$.

We note $W+$ the set of all the hereditary sets of $w \in W$.

3. Belief Function on Kripke-Like Semantics.

Before introducing a variation of belief functions, we have to define a set K. Actually, belief functions instead of being applications from 2W to [0,1], will be application from W+ to K.

Definition of world level Consider a model (W, R, i). Let n be the maximal path length of the graph associated to W and let n be, for each $w \in W$, the maximal length of a path between w and an initial world. We call *initial-level* of w, the integer n-nw.

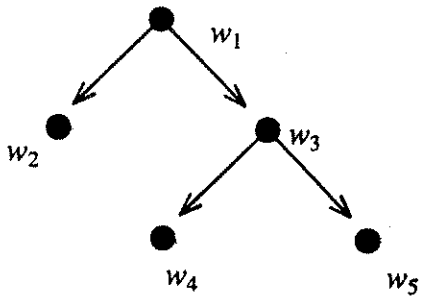
Definition of cost of world w The cost of w is an application m which associates the heredity set of w with an n-uplet $m(w\ ) = (0, \ldots, 0, c(w), 0, \ldots, 0)$ where the rank of $c(w)$ in the n-uplet equals the initial-level of w.

Definition 3: Let K be the closure by addition of the set of the $m(w+)$'s to which we add the null n-uplet, $K\ = (0,\ldots,0)$:

$$K = K0 \cup \{ \sum_{w_i} m(wj^{+}) : w \in W, J \subset \mathbb{N} \}.$$

We note $K1 = \sum_{w_i} m(w^{+})$, $K\ = (k1, \ldots, kn)$ the maximal element of K.

Consider the following example :



We note ci the cost of a world wi.

We can deduce that :

$m(w1\ ) = (0, 0, c\ )$
$m(w2\ ) = (0, c\ , 0)$
$m(w3\ ) = (0, c\ , 0)$
$m(w4\ ) = (c\ , 0, 0)$
$m(w5\ ) = (c\ , 0, 0)$.

Then, $K1 = (c5+c4, c3+c2, c1)$, $K0 = (0, 0, 0)$.

It is possible to define the belief function in W .

Definition 4 Belief Function: We call *belief function* of $A \in W+$ the application Bel : $W+ \rightarrow K$ defined by :

$$\mathrm{Bel}(A) = \sum_{B \subset A} m(B)$$

As $(W+, \cup, \cap, \neg, \Rightarrow, \varnothing, W)$ has a PI-algebra structure, for all A, B of $W+$, Bel($\neg A$), Bel($A \cup B$), Bel($A \cap B$), Bel($A \Rightarrow B$) are well-defined.

As it is done in the theory of evidence, we can define the doubt function and the higher probability function :

doubt function : Dou(A) = Bel($\neg A$)

higher probability function : P*(A) = K − Dou(A)

degree of non-contradiction : P**(A) = Bel($\neg\neg A$).

It is possible to define an order relation on K.

Definition : Let $g = (g1, \ldots, gn) \in K$ and $h = (h1, \ldots, hn) \in K$. We say that $g > k$ iff for the smallest i such as, for all $j > i$, $gj = hj$, we have $gi > hi$. gi and hi will be called meaningful terms of the comparison.

We have the following properties :

Bel($\varnothing$) = $K0$,

Bel(W) = $K1$,

if $A \subset B$ then Bel(A) < Bel(B),

Bel($A \cap \neg A$) = $K0$,

Bel(A) + Bel($\neg A$) = Bel($A \cup \neg A$) ≤ Bel(W) = $K1$,

If Bel($A \Rightarrow B$) = $K1$ then Bel(A) ≤ Bel($W \Rightarrow B$),

Bel($A \Rightarrow \neg\neg A$) = $K1$,

Bel($\neg\neg A$) ≥ Bel(A),

$$\forall n \quad \forall A_1, \Lambda, A_n \in W^+ \quad \mathrm{Bel}\left(\bigvee_j A_j\right) = \sum_{\substack{I \subset \{1, \Lambda, n\} \\ I \neq \varnothing}} (-1)^{|I|+1} \mathrm{Bel}\left(\bigcap_{i \in I} A_i\right),$$

$$P^*(A) = \sum_{B \cap A \neq \varnothing} m(B),$$

$$P^{**}(A) = \sum_{B \cap \neg A = \varnothing} m(B).$$

In the previous example, we can establish that :

	Bel	Dou	P**
$w1$	(c +$c4$, $c3$+$c2$, $c1$)	(0, 0, 0)	($c5$+$c4$+$c2$, $c3$+$c2$, $c1$)
$w2$	(0, c , 0)	($c5$+$c4$, $c3$, 0)	(0, $c2$, 0)
$w3$	(c +$c4$, $c3$, 0)	(0, $c2$, 0)	($c5$+$c4$, $c3$, 0)
$w4$	(c , 0, 0)	($c5$, $c2$, 0)	($c4$, 0, 0)
$w5$	(c , 0, 0)	($c4$, $c2$, 0)	($c5$, 0, 0)
$w4 \cup w$	(c +$c4$, 0, 0)	(0, $c2$, 0)	($c5$+$c4$, $c3$, 0)

For instance, we verify that Bel($w1$) > Bel(w) (c > 0). But, we can not compare Bel($w2$) and Bel(w) as c and $c3$ are incomparable. Suppose that $c2 < c3$ (i.e. it is 'easier' to demonstrate $w3$ than $w2$), consequently, Bel($w2$) < Bel(w). Although Bel(w) > Bel(w $\cup w$).

The notion of conditional belief function can also be defined. It won't be defined using classical methods. But instead, we will recalculate the value. We will see in the example that this evaluation allows a non-monotony reasoning.

Definition : We call *conditional belief function* of A given B, the function Bel(A/B) = Bel (A) where Bel$B$ is the restriction of Bel to B.

In the previous example, if B = $w3$, then we verify :

Bel $(w3\ ) = (c\ +c4, c3)$ $\quad$ Bel$B(w4\ ) = (c\ , 0)$
Bel$B(w5\ ) = (c\ , 0)$ $\quad$ Bel$B\ (w5\ \cup\ w\ ) = (c\ +c4, 0)$.
Bel$B(w2\ )$ is not defined as $[w\ ) \cap [w3) = \varnothing$.

These notions can be extended to the set of formulas. Actually in a model $M = (W, R, i)$, we associate with a formula φ the belief function of $(i(\varphi))$.
More precisely, we define Bel which associates to each well-formed formula its W-valuation :

$$\text{Bel}+(\varphi) = \text{Bel}(i(\varphi)).$$

Using the algebraic properties of $(W+, \cup, \cap, \neg, \Rightarrow)$, we can deduce that :

$$\text{Bel}+(\neg\ \varphi) = \text{Dou}(i(\varphi)),$$
$$\text{Bel}+(\varphi \vee \psi) = \text{Bel}(i(\varphi) \cup\ i(\psi)),$$
$$\text{Bel}+(\varphi \wedge \psi) = \text{Bel}(i(\varphi) \cap\ i(\psi)),$$
$$\text{Bel}+(\varphi \rightarrow \psi) = \text{Bel}(i(\varphi) \Rightarrow i(\psi)).$$

Then, we introduce a new terminology of truth values for a well-formed formula φ :

" truth value " of φ	
$\text{Bel}+(\varphi) = K$	true
$\text{P}^{**}(i(\varphi)) = K1$	non-contradictory
$\text{Bel}\ (\varphi) > \text{Dou}(i(\varphi))$	locally (possibly) true
$\text{P}^{**}(\ i(\varphi)) > \text{Dou}(i(\varphi))$	locally probable
$\text{Dou}(i(\varphi)) > \text{P}^{**}(\ i(\varphi))$	locally improbable
$\text{Dou}(i(\varphi)) > \text{Bel}+(\varphi)$	locally (possibly) false
$\text{Dou}(i(\varphi)) = K$	false

Remark : In the example, if $A = w4\ \cup w$ then $\neg\ A = w$ and $\neg\neg\ A = w$. Therefore, we verify that $\text{P}^{**}(A) > \text{Dou}(A) > \text{Bel}(A)$. A proposition can be, in the same time, locally possibly false and locally probable. (This system is then incomparable with multi-valued systems (Belnap 77)).

4. Example : murderers

We will illustrate our theory with an example from the Theory of Evidence of Dempster and Shafer. A murder has been committed, three men are suspected : Pierre, Marie and Jacques is guilty. They are, respectively, tall, small and small. Moreover, it is known that the murderer acted alone. The police has three testimonies :

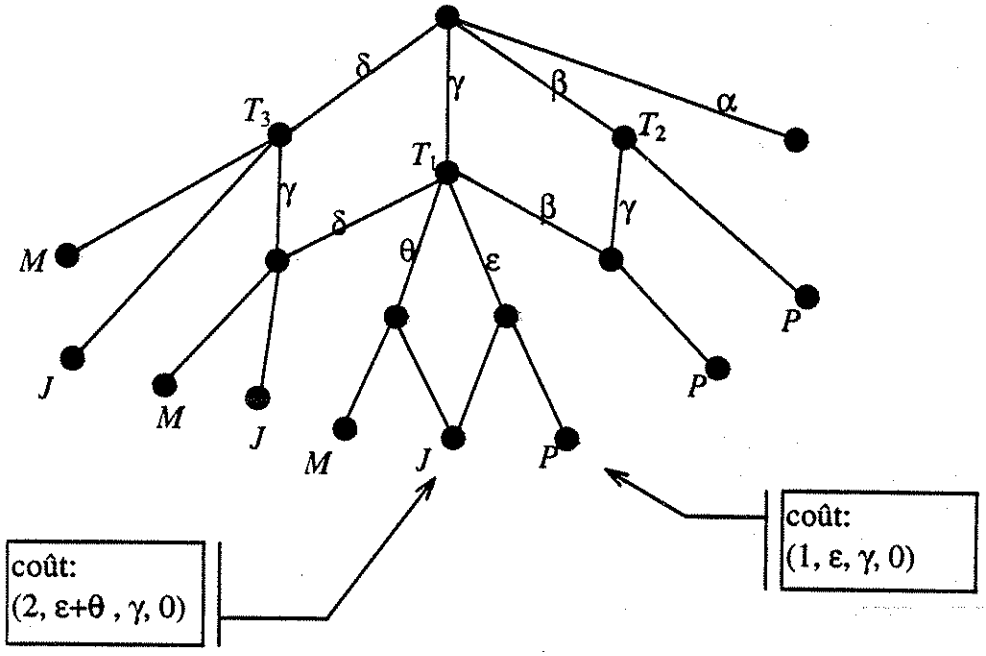
testimony T : a man has seen a small man from afar,
testimony $T2$: an old woman with glasses has seen a tall person,
testimony $T3$: Pierre's wife maintains that Pierre had been at home.

