CHAPTER 10:
Temporal Belief Logics for Modelling Distributed Artificial Intelligence Systems

Michael Wooldridge

Department of Computing
Manchester Metropolitan University
Chester Street
Manchester M1 5GD
United Kingdom

EMAIL M.Wooldridge@doc.mmu.ac.uk

TEL (+44 61) 247 1531

FAX (+44 61) 247 1483

Abstract

In this article, we look at formal methods for reasoning about multi-agent systems, with particular emphasis on the twin problems of *specifying* and *verifying* such systems. We begin by critically reviewing several types of formalism, from both AI and mainstream computer science, which suggest themselves as possible candidates for the task. This review leads us to conclude that none of these formalisms are ideal, but that a hybrid approach might be suitable. The method we propose involves constructing a model of multi-agent systems, and then defining an execution model, which describes how agents may act and interact. As execution proceeds in the manner defined by the execution model, a system traces out a history which may be used as the basis for a *temporal belief logic*. This logic can then be used to reason about the systems described by the model of multi-agent systems. We outline the development of a model of agents and multi-agent systems, and give an example of a temporal belief logic developed to reason about this model. We then discuss the use of the logic and point to future research topics.

10.1 Introduction

This article discusses the use of *formal methods* for reasoning about DAI systems. By formal, we really mean *mathematical*, and in particular, this article is concerned with using *logics* for reasoning about multi-agent systems. Now the use of formal methods to reason about computer systems is not universally accepted as both worthwhile and practicable, even in mainstream computer science, (where such techniques have been the object of study since the late 1960s). The article therefore begins, in the following section, by motivating the work. Following this, a critical review is presented of three types of formalism that suggest themselves as being appropriate to the task. These formalisms are: (i) intentional logics, developed by researchers in AI and philosophy to describe systems with beliefs, goals, intentions, and so on; (ii) temporal logics, developed by researchers in computer science for reasoning about 'reactive' systems; and (iii) process algebras, developed for modelling concurrent activity. The review concludes that none of these types of formalism are ideal for the task. However, it is suggested that a *hybrid* approach might be suitable. Specifically, the article proposes the following three-stage research programme:

1. Develop a formal model of the type of multi-agent system you wish to reason about;

2. For the model developed in (1), construct an *execution model*, which states formally how the execution of such a system may proceed;

3. Use the *histories* traced out in the execution of a system as defined in (2) as the semantic basis for a logic, which can then be used to represent and reason about the systems modelled in (1).

In section 10.3, we give an example, worked in as much detail as space allows, to illustrate this programme; this section results in the definition of a *temporal belief logic* called AL, which, it is claimed, can be used to express the properties of multi-agent systems. The use of this logic is discussed in section 10.4, and some conclusions are presented in section 10.5.

10.1.1 Motivation

There are a number of reasons why one should study formal methods for multi-agent systems. The first, and most obvious reason, is that they might provide a tool to help

manage the complexity of such systems. Such a tool is highly desirable, as designing, implementing and debugging multi-agent systems is not easy: in a 1987 article, Gasser *et al.* observed that

> 'Concurrency, problem domain uncertainty and non-determinism in execution together conspire to make it very difficult to understand the activity within a distributed intelligent system'. (Gasser et al., 1987, p148)

To counter such problems, the authors advocated the development of

> '…graphic displays of system activity linked to intelligent model based tools which help a developer reason about expected and observed behaviour'. (Gasser et al., 1987, p148)

The work described by Gasser and colleagues is firmly and unashamedly in the experimental tradition of AI: hence the call for practical tools to aid understanding. However, given the software experience of the past two decades, it seems surprising that no similar plea was made for principled techniques for specifying and verifying multi-agent systems.

In addition to providing a tool to help understand and manage multi-agent systems, it is to be expected that a suitable formalism would provide a framework in which general questions about cooperation and social interaction might be posed, and solutions developed. For example, a number of theories of social activity have been developed by workers in DAI, of which probably the best-known is the Levesque-Cohen model of joint intentions (Levesque et al., 1990). And yet such theories are typically expressed in formalisms that cannot be related to 'real' DAI systems, as they make unreasonable assumptions about the deductive capabilities of agents. So, more realistic formalisms for representing multi-agent systems might be used as the basis of more pragmatic theories of social activity.

10.2 Background

If one aims to develop formal methods for reasoning about multi-agent systems, a good place to start is by observing how the mainstream AI/DAI community has gone about building intelligent (social) agents. Unfortunately, one immediately runs into difficulties, as the issue of 'intelligent agent architecture' is the subject of a somewhat

heated ongoing debate in AI. Interesting and important though this debate is, it is not the aim of this article to become embroiled in it (see, e.g., (Brooks, 1991) for one view). Despite the intense interest — and controversy — that alternative approaches have evoked, the majority of work in DAI lies in the so-called 'classical', or 'deliberative' camp, and it is on such work that this article focusses. Briefly, this camp proposes that individual agents are symbolic AI systems in the classical sense, with some 'knowledge', (expressed in a symbolic language), some reasoning ability, and so on. Examples of classical approaches to DAI are AGENT0 (Shoham, 1993), Concurrent METATEM (Fisher and Wooldridge, 1993a), MACE (Gasser et al., 1987), and MCS/IPEM (Doran et al., 1991).

How is one to go about reasoning about such systems? What techniques are appropriate, and/or available for the task? There are three resources which initially suggest themselves as suitable:

- *Intentional logics.*

  Researchers in AI, philosophy, and economics have developed many logics for representing and reasoning about the so-called *intentional* notions: belief, desire, and so on. If it is accepted that the agents we wish to reason about may be described in such terms (see, e.g., (Shoham, 1993)), then one might use such a logic to reason about them.

