Domain Semantics for Agent-Oriented Programming

Roger Norton and Rebecca Thomas
School of Computer Science and Mathematics
Marist College
Poughkeepsie, NY 12601-1387
Roger.Norton@Marist.edu

Abstract

This paper describes a novel semantic framework for an agent architecture. Drawing on Shoham's work on AGENT0 (Shoham 1993), we replace Shoham's modal logic semantics with a new framework based on category theory. In particular, we use a consistently complete ω-algebraic complete partial order as our semantic structure, and briefly explore the consequences of this choice. Most importantly, we can now speak about the dynamic evolution of an agent's mental state, while Shoham's original work could only model the static mental state.

1. Introduction

Much recent work has been devoted to the study of agents (Rao & Georgeff 1991, Wooldridge & Jennings 1994, Wooldridge, Müller, & Tambe 1995, Wooldridge & Rao 1999), and in particular to modeling the internal states of these agents. These internal states should allow for the recording of the information necessary for the agent in its mundane operations. This information will typically include

- information about the world and other agents, modeled as an agent's *beliefs*
- information about which actions are available to the agent, modeled as the agent's *capabilities*
- information about what the agent's actions are meant to accomplish, modeled variously as the agent's *plans, intentions, commitments* or *goals*

This paper reworks the agent model presented in (Shoham 1993), which presents both a high-level model of *agent-oriented programming* (AOP) and a simple AOP language, called AGENT0. AGENT0 agents have beliefs, abilities to perform actions (which may be relative to what is true in the world), and commitments made to other agents or to themselves concerning actions to be taken at particular future times. Unlike Shoham's original formulation, which used a standard modal logic semantics, this new formalization uses domain theory to provide the semantic basis for the AGENT0 language.

We believe that it will be productive to move from a modal logic-based semantic framework to a domain-theoretic framework for agent modeling because of the greater generality afforded by this new framework. By the nature of the standard possible-worlds semantics for modal logics, agent models that use these logics suffer from well-known problems, most importantly the problem of logical omniscience. In addition, Shoham's original work and the later work of others, e.g. (Thomas 1993, Thomas 1999), modeled only the static mental state of agents. The dynamic evolution of an agent's mental state was prescribed by the agent's program, but no formal model of this evolution was given as part of the work on AOP.

This paper may frustrate mathematicians, since we are not breaking new ground in category theory. It may frustrate researchers in artificial intelligence, since the use of category theory allows great generality of results. We believe, however, that category theory can usefully be applied to the problem of specifying agent semantics, and that the semantic framework so derived has real advantages over the more standard models.

2. Review of AGENT0

In the AGENT0 AOP scheme, the mental states of agents consist of beliefs, abilities, obligations, and decisions. The beliefs are typical of most agent models; they represent the information that an agent has available for its reasoning. Abilities, in AGENT0, refer to facts, rather than to actions. The intuition is that an action will make some fact true, and that typically that is the point of the action. These abilities may be conditional on certain facts about the world; for example, an agent may be capable of opening a door if that door is unlocked, and not otherwise. (AGENT0 also introduces the notion of a primitive action, an action which cannot be broken down into component piece.) Obligations, which were called commitments in earlier versions of AGENT0, are essentially promises made to some agent that some fact will be made true. Decisions (earlier called choices) are simply obligations to oneself.

An AGENT0 program specifies (1) the (unchanging) set of capabilities of the agent, (2) the initial beliefs of that agent, and (3) a set of commitment rules, which specify the conditions under which the agent should make new commitments. The commitment rules specify a condition which must be true of the agent's mental state, and a condition which must be true of some newly-received message. If both conditions are met, then the obligation(s) listed in the rule are all established and must be carried out by the agent.

In our model, we have made some minor changes. We have replaced the idea of obligations with non-social commitments, that is, with commitments that are not made to any particular agent. (While it is useful to have social commitments, we are confident that only a small change will be required to our model to incorporate this notion.) We divide commitments into two types, simple commitments, in which an agent agrees to establish some fact at a given time, and conditional commitments, in which the agent agrees that it will establish the given fact at the given time if some condition is then true. For example, our agent may promise to open the door to the seminar room at 3:00pm, if the door is then unlocked.