1. *Construction of the structure*

We define the different propositions : $T1$, $T2$, $T3$ (testimonies) and P, J and M (respectively, Pierre, Jacques and Marie is guilty).
Following the testimonies, $T1$ implies that if it is a man then Pierre or Jacques is guilty and if the murderer is small then Jacques or Marie. $T2$ accuses Pierre and $T3$, by clearing Pierre, accuses Jacques or Marie. We assign the following costs :

α	all the testimonies are false,
β	testimony $T1$ is true,
γ	testimony $T2$ is true,
δ	testimony $T3$ is true,
ε	in $T1$, the guilty party is a man,
θ	in $T1$, the guilty party is small.

We associate costs to arcs leading to the worlds forcing the truth of propositions. The other arcs have a cost equal to 1. We calculate the cost of a world by adding the cost of all the arcs leading to that world.



• ***syntactic properties***

From this representation, we can deduce several facts :

$T2 \rightarrow \neg\neg P$ From $T2$, we can conclude that Pierre can be guilty.
$\neg(\neg\neg P \rightarrow T2)$ The guiltiness of Pierre doesn't lead to the truth of $T2$.
$T1 \rightarrow \neg\neg(P \vee J \vee M)$ $T1$ doesn't clear anybody.
$T3 \rightarrow \neg\neg(J \vee M)$ From $T3$, we can conclude that Jacques or Marie can be guilty.
$T1 \wedge T2 \rightarrow P \wedge \neg J \wedge \neg M$ $T1$ et $T2$ imply that Pierre is guilty.
$\neg(T2 \wedge T3)$ $T2$ et $T3$ are incompatible.

• ***local truths***

As none of atomic formulas are true, we are going to estimate the value of their belief functions.

			meaningful terms			
	Bel	Dou	Bel	Dou		
$T1$	$(6, \theta+\varepsilon+\beta+2\gamma, \gamma, 0)$		$(3, 0, \alpha, 0)$		γ	α
$T2$	$(2, \beta+\gamma, \beta, 0)$		$(7, \gamma+\delta+\theta+\varepsilon, \alpha+\delta, 0)$		β	$\alpha+\delta$
$T3$	$(4, \gamma+\delta, \delta, 0)$		$(5, \theta+\varepsilon+\beta+\gamma, \beta+\alpha, 0)$		δ	$\beta+\alpha$

•firstly, suppose that the testimony $T1$ is good then Bel($T1$) > Dou($T1$). Then $\gamma < \alpha$.
•Moreover, if we admit that from a long distance, it is easier to recognize a man than his stature, then $\varepsilon > \theta$.
•Secondly, we suppose that the testimony of the old woman can be either true or false : $\beta = \alpha+\delta$.
•At last, Pierre's wife may lie : $\delta < \beta+\alpha$.

Therefore, we can estimate the truth values of P, J and M :

				meaningful terms					
	Bel	Dou	P^{**}	Bel	Dou	P**			
p	(3, 0, 0, 0)		(6, $\gamma+\delta+\theta$, $\delta+\alpha$, 0)			(3, $\beta+\gamma$, 0, 0)	0	$\gamma+\delta+\theta$	$\beta+\gamma$
J	(3, 0, 0, 0)		(6, $\beta+\gamma$, $\beta+\alpha$, 0)	(3, 0, 0, 0)		0	$\beta+\gamma$	0	
M	(3, 0, 0, 0)		(6, $\varepsilon+\beta+\gamma$, $\beta+\alpha$, 0)			(3, 0, 0, 0)	0	$\varepsilon+\beta+\gamma$	0

with Bel (W) = $K1$ = (9, $2\gamma+\delta+\theta+\varepsilon+\beta$, $\delta+\gamma+\beta+\alpha$,0)

Simple comparison

Concerning Pierre :

We have $\gamma+\delta+\theta > 0$, but $\gamma+\delta+\theta$ is not comparable with $\beta+\gamma$. We do not know if P is locally true or false.

concerning Jacques :

$\beta+\gamma > 0$ then J is locally false.

Identically, M is locally false.

At this state of knowledge, we can conclude that Jacques and Marie may not be guilty and although Pierre seems to be guilty, it is impossible to prove his guiltiness.

Crossed comparison

We can compare the different values of the doubt function for the suspects using meaningful terms :

Jacques and Marie :

Their meaningful terms are both equal to $\beta+\alpha$. Consequently, we will compare following terms, i.e. $\beta+\gamma$ for Jacques and $\varepsilon+\beta+\gamma$ for Marie. Then we verify that Dou(J) < Dou(M). Then, it is easier to prove that Marie is innocent than to prove that Jacques is innocent.

Pierre and Marie :

We compare $\delta+\alpha$ (Pierre) and $\beta+\alpha$ (Marie).

As $\beta = \delta+\alpha$, Dou(M) > Dou(P).

Pierre et Jacques :

We compare $\delta+\alpha$ (Pierre) and $\beta+\alpha$ (Jacques).

As $\beta = \delta+\alpha$, Dou(J) > Dou(P).

Consequently, it is easier to prove that Marie or Jacques is innocent than that Pierre is innocent.

conditional comparison

If we suppose that $T1$ is true, we have the following values :

	Bel$T1$	Dou$T1$	P**$T1$
P	(2, 0)	(4, $\delta+\theta$)	(2, β)
J	(2, 0)	(4, β)	(2, 0)
M	(2, 0)	(4, $\varepsilon+\beta$)	(2, 0)

The conclusion of simple comparisons and crossed comparisons will not change except for the comparison of the doubt function of P and J. Actually, it is then impossible to compare β and $\delta+\theta$. Consequently, we are not able to conclude that it is easier to prove that Jacques is innocent than that Pierre is innocent.

If we suppose $T1$ and $T2$ true, we have :

	Bel$T1$&$T2$		Dou $T1$&$T2$	P** $T1$&$T2$
P	(1, 0)	(0, 0)	(1, 1)	
J	(0, 0)	(1, 1)	(0, 0)	
M	(0, 0)	(1, 1)	(0, 0)	

Now, P is locally true and J and M are false. These results confirm a theorem of the model :

$$T1 \wedge T2 \rightarrow P \wedge \neg J \wedge \neg M.$$

To conclude, we remark that initially Pierre is the ideal guilty guy. But, if the testimony $T1$ is true, then although Pierre seems to be guilty, it is not obvious that Jacques is guiltless. Finally, if we suppose $T2$ true the guiltiness of Pierre is definitively confirmed. The conclusions evolved following hypothesis.
Moreover, the symbolic formalization of costs allows a kind of indecision. Actually, if we use numeric values in the case in which $T1$ is true, it would be possible to conclude that Pierre is guilty (proving that Jacques is guiltless would be comparable with proving that Pierre is guiltless).

5. Conclusion

This article presents an application of Belief functions to the Kripke-Like Semantics of pseudo-intuitionistic logic. This theory allows a good expressiveness of knowledge representation by combining symbolic value with logical formulas. We have presented here one example but some more complicated examples exist using local inconsistency (default values and absurd world [8]).

References

[1] Behrostaghi M.A, "Aspect of Basic Logic", Ph.d*D. Thesis University of Milwaukee, Wisconsin, 1995.
[2] Brouwer, L.E.J. " De ondetrouwbaarheid der logishe principes" Tidjdshrift voor wijsbegeerte, 1908.
[3] De Glas M. "Pensée logico-mathématique et intelligence artificielle" in "Pensée Logico-Mathématique" PUF
[4] De Glas M. & Thienot C. & Jacquet J.P. "implication doesn't imply inclusion" Logic Colloquium, San Sebastian, July 1996.
[5] Epstein R.L. " *the semantics foundation if logic" Vol.1 Propositional Logig, Kluwer Academic Publishers, 1990.
[6] Heyting A. "Intuitionism, An introduction" North Holland, 1971.
[7] Belnap N.D. "A useful four-valued logic", in: G. Epstein, J.M. Dunn (Eds)., Modern uses of Multiple-Valued Logic, Reidel Publishing Company, Boston, 1977, pp.7-73.
[8] Restall, G. "Subintuitionistic Logics", Notre Dame of Formal Logic, Vol 35, number 1 ,winter 1994.
[9] shafer, G. & Dempster " A mathematical theory of evidence" Princepton University Press, 1976.
[10] Thienot C."A logic without modus ponens", Logic Colloquium, Leeds, July 1997.

Progress Towards a Formal Theory of Practical Reasoning: Problems and Prospects

Richmond H. Thomason
AI Laboratory
130 ATL Building
University of Michigan
Ann Arbor, MI 48109–2210

rich@thomason.org

Abstract

From its beginnings in Aristotle, logic was intended to account not only for reasoning that is *theoretical* (or conclusion-oriented), but for reasoning that is *practical* (or action-oriented). However, despite an interest in the topic that continues to the present, the practical side of reasoning has remained broadly speculative. At least in some domains (mathematics, in particular), there are well developed proof-theoretic and semantic theories that yield quite detailed models of correct reasoning, and these models are useful for both theoretical and practical purposes. In contrast, the logical work on practical reasoning has remained broadly speculative and disengaged from applications. Logical formalisms have not been forthcoming that would be useful either in designing an agent that needs to act intelligently, or in helping an intelligent agent to evaluate its reasoning about action.

The decision-theoretic paradigm that has dominated economic thinking, on the other hand, certainly has produced applicable models of correct decision making. And, though decision theory and logic are certainly different subjects, it is easy to find areas of overlap in the concepts and techniques, as well as people who have made fundamental contributions to both fields.

Despite these similarities, I think it would be wrong to think of decision theory as the realization, within a different academic discipline, of a logical theory of practical reasoning. The reason is that *correct inference* is central to the logical approach to a subject matter, and correct inference is largely neglected in the decision theoretic paradigm.

The absence of a logical theory of practical reasoning is largely due to the unavailability of appropriate inference procedures. To handle even the simplest cases of practical reasoning, it is essential to have a reasoning mechanism that allows for practical conclusions that are nonmonotonic in the agent's beliefs. If an agent believes that he is out of milk, he may well conclude to walk to the store. If he then adds the belief that the store is closed, he will then have to withdraw his conclusion. And, until recently, probability functions have provided the only way to formalize inference procedures with these characteristics.