- *Temporal logics.*

  Researchers in mainstream computer science have for some time used temporal logics for reasoning about *reactive* systems, of which DAI systems are a subset (see below). Therefore, some variant of temporal logic might be suitable for reasoning about DAI systems.

- *Process algebras.*

  In order to model concurrency, a variety of process algebras have been developed; since DAI systems are by definition concurrent, it may be that some formalism of this type may be used.

In the remainder of this section, we discuss these formalisms in more detail.

Intentional Logics

There is a well established tradition in AI/philosophy of devising logics of the mentalistic, *intentional* notions: belief, knowledge, intention, and so on (Hintikka, 1962; Moore, 1985; Konolige, 1986a; Cohen and Levesque, 1990). Such logics identify an agent with an *intentional system*:

> '*intentional systems* ... [are] entities whose behaviour can be predicted by the method of attributing beliefs, desires, and rational acumen ...' (Dennett, 1987, p49)

The prevalent method for defining the semantics of intentional logics has been to give them a *possible worlds* interpretation (Halpern and Moses, 1992). Possible worlds semantics have the advantage of a well established, theoretically attractive foundation, (see, e.g., (Chellas, 1980)), but suffer from several disadvantages. Chief among these is the famous 'logical omniscience' problem, which seems to imply that agents are perfect reasoners. Another difficulty is that unless the worlds in the semantics are in some way 'grounded', (i.e., given some concrete interpretation in terms of agents), then they must remain a theoretical nicety. Given that standard possible worlds semantics are not suited to the task of reasoning about multi-agent systems, one seems to be faced by three options. If one wishes to retain possible worlds semantics, (they are, after all, highly attractive from a theoretical point of view), then one might attempt to find some way of *grounding* the possible worlds. The second option is to try to weaken the possible worlds model in some way. The final option is to reject possible worlds altogether, and seek an alternative semantic base.

Epistemic logics with grounded possible worlds semantics have recently become the object of study for researchers in mainstream computer science, who found that the notion of knowledge is a valuable one for analyzing distributed systems and protocols (see (Halpern, 1987) for an overview). However, it is not clear how these approaches are related to the classical model of agents — if at all. So, while distributed systems models of knowledge are an important research topic in their own right, they are, at the moment, only of tangential interest to DAI.

Another possibility is to weaken possible worlds semantics in some way; Levesque suggested one way of doing this, (Levesque, 1984), by borrowing some ideas from situation semantics (Barwise and Perry, 1983). Another scheme is described in (Fagin and Halpern, 1985). However, these methods have been criticised for their essentially

ad hoc nature (Konolige, 1986b). Also, they suffer from one of the key problems of standard possible worlds approaches: they are not grounded.

Some researchers have rejected possible worlds altogether, and looked instead to the possibility of developing an alternative semantics. The best-known example of this work is the *deduction model of belief* developed by Kurt Konolige (Konolige, 1986a). The deduction model defines a belief system as a tuple containing a 'base set' of formulae in some internal, logical language of belief, together with a set of *deduction rules* for deriving new beliefs. Konolige argued that an agent with such a belief system could be said to believe something if it was possible to derive that thing from its base set using its deduction rules. Logically incomplete reasoning may be modelled by giving the agent logically incomplete deduction rules. Interestingly, the deduction model can be viewed as an abstract model of the beliefs of classical AI systems. For this reason, the deduction model seems to be the best candidate out of all the intentional logics mentioned so far for the purposes of this research.

Temporal Logics

The second resource that may possibly be used for reasoning about multi-agent systems is the Pnuelian tradition of using temporal logics to reason about *reactive systems*[1]:

> 'Reactive systems are systems that cannot adequately be described by the *relational* or *functional* view. The relational view regards programs as functions … from an initial state to a terminal state. Typically, the main role of reactive systems is to maintain an interaction with their environment, and therefore must be described (and specified) in terms of their on-going behaviour … [E]very concurrent system … must be studied by behavioural means. This is because each individual module in a concurrent system is a reactive subsystem, interacting with its own environment which consists of the other modules.' (Pnueli, 1986)

[1] There are at least three current usages of the term *reactive system* in computer science. The first, oldest usage is that by Pnueli and followers (see, e.g., (Pnueli, 1986), and the description above). Second, researchers in AI planning take a reactive system to be one that is capable of responding dynamically to changes in its environment — here the word 'reactive' is taken to be synonymous with 'responsive' (see, e.g., (Kaelbling, 1986)). More recently, the term has been taken to denote systems which respond directly to the world, rather than reason explicitly about it (see, e.g., (Connah and Wavish, 1990)). In this article the term is used in its Pnuelian sense.

There are good reasons for supposing that multi-agent systems of the type this article is interested in modeling are reactive:

- the applications for which a multi-agent approach seems well suited (e.g., air traffic control (Cammarata et al., 1983)) are typically non-terminating, and therefore cannot be described by the functional view;
- multi-agent systems are necessarily concurrent, and as Pnueli observes (above) each agent should therefore be considered a reactive system.

In a 1977 article, Pnueli proposed the use of temporal logic for reasoning about reactive systems (Pnueli, 1977). Much research effort has subsequently been devoted to investigating this possibility (see, e.g., (Emerson, 1990) for a good overview and references). Unfortunately, naive attempts to adapt such techniques to DAI seem doomed to failure, as Pnuelian models of concurrency typically deal with the execution of individual program instructions, a grain size too fine for our purposes.