3. Intuitive example

While the mathematics underlying our model will be presented in detail below, it seems appropriate to give some intuitive description in order to illustrate our approach.

We model our agent's mental state with a collection of elements, which are arranged to form a complete partial order (see Figure 1). Each element is a potential belief state of the agent; that is, it can be seen as a collection of facts. Some elements of the partial order correspond to messages received by the agent; the contents of a message can be seen as a collection of facts. One element is singled out as the initial belief state of the agent. Whenever the agent receives a message, it moves to a new belief state. This new belief state is found by taking the least upper bound of the current belief state and the just-received message, subject to consistency constraints.

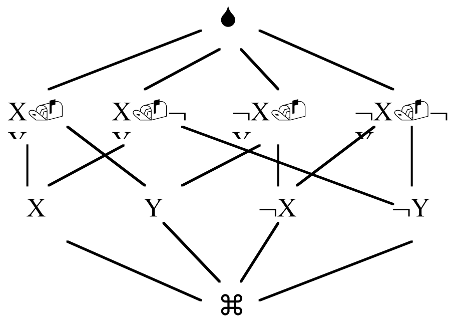

Figure 1

In Figure 1, let the f state be [X]. If the agent receives a message with content [Y], its new belief state will be the least upper bound of [X] and [Y], which is in this case [X📫Y]. Should the agent next receive the message [X], its new belief state will remain [X📫Y]. Thus this diagram captures the evolution of the agent's beliefs in response to messages it might receive.

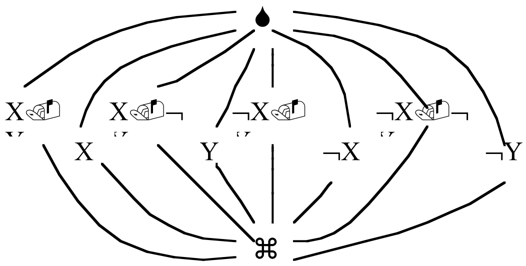

Figure 2

Let us examine anot set of elements, which will give us different results. Figure 2 shows the mental state of a nonlearning agent. Starting from belief state "⊥" this agent can learn one thing. Any further learning will put it into an inconsistent state.

Obviously, both Figure 1 and Figure 2 describe very simple agents. This same framework is very general and can capture many different styles of agent reasoning.

4. A Mathematical Structure for Defining the Semantics of Agents

When investigating what mathematical structure would be appropriate to model both the static and dynamic nature of an agent we initially tried utilizing Kripke diagrams. These proved to be adequate for modeling the static nature of an agent but we had difficulty in dealing with the agent's dynamic nature. We briefly investigated Montague semantics, which avoid some problems of Kripke models by weakening the assumed reasoning power of agents, but found them equally problematic in terms of representing the dynamic mental state. We next turned to category theory, specifically to a fairly general category that is often used in semantic models, namely a Cartesian closed category (MacLane 1971). This proved to be adequate to model both the static and dynamic nature but had the disadvantage of being too general. There were aspects of a Cartesian closed category that did not correspond to any characteristic of an agent within our denotation. We next honed in on what aspects of these categories were in fact needed for our model to work. What we concluded was that a form of a Scott Domain (Scott 1982**),** namely, a *consistently complete ω-algebraic complete partial order*, henceforth called simply a domain, was what was needed.

Before we proceed with the details of our model, let us first enumerate a few of the properties of the category of domains that will play a prominent role in our model:

1. this category is in fact Cartesian closed and so **terminal elements**, **products**, and **function types** are available.
2. the function **U**:Dx$D \rightarrow D$, which maps two elements to their least upper bound, is continuous.

.