An approach to practical reasoning based on probability relies on numerical calculation rather than qualitative inference, so it needs quantities, not only for probabilities, but for utilities. Leonard Savage called the problem of constructing a quantitative model the *small worlds problem*. A good solution to the small worlds problem is great when you can get one. But you can't always do that. Trying to deal with decision problems in the absence of a qualitative model raises a number of difficult questions.

1. How to represent the reasoning process (rather than just the outcome).
2. How to make use of large amounts of knowledge, in open-ended decision situations. (In practice, the decision theoretic models are limited to outcomes that depend on no more than a dozen or so variables.)
3. How to make use of reasonable assumptions that are known in some sense, but cannot readily be assigned a probability in a many contexts.
4. How to construct a decision-theoretic microworld.
5. How to learn an agent's preferences from readily available information.
6. How to deal with conflicting goals.
7. How to model cases in which the agent is to some extent distributed, without complete agreement or communication among sub-agents.
8. What to do about problems of real-time, resource limited reasoning.

To deal with these problems, we need an alternative theory of a decision-making agent with the following characteristics.

Regarding belief the theory should:

1. Relax the quantitative commitments of decision theory.
2. Provide for belief kinematics, in allowing an update function to be defined.
3. Be engineerable. In particular, the information needed to support and update beliefs should be acquirable in some practicable way.

Regarding desire, the theory should:

1. Retain the idea that desires are immediate, with a source that is external to practical reasoning (below, I will call these immediate desires *wishes*), and that there are reasoned desires that depend on wishes and beliefs (below, I will call these *wants*). It is assumed that wants are like intentions, which are more or less connected to actions.
2. Treat practical reasoning as a process that creates considered desires by transforming wishes into wants.
3. Allow for the creation, cancellation, and reprioritization of wants in light of changing beliefs.
4. Treat the outcome of practical reasoning as nonunique. Agents with the same beliefs and wishes could reach different conclusions, even while conforming to the full principle of rationality.

Developing such a theory makes for a large-scale challenge. However, new ideas from many disciplines (and especially from Artificial Intelligence) provide a real opportunity for meeting this challenge.

In my talk, I will try to provide a sketch of these opportunities.

Risk Parameters for Utilitarian Desires (extended abstract)

Leendert van der Torre
Dept of AI, Vrije Universiteit
de Boelelaan 1081a, 1081 HV Amsterdam
The Netherlands
torre@cs.vu.nl

Emil Weydert
Max Planck Institute for Computer Science
Im Stadtwald, D-66123 Saarbrücken
Germany
weydert@mpi-sb.mpg.de

Abstract

In qualitative decision-theoretic planning desires – qualitative abstractions of utility functions – are combined with defaults – qualitative abstractions of probability distributions – to calculate the expected utilities of actions. In this paper we consider Lang's framework of qualitative decision theory, in which utility functions are constructed from desires. Unfortunately there is no consensus about the desired logical properties of desires, in contrast to the case for defaults. To do justice to the wide variety of desires we define parameterized desires in an extension of Lang's framework. There are three parameters. The strength parameter encodes the importance of the desire, the lifting parameter encodes how to determine the utility of a set from the utilities of its elements, and the polarity parameter encodes the relation between gain of utility for rewards and loss of utility for violations. The parameters influence how desires interact, and they thus increase the control on the construction process of utility functions from desires.

1 Introduction

Classical decision theory [Luce and Raiffa, 1957; Jeffrey, 1983; Keeney and Raiffa, 1976] has been developed to describe and prescribe rational human decision making. However, due to so-called 'human irrationality', the description task is complicated so that its use may be restricted to decision making by artificial agents. For example, in decision-theoretic planning a robot receives our requirements or imperatives, tries to figure out the set of admissible utility functions and probability distributions, calculates the expected utilities and acts accordingly. However, a new problem arises for this application domain of decision theory. In planning it is assumed that we cannot completely impose our preferences and beliefs, because either we do not know them or it is computationally too expensive to elicitate and communicate them. These requirements are therefore as well *heuristic approximations* [Doyle and Wellman, 1991] as ways to *compactly* communicate our preferences and beliefs [Haddawy and Hanks, 1992] that only refer to *qualitative abstractions* of utility functions and probability distributions (the latter are sometimes called plausibilities). In qualitative decision theory these qualitative counterparts of preferences and beliefs are called desires and defaults. We summarize the terminology used in this paper in Table 1 below.

utilities		probabilities	
quantitative	qualitative	quantitative	qualitative
preference	desire	belief	default

Table 1: Requirements in decision-theoretic planning

In this paper we propose a logic of utilitarian desires that builds on previous work of Boutilier [1994] and Lang [1996]. This logic is concerned with two problematic issues.

- First, as observed and discussed by Lang, the logic should not only characterize deductive relations between the desires – the logic of norms, imperatives and obligations called deontic logic for example also does so – but it should also determine the decision making process of the agent. As a consequence, Lang is more interested in the admissible utility functions than in the derivable desires. In other words, the semantics is more important than the syntactic or proof-theoretic counterpart.
- Secondly, not discussed or dealt with by either Boutilier or Lang, there are multiple intuitions about the logical properties of preferences and desires [Mullen, 1979; Pearl, 1993; Bacchus and Grove, 1996]. In other words, which desires can be derived intuitively sometimes depends on the meaning of the propositions. This multitude of intuitions hinders the effective use of desire specifications in a qualitative decision theory.

We give the robot's owner a tool to guide the robot's construction process of the intended utility functions by introducing several parameters.

The strength parameter encodes the importance of the desire,

The lifting parameter determines how to construct the utility of a set from the utilities of its elements,

The polarity parameter encodes the proportion between gain of utility for rewards and loss of utility for violation.

Decision theory explains the different intuitions about utilitarian desires and justifies our parameters. Rational agents base their decisions on the expected utility of their actions, i.e. they multiply the utility of the outcomes of possible actions by their probability and then choose the action that maximizes this expected utility. The intuitions differ due to the fact that utilities encode values as well as the agent's attitude towards risk, whereas probabilities only encode frequencies. They act *as if* they have an utility function, but they are not assumed to be aware of their compact values+risk representation. In classical decision theory, this unawareness is reflected by the contrived status of utility functions. To get some feeling for the different status of probabilities and utilities, consider the following two heuristics for requirements based on expected utilities. The first heuristic only considers the most likely states in the expected utility calculations, and the second heuristic only considers the most preferred states. The two heuristics are in an obvious way symmetric, but they have completely different consequences. The first heuristic cannot explain that people insure themselves for unlikely but grave events, see e.g. [Tan and Pearl, 1994a], and the second heuristic has the disadvantage that if the most preferred states are very unlikely, such as winning a lottery, then the requirement does not have an impact on the expected utilities and therefore not on the decisions.

With the parameters the risk component of each desire can be fit to the preference it encodes – we therefore call them risk parameters. The risk parameterization we propose for desires is not appropriate for defaults, though Boutilier's and Lang's logics are analogous to formalisms proposed for defaults, as we show in detail for Lang's framework and Weydert's framework for defaults. (They have as such been criticized by for example [Tan and Pearl, 1994b; Bacchus and Grove, 1996]). Our extension of the logic of utilitarian desires thus highlights a distinction between utilitarian desires and probabilistic defaults not found in the original proposals; we call it bipolarity.

2 The logic: explicit strengths

In this section we introduce the first parameter, that represents the strength s of the desire. Weydert has introduced explicit strength parameters in $\models_{\geq 1}$ or in $\models_{>0}$ satisfaction, based on the following truth conditions for parameterized conditionals, where u is a real-valued function on worlds.

$u \models D_{\geq s}(a|b)$ if $\max_{w \models a \wedge b} u(w) \geq s + \max_{w \models \neg a \wedge b} u(w)$
$u \models D_{>s}(a|b)$ if $\max_{w \models a \wedge b} u(w) > s + \max_{w \models \neg a \wedge b} u(w)$

There are no intuitive arguments supporting either one or the other because the two constraints are nearly identical. We have $u \models D_{>s}(a \mid b)$ if there is some small number ϵ such that $u \models D_{\geq s+\epsilon}(a \mid b)$. From the perspective of intuition, it is an arbitrary choice. However, there are technical distinctions. First, as we already remarked, $\models_{>0}$ determines a rational inference relation whereas $\models_{\geq 1}$ does not. Moreover, several constructions Weydert has investigated are easier defined in an extension of $\models_{\geq 1}$ satisfaction than in an extension of $\models_{>0}$ satisfaction. We therefore choose the former, abbreviating $D_{\geq s}(a|b)$ by $D_s(a|b)$. The results of this paper carry over to the other case.

2.1 The logic: the lifting problem

Consider the nonempty set of worlds that satisfy the proposition p and an utility function u that assigns utility to each of these worlds. What can we say about the utility of the set of worlds, i.e. the utility of p? This has been called the lifting problem (see e.g. [Thomason and Horty, 1996]), because the problem is how to lift a property of worlds to a property of sets of worlds.

Without knowing the probability of the individual worlds, the obvious choice is to consider the maximal or minimal utility of its elements. Let us call these operators $u_M(p)$ and $u_m(p)$, or $Mu(p)$ and $mu(p)$. $Mu(p)$ and $mu(p)$ are the poles of the set of utility values of the p worlds, in the sense that for each world w that satisfies p we have that $Mu(p) \geq u(w) \geq mu(p)$. If we know that we are in a p state, then assuming $Mu(p)$ is optimistic (the best case arises) and assuming $mu(p)$ is pessimistic (the worst case arises).

$$\begin{array}{lcl} Mu(p) & = & \max_{w \models p} u(w) \\ mu(p) & = & \min_{w \models p} u(w) \end{array}$$

$Mu(p)$ and $mu(p)$ can be used to define different types of constraints for desires (with strength s). The two poles can be compared in the following four ways, assuming there are a_1 and a_2 worlds.

$u \models a_1 \succ_{mM:s} a_2$
$\Leftrightarrow mu(a_1) \geq s + Mu(a_2)$
$\Leftrightarrow \min_{w \models a \wedge b} u(w) \geq s + \max_{w \models \neg a \wedge b} u(w)$
$u \models a_1 \succ_{MM:s} a_2$
$\Leftrightarrow Mu(a_1) \geq s + Mu(a_2)$
$\Leftrightarrow \max_{w \models a \wedge b} u(w) \geq s + \max_{w \models \neg a \wedge b} u(w)$
$u \models a_1 \succ_{mm:s} a_2$
$\Leftrightarrow mu(a_1) \geq s + mu(a_2)$
$\Leftrightarrow \min_{w \models a \wedge b} u(w) \geq s + \min_{w \models \neg a \wedge b} u(w)$
$u \models a_1 \succ_{Mm:s} a_2$
$\Leftrightarrow Mu(a_1) \geq s + mu(a_2)$
$\Leftrightarrow \max_{w \models a \wedge b} u(w) \geq s + \min_{w \models \neg a \wedge b} u(w)$

In Definition 1 below a desire $D(a|b)$ is defined as usual by $a \wedge b \succ \neg a \wedge b$. If either $a \wedge b$ or $\neg a \wedge b$ is inconsistent, i.e. if there are no worlds satisfying it, then we assume that the desire is vacuously true.