Process Algebras

Process algebras are model-based formalisms developed by researchers in computer science to allow reasoning about concurrency in systems. Probably the best-known examples of process algebras are Milner's Calculus of Communicating Systems (CCS) (Milner, 1989) and its recent offspring, the π-calculus (Milner et al., 1992). In a formalism such as CCS, the behaviour of an agent[2] is defined as a set of equations; an algebra is provided for manipulating these equations, and so proving properties of agents. CCS and its relatives are the subject of much ongoing work in computer science, and together they form a rich family of techniques for modelling and understanding concurrency. However, the abstract properties of systems that we wish to reason about are often difficult to express using process algebras. Also, at the time of writing, process algebras are not in the mainstream of (D)AI research at all; in fact the author is aware of no published work which applies them to the problems of (D)AI. For these reasons, we do not consider CCS-like formalisms any further in this article.

[2]We stress that in process algebras, the term 'agent' is used in a much looser sense than it is used here.

Comments

None of the formalisms described above are directly applicable to our problem. However, there are elements of both intentional logics and temporal logics that are appealing. In an ideal formalism, a temporal component would seem useful to describe the reactive nature of DAI systems; and a model of belief such as Konolige's deduction model could be used to represent the beliefs that agents have. However, simply combining formalisms in an unprincipled way would not be helpful. Instead, we must be careful to develop a temporal belief logic and establish a precise relationship between that logic and the systems we wish to reason about. This leads us to the three-stage research programme mentioned earlier.

10.3 An Approach to Reasoning about Multi-Agent Systems

Recall the three stage programme of research we sketched out earlier: (i) define a formal model of the multi-agent systems about which we wish to reason; (ii) define an execution model for such systems, which describes how agents may act and interact; and (iii) use runs of such a system as a model for a temporal-based logic. This programme is illustrated, by means of a worked example, in this section, which concludes with the definition of a logic called AL. Although AL has superficial syntactic similarities to logics such as that described in (Cohen and Levesque, 1990), it is closely linked to the systems we are modelling. We can thus begin to realistically claim that we have a logic for reasoning about the kind of system we might actually build.

10.3.1 A Model of Multi-Agent Systems

In this section, we define a simple formal model that captures the following key properties of classical multi-agent systems:

- *Agents have names.* Each agent is uniquely identified by an *agent id*, drawn from the set Ag. We usually write i and j for agent ids.
- *Agents have beliefs.* As we observed above, it is reasonable to model the 'knowledge' of a classical AI system as a belief set, after (Konolige, 1986a). We assume for our model that agent's beliefs are expressed in some *internal* language L; this language could be a frame or semantic net language, but we shall suppose it is

a *logical language*. We write $Form(L)$ for the set of (well-formed) formulae of L, and we let BS be the set of possible belief sets that an agent could have:

$$BS \stackrel{\text{def}}{=} \wp(Form(L)).$$

- *Agents can perform actions*. The problem of modelling and reasoning about actions performed in the physical world is the subject of much ongoing work in AI. To avoid the problems inherent in any treatment of such actions, we shall assume that agents can only perform *private* actions; that is, they can only perform actions which operate on their own state. We let Ac be the set of all such actions.

- *Agents can send messages*. Although communication is not universally assumed in DAI, it is, nevertheless, a common assumption. We therefore suppose that agents can communicate by sending messages. A message is a triple $\langle i, j, \varphi \rangle$, where $i \in Ag$ is the *sender* of the message, $j \in Ag$ is the *recipient*, and $\varphi \in Form(L)$ is the message content. (We could assume a different communication language if required.) Let $Mess$ be the set of all messages:

  $$Mess \stackrel{\text{def}}{=} \{\langle i, j, \varphi \rangle \mid i, j \in Ag \text{ and } \varphi \in Form(L)\}.$$

  It is useful to assume a function rcv, with the signature

  $$rcv : Ag \times powerset(Mess) \rightarrow \wp(Mess)$$

  such that if $i \in Ag$ and $m \subseteq Mess$, then $rcv(i, m)$ is that subset of m in which i is the recipient.

We now define an agent to be a 4-tuple:

$$\langle \mathcal{B}^0, \mathcal{A}, \mathcal{M}, \mathcal{N} \rangle$$

where

- $\mathcal{B}^0 \in BS$ is the agent's *initial belief set*;
- $\mathcal{A} : BS \rightarrow Ac$ is the agent's *action function*;
- $\mathcal{M} : BS \rightarrow \wp(Mess)$ is the agent's *message generation function*;
- $\mathcal{N} : BS \times Ac \times \wp(Mess) \rightarrow BS$ is the agent's *next state function*.

The idea is that on the basis of its initial beliefs, an agent selects an action to perform using the function $\mathcal{A}$, and some messages to send, using the function $\mathcal{M}$. The function $\mathcal{N}$ then transforms the agent from one state to another, on the basis of the messages it receives, and action it has just performed.

A multi-agent system is then just an indexed set of such agents:

$$\{\langle \mathcal{B}_i^0, \mathcal{A}_i, \mathcal{M}_i, \mathcal{N}_i \rangle : i \in Ag\}.$$

We must now consider how the execution of such a system may proceed.

10.3.2 Execution Models

The execution model we present in this section hinges on the notion of the *state* of a system, and of changes in state being caused by *transitions*. Crudely, a state is a 'snapshot' of the belief set of every agent in the system at some moment in time. These belief sets are assumed to be in some kind of 'equilibrium' (cf. (Gärdenfors, 1988)). A state change, or transition, occurs when one or more agents receive some messages and perform actions.