3. every domain D contains a set of elements $\texttt{fin}(D)$, called the **finite elements**, which satisfy the property that if $x \in \texttt{fin}(D)$, $M \subseteq D$ is a directed set, and $x \leq \bigcup M$, then $x \leq y$ for some $y \in M$. (Winskel & Larsen 1984)
4. for each element $x \in D$ we define $\uparrow x = \{a \in \texttt{fin}(D) \mid a \leq x\}$ and have that $\uparrow x$ is directed and $\bigcup(\uparrow x) = x$, so in fact a domain is determined by its finite elements.
5. Given a function $f{:}\texttt{fin}(D) \rightarrow D$ that is monotonic, it can always be extended to a function $\hat{f}{:}D \rightarrow D$ that is continuous. This tells us that continuous functions are completely characterized by their actions on the finite elements.
6. Each domain D has a bottom element $\perp$. We will also require the existence of a top element ♠.

4.1 The Semantic Domains of Agents

In order to define our model we will need to introduce the following domains:

- **Time $\mathtt{T}$** = $\mathtt{N}^{\infty}$, which is the domain of natural numbers with an element ∞ attached to the top of the order, or $\mathtt{T} = \underline{\mathbf{n}}$, which is the domain of natural numbers up to and including n. On each of these domains is defined a continuous function *next*: $\mathtt{T} \rightarrow \mathtt{T}$ which maps each element to the next element in the partial order.[1]
- **Domain of Discourse** $\mathtt{D}$. This domain models the world of all the agents and will usually be, but not required to be, defined using a predicate logic over a basis in which the ordering is based on implication. Each element ϕ in this domain would probably be a consistently complete set of predicates with $\phi \leq \varphi$ iff all models of ϕ are also models of φ.
- **Negation**. $\neg$**:** $\mathtt{D} \rightarrow \mathtt{D}$, an idempotent operator satisfying $\bigcup(\neg\phi,\phi)$ **= ♠** .
- **Abilities** $\mathtt{A}_a \subseteq \mathtt{D}$. These domains, one for each agent, will be subdomains of the domain $\mathtt{D}$ and represent the abilities of an individual agent. Since the abilities of individual agents differ, we cannot utilize $\mathtt{D}$ to model the abilities of all agents. (Note that we follow Shoham in defining abilities to relate to facts rather than actions.)
- **Primitives** $\mathtt{P}_a = \texttt{fin}(\mathtt{A}_a)$. The primitives correspond to the primitive actions (facts) associated with a particular agent. All abilities in $\mathtt{A}_a$ are determined by these primitives as discussed in (4) above.
- **Beliefs** $\mathtt{B}_a$ **:**$\mathtt{T} \rightarrow \mathtt{A}_a$, correspond to the beliefs of an agent represented as a continuous function of time. Since continuous functions are monotonic, $\mathtt{B}_a$ will be monotonic with respect to the ordering on $\mathtt{T}$ and $\mathtt{A}_a$. This function captures a part of the dynamic nature of an agent.
- **Simple Commitments** $\mathtt{S}_a$**:** $\mathtt{T} \rightarrow \mathtt{A}_a$**.** These are the commitments that an agent has made represented as a continuous function of time.
- **Conditional Commitments** $\mathtt{C}_a$**:** $\mathtt{T} \rightarrow (\ \mathtt{P}_a \ \rightarrow \mathtt{A}_a)$. These are the conditional commitments that an agent has made represented as a continuous function of time. The condition associated with a conditional commitment must be a primitive.

4.2 The Semantic Functions

Initialization: When an agent is specified, an initialization element $\phi \in \mathtt{A}_a$ is given. This element specifies the initial beliefs of the agent and thus defines an initial value for $\mathtt{B}_a$ given by the formula $\mathtt{B}_a(t) = \phi$ for all $t \in \mathtt{T}$[2]. Also specified at this time are a finite number of initial commitments, both simple and conditional. These result in initial values for the functions $\mathtt{S}_a$ and $\mathtt{C}_a$ with the restrictions $\mathtt{S}_a(t) = \mathtt{S}_a(next(t))$ for $t \geq t_0$ for some $t_0 \in \mathtt{T}$ and $\mathtt{C}_a(t) = \mathtt{C}_a(next(t))$ for $t \geq t_1$ for some $t_1 \in \mathtt{T}_a$. This condition ensures that only a finite number of commitments were made at initialization.