Definition 1 (Logic of parametrized desires) *A* (parametrized) desire *is defined by a pair of propositional formulas a and b together with a real $s > 0$ for strength and an index $l \in \{mM, MM, mm, Mm\}$ for lifting, and is denoted $D_{l:s}(a \mid b)$. A* (parameterized) desire specification $DS = \{D_{l_1:s_1}(a_1|b_1), \ldots, D_{l_n:s_n}(a_n|b_n)\}$ *is a finite set of parameterized desires. An* utility function *u, a map from W to the reals $\mathbb{R}$, satisfies the desire $D_{l:s}(a|b)$, written as $u \models D_{l:s}(a|b)$, if and only if there are no $a \wedge b$ worlds, or there are no $\neg a \wedge b$ worlds, or according to the following truth conditions.*

$$
\begin{array}{l}
u \models D_{mM:s}(a|b) \\
\Leftrightarrow mu(a \wedge b) \geq s + Mu(\neg a \wedge b) \\
\Leftrightarrow \min_{w \models a \wedge b} u(w) \geq s + \max_{w \models \neg a \wedge b} u(w) \\
u \models D_{MM:s}(a|b) \\
\Leftrightarrow Mu(a \wedge b) \geq s + Mu(\neg a \wedge b) \\
\Leftrightarrow \max_{w \models a \wedge b} u(w) \geq s + \max_{w \models \neg a \wedge b} u(w) \\
u \models D_{mm:s}(a|b) \\
\Leftrightarrow mu(a \wedge b) \geq s + mu(\neg a \wedge b) \\
\Leftrightarrow \min_{w \models a \wedge b} u(w) \geq s + \min_{w \models \neg a \wedge b} u(w) \\
u \models D_{Mm:s}(a|b) \\
\Leftrightarrow Mu(a \wedge b) \geq s + mu(\neg a \wedge b) \\
\Leftrightarrow \max_{w \models a \wedge b} u(w) \geq s + \min_{w \models \neg a \wedge b} u(w)
\end{array}
$$

An utility function u satisfies the desire specification DS, written as $u \models DS$, if and only if it satisfies each desire in DS.

The four types of desires directly imply the properties written below, in which we say that 'world w_1 is better than world w_2' if we have $u(w_1) > u(w_2)$.

$u \models D_{mM:s}(a\|b)$	each $a \wedge b$ world is better than all the $\neg a \wedge b$ worlds,
$u \models D_{MM:s}(a\|b)$	the best b worlds are a worlds, or there are no b worlds,
$u \models D_{mm:s}(a\|b)$	the worst b worlds are $\neg a$ worlds, or there are no b worlds,
$u \models D_{Mm:s}(a\|b)$	there is an $a \wedge b$ world that is better than a $\neg a \wedge b$ world, or there are no b worlds.

The following proposition shows the relations between the different types of desires.

Proposition 2 (Relations between param. desires) *We have the following relations between the parameterized desires based on the different values for the lifting parameter.*

- *if $u \models D_{mM:s}(a \mid b)$ then $u \models D_{MM:s}(a \mid b)$, $u \models D_{mm:s}(a|b)$ and $u \models D_{Mm:s}(a|b)$, and*
- *if $u \models D_{mM:s}(a \mid b)$, $u \models D_{MM:s}(a \mid b)$ or $u \models D_{mm:s}(a|b)$ then $u \models D_{Mm:s}(a|b)$,*
- *$u \models D_{MM:s}(a|b)$ does not imply $u \models D_{mm:s}(a|b)$ or vice versa.*

These relations are represented in Figure 1 below.

Proof *Follows directly from the fact that all truth conditions are universally quantified constraints on pairs of worlds, together with the fact that $Mu(a) \geq mu(a)$.*

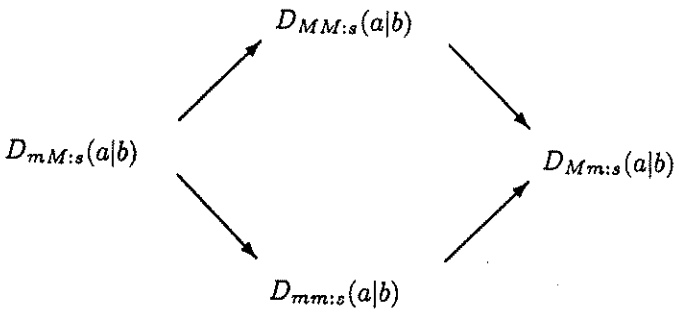


Figure 1: Relations between the four types of desires

Consider the additional assumption that the lifting parameters of all desires of the desire specification have the same value. In that case the lifting parameter is not a property of the individual desires but may be seen as a way we reason about desires. This is represented by indexing the satisfiability relation by the used lifting parameter, e.g. $\models_{mM}$, instead of the individual desires. In the following definition we say for the four lifting values l that DS is a l-conflict set if DS is inconsistent with respect to $\models_l$.

Definition 3 (Conflicts) *A desire specification DS (with only strength parameters) is an l-conflict set if there is no u with $u \models D_{l:s}(a|b)$ for each $D_s(a|b) \in DS$. DS is called conflict-free if it is not an mM-conflict set.*

We end this section with a brief discussion and illustration of the new types of desires. First, the desire $D_{mm:s}(a|b)$ is the dual of $D_{MM:s}(a|b)$ and has similar properties. As we already observed above, $D_{mm:s}(a|b)$ reflects a pessimistic view in the sense that it only considers the worst b states, whereas $D_{MM:s}(a|b)$ only considers the best b states.

Second, the desire $D_{mM:s}(a|b)$ induces a constraint on utility functions that is in the present setting too strong to be of much use, because it is rare that each $a \wedge b$ world is better than all $\neg a \wedge b$ worlds. For example, two desires 'to be healthy' $D_{s_1}(h \mid \top)$ and 'to be wealthy' $D_{s_2}(w|\top)$ are a mM-conflict set. Utility functions cannot satisfy the strong constraints if there are $w \wedge \neg h$ and $\neg w \wedge h$ worlds, because the first constraint prefers the first world to the second one and the second constraint vice versa. Moreover, a specificity set (there is a preference of no surgery over surgery, but this is inverse if surgery improves one's long term health [Bacchus and Grove, 1996]) is an mM-conflict set.

There is a set of examples for which the strong desires can be used, though. In other words, there are non-trivial conflict-free sets of desires. An example is the transitivity set discussed below, together with two other conflict free sets of desires. In this example we use the

fact that the set of worlds can be restricted in the obvious way to all worlds which satisfy a set of formulas called the background knowledge – see [Lang, 1996] for details.

Example 4 (Transitivity) *Consider the following three desire specifications, together with the background knowledge* $\neg(p \wedge c)$, $\neg(p \wedge h)$, $\neg(c \wedge h)$ *and* $\top \leftrightarrow (p \vee c \vee h)$. *This background knowledge encodes that the three variables* p, c *and* h *are mutually exclusive and exhaustive. Hence, there are only* $p \wedge \neg c \wedge \neg h$, $\neg p \wedge c \wedge \neg h$ *and* $\neg p \wedge \neg c \wedge h$ *worlds in* W. *We also give the representation based on* $\succ$ *operators, because they are the most readible. CTD and ATD represent contrary-to-duty and according-to-duty examples extensively discussed in the logic of obligations, see e.g. [van der Torre and Tan, 1999].*

TRANS	$D_{mM:1}(p \mid p \vee c)$	$p \succ_{mM:1} c$
	$D_{mM:1}(c \mid c \vee h)$	$c \succ_{mM:1} h$
CTD	$D_{mM:1}(p \mid p \vee c \vee h)$	$p \succ_{mM:1} c \vee h$
	$D_{mM:1}(c \mid c \vee h)$	$c \succ_{mM:1} h$
ATD	$D_{mM:1}(p \mid p \vee c)$	$p \succ_{mM:1} c$
	$D_{mM:1}(p \vee c \mid p \vee c \vee h)$	$p \vee c \succ_{mM:1} h$

The three sets of constraints are equivalent. For all worlds w_1, w_2, w_3 *such that* $w_1 \models p$, $w_2 \models c$ *and* $w_3 \models h$ *we have that* $u(w_1) \geq 1 + u(w_2) \geq 2 + u(w_3)$.

Finally, we consider the weakest desire $D_{Mm:s}(a|b)$. It seems to be too weak to be of any use, because there is nearly always an $a \wedge b$ world that is better than some $\neg a \wedge b$ world. However, some examples suggest that the three other constraints are too strong. One example is the marriage of Sue example of Bacchus and Grove [Bacchus and Grove, 1996].

Example 5 (Marriage) *Consider the desire specification DS that consists of the following three desires.*

$D_1(j \mid \top)$	*Sue prefers to be married to John*
$D_1(f \mid \top)$	*Sue prefers to be married to Fred*
$D_1(\neg(j \wedge f) \mid \top)$	*Sue prefers to be married to neither*

DS is an mM-, MM- and mm-conflict set. For example, the desire specification

$$\{D_{MM:1}(j|\top), D_{MM:1}(f|\top), D_{MM:1}(\neg(j \wedge f)|\top)\}$$

is inconsistent, because there is not a single world that satisfies the materializations of all three desires (j, f *and* $\neg(j \wedge f)$). *In other words, each world violates at least one desire* ($\neg j$, $\neg f$ *or* $j \wedge f$). *However, DS is not an Mm-conflict set. An example of an utility function that satisfies the three desires* $D_{Mm:1}(j|\top)$, $D_{Mm:1}(f|\top)$ *and* $D_{Mm:1}(\neg(j \wedge f)|\top)$ *is*

$$u(w) = \begin{cases} 0 & \text{if } w \models j \leftrightarrow \neg f \\ -1 & \text{if } w \models j \leftrightarrow f \end{cases}$$

We have $u \models D_{Mm:1}(j|\top)$ *because* $j \wedge \neg f$ *worlds are better than* $\neg j \wedge \neg f$ *worlds, we have* $u \models D_{Mm:1}(f|\top)$ *because* $\neg j \wedge f$ *worlds are better than* $\neg j \wedge \neg f$ *worlds, and we have* $u \models D_{Mm:1}(\neg(j \wedge f)|\top)$ *because* $j \leftrightarrow \neg f$ *worlds are better than* $j \wedge f$ *worlds.*

A second example that is only consistent with the weakest constraint is the following desire specification.