Let us return to our model of multi-agent systems. The initial state of the system is given by every agent i having its initial belief set $\mathcal{B}_i^0$. The next state of agent i depends upon the action that i performed, the messages that were sent to it, and its initial state. Formally, the state $\mathcal{B}_i^1$ of agent i at time 1 is given by the following equation:

$$\mathcal{B}_i^1 \stackrel{\text{def}}{=} \mathcal{N}_i(\mathcal{B}_i^0, \mathcal{A}_i(\mathcal{B}_i^0), rcv(i, \bigcup_{j \in Ag} \mathcal{M}_j(\mathcal{B}_j^0))).$$

We can thus deduce the state of the system at time 1. This equation can easily be generalised to give the belief set of agent i for an arbitrary time $u \in \mathbb{N}$ such that $u > 0$:

$$\mathcal{B}_i^u \stackrel{\text{def}}{=} \mathcal{N}_i(\mathcal{B}_i^{u-1}, \mathcal{A}_i(\mathcal{B}_i^{u-1}), rcv(i, \bigcup_{j \in Ag} \mathcal{M}_j(\mathcal{B}_j^{u-1}))). \tag{10.1}$$

The execution of a system thus proceeds, according to this very simple execution model, with each agent picking an action to perform, sending messages, receiving messages, shifting into its next state, and so on. Note that this execution model does not really describe *concurrency* at all, as it assumes that agents act in synchrony (see section 10.5).

As it executes, a system traces out an *execution history*, which describes each agent's state, the actions it performed, and the messages it sent at each moment in time.

Without going into formal details, we shall assume that Σ is the set of all such execution histories, and we shall use σ to denote a member of this set. If we write s_u for the uth state of σ, and τ_u for the uth transition of σ, then we can visualise σ as follows.

$$\sigma : s_0 \xrightarrow{\tau_0} s_1 \xrightarrow{\tau_1} s_2 \xrightarrow{\tau_2} s_3 \xrightarrow{\tau_3} \cdots \xrightarrow{\tau_{u-1}} s_u \xrightarrow{\tau_u} \cdots$$

It is assumed that executions are non-terminating.

10.3.3 A Linear Time Temporal Belief Logic

This section introduces a logic for reasoning about multi-agent systems of the type described by the model developed above. The logic is called AL, which stands for 'Agent Logic'. The logic is propositional, in that it does not allow quantification[3], and it contains three atomic operators: Bel, for describing the beliefs of agents; Send, for describing the messages that agents send; and Do, for describing the actions that agents perform. Additionally, AL contains a set of modal temporal operators, which allow the description of the dynamic properties of agents. Finally, note that AL is based on a model of time that is *linear* (i.e., each moment in time is assumed to have just one successor), *bounded in the past* (i.e., there is a 'beginning of time'), and *infinite in the future* (i.e., there is no 'last moment' of time). For alternative versions of AL, based on a *branching* model of time, see (Wooldridge, 1992; Wooldridge and Fisher, 1992).

Syntax

AL is intended to allow reasoning about agents: their beliefs, their actions, and the messages they send. Since it isn't possible to actually put an action or agent directly into a formula of the language, there must be a way of referring to them. This is achieved by putting symbols into the language whose denotation is an agent identifier or action. Since quantification isn't allowed in AL, these symbols will be constants. Also, to express the beliefs of agents, the internal language L must appear in AL somewhere.

More formally, the language of AL based on L contains the following symbols:

1. The symbols {true, Bel, Send, Do};

[3]There is no special reason why we should not define a quantified language other than the space restrictions of this article; for quantified versions of AL, see (Wooldridge, 1992).

2. A countable set of constant symbols *Const* made up of the disjoint sets $Const_{Ag}$ (agent constants), and $Const_{Ac}$ (action constants);

3. All closed formulae of the internal language L;

4. The unary propositional connective '$\neg$' and binary propositional connective '$\vee$';

5. The unary temporal connectives $\{\bigcirc, \text{◎}\}$ and the binary temporal connectives $\{\mathcal{U}, \mathcal{S}\}$;

6. The punctuation symbols $\{), (\}$.

AL is thus parameterized by the internal language, L. The set of (well-formed) formulae of AL based on internal language L is defined by the following rules:

1. If i, j are agent ids, φ is a closed formula of L, and α is an action constant, then the following are atomic formulae of AL:

$$\text{true} \qquad (\text{Bel } i\ \varphi) \qquad (\text{Send } i\ j\ \varphi) \qquad (\text{Do } i\ \alpha)$$

2. If φ, ψ are formulae of AL, then the following are formulae of AL:

$$\neg\varphi \qquad \varphi \vee \psi$$

3. If φ, ψ are formulae of AL, then the following are formulae of AL:

$$\bigcirc\varphi \qquad \text{◎}\varphi \qquad \varphi\,\mathcal{U}\,\psi \qquad \varphi\,\mathcal{S}\,\psi$$

Table 10.1 summaries the meaning of the non-standard operators in AL.