Effects of Commitments on Beliefs: As time passes, an agent's beliefs changes based upon information it receives from other agents, and based upon commitments made by the agent to itself and to other agents. These commitments also cause actions to be performed by the agent as a result of "side effects".

The changes in the beliefs of an agent due to its simple commitments at time *next*(*t*), where $t \in \mathtt{T}$, is described by the formula

$$\mathtt{B}_a(next(t)) = \mathtt{B}_a(t) \bigcup \mathtt{S}_a(t).$$

This formula correspond to the following commutative diagram:

$$\begin{tikzcd}
\mathtt{T} \arrow[r, "next"] \arrow[d, "\mathtt{B}_a \mathtt{x} \mathtt{S}_a"'] & \mathtt{T} \arrow[d, "\mathtt{B}_a"] \\
\mathtt{A}_a \mathtt{x}\ \mathtt{A}_a \arrow[r, phantom, "\bigcup"] & \mathtt{A}_a
\end{tikzcd}$$

[1] . We have investigated models which included more complex domains of time which would allow for each agent to function on a different time scale $\mathtt{T}_a$. This would require the existence of a functor $\boldsymbol{F}_{ab}{:}\mathtt{T}_a \rightarrow \mathtt{T}_b$, for pairs of agents **a** and **b**. We do not discuss this more elaborate model of time in this paper.

[2] Since $\mathtt{B}_a$ is required to be continuous, and thus monotonic, we cannot simply define $\mathtt{B}_a(0) = \phi$.

.

In the case where our domain corresponds to some predicate logic, the above asserts that at time *next(t)*, what we believe is what we believed at time *t* anded with what we were committed to believe at time *t*. These changes also result in primitive actions being performed by the agent. We specify these actions as side effects of the primitive elements in our domains. Thus at time *next*(*t*) an agent performs the actions:

$$\{\mathbf{Do}\ p \in P_a \mid \quad p \leq S_a(t)\ \&\ p \text{ 🔔 } B_a(t)\ \}$$

Thus, if an agent commits to believing at time t that it has just performed the open-door action, then at time t it will add such a belief to its knowledge base and will perform the open-door action, unless the door is open already.

Similarly, the changes in the beliefs of an agent due to its conditional commitments at time *next*(*t*), where $t \in T$, is described by the formula

$$B_a(next(t)) = B_a(t) \cup C_a(t)(\ B_a(t)).$$

Note here that the commitments that are acted upon are the ones determined by both the time *t* and the current beliefs $B_a(t\ )$. This formula corresponds to the following commutative diagram:

$$\begin{tikzcd}
T \arrow[rr, "next"] \arrow[d, "B_a\ \times C_a \times B_a"] & & T \arrow[d, "B_a"] \\
A_a \times (A_a \to A_a) \times A_a \arrow[r, "id \times apply"'] & A_a \times A_a \arrow[r] & A_a \arrow[r, "\cup"'] & {}
\end{tikzcd}$$

These changes cause the following primitive actions to be performed by the agent

$$\{\mathbf{Do}\ p \in P_a \mid \quad p \leq C_a(t)(\ B_a(t))\ \&\ p \text{ 🔔 } B_a(t)\}\ .$$

Incoming Messages: When a message is sent to an agent it will be intercepted by the system interface and converted to an appropriate form for processing by the agent. The messages that are received are of two forms: **Informational** and **Requests**.