Example 6 (Fence and dog) *Consider the desire specification DS that consists of the following three desires.*

$D_1(\neg f \mid \top)$	*preference for no fence*
$D_1(f \mid d)$	*preference for fence if there is a dog*
$D_1(d \mid \top)$	*preference for a dog*

DS is an mM-, MM- and mm-conflict set, but it is not an Mm-conflict set. An example of an utility function that satisfies the three desires $D_{Mm:1}(\neg f \mid \top)$, $D_{Mm:1}(f|d)$ *and* $D_{Mm:1}(d|\top)$ *is*

$$u(w) = \begin{cases} 0 & \text{if } w \models f \leftrightarrow d \\ -1 & \text{if } w \models f \leftrightarrow \neg d \end{cases}$$

We have $u \models D_{Mm:1}(\neg f \mid \top)$ *because* $\neg f \wedge \neg d$ *worlds are better than* $f \wedge \neg d$ *worlds, we have* $u \models D_{Mm:1}(f|d)$ *because* $f \wedge d$ *worlds are better than* $\neg f \wedge d$ *worlds, and we have* $u \models D_{Mm:1}(d|\top)$ *because* $f \wedge d$ *worlds are better than* $f \wedge \neg d$ *worlds.*

Summarizing, there are desire specifications which can be analyzed with the strongest desires, and there are desire specifications which can only be analyzed with the weakest desires. However, most examples can more naturally be formalized with D_{MM}, i.e. with the semantics used in Boutilier's, Lang's and Weydert's frameworks. This will therefore be our standard representation.

2.2 The nonmonotonic construction

In this section we introduce our third parameter. We call it the polarity parameter p and we express desires with polarity by $D^p_{l:s}(a|b)$. It is used in the local utility functions, i.e. in the construction of the distinguished utility functions. Consider a local utility function that not only considers loss of utility for violations, as in Lang's construction, but also gain of utility for rewards. That is, the real valued function u is a local utility function of $D_{l:s}(a \mid b)$ - $u_{a|b}$ in Lang's notation - if there exists an $\alpha \geq 0$ (its utility loss) and a $\beta \geq 0$ (its utility gain) with $\alpha + \beta \geq s$ such that

$$u(w) = \begin{cases} \beta & \text{if } w \models a \wedge b \\ 0 & \text{if } w \models \neg b \\ -\alpha & \text{if } w \models \neg a \wedge b \end{cases}$$

For representational convenience we represent this utility function below by $u = u^{\beta}_{a \wedge b} + u^{-\alpha}_{\neg a \wedge b}$. The two reals β and $-\alpha$ are the two poles of this local utility function, in the sense that for all worlds w we have that $\beta \geq u(w) \geq -\alpha$. The polarity parameter is defined by $p = \frac{\alpha}{\alpha+\beta}$, and thus restricts the relative values of α and β. Obviously we have $0 \leq p \leq 1$. For example,

mixed gain-loss desires with polarity 0.5 have their set of local utility functions u defined for $\alpha \geq 0.5 \times s$ with $u = u^{\alpha}_{a \wedge b} + u^{-\alpha}_{\neg a \wedge b}$, i.e.

$$u(w) = \begin{array}{ll} \alpha & \text{if } w \models a \wedge b \\ 0 & \text{if } w \models \neg b \\ -\alpha & \text{if } w \models \neg a \wedge b \end{array}$$

If the polarity of a desire is 0 then we call the desire a gain desire, because its utility loss α is zero. Likewise, if its polarity is 1 then we call it a loss desire, because its utility gain β is zero.

The philosophy of Lang's framework is to define the utility functions of a set of desires as a function of the utility functions of elements of this set; the latter are called their local utility functions. The same philosophy underlies multi-attribute utility theory with the use of additive independence [Keeney and Raiffa, 1976; Wellman and Doyle, 1992; Bacchus and Grove, 1996]. There are several different ways to represent this idea of defining the utility functions of a set of desires as a function from the utility functions of its elements. In this paper we follow a standard model preference semantics, similar to the one adopted by Weydert. Our reformulation of Lang's framework in standard model preference semantics has some advantages. Most importantly, in his definition it is unclear that there is a *set* of local utility functions associated with each desire, and that for the construction of the global utility function we have to choose elements from these sets. The representation in Definition 7 below also facilitates Proposition 8 afterwards. A second minor advantage is that logical notions such as inference relations are defined in the standard way.

Local and distinguished utility functions are defined in two steps. First the set of constructible utility functions is defined, represented by $CONS(DS)$, and thereafter the distinguished utility functions, represented by U_J to refer to Jeffrey conditionalization. Due to this two step definition the distinguished utility functions are *not* simple additions of local utility functions. Instead, in Proposition 8 we show that they are *weighted* additions of local functions. Moreover, due to this two-step definition the desires can be redundant, because a desire does not add anything to the distinguished utility function when its constructible utility function ranks all worlds 0.

Definition 7 (Nonmonotonic extension) *A* (parameterized) desire *is defined by a pair of propositional formulas a and b together with the real $0 \leq p \leq 1$ for polarity, $l \in \{mM, MM, mm, Mm\}$ for lifting, and the real $s > 0$ for strength, and is denoted $D^{p}_{l:s}(a \mid b)$. A* (parameterized) desire specification *$DS = \{D^{p_1}_{l_1:s_1}(a_1 \mid b_1), \ldots, D^{p_n}_{l_n:s_n}(a_n \mid b_n)\}$ is a finite set of parameterized desires. The set of utility functions of DS, denoted by $U(DS)$, is the set of its models as given in Definition 1.*

$$U(DS) = \{u \mid u \models D_{l_1:s_1}(a_1|b_1), \ldots, u \models D_{l_n:s_n}(a_n|b_n)\}$$

The preferred or distinguished utility functions of a single desire, also called its local utility functions, are defined in two steps as follows. Let u^{α}_{a} be the utility function such that $u(w) = \alpha$ if $w \models a$, $u(w) = 0$ otherwise.

$$CONS(D^{p}_{l:s}(a|b)) = \{\{u^{\beta}_{a \wedge b} + u^{-\alpha}_{\neg a \wedge b} \mid \frac{\alpha}{\alpha + \beta} = p \text{ and } \alpha, \beta \geq 0\}$$

$$U_J(D^{p}_{l:s}(a|b)) = U(\{D^{p}_{l:s}(a|b)\}) \cap CONS(D^{p}_{l:s}(a|b)) = \{u^{\beta}_{a \wedge b} + u^{-\alpha}_{\neg a \wedge b} \mid \frac{\alpha}{\alpha + \beta} = p, \alpha, \beta \geq 0, \alpha + \beta \geq s\}$$

The preferred or distinguished utility functions of a desire specification DS are constructed as follows.

$$CONS(DS) = \left\{ u = u_1 + \ldots + u_n \;\middle|\; \begin{array}{l} u_1 \in CONS(D^{p_1}_{l_1:s_1}(a_1|b_1)), \\ \ldots, \\ u_n \in CONS(D^{p_n}_{l_n:s_n}(a_n|b_n)) \end{array} \right\}$$

$$U_J(DS) = U(DS) \cap CONS(DS)$$

The following proposition illustrates the formal construction by considering equivalent weighted additions, and it shows how to construct distinguished utility functions from single local utility functions instead of sets of them.

Proposition 8 (Weighted additions) *The constructible utility functions of*

$$DS = \{D^{p_1}_{l_1:s_1}(a_1|b_1), \ldots, D^{p_n}_{l_n:s_n}(a_n|b_n)\}$$

are weighted additions of local utility functions.

$$CONS(DS) = \left\{ u = k_1 \times u_1 + \ldots + k_n \times u_n \;\middle|\; \begin{array}{l} u_1 \in U_J(D^{p_1}_{l_1:s_1}(a_1|b_1)), \\ \ldots, \\ u_n \in U_J(D^{p_n}_{l_n:s_n}(a_n|b_n)), \\ k_1 \geq 0, \ldots, k_n \geq 0 \end{array} \right\}$$

The constructible utility functions of DS are weighted additions of the minimal local utility functions $U_{min}(D^{p}_{l:s}(a|b)) = u^{s \times (1-p)}_{a \wedge b} + u^{-s \times p}_{\neg a \wedge b}$.

$$CONS(DS) = \left\{ u = k_1 \times u_1 + \ldots + k_n \times u_n \;\middle|\; \begin{array}{l} u_1 = U_{min}(D^{p_1}_{l_1:s_1}(a_1|b_1)) \\ \ldots \\ u_n = U_{min}(D^{p_n}_{l_n:s_n}(a_n|b_n)) \\ k_1 \geq 0, \ldots, k_n \geq 0 \end{array} \right\}$$

Proof *We first consider the first equivalence, and we prove that* $CONS_1(DS) = CONS_2(DS)$ *where* $CONS_1$ *is the construction defined in Definition 7 and* $CONS_2$ *is the first weighted addition defined above. That is, for each utility function in one construction we show for which variables* α, β *and* k *this utility function is also part of the other construction.*

$\Rightarrow$ *For each desire, define* α, β *and* k_i *in* $CONS_2$ *by* $\alpha \times \frac{s}{\alpha+\beta}$, $\beta \times \frac{s}{\alpha+\beta}$ *and* $\frac{\alpha+\beta}{s}$ *for* α *and* β *in* $CONS_1$. *The local utility functions used in* $CONS_2$ *satisfy the constraints, because* $\alpha \times \frac{s}{\alpha+\beta} + \beta \times \frac{s}{\alpha+\beta} = s$.

$\Leftarrow$ *For each desire, define* α *and* β *in* $CONS_1$ *by* $k_i \times \alpha$ *and* $k_i \times \beta$ *for* k_i, α *and* β *in* $CONS_2$.

We now consider the second equivalence, and we prove that $CONS_2(DS) = CONS_3(DS)$ *where* $CONS_2(DS)$ *is again the first weighted addition defined above and* $CONS_3$ *is the second one. This follows directly from the fact that the utility function we constructed in the previous item is in fact the minimal one.*

$\Rightarrow$ *The* $\Leftarrow$*-part of the previous item shows how to construct an element of* $CONS_1$ *from an element of* $CONS_3$, *and the* $\Rightarrow$*-part of the previous item shows how to construct an element of* $CONS_3$ *from an element of* $CONS_1$.