Semantics

A model for AL is a structure:

$$M = \langle \sigma, Ag, Ac, bel, action, sent, I \rangle$$

where

- $\sigma \in \Sigma$ is an execution history;
- Ag is a set of agent ids;

$(\mathsf{Bel}\ i\ \varphi)$	Agent i believes φ
$(\mathsf{Send}\ i\ j\ \varphi)$	Agent i sent j message φ
$(\mathsf{Do}\ i\ \alpha)$	Agent i performs action α
$\bigcirc \varphi$	Next φ
$\textcircled{\scriptsize S} \varphi$	Last φ (strong)
$\varphi \mathcal{U} \psi$	φ Until ψ (not strict)
$\varphi \mathcal{S} \psi$	φ Since ψ (strict)

Table 10.1: *Non-standard Operators in AL*

- Ac is a set of actions;
- $bel : \Sigma \times Ag \times \mathbb{N} \rightarrow BS$

  is a function that takes an execution history, an agent identifier, and a time, and returns the belief set of the agent in the execution history at that time;
- $action : \Sigma \times Ag \times \mathbb{N} \rightarrow Ac$

  is a function that takes an execution history, an agent identifier, and a time, and returns the action performed by the agent in the execution history at that time;
- $sent : \Sigma \times \mathbb{N} \rightarrow \wp(Mess)$

  is a function that takes an execution history and a time, and returns the set of messages sent in the execution history at that time;
- $I : Const \leftrightarrow (Ag \cup Ac)$

  interprets constants.

There is a close relationship between the formal model of multi-agent systems developed earlier, and logical models for AL. We can characterise this relationship by stating precisely the conditions under which a model for AL can be considered to represent an execution of a multi-agent system:

$\langle M, u\rangle \models \mathsf{true}$		
$\langle M, u\rangle \models (\mathsf{Bel}\ i\ \varphi)$	iff	$\varphi \in bel(\sigma, I(i), u)$
$\langle M, u\rangle \models (\mathsf{Do}\ i\ \alpha)$	iff	$action(\sigma, I(i), u) = I(\alpha)$
$\langle M, u\rangle \models (\mathsf{Send}\ i\ j\ \varphi)$	iff	$\langle I(i), I(j), \varphi\rangle \in sent(\sigma, u)$
$\langle M, u\rangle \models \neg\varphi$	iff	$\langle M, u\rangle \not\models \varphi$
$\langle M, u\rangle \models \varphi \vee \psi$	iff	$\langle M, u\rangle \models \varphi$ or $\langle M, u\rangle \models \psi$
$\langle M, u\rangle \models \bigcirc\varphi$	iff	$\langle M, u+1\rangle \models \varphi$
$\langle M, u\rangle \models \text{◎}\varphi$	iff	$u > 0$ and $\langle M, u-1\rangle \models \varphi$
$\langle M, u\rangle \models \varphi\,\mathcal{U}\,\psi$	iff	$\exists v \in \mathbb{N}$ s.t. $v \geq u$ and $\langle M, v\rangle \models \psi$ and $\forall w \in \mathbb{N}$ s.t. $u \leq w < v, \langle M, w\rangle \models \varphi$
$\langle M, u\rangle \models \varphi\,\mathcal{S}\,\psi$	iff	$\exists v \in \{0, \ldots, u-1\}$ s.t. $\langle M, v\rangle \models \psi$ and $\forall w \in \mathbb{N}$ s.t. $v < w < u, \langle M, w\rangle \models \varphi$

Figure 10.1: *Semantic Rules for AL*

A model

$$\langle \sigma, Ag, Ac, bel, action, sent, I\rangle$$

for AL represents a run of the system

$$\{\langle \mathcal{B}_i^0, \mathcal{A}_i, \mathcal{M}_i, \mathcal{N}_i\rangle : i \in Ag\}$$

iff the following condition obtains:

$$\forall u \in \mathbb{N} \cdot \forall i \in Ag \cdot$$
$$bel(\sigma, i, u) = \mathcal{B}_i^u \wedge action(\sigma, i, u) = \mathcal{A}_i(\mathcal{B}_i^u) \wedge sent(\sigma, u) = \bigcup_{j \in Ag} \mathcal{M}_j(\mathcal{B}_j^u).$$

(Recall that $\mathcal{B}_i^u$ is defined in equation (10.1).) The satisfaction relation '$\models$' for AL holds between pairs of the form $\langle M, u\rangle$, (where M is a model for AL, and $u \in \mathbb{N}$ is a temporal index into M), and formulae of AL. The semantic rules for AL are given in Figure 10.1.

$\bullet\varphi$	$\stackrel{\text{def}}{=}$	$\neg \bigcirc \neg\varphi$	(Weak) last φ
start	$\stackrel{\text{def}}{=}$	$\neg \bigcirc \mathsf{true}$	Initially …
$\Diamond\varphi$	$\stackrel{\text{def}}{=}$	$\mathsf{true}\,\mathcal{U}\,\varphi$	Sometime φ
$\Box\varphi$	$\stackrel{\text{def}}{=}$	$\neg\Diamond\neg\varphi$	Always φ
$\varphi\,\mathcal{W}\,\psi$	$\stackrel{\text{def}}{=}$	$\Box\varphi \vee \varphi\,\mathcal{U}\,\psi$	φ Unless ψ
$\blacklozenge\varphi$	$\stackrel{\text{def}}{=}$	$\mathsf{true}\,\mathcal{S}\,\varphi$	Was φ
$\blacksquare\varphi$	$\stackrel{\text{def}}{=}$	$\neg\blacklozenge\neg\varphi$	Heretofore φ
$\varphi\,\mathcal{Z}\,\psi$	$\stackrel{\text{def}}{=}$	$\blacksquare\varphi \vee \varphi\,\mathcal{S}\,\psi$	φ Zince ψ
$\varphi\mathcal{B}\psi$	$\stackrel{\text{def}}{=}$	$\neg((\neg\varphi)\,\mathcal{U}\,\psi)$	φ Before (precedes) ψ

Table 10.2: *Derived Temporal Operators*

The first four rules deal with atomic formulae, (or *atoms*), of AL. The formula true is a logical constant for truth; it is always satisfied. The formula (Bel i φ) is read 'agent i believes φ'. Bel is essentially the belief operator from Konolige's logic L^B (Konolige, 1986a). The formula (Do i α) is read 'agent i performs the action α'. The formula (Send i j φ) describes the sending of messages, and will be satisfied if agent i has sent j a message with content φ.