Informational messages are simply informing agent$_a$ of some fact ϕ. The agent will then, depending on the time t_0 when the message is received, adjust its beliefs accordingly. This adjustment is represented by the formula

$$B_a(t) = B_a(t) \cup \phi \text{ for } t \geq t_0.$$

Requests are messages requesting agent$_a$ to commit to some fact ϕ at a time t_0. This request may be conditional on some primitive fact p being true at the time. Again the system interface intercepts these messages and converts them to an appropriate form for processing by the agent. In the case of a simple request (t_0, ϕ) the adjustment is represented by the formula

$$S_a(t) = S_a(t) \cup \phi \text{ for } t \geq t_0,$$

and for a conditional request (t_0, (p, ϕ)) the adjustment is represented by the formula

$$C_a(t) = C_a(t) \cup \hat{f}_{(p,\phi)} \text{ for } t \geq t_0,$$

where we have $f_{(p,\phi)}$: $P_a \to A_a$ is defined by

$$f_{(p,\phi)}(q) \ = \begin{cases} \phi & \text{for } q \geq p. \\ \perp & \text{otherwise.} \end{cases}$$

The function $\hat{f}_{(p,\phi)}$: $A_a \to A_a$ is the continuous extension of $f_{(p,\phi)}$: $P_a \to A_a$ that was discussed in (5) above.

Informational Relevancy: In the scenario described above an agent did not examine the content of a fact ϕ to insure that its content was relevant to the domain of the agent. This scenario thus assumes that each agent knows the abilities of the other agents, or at least the ones for which it is interested in communicating with. If an agent does not know about the abilities of another agent, it really has no vocabulary for even representing any information about these unknown abilities. In such cases, we can always adjust any incoming messages to fit within the domain of abilities of the particular agent. So for example, if agent *a* receives a fact ϕ from agent$_b$, the system can then pass on to the agent$_a$ simply the fact $\phi' = \bigcup\{p \in P_a \mid p \in \uparrow\phi\}$, which of course may simply be $\perp$.

Beliefs: In our world of agents we are interested in what our agents "believe" at some time *t*. This of course is determined by investigating $B_a(t)$. To determine whether agent$_a$ believes fact ϕ at time *t*, we simply determine if $\phi \leq B_a(t)$. If it is, we say that agent$_a$ believes ϕ at time *t* and write $B_a(t)(\phi)$. Since we require that $\bigcup (\neg\phi, \phi) =$ 🌢, if agent$_a$ believes both $\neg\phi$ and ϕ at time *t*, then agent$_a$ believes 🌢 and thus believes all things. We will refer to this as agent$_a$ being in a state of ***inconsistency***. What about the question of what agent$_a$ does not believe?

Definition: We define $\neg B_a : T \to PowerSet(P_a)$ where $\neg B_a(t) = \{p \in P_a \mid p \not\in B_a(t)\ \}$. We then say that agent$_a$ **does not believe** fact ϕ at time *t* iff $(\uparrow\phi \cap \neg B_a(t)) \neq \varnothing$ and will write $\neg B_a(t)(\phi)$.

Lemma: If agent$_a$ is in a consistent state, then we have that $B_a(t)(\ \neg\phi)$ implies that $\neg B_a(t)(\phi)$.

Consistency. We have not imposed any consistency constraints on our agents' beliefs. Thus it is possible for an agent to believe both ϕ and $\neg\phi$, and hence be in a state of inconsistency. This may be an undesirable situation. We can however, easily remedy this by imposing a few consistency constraints. For instance we can impose

1. Any initialization element ϕ be consistent, so $\phi \neq$ 🌢.
2. For commitments we would have

$$\begin{cases} B_a(t) \cup S_a(t) & \text{if } B_a(t) \cup S_a(t) \neq \text{🌢} \\ & \end{cases}$$

.

$\mathrm{B}_a(next(t)) =$

$\mathrm{B}_a(t)$ otherwise

3. For conditional commitments we would have

$$\mathrm{B}_a(next(t)) = \begin{cases} \mathrm{B}_a(t) \cup \mathrm{C}_a(t)(\mathrm{B}_a(t)) & \text{if } \mathrm{B}_a(t) \cup \mathrm{C}_a(t)(\mathrm{B}_a(t)) \neq \blacklozenge \\ \mathrm{B}_a(t) & \text{otherwise} \end{cases}$$

We could similarly impose consistency constraints on the incoming messages processed by our agent.