$\Leftarrow$ *Trivial since* U_{min} *is an element of* U_J.

The following proposition shows that the existence of distinguished utility functions of a desire specification does not follow from the existence of utility functions. Weydert has proven this for his defaults, i.e. for loss D_{MM} desires. It is an open problem whether it can be proven in a more general context, e.g. for all D_{MM} desires. This property is considered very desirable in reasoning about defaults (see [Kraus *et al.*, 1990]), but it is not clear whether it plays a similar role in reasoning about desires.

Proposition 9 *Let* DS *be a desire specification.* $U(DS) \neq \emptyset$ *does* not *imply* $U_J(DS) \neq \emptyset$.

Proof *Two counterexamples are the desire specification* $DS = \{D_{Mm}(p), D_{Mm}(\neg p)\}$ *and the desire specification* $DS = \{D_{MM}(p), D_{mm}(\neg p)\}$. *Both have models but no preferred models.*

3 Conclusions

In this paper we have studied and extended the logic of desires in Lang's framework for qualitative decision theory. We introduced three parameters for the utilitarian desires that reflect its strength and the risk attitude of the agent, because utilities represent besides values also the agent's risk attitude. The parameterized desires can deal with the class of intuitions about the logical properties of desires by changing the parameter values for the requirements at hand. Despite the fact that the mechanisms developed in reasoning about defaults could be used for desires, it seems very unlikely that our logic of desires can be used to formalize defaults. In reasoning about uncertainty there is no formal counterpart of risk.

Subjects for further research are studies of minimization principles introduced in [Weydert, 1995; 1996; 1998] in the logic for desires, of existence theorems for fragments of the logic, and the search for general guidelines or heuristics for the values of the parameters (such as particular combinations of them) and for the determination of the parameter values in an interactive system.

References

[Bacchus and Grove, 1996] F. Bacchus and A.J. Grove. Utility independence in a qualitative decision theory. In *Proceedings of KR'96*, pages 542–552, 1996.

[Boutilier, 1994] C. Boutilier. Toward a logic for qualitative decision theory. In *Proceedings of the KR'94*, pages 75–86, 1994.

[Doyle and Wellman, 1991] J. Doyle and M.P. Wellman. Preferential semantics for goals. In *Proceedings of AAAI-91*, pages 698–703, Anaheim, 1991.

[Haddawy and Hanks, 1992] P. Haddawy and S. Hanks. Representations for decision-theoretic planning: Utility functions for dead-line goals. In *Proceedings of the KR'92*, Cambridge, MA, 1992.

[Jeffrey, 1983] R. Jeffrey. *The Logic of Decision.* University of Chicago Press, 2nd edition, 1983.

[Keeney and Raiffa, 1976] R.L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Trade-offs.* Wiley and Sons, New York, 1976.

[Kraus *et al.*, 1990] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, 44:167–207, 1990.

[Lang, 1996] J. Lang. Conditional desires and utilities - an alternative approach to qualitative decision theory. In *Proceedings of the ECAI'96*, pages 318–322, 1996.

[Luce and Raiffa, 1957] R.D. Luce and H. Raiffa. *Games and Decisions.* John Wiley, New York, 1957.

[Mullen, 1979] J.D. Mullen. Does the logic of preference rest on a mistake? *Metaphilosophy*, 10:247–255, 1979.

[Pearl, 1993] J. Pearl. From conditional ought to qualitative decision theory. In *Proceedings of the UAI'93*, pages 12–20, 1993.

[Tan and Pearl, 1994a] S.-W. Tan and J. Pearl. Qualitative decision theory. In *Proceedings of the AAAI'94*, 1994.

[Tan and Pearl, 1994b] S.-W. Tan and J. Pearl. Specification and evaluation of preferences under uncertainty. In *Proceedings of the KR'94*, pages 530–539, 1994.

[Thomason and Horty, 1996] R. Thomason and R. Horty. Nondeterministic action and dominance: foundations for planning and qualitative decision. In *Proceedings of the TARK'96*, pages 229–250. Morgan Kaufmann, 1996.

[van der Torre and Tan, 1999] L.W.N. van der Torre and Y.H. Tan. Contrary-to-duty reasoning with preference-based dyadic obligations. *Submitted*, 1999.

[Wellman and Doyle, 1992] M.P. Wellman and J. Doyle. Modular utility representation for decision-theoretic planning. In *Proceedings of the first international conference on artificial intelligence planning systems (AIPS92)*, pages 236–242, 1992.

[Weydert, 1995] E. Weydert. Default entailment a preferential construction semantics for defeasible inference. In Ipke Wachsmuth, Claus-Rainer Rollinger, and Wilfried Brauer, editors, *Annual German Conference on Artificial Intelligence (KI-19) : Bielefeld, Germany, September 11-13, 1995; proceedings*, volume LNAI 981, pages 173–184, Berlin, 1995. Springer.

[Weydert, 1996] E. Weydert. System J – revision entailment: Default reasoning through ranking measure updates. In Dov Gabbay and Hans Jürgen Ohlbach, editors, *Practical Reasoning - International Conference on Formal and Applied Practical Reasoning, FAPR'96*, volume 1085 of *Lecture Notes in Computer Science*, pages 637–649, Bonn, Germany, June 1996. Springer.

[Weydert, 1998] E. Weydert. System jz: How to build a canonical ranking model of a default knowledge base. In *Proceedings of the Seventh International conference on Knowledge Representation and Reasoning (KR'98)*, pages 190–201, 1998.

Full Acceptance Through Argumentation - a Preliminary Report

Renata Wassermann
Institute for Logic, Language and Computation
University of Amsterdam
Plantage Muidergracht 24
1018TV - Amsterdam, The Netherlands
e-mail: renata@wins.uva.nl

Abstract

When an agent receives new pieces of information, these may contradict his previous beliefs. The agent must decide how to solve this contradiction. Most frameworks dealing with the problem of belief revision attach higher priority to incoming information, i.e., they may give up some part of the old beliefs in order to accommodate the new piece of information and keep consistency. In this paper, we propose the use of argumentation theory to decide whether incoming information should be accepted or not.

1 Introduction

The problem of belief revision, i.e., of how the beliefs of an agent should change in the presence of new information, has been recently addressed by various authors. In most approaches, specially those following the AGM paradigm [Alchourrón *et al.*, 1985], the agents are idealized in that they are assumed to have perfect recall and to hold only consistent beliefs, which are furthermore assumed to be closed under logic consequence.

Incoming information is usually given the highest priority, so that if a contradiction arises, some of the previous beliefs have to be given up. In approaches to non-prioritized belief revision [Hansson, 1997], i.e., revision in which the new piece of information does not have the highest priority, the decision whether to accept or not new information is taken by extra-logical means such as selection functions or incision functions, but there is no real recipe of how to choose these functions. In this paper, we explore a different idea - using argumentation theory for deciding whether new information is acceptable.

In [Wassermann, 1999b] we have developed a framework for belief revision which takes into account the effects of both limited memory and limited capacities of inference. In this model, the belief state of an agent is represented by a structure that distinguishes different kinds of beliefs: beliefs that are explicit or basic, beliefs that are implicit or merely derived, and beliefs that are active, i.e., in use. Besides the beliefs of the agent, the structure also represented "provisional beliefs", i.e., sentences for which the agent has some evidence but is not yet sure whether to accept them or not. In [Hansson and Wassermann, 1999] and [Wassermann, 1999a], some ways of deciding which beliefs were active during a certain operation of belief change were explored. In this paper, we turn to another question left open by the model in [Wassermann, 1999b], namely how to decide whether a provisional belief should be accepted.

According to [Dung, 1995], a formula is believable "if it can be argued successfully against attacking arguments". Dung also says that reasoning abount one's own beliefs is like performing an internal argument. Our concept of provisional beliefs is based on Harman's idea of *tentative hypotheses*. In order to be fully accepted, a tentative hypothesis has to survive the best attempts to refute it [Harman, 1986]. In our case, "best attempts" are as good as the agent is capable given his limitations.

This is reflected in the framework for resource-bounded argumentation given in [Loui, 1998]. Loui describes a very general framework where there are a number of parties involved, some of which (the players) are allowed to make locutions, the others being advocates. Each of the players try to get the current opinion to be in his favour by presenting arguments. A vector represents the resources consumed at each move.

A protocol for disputation has to be defined and depends on the application. These are the real "rules of the game", which determine what is allowed as a move, who is allowed to take next move, how the moves affect the current opinion, and what the conditions for termination are. In [Loui, 1998], some protocols are presented, which can be chosen according to the intended application.

In the next section we will present the framework for resource-bounded belief revision introduced in [Wassermann, 1999b]. Then we will present the theory of argumentation that we use, based on [Loui, 1998]. In section 4 we present our proposal for using the theory sketched in section 3 to enrich the framework presented in section 2.

2 Belief States and Change Operations

In this section, we are going to briefly present the framework for resource-bounded belief revision introduced in

[Wassermann, 1999b]. We start by introducing some distinctions between different kinds of beliefs. The example below motivates the distinctions.

Consider the following situation: Mary is going out, and her mother tells her that she should take an umbrella. Besides beliefs about other subjects, Mary holds the belief that if she is going to be outside for a long time, then she should take the umbrella. She also believes that she will be outside the whole day. If her mother had not mentioned the umbrella, Mary would not have thought of it. Upon it being brought to her notice, she concludes she should indeed take the umbrella.

Following Harman [Harman, 1986], we will assume that there are beliefs that are explicitly represented. We call *implicit beliefs* those beliefs that can be inferred from the set of the agent's explicit beliefs, according to the agent's logical ability. We will not concentrate in one particular inference operator, but use Inf to denote what an agent can infer in one step. The set of implicit beliefs is given by what the agent would be able to infer if he was given unlimited time, i.e., the result of applying Inf an unlimited number of times.

Not all of the agents beliefs are available at the same time. We call *active beliefs* the set containing beliefs that are available for use and things about which the agent is not yet sure. These last are called *provisional beliefs*. Provisional beliefs are not real beliefs, since they are still under inspection. They are outside the set of explicit beliefs.