The propositional connectives $\neg$ (not) and $\vee$ (or) have standard semantics. The remaining propositional connectives ($\Rightarrow$ (if … then …), $\wedge$ (and), and $\Leftrightarrow$ (iff)) are defined as abbreviations in the usual way.

For convenience, the abbreviations in Table 10.2 are assumed. '$\bigcirc$' and '$\bigcirc$' are the *next* and (strong) *last* operators respectively: $\bigcirc\varphi$ is satisfied if φ is satisfied at the next time; '$\bigcirc$' is the past time version of this operator, so that $\bigcirc\varphi$ is satisfied if φ was satisfied at the last time. '$\bigcirc$' is said to be *strong* because it is never satisfied at the beginning of time. The '$\bullet$' operator is the weak version of '$\bigcirc$': it is alway satisfied at the beginning of time. The formula 'start' is *only* satisfied at the beginning of time.

'$\Box$' and '$\blacksquare$' are the *always* and *heretofore* operators respectively: $\Box\varphi$ will be satisfied iff φ is satisfied now and in all future moments; '$\blacksquare$' is the past time version. '$\Diamond$' and '$\blacklozenge$' are the *sometime* and *was* operators respectively: $\Diamond\varphi$ will be satisfied if φ is satisfied now, or becomes satisfied at least once in the future; '$\blacklozenge$' is the past version,

so $\blacklozenge\varphi$ will be satisfied if φ was satisfied at least once in the past. '$\mathcal{U}$' and '$\mathcal{W}$' are the *until* and *unless* operators respectively ('$\mathcal{W}$' is sometimes called the *weak until* operator): $\varphi\,\mathcal{U}\,\psi$ will be satisfied if φ is satisfied at all times until ψ becomes satisfied — ψ must eventually be satisfied. '$\mathcal{W}$' is similar to '$\mathcal{U}$', but allows for the possibility that the second argument never becomes satisfied. Finally, the '$\mathcal{S}$' (since) and '$\mathcal{Z}$' (zince) operators are the past time versions of '$\mathcal{U}$' and '$\mathcal{W}$' respectively.

Satisfiability and *validity* for AL are defined in the usual way.

Proof Theory

The proof theoretic aspects of the linear discrete temporal logic upon which AL is based have been examined at length elsewhere, and we will not, therefore, discuss them here. The reader is referred to (Gough, 1984; Fisher, 1991; Manna and Pnueli, 1992; Emerson, 1990) for references and overviews.

10.4 Reasoning about Multi-Agent Systems

Having developed, in the preceding sections, a logic which may be used to represent the properties of multi-agent systems, we now turn to the issue of actually *using* the logic. There are two obvious ways in which the logic might be used: for *specification*, and for *verification*.

Specification

A system's specification is a statement describing those properties that it is intended the system should exhibit. A *formal* specification is one which is expressed (for the most part) in the language of mathematics. It is possible to use AL as a formal specification language for multi-agent systems, in a way that we shall now illustrate[4].

It was pointed out in section 10.2 that temporal logics have for some time been used as a specification language for reactive systems, of which DAI systems are a subset. There are two types of properties of such systems that we normally wish to specify:

- *liveness* properties, which assert that 'something good will happen', and
- *safety* properties, which assert that 'nothing bad happens'.

[4]Examples are actually presented using the quantified version of AL developed in (Wooldridge, 1992).

To illustrate such properties, and the way in which they might be expressed using AL, we shall consider a trivial example of a protocol for cooperative activity. In this protocol, agents communicate through just two message types (cf. (Shoham, 1993)):

$Request(\alpha)$	a request for action α to be performed
$Inform(\varphi)$	sender informs recipient of 'fact' φ

Two domain predicates are assumed:

$Friend(i)$	agent i is a friend
$Trust(i)$	agent i is trusted

An agent that receives a *Request* from a friendly agent to do some action will eventually do it, but not otherwise (i.e., actions will *only* be done at the request of a friendly agent). If an agent receives an *Inform* message, it will add the 'fact' to it beliefs only if the sender is trusted.

This simple protocol may specified with ease. First, we state that an agent that receives a *Request* from a friendly agent will do the action; this is a liveness property.

$$\Box \forall i \cdot \forall j \cdot \forall \alpha \cdot (\mathsf{Send}\ i\ j\ Request(\alpha)) \wedge (\mathsf{Bel}\ j\ Friend(i)) \Rightarrow \Diamond (\mathsf{Do}\ j\ \alpha)$$

That is, it is always true that if i sends j a *Request* for α, and j believes i to be friendly, then j will eventually do α.

The following expresses a safety property, namely that an if an agent does some action, then the performance of the action must have been preceded by a *Request* from a friendly agent to do the action.

$$\Box \forall i \cdot \forall j \cdot \forall \alpha \cdot \left[\begin{array}{c} \neg((\mathsf{Send}\ i\ j\ Request(\alpha)) \wedge (\mathsf{Bel}\ j\ Friendly(i))) \\ \mathcal{Z} \\ (\mathsf{Do}\ j\ \alpha) \wedge \neg(\mathsf{Send}\ i\ j\ Request(\alpha)) \end{array} \right] \Rightarrow \neg(\mathsf{Do}\ j\ \alpha)$$

We now move on to the properties of *Inform* messages. The following liveness property asserts that an agent receiving an *Inform* message from a trusted agent will eventually come to believe the content of the message.

$$\Box \forall i \cdot \forall j \cdot (\mathsf{Send}\ i\ j\ Inform(\varphi)) \wedge (\mathsf{Bel}\ j\ Trusted(i)) \Rightarrow \Diamond (\mathsf{Bel}\ j\ \varphi)$$

In this case, there is no corresponding safety property: we do not want to specify that an agent will only believe something if it has previously been informed of it by a trusted agent.