5. Example

Let us present an example in which an agent is to find its way out of a maze. The agent will be programmed to use the algorithm in which it figuratively places its right hand on the wall and moves in such a way as to keep contact with the wall at all times. That is, if the agent can turn right, it will; if it cannot turn right but can go straight, it will. If it cannot turn right or go straight but can turn left, it will, and if it can do none of these things, it will turn around and return to its previous position (but now facing the other way).

We will model this example using a modus ponens domain. In this structure, we have a set of predicates. We will not need conjunction or disjunction for this example, but we will need implication. An element of the domain is just a set of well-formed formulas (wffs), where each wff is either a predicate or an implication.

We'll assume that our agent has sensors that allow it to recognize whether there is a wall in front of it, on its left, on its right, and behind it. We will encode this information using a 5-ary state predicate S(R, F, L, B, T). The first four terms of the predicate are Boolean; they are 1 if there is a wall respectively to the right of, in front of, to the left of, or behind the agent. The final term, T, is a time. So S(1,0,0,1,15) will be interpreted to mean that at time 15, the agent had walls to its right and behind it, but not to its left or in front of it. We will have four more predicates: MR(T) means that the agent turns to its right and moves one unit forward at time T. Similarly, $MF(T_1)$, $ML(T_2)$, and $MB(T_3)$ mean that the agent moves forward, left, and back at the given times. These predicates have side effects; if the agent adopts a belief in, say, MR(T) then it will execute the corresponding action at time T. (Note that the agent can only adopt this belief if it is consistent with the agent's other beliefs.)

Given this language, we can write rules that will allow the agent to execute the algorithm described above. Note that although our language for this example does not allow quantifiers, we will use them as an abbreviation. In fact, each element in the domain will have an infinite number of implication instances corresponding to each rule given here.

$$\forall X,Y,Z,T\ \mathrm{B}_a\,(T)(S(0,X,Y,Z,T)) \Rightarrow MR(T)$$
$$\forall Y,Z,T\ \mathrm{B}_a\,(T)(S(1,0,Y,Z,T)) \Rightarrow MF(T)$$
$$\forall Z,T\ \mathrm{B}_a\,(T)(S(1,1,0,Z,T)) \Rightarrow ML(T)$$
$$\forall T\ \mathrm{B}_a\,(T)(S(1,1,1,0,T)) \Rightarrow MB(T)$$

Of course, our semantic framework is more general than the maze example demonstrates. As a second example, we present the standard nonmonotonic reasoning problem concerning whether Tweety the bird can or cannot fly. We can draw a simple domain for this reasoning problem; see Figure 3.

The initialization element for this example is {bird(Tweety)→flies(Tweety), dead(Tweety)→ ¬flies(Tweety)}; these are the only things the agent believes when it first begins operation. Note that the → symbol now represents defeasible implication. Should it now learn that Tweety is a bird, it will conclude that Tweety can fly. If instead it learns that Tweety is dead, it will conclude that Tweety cannot fly. Should it learn that Tweety is both dead and a bird, it will still conclude that Tweety cannot fly, regardless of the order in which it learns these two facts.

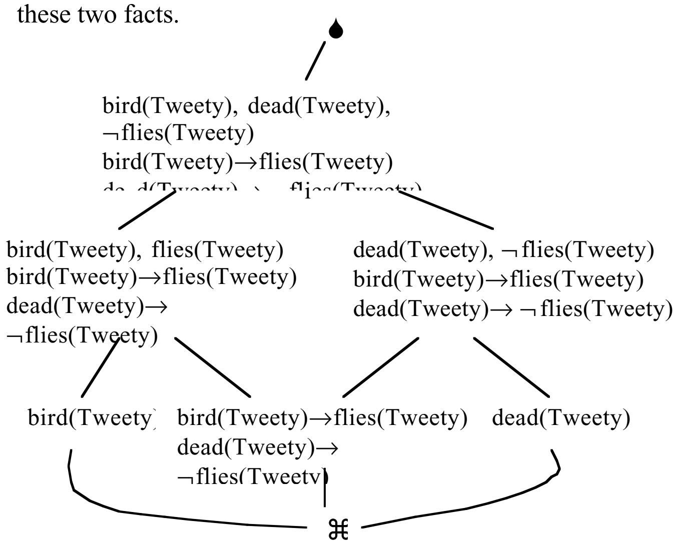
♦
bird(Tweety), dead(Tweety),
¬flies(Tweety)
bird(Tweety)→flies(Tweety)
bird(Tweety), flies(Tweety)
bird(Tweety)→flies(Tweety)
dead(Tweety)→
¬flies(Tweety)
dead(Tweety), ¬ flies(Tweety)
bird(Tweety)→flies(Tweety)
dead(Tweety)→ ¬ flies(Tweety)
bird(Tweety)
bird(Tweety)→flies(Tweety)
dead(Tweety)→
¬flies(Tweety)
dead(Tweety)
⌘