A belief state is a structure $\beta = \langle E, Inf, A \rangle$, where E is the set of the agent's explicit beliefs, Inf is the agents inference function and A is the set of the agent's active beliefs. The set of implicit beliefs is given by: $I = Inf^*(E) = Inf(E) \cup Inf^2(E) \cup Inf^3(E) \cup .....$

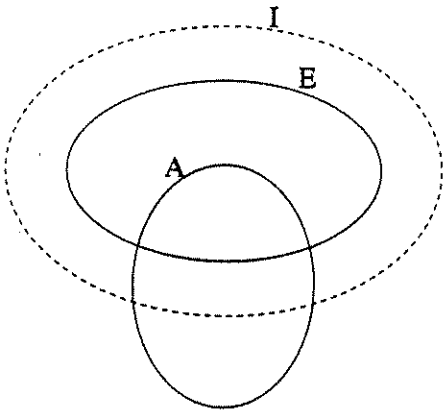


Figure 1: Structure of an agent's beliefs

At this point it may be useful to return to our small example to illustrate the difference between explicit and active beliefs. Mary's belief that if she is going to be outside for a long time, then she should take the umbrella is part of her explicit beliefs and so is her belief that she will be outside the whole day. These beliefs only become active when her mother mentions the umbrella. When Mary thinks of it, she infers that she should take the umbrella. This example shows an argument against representing belief states as logically closed sets. Mary did not hold the belief that she should take the umbrella until the time at which the inference was made. It also shows that not all beliefs are active at the same time. Let p stand for "Mary should take an umbrella" and q for "Mary will be outside for a long time". Before talking to her mother, Mary's explicit beliefs contain, among others, the beliefs q and $q \rightarrow p$. The implicit beliefs contain, among others, p. The set of active beliefs is empty (actually it could probably contain some remains of other reasoning, but this is not relevant for this argument). When the mother says that Mary should take an umbrella, p becomes a provisional active, but not explicit, belief. Mary does not necessarily believe everything her mother says immediately, so that she has to think about it. This is as if she were asking herself whether she should take the umbrella. The beliefs q and $q \rightarrow p$ become active, since they are relevant for deciding whether to accept p. When Mary eventually decides to accept p, this belief is made explicit and the set of active beliefs may get new elements according to new input.

Given our representation of belief state, the next step is to define operations that can be applied to belief states to modify them.

In AGM theory, three operations are defined on belief states: expansion, contraction, and revision. Expansion consists in adding a new belief to the belief state without checking the consistency of the resulting state, contraction consists in deleting a belief from a belief state in a way that the resulting state does not imply the deleted belief, and revision consists in adding a belief to a belief state in such a way that the resulting belief state is consistent. Traditionally, revision is seen as a sequence of a contraction and an expansion (in any order). But this is not a division into simpler steps, since contraction is (computationally) as complicated as revision. We want to decompose revision and contraction in simple operations that show what happens with an agent's belief state in each step, instead of only analyzing the initial and final states.

The set of active beliefs is based on the concept of a short-term memory. Beliefs that are active can be forgotten or stored as explicit (but inactive) beliefs. Since the set of active beliefs is assumed to be very limited in size, there must be a mechanism that, in cases of overflow, selects which beliefs will be forgotten or stored.

The first operation we define is similar to AGM expansion in the sense that it consists in simply adding new information to a set without checking for consistency. But the operation takes the limited size of the set

into account.[1] When trying to add something to a set that is already at its maximum size, some elements of the set have to be given up. This can be seen as a kind of "forgetting".

If X is a set with maximum size m and α is an element we want to add to X, then:

$$X \cup^* \{\alpha\} = X' \cup \{\alpha\}, \text{ where } X' \subseteq X, |X'| < m.$$

Note that this operation reduces to a simple union as long as the set is not "full". Since the size m of the set is given as a parameter, the operation is more accurately denoted as $\cup^*_m$. When the set is already at its maximum size, something has to be discarded. If the set X is ordered (for example by the last time the beliefs were recalled), we can stipulate that the minimal elements of the set are the first to be dismissed, i.e., we want to ensure that if an element is dismissed, then there is no other element which is retained and that is less than the dismissed one in the order:

$$\forall y(y \in X \setminus X' \rightarrow \neg\exists x(x \in X' \wedge x < y)).$$

We now define six operations that can be applied on belief states to change the status of beliefs.

Definition 2.1 *Let $\langle E, Inf, A\rangle$ be a belief state and α a formula. We define the following operations on $\langle E, Inf, A\rangle$ (we will omit the second argument Inf since the operations defined do not affect it):*

1. *Observation ($+_o$): adds an external input to the set of active beliefs.*
   $\langle E, A\rangle +_o \alpha = \langle E, A \cup^* \{\alpha\}\rangle$
2. *Retrieval ($+_r$): retrieves an explicit belief into the set of active beliefs.*
   $$\langle E, A\rangle +_r \alpha = \begin{cases} \langle E, A \cup^* \{\alpha\}\rangle, & \text{if } \alpha \in E \\ \langle E, A\rangle & \text{otherwise} \end{cases}$$
3. *Acceptance ($+_a$): makes an active belief explicit.*[2]
   $$\langle E, A\rangle +_a \alpha = \begin{cases} \langle E \cup^* \{\alpha\}, A \setminus \{\alpha\}\rangle, & \text{if } \alpha \in A \\ \langle E, A\rangle & \text{otherwise} \end{cases}$$
4. *Inference ($+_i$): infers something from active beliefs.*
   $$\langle E, A\rangle +_i \alpha = \begin{cases} \langle E, A \cup^* \{\alpha\}\rangle, & \text{if } \alpha \in Inf(A) \\ \langle E, A\rangle & \text{otherwise} \end{cases}$$
5. *Doubting ($+_d$): a belief that was accepted is questioned, becoming provisional.*
   $$\langle E, A\rangle +_d \alpha = \begin{cases} \langle E \setminus \{\alpha\}, A\rangle, & \text{if } \alpha \in A \cap E \\ \langle E, A\rangle & \text{otherwise} \end{cases}$$
6. *Rejection ($+_c$): rejects an active belief.*
   $$\langle E, A\rangle +_c \alpha = \begin{cases} \langle E, A \setminus \{\alpha\}\rangle, & \text{if } \alpha \in A \\ \langle E, A\rangle & \text{otherwise} \end{cases}$$

[1] When we talk about the size of a set of formulas, we mean something like its complexity. The sets $\{p, q\}$ and $\{p \wedge q\}$ should have the same size. We could, for example, count the occurrence of atoms.

[2] Acceptance could also be defined without deleting the accepted belief from A, which seems to be more intuitive for human agents. The choice made here reflects our interest in artificial agents.

The six operations defined above can be combined to model more complex operations. As an example of such a composition, consider what happens when an agent gets new information via observation. The belief will first come into the set of active beliefs through the operation $+_o$ and then the agent may accept it ($+_a$). Another example is the case of an explicit belief that becomes active (retrieval: $+_r$), when it would be expected that some implicit beliefs will also become active, i.e., the retrieval operation will be followed by an inference ($+_i$).

3 Argumentation

In this section, we introduce the basic concepts of argumentation theory that we will need in this paper. This section is based on [Loui, 1998].

Argumentation has been investigated by researchers in the area of philosophy and artificial intelligence. Recently, it became clear that argumentation can be seen as a kind of non-monotonic reasoning. Arguments are not proof, but some kind of justification for a claim, usually defeasible. An argumentation process usually follows some protocol that defines what the possible moves are, how a move affects the current state of the disputation, who is allowed to move, etc. Once the parties involved in the disputation agree about the protocol, the outcome of an argumentation process following the protocol is considered fair.

Disputations are highly non-monotonic. The outcome depends on the particular way in which the argumentation process took place and if the process continues, the outcome may change. Nevertheless, the process is fair (provided the disputants agreed about the protocol) and the outcome is warranted.

An argument is usually a pair formed by a set of formulas and one special formula, the claim. The set of formulas serves as a justification for the claim. Arguments are related to each other is several ways. Arguments can interfere with each other, in case their claims (or subclaims) are inconsistent.

Loui [Loui, 1998] defines a very general framework for argumentation that has to be "filled in" in order to model particular kinds of disputation.

An argumentation process is a sequence of locutions, where each locution is a triple formed by one party, the argument and the resources consumed. The participants of the argumentation process do not have necessarily access to the same information. They may also have different shares of resources at their disposal. In our case, we will use argumentation processes where only two parties are involved, **pro** and **con**. A variable *current.opinion* stores the party which is winning the disputation at a certain point. The parties try to switch the current opinion in their favour by advancing locutions. Since we are modeling an internal argumentation process, where a single agent is involved and plays the roles of **pro** and **con**, we can assume that both parties have access to the same information.

4 Using Argumentation for Accepting Beliefs

In this section we present our proposal for using argumentation in order to decide whether a provisional belief should be accepted or not. In our case, the argumentation is an internal process where a single agent plays the role of **pro** and **con**, analyzing the arguments for and against a given provisional belief. Since we are dealing with resource-bounded agents, this internal argumentation will not always succeed in examining all reasons for accepting or rejecting a belief. By defining a protocol for this process, we have to take care that the outcome can be considered fair.

There are two ways in which a sentence can become a provisional belief:

1. New information may be acquired by an operation of observation, i.e., come from the outside world. This new piece of information has to be checked before being fully accepted. In this case, **con** tries to argue against it. If he fails, the provisional belief is accepted, since it has survived the best attempts to refute it. If **con** succeeds, the provisional belief is rejected.
2. A sentence that was previously accepted, an explicit belief, may become provisional if the agent gets evidence against it. In this case, inquiry is reopened ([Harman, 1986]) and **pro** tries to argue for the sentence. If he fails, the provisional belief is rejected. If **pro** succeeds, the provisional belief becomes fully accepted again.

In the framework presented in section 2, there are two clearly limited resources: the size of the set of active beliefs and the number of basic operations used in the disputation process. Since in our case a single agent is playing the roles of **pro** and **con**, the set of active beliefs is a shared resource, both **pro** and **con** have access to the whole set.

All the sentences in the arguments presented become active. The elements of the set of active beliefs are ordered according to the order in which they were introduced in the argumentation. When the set gets too big, the oldest elements are "forgotten". If the discarded elements were explicit beliefs that were retrieved, they remain in the set of explicit beliefs but become inactive. If they were only provisional beliefs, then they are irremediably forgotten and dismissed from the whole structure.

An argument for us will be a sequence of elements of the set of explicit beliefs which is a derivation for its claim according to a finite (small) number of applications of inference rules known by the agent. An argument *arg* of player p (=**pro** or **con**) is counterargued when the other player presents an argument against one of the elements of *arg* (its subclaims). An argument *arg* of player p is defeated if it is counterargued by *arg'* and p does not manage to counterargue *arg'* (either because there are no counterarguments or because the resources are exhausted).