This example is admittedly simple, but at least demonstrates how one might go about the specification process. In (Wooldridge, 1992), several more detailed examples are provided.

Verification

Verification is the process of showing that an implemented system is correct with respect to its specification. *Formally* verifying a system involves proving (in the mathematical sense) that it is correct. Just as it is possible to use AL for specification, so it is possible to use it for verification. However, verification is generally regarded as a much more complex process than specification, and although the principles of formal verification have been fairly well understood for some time, it is quite rare to find examples of its use outside textbooks and research papers. We will therefore only touch on the subject here.

To verify that a system $\mathcal{SYS}$ is correct with respect to a specification $\mathcal{SPEC}$ (where $\mathcal{SPEC}$ is expressed as a set of formulae of AL, as described above), one first derives the *theory* of $\mathcal{SYS}$, which we shall denote by $\mathcal{TH}(\mathcal{SYS})$. Then, one attempts to prove that $\mathcal{SPEC}$ follows from $\mathcal{TH}(\mathcal{SYS})$, i.e., that $\mathcal{TH}(\mathcal{SYS}) \vdash \mathcal{SPEC}$, using the proof theory of AL. $\mathcal{TH}(\mathcal{SYS})$ will be a set of formulae of AL which capture both *general* properties of the type of system being modelled, and *specific* properties of $\mathcal{SYS}$. Examples of how one goes about deriving the theory of a system are provided in both (Wooldridge, 1992, Chapter 6) and (Fisher and Wooldridge, 1993b).

10.5 Discussion

This article has described a research programme, directed at developing formal methods for reasoning about multi-agent systems. This programme was illustrated by means of a simple worked example. The emphasis throughout the article has been on developing the ideas in an intelligible way. This has meant that in order to avoid confusion, a number of important issues have been glossed over, or ignored completely. Future work must focus particularly on the following topics:

- *Realistic treatments of concurrency.*

  The issue of concurrency has scarcely been addressed in formal treatments of DAI, and yet concurrency is at the very heart of the area. The reluctance to

address this issue is understandable, however: DAI systems are *not* simply concurrent systems. DAI researchers have quite rightly focussed on such issues as coordination, as these are peculiar to the domain. Nevertheless, if we ever hope to claim that we have a formalism which truly captures the properties of a DAI system, then we must deal with this issue. There are some signs that work is being done in this area (Burkhard, 1993).

- *Modelling more complex and realistic agents and systems.*

  The model of agents presented in this article was intended to be general, and yet its very generality makes it unrealistically simple. All but the most basic applications have a much richer internal structure. Our theories and models must be able to represent this structural and operational complexity, while remaining conceptually tractable. In this area too, there are signs that some work is being done (Rao and Georgeff, 1992; Wooldridge, 1994).

- *(Semi-)automated theorem proving.*

  We cannot hope to use logics such as AL for reasoning about realistic multi-agent systems without some kind of theorem proving support for them. Unfortunately, this kind of logic tends to be extraordinarily complex in computational terms; some similar logics are so complex as to be undecidable, *even in the propositional case* (Halpern and Vardi, 1989). However, work is underway to find proof methods for simplied variants of these logics (Wooldridge and Fisher, 1994).

- *Bridging the gap between theory and practice.*

  There is an intrinsic value in formal models and theories of the world, that makes them a worthwhile exercise even if they have no immediately obvious practical application. However, it is hoped that the research programme sketched in this article *does* have some practical applications. One important possibility for the near future involves bringing the practice of DAI closer to its theory by using logics such as AL to program agents directly; this is, in essence, what the 'agent-oriented' paradigm is all about (Shoham, 1993; Thomas, 1993; Fisher and Wooldridge, 1993a).

Acknowledgements

This article is a condensed and simplified version of part of the author's PhD thesis, (Wooldridge, 1992), and as such this work was supported by (what was) the SERC; a revised and much extended version of the review material presented here has appeared as (Wooldridge and Jennings, 1994). Thanks to Michael Fisher for his time and suggestions, and to Greg O'Hare for his encouragement and enthusiastic support.

References

Barwise, J. and Perry, J. (1983). *Situations and Attitudes*. The MIT Press.

Brooks, R. (1991). Intelligence without reason. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 569–595, Sydney, Australia.

Burkhard, H.-D. (1993). Liveness and fairness properties in multi-agent systems. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, pages 325–330, Chambéry, France.

Cammarata, S., McArthur, D., and Steeb, R. (1983). Strategies of cooperation in distributed problem solving. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83)*, Karlsruhe, Federal Republic of Germany.

Chellas, B. (1980). *Modal Logic: An Introduction*. Cambridge University Press.

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

Connah, D. and Wavish, P. (1990). An experiment in cooperation. In Demazeau, Y. and Müller, J.-P., editors, *Decentralized AI — Proceedings of the First European Workshop on Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW-89)*, pages 197–214. Elsevier/North Holland.

Dennett, D. C. (1987). *The Intentional Stance*. The MIT Press.