Figure 3

6. Discussion

As a result of the semantic structure given above, an agent's beliefs can change through receiving messages and through side effects of commitments and conditional commitments. It may seem unintuitive that an agent should be able to make a commitment to believe something in the future. Often, however, the agent will make a commitment to believe that it took some action at a given time; as a side effect, then, the agent will take the action. The agent may have information about the effects of its actions, but it need not conclude that its actions are

.

always successful. For example, an agent program may specify that an agent commit to believing at 4:00pm that at 3:59pm it performed an open-door action on the door of the seminar room. Its program may further specify that it should always believe that such an action entails the door being open afterward, or its program may instead specify that once the action is performed, the agent should adopt a belief that the door is open only if its sensors confirm that fact. Either approach is consistent with the AOP paradigm, since Shoham purposely did not specify how agents should reason about the results of their actions.

AOP was intended to be a very general approach to programming agents. We believe that we have further generalized the approach by replacing the Kripke-style semantic structures used by Shoham (Shoham 1993) and Thomas (Thomas 1993, Thomas 1999) with a Scott Domain. Within this framework, the user is free to specify that an agent should be a perfect reasoner, but the user is equally free to specify that an agent is incapable of reasoning at all, or can perform only certain kinds of reasoning. Such reasoning abilities are encoded in the structure of the domain, specifically in the elements and the function **U** (which maps two elements to their least upper bound).

7. Future Work

In this paper we have introduced two fairly conventional forms of commitments, namely $\mathtt{S}_{\mathbf{a}}\mathbf{:}\ \mathtt{T}\rightarrow\mathtt{A}_{\mathbf{a}}$ and $\mathtt{C}_{\mathbf{a}}\mathbf{:}\ \mathtt{T}\rightarrow(\ \mathtt{A}_{\mathbf{a}}\rightarrow\mathtt{A}_{\mathbf{a}})$. However, due to the fact that our category of domains is Cartesian closed, we can easily introduce more complex versions of commitments. For instance we could define a recursive form of commitment that would allow us to commit to at time t_0 to commit to at time t_1 to believe in fact ϕ. This could be accomplished by defining a domain of high level commitments $\mathtt{H}_{\mathrm{a}} = \mathtt{S}_{\mathbf{a}} + \mathtt{C}_{\mathbf{a}} + \mathtt{H}_{\mathrm{a}}$. In our current investigations we are exploring these high level commitments as well as investigating the use of a more complex domain of time $\mathtt{T}_{\mathrm{a}}$.

One problem with our model as it stands is that we have not discussed nested belief operators; that is, we have no way of representing what an agent believes about its own beliefs or the beliefs of other agents. Such nesting of belief operators is crucial in many domains, and we are currently investigating extensions to our work that will allow such nesting.

References

(Apt 1981) Apt, K.R. “Ten Years of Hoare's Logic: A Survey --Part I,” *ACM Transactions on Programming Languages and Systems*, 3: 4, ACM, 1981.

(Attardi & Simi 1981) Attardi, A. and Simi, M. “Semantics of Inheritance and Attributes in the Description System Omega,” MIT A.I. Memo 642, 1981.

(Cardelli 1988) Cardelli, L. “A Semantics of Multiple Inheritance,” *Information and Computation*, number 76, pages 138-164. Academic Press, 1988.