When an argument is introduced by one of the players, the beliefs that are part of it are retrieved into the set of active beliefs. When an argument is counterargued, its claim becomes provisional. If an argument is defeated, its claim is rejected.

The protocol we will be using assumes that the resources are equally divided, i.e., if player p_1 has exhausted his share of resources but p_2 has not, then p_2 is still allowed one move. Except for this situation, the players alternate the moves. No repetition of counterargued (sub-)arguments is allowed.

Suppose a sentence α is observed. The current opinion is set to **pro** and **con** tries to find an argument for $\neg\alpha$. If he fails, then α is accepted, otherwise, current opinion is set to **con** and **pro** tries to either counterargue the last argument or present a new argument for α. If **pro** fails, then α is rejected. Otherwise, current opinion is set to **pro** and **con** tries to either counterargue the last argument or present a new argument for $\neg\alpha$. The process continues until resources are exhausted. The player favoured by current opinion wins.

5 Example

We will now see an example of application of the protocol described in section 4.

We first have to give some more details about the procedure. The claim to be verified, a provisional belief, remains active during the whole argumentation. It cannot be dismissed due to overflow in the set of active beliefs. The set of active beliefs is ordered by recency, i.e., beliefs that have been used first are the first to be forgotten in case of overflow. However, if an active belief is reused, it becomes more recent and changes place in the order. This agrees with cognitive models of memory, as for example in [Anderson, 1980].

The claim which is being verified and claims of arguments that have been counterargued cannot be used in new arguments.

The size of the set of active beliefs, one of the limited resources, is given by the number of atoms occurring in its formulas. Part of the history of the process is kept in the form of arguments advanced, so that there is no repetition. This set can also be limited in size like the set of active beliefs, but in the example we will ignore this fact.

We will use the following logic for the example:

1. atoms $a, b, c, ..., p$ standing for "*a*lbert comes to the party", "*b*etty comes to the party", "*c*harles comes to the party", ..., "*p*atrick comes to the party".
2. formulas $x \rightarrow y$ standing for "If x comes to the party, then y comes to the party" ; $x \rightarrow \neg y$ standing for "If x comes to the party, then y does not come to the party", etc.
3. inference rules modus ponens $(x, x \rightarrow y \Rightarrow y)$ and inversion $(x \rightarrow y \Rightarrow \neg y \rightarrow \neg x)$.

Depending on who likes whom and who dislikes whom, we know who is (or is not) going to come to the party

given who is (or is not) coming. Moreover, we know of some people that are coming (*a*lbert, *f*erry, *h*arold, *k*ate, and *o*swald). Our initial set of explicit beliefs is:

$E = \{a, a \rightarrow b, b \rightarrow c, c \rightarrow d, d \rightarrow g, f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p, h, h \rightarrow i, i \rightarrow j, j \rightarrow \neg e, k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i, o, o \rightarrow p, p \rightarrow \neg l, \neg l \rightarrow \neg b\}$.

We assume that the maximum size of the set of active beliefs is 20. We want to know whether *i*van is coming to the party:

- step 1: **con** tries to refute i, presenting an argument for $\neg i$. The formulas in the argument are retrieved from the set of explicit beliefs and stored as active beliefs. Inference is applied four times in order to get to the claim $\neg i$ from the argument.
  - **con** presents argument $\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}$ for $\neg i$.
  - 9 basic operations: retrieval $\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}$; inference $\{l, m, n, \neg i\}$
  - $A = \{i, k, k \rightarrow l, l, l \rightarrow m, m, m \rightarrow n, n, n \rightarrow \neg i, \neg i\}$; $|A|$=14
  - History: $\{\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}\}$
  - current.opinion = **con**
- step 2: **pro** advances an argument against one of the subclaims of the previous argument. The previous argument is counterargued, but not yet defeated, since **con** may counterargue this present argument. The set of active beliefs grows to its maximum size, 20. The oldest active belief besides the claim (k) is dismissed to make space for the new activated beliefs.
  - **pro** presents counterargument $\{o, o \rightarrow p, p \rightarrow \neg l\}$ against l.
  - 5 basic operations: retrieval $\{o, o \rightarrow p, p \rightarrow \neg l\}$; inference $\{p, \neg l\}$.
  - $A = \{i, k \rightarrow l, l, l \rightarrow m, m, m \rightarrow n, n, n \rightarrow \neg i, \neg i, o, o \rightarrow p, p, p \rightarrow \neg l, \neg l\}$; $|A|$=20
  - History: $\{\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}, \{o, o \rightarrow p, p \rightarrow \neg l\}\}$
  - current.opinion = **pro**
- step 3: **con** counterargues the previous argument. The oldest elements of the set of active beliefs (except i) are dismissed to make space for the new beliefs retrieved.
  - **con** presents counterargument $\{f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p\}$ against p.
  - 7 basic operations: retrieval $\{f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p\}$; inference $\{e, \neg c, \neg p\}$.
  - $A = \{i, \neg i, o, o \rightarrow p, p, p \rightarrow \neg l, \neg l, f, f \rightarrow e, e, e \rightarrow \neg c, \neg c, \neg c \rightarrow \neg p, \neg p\}$; $|A|$=19
  - History: $\{\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}, \{o, o \rightarrow p, p \rightarrow \neg l\}, \{f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p\}\}$
  - current.opinion = **con**
- step 4: **pro** counterargues the previous argument. Again, some elements of the set of active beliefs must be dismissed.
  - **pro** presents counterargument $\{a, a \rightarrow b, b \rightarrow c\}$ against $\neg c$.
  - 5 basic operations: retrieval $\{a, a \rightarrow b, b \rightarrow c\}$; inference $\{b, c\}$
  - $A = \{i, \neg l, f, f \rightarrow e, e, e \rightarrow \neg c, \neg c, \neg c \rightarrow \neg p, \neg p, a, a \rightarrow b, b, b \rightarrow c, c\}$; $|A|$=19
  - History: $\{\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}, \{o, o \rightarrow p, p \rightarrow \neg l\}, \{f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p\}, \{a, a \rightarrow b, b \rightarrow c\}\}$
  - current.opinion = **pro**
- step 5: **con**'s arguments were defeated, since he cannot counterargue the previous arguments advanced by **pro** anymore. **con** advances a new argument against i. Some of the beliefs used in this argument ($f, f \rightarrow e, e \rightarrow \neg j$) are already active so they do not need to be retrieved. They only change place in the set of active beliefs.
  - **con** presents counterargument $\{f, f \rightarrow e, e \rightarrow \neg j, \neg j \rightarrow \neg i\}$ against i.
  - 6 basic operations: retrieval $\{j \rightarrow \neg e, i \rightarrow j\}$; inferences $\{e \rightarrow \neg j, \neg j, \neg j \rightarrow \neg i, \neg i\}$
  - $A = \{i, b, b \rightarrow c, c, f, f \rightarrow e, e, j \rightarrow \neg e, e \rightarrow \neg j, \neg j, i \rightarrow j, \neg j \rightarrow \neg i, \neg i\}$; $|A|$=19
  - History: $\{\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}, \{o, o \rightarrow p, p \rightarrow \neg l\}, \{f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p\}, \{a, a \rightarrow b, b \rightarrow c\}, \{f, f \rightarrow e, e \rightarrow \neg j, \neg j \rightarrow \neg i\}\}$
  - current.opinion = **con**
- step 6: **pro** cannot counterargue the previous argument, but presents instead a new argument for i. Since **con** does not have any other counterarguments or arguments for $\neg i$, **pro** wins the disputation.
  - **pro** presents argument $\{h, h \rightarrow i\}$ for i.
  - 3 basic operations: retrieval $\{h, h \rightarrow i\}$; inference $\{i\}$.
  - $A = \{i, c, f, f \rightarrow e, e, j \rightarrow \neg e, e \rightarrow \neg j, \neg j, i \rightarrow j, \neg j \rightarrow \neg i, \neg i, h, h \rightarrow i\}$; $|A|$=19
  - History: $\{\{k, k \rightarrow l, l \rightarrow m, m \rightarrow n, n \rightarrow \neg i\}, \{o, o \rightarrow p, p \rightarrow \neg l\}, \{f, f \rightarrow e, e \rightarrow \neg c, \neg c \rightarrow \neg p\}, \{a, a \rightarrow b, b \rightarrow c\}, \{f, f \rightarrow e, e \rightarrow \neg j, \neg j \rightarrow \neg i\}, \{h, h \rightarrow i\}\}$
  - current.opinion = **pro**

6 Conclusions

We have presented some ideas on how to use argumentation theory in order to decide which beliefs should be fully accepted. These ideas enrich the framework presented in [Wassermann, 1999b].

Although the protocol defined and the example are quite simple, they illustrate the internal process of

"weighting" the arguments in favour and against a certain claim that takes place when an agent is confronted with information about which he is not sure.

Further work includes examining existing implemented argumentation systems in order to refine the protocol of the argumentation process. One such system is presented in [Simari and Loui, 1992], together with a mathematical treatment of the relations between arguments.

Acknowledgments: I would like to thank Daniela Carbogim and Frans Voorbraak for comments on an earlier draft. This work is supported by a grant from the Brazilian funding agency CAPES.

References

[Alchourrón *et al.*, 1985] Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change. *Journal of Symbolic Logic*, 50(2):510–530, 1985.

[Anderson, 1980] John R. Anderson. *Cognitive Psychology and its Implications*. W.H.Freeman, 1980.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-persons games. *Artificial Intelligence*, 77:321–357, 1995.

[Hansson and Wassermann, 1999] Sven Ove Hansson and Renata Wassermann. Local change. In preparation (a preliminary version appeared in the Fourth Symposium on Logical Formalizations of Commonsense Reasoning, London, 1998), 1999.

[Hansson, 1997] Sven Ove Hansson, editor. *Theoria - special issue on non-prioritized belief revision*, volume 63, 1997.

[Harman, 1986] Gilbert Harman. *Change in View - Principles of Reasoning*. MIT Press, 1986.

[Loui, 1998] Ronald P. Loui. Process and police: Resource-bounded non-demonstrative reasoning. *Computational Intelligence*, 1998.

[Simari and Loui, 1992] Guillermo R. Simari and Ronald P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, 53(2-3):125–157, 1992.

[Wassermann, 1999a] Renata Wassermann. On structured belief bases. In Hans Rott and Mary-Anne Williams, editors, *Frontiers in Belief Revision*. Kluwer, 1999. to appear.

[Wassermann, 1999b] Renata Wassermann. Resource-bounded belief revision. *Erkenntnis*, 1999. to appear.