Doran, J., Carvajal, H., Choo, Y. J., and Li, Y. (1991). The MCS multi-agent testbed: Developments and experiments. In Deen, S. M., editor, *CKBS-90 — Proceedings of*

the International Working Conference on Cooperating Knowledge Based Systems, pages 240–254. Springer-Verlag.

Emerson, E. A. (1990). Temporal and modal logic. In van Leeuwen, J., editor, *Handbook of Theoretical Computer Science*, pages 996–1072. Elsevier.

Fagin, R. and Halpern, J. Y. (1985). Belief, awareness, and limited reasoning. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence (IJCAI-85)*, pages 480–490, Los Angeles, CA.

Fisher, M. (1991). A resolution method for temporal logic. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, Sydney, Australia.

Fisher, M. and Wooldridge, M. (1993a). Executable temporal logic for distributed A.I. In *Proceedings of the Twelfth International Workshop on Distributed Artificial Intelligence (IWDAI-93)*, pages 131–142, Hidden Valley, PA.

Fisher, M. and Wooldridge, M. (1993b). Specifying and verifying distributed intelligent systems. In Filgueiras, M. and Damas, L., editors, *Progress in Artificial Intelligence — Sixth Portuguese Conference on Artificial Intelligence (LNAI Volume 727)*, pages 13–28. Springer-Verlag.

Gärdenfors, P. (1988). *Knowledge in Flux*. The MIT Press.

Gasser, L., Braganza, C., and Hermann, N. (1987). MACE: A flexible testbed for distributed AI research. In Huhns, M., editor, *Distributed Artificial Intelligence*, pages 119–152. Pitman/Morgan Kaufmann.

Gough, G. D. (1984). Decision procedures for temporal logic. Master's thesis, Department of Computer Science, Manchester University, Oxford Rd., Manchester M13 9PL, UK.

Halpern, J. Y. (1987). Using reasoning about knowledge to analyze distributed systems. *Annual Review of Computer Science*, 2:37–68.

Halpern, J. Y. and Moses, Y. (1992). A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379.

Halpern, J. Y. and Vardi, M. Y. (1989). The complexity of reasoning about knowledge and time. I. lower bounds. *Journal of Computer and System Sciences*, 38:195–237.

Hintikka, J. (1962). *Knowledge and Belief*. Cornell University Press.

Kaelbling, L. P. (1986). An architecture for intelligent reactive systems. In Georgeff, M. P. and Lansky, A. L., editors, *Reasoning About Actions & Plans — Proceedings of the 1986 Workshop*, pages 395–410. Morgan Kaufmann Publishers, Inc.

Konolige, K. (1986a). *A Deduction Model of Belief*. Pitman/Morgan Kaufmann.

Konolige, K. (1986b). What awareness isn't: A sentential view of implicit and explicit belief (position paper). In Halpern, J. Y., editor, *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning About Knowledge*, pages 241–250. Morgan Kaufmann Publishers, Inc.

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

Levesque, H. J., Cohen, P. R., and Nunes, J. H. T. (1990). On acting together. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, pages 94–99, Boston, MA.

Manna, Z. and Pnueli, A. (1992). *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag.

Milner, R. (1989). *Communication and Concurrency*. Prentice Hall.

Milner, R., Parrow, J., and Walker, D. (1992). A calculus of mobile processes. *Information and Computation*, 100(1):1–77.

Moore, R. C. (1985). A formal theory of knowledge and action. In Hobbs, J. R. and Moore, R. C., editors, *Formal Theories of the Commonsense World*. Ablex Publishing Corporation.

Pnueli, A. (1977). The temporal logic of programs. In *Proceedings of the Eighteenth Symposium on the Foundations of Computer Science*.

Pnueli, A. (1986). Specification and development of reactive systems. In *Information Processing 86*. Elsevier/North Holland.

Rao, A. S. and Georgeff, M. P. (1992). An abstract architecture for rational agents. In Rich, C., Swartout, W., and Nebel, B., editors, *Proceedings of Knowledge Representation and Reasoning (KR&R-92)*, pages 439–449.

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

Thomas, S. R. (1993). *PLACA, an Agent Oriented Programming Language*. PhD thesis, Computer Science Department, Stanford University, Stanford, CA 94305. (Available as technical report STAN–CS–93–1487).

Wooldridge, M. (1992). *The Logical Modelling of Computational Multi-Agent Systems*. PhD thesis, Department of Computation, UMIST, Manchester, UK. (Also available as Technical Report MMU–DOC–94–01, Department of Computing, Manchester Metropolitan University, Chester St., Manchester, UK).

Wooldridge, M. (1994). This is MyWorld: The logic of an agent-oriented testbed for DAI. In Wooldridge, M. and Jennings, N. R., editors, *Pre-proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages*, pages 147–163, Amsterdam, The Netherlands.

Wooldridge, M. and Fisher, M. (1992). A first-order branching time logic of multi-agent systems. In Neumann, B., editor, *Proceedings of the Tenth European Conference on Artificial Intelligence (ECAI-92)*, pages 234–238. John Wiley & Sons.

Wooldridge, M. and Fisher, M. (1994). A decision procedure for a temporal belief logic. In Gabbay, D. M. and Ohlbach, H. J., editors, *Temporal Logic — Proceedings of the First International Conference (LNAI Volume 827)*, pages 317–331. Springer-Verlag.

Wooldridge, M. and Jennings, N. R. (1994). Agent theories, architectures, and languages: A survey. In Wooldridge, M. and Jennings, N. R., editors, *Pre-proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages*, pages 1–32, Amsterdam, The Netherlands.