(Goguen et al, 1973) Goguen, J. A., Thatcher, J. W., Wagner, E. G., and Wright, J. “A Junction Between Computer Science and Category Theory, I: Basic Concepts and Example(Part I).” IBM Research Report, 1973.

(Goguen et al, 1976) Goguen, J. A., Thatcher, J. W., Wagner, E. G., and Wright, J. “An Introduction to Categories, Algebraic Theories, and Algebras.” IBM Research Report, 1976.

(Goguen et al, 1977) Goguen, J. A., Thatcher, J. W., Wagner, E. G., and Wright, J. “A Junction Between Computer Science and Category Theory, I: Basic Concepts and Example(Part II).” IBM Research Report, 1977.

(Hoare 1969) Hoare, C. A. R. “An Axiomatic Basic for Computer Programming,” *Communications of the ACM*, 12:10, 1969.

(Liskov & Guttag 1986) Liskov, B. H. and Guttag, J. *Abstraction and Specification in Program Development*, MIT Press, Cambridge, 1986.

(MacLane 1971) Mac Lane, S. *Categories for the Working Mathematician*, Graduate Texts in Mathematics, Springer-Verlag, New York, 1971.

(Norton 1987) Norton, R. L. “Information Systems as Implementations,” Proceedings of the 15th Annual Computer Science Conference, St. Louis, Missouri, 1987.

(Norton 1989) Norton, R. L. “Predicate Transformers as Homomorphisms on Scott's Information Systems,” Proceedings of the 17th Annual Computer Science Conference, Louisville, Kentucky, 1989.

(Norton 1991) Norton, R. L. “A Categorical Treatment of Classes in an Object-Oriented Language,” Marist Working Papers, V.7, 1991.

(Norton 1991) Norton, R. L. “A Categorical Treatment of Classes and Multiple Inheritance in an Object-Oriented Programming Language,” Dissertation, Syracuse University, 1991.

(Rao & Georgeff 1991) Rao, A. S., and Georgeff, M. P. “Modeling rational agents within a BDI-architecture”. In J. Allen, R. Fikes, and E. Sandewall, eds, Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference, pp. 473-484, 1991.

(Scott 1982) Scott, D. S. “Domains for Denotational Semantics,” in M. Nielsen and E. M. Schmidt, eds., International Colloquium on Automata, Languages and Programs, Volume 140 of Lecture Notes in Computer Science, Springer, 1982.

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

(Thomas 1993) Thomas, S. R., PLACA, An Agent Oriented Programming Language. Ph.D. Thesis, Stanford University, 1993. Available as Stanford University Computer Science Department Technical Report STAN-CS-93-1387.

(Thomas 1999) Thomas, S. R., “A Survey of Agent-Oriented Programming,” in M. Wooldridge and A. Rao, eds, Foundations of Rational Agency, Volume 14 of Applied Logic Series, Kluwer Academic Publishers, pp.263-274, 1999.

(Wagner 1981) Wagner, E. G. “Lecture Notes on the Algebraic Specification of Data Types"“, Proceedings of the Seventh National School for Young Scientists, 1981.

(Winskel & Larsen 1984) Winskel, G. and Larsen, K. G. “Using Information Systems to Solve Recursive Domain Equations Effectively,” Technical Report, University of Cambridge Laboratory, 1984.

(Wooldridge & Jennings 1994) Wooldridge, M. J., and Jennings, N. R., eds., Intelligent Agents: ECAI-94 Workshop on Agent Theories, Architectures, and Languages, Volume 890 of Lecture Notes in Artificial Intelligence, Springer-Verlag, 1994.

(Wooldridge, Müller, & Tambe 1995) Wooldridge, M. J., Müller, J. P., and Tambe, M., eds., Intelligent Agents II: Agent Theories, Architectures, and Languages, Volume 1037 of Lecture Notes in Artificial Intelligence, Springer-Verlag, 1995.

(Wooldridge & Rao 1999) Wooldridge, M., and Rao, A., eds., Foundations of Rational Agency, Volume 14 of Applied Logic Series, Kluwer Academic Publishers, 1999.

.