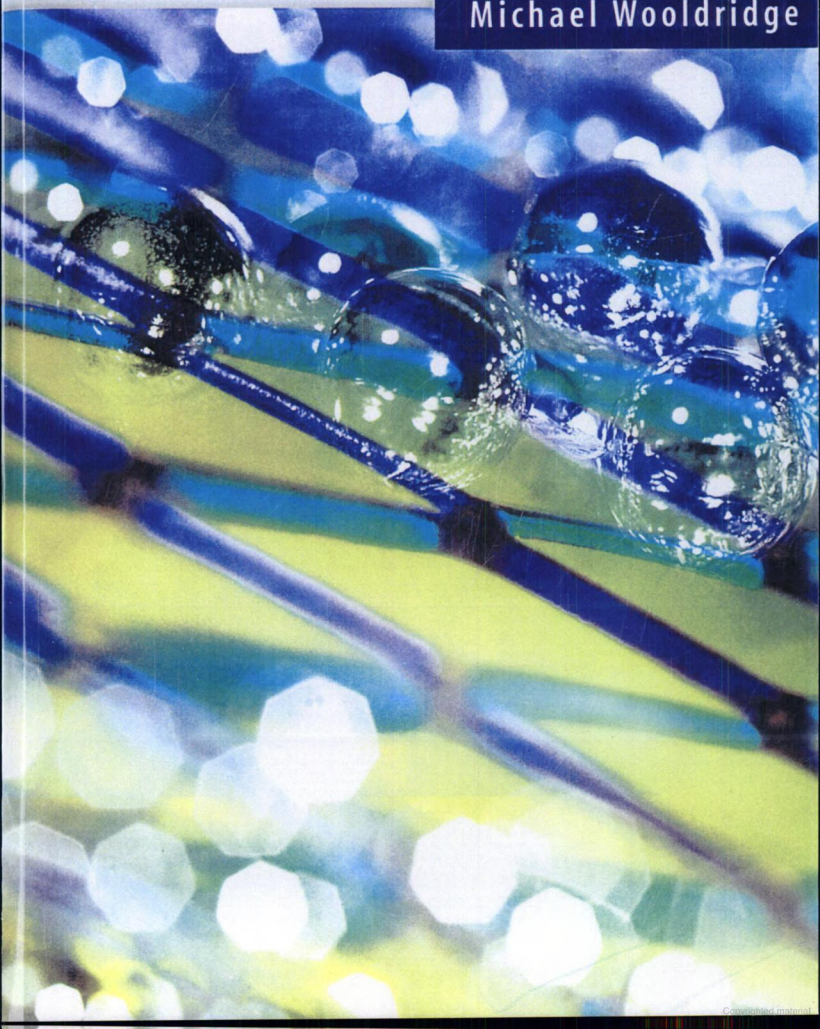
REASONING ABOUT RATIONAL AGENTS

Michael Wooldridge



Reasoning about Rational Agents

Michael Wooldridge

The MIT Press
Cambridge, Massachusetts
London, England

This book was set in Times Roman by the author using the LaTeX document preparation system and printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Wooldridge, Michael J., 1966–
Reasoning about rational agents / Michael Wooldridge.
p. cm.—(Intelligent robots and autonomous agents)
Includes bibliographical references and index.
ISBN 0–262–23213–8 (hc: alk. paper)
1. Intelligent agents (Computer software). I. Title. II. Series.
QA76.76I58W66 2000
006.3—dc21 00-035474
CIP

10 9 8 7 6 5 4 3

1 Rational Agents

The world we inhabit is full of agents. We encounter them every day, as we go about our lives. By the term "agent," I mean an entity that acts upon the environment it inhabits. Agents are not merely observers of their environment, nor are they the passive recipients of actions performed by other entities. Rather, agents are the active, purposeful originators of action. These actions are performed in order to modify and shape the environment inhabited by the agent. People like you and I are the most obvious examples of agents in the real world, but there are many others, such as legal entities like governments, companies, and corporations.

In our dealings with the world, we often make a distinction between agents that are *rational* and those that are not, in the following sense. An agent is said to be rational if it chooses to perform actions that are in its own best interests, given the beliefs it has about the world. For example, if I have a goal of staying dry, and I believe it is raining, then it is rational of me to take an umbrella when I leave my house. Taking the umbrella will enable me to satisfy my goal of staying dry, and in this sense, it is in my best interest. You would still be inclined to refer to my behavior as rational even if I was mistaken in my belief that it was raining: the point is that I made a decision that, if my beliefs were correct, would have achieved one of my goals.

In one sense, human agents are notoriously *irrational* when they make decisions. From time to time, we all make decisions that, with the benefit of hindsight, are self-evidently poor. When evaluated against strict metrics of rationality, humans rarely perform well. But this paints a misleading picture. The fact is that, overall, humans are very *good* decision makers. We inhabit a continuously changing physical world of extraordinary richness, complexity, and diversity. Our plans are constantly thwarted, both by random quirks of nature and, occasionally, by the deliberately malicious intervention of our peers. We are forced to make decisions for which we are ill prepared, both in terms of prior experience and in terms of the information upon which to base decisions. And yet, despite all this, we prosper. As decision makers, we seem to be quite robust to the vagaries of our environment. We successfully interact with our peers even when they are in conflict with us, and we cope with uncertainty about the world, failed actions and plans, and the complexity of our environment.

One goal of the artificial intelligence (AI) community is to engineer computer programs that can act as autonomous, rational agents. We wish to build computer programs that can *independently* make good decisions about what actions to perform. But it is not enough to have programs that *think* of a good action to perform — we wish to have them actually *execute* these actions. In other words, we want to create rational agents that are *embodied* in some environment — that inhabit and act upon some environment in the same way that we inhabit and act upon ours.

Ideally, we would like to engineer agents that are as good at making decisions and acting upon them as we are. Of course, this goal is a long way from being achieved. Unfortunately, like so many other problems encountered throughout the history of AI, it has proven extraordinarily difficult to engineer agents that can select and execute good decisions for moderately complex environments. Nor is the problem likely to be solved in any meaningful sense for some time yet.

1.1 Properties of Rational Agents

In order to understand why it is hard to engineer a rational agent, let us consider what sorts of properties we expect a rational agent to exhibit. First, it is important to understand that we usually consider agents to be systems that are *situated* or *embodied* in some environment — agents are not *disembodied* systems. By this, I mean that agents are capable of *sensing* their environment and have a repertoire of possible *actions* that they can perform in order to modify the environment. This leads to the view of an agent as shown in Figure 1.1. The requirement that agents must be embodied implies that many of the systems that have been developed to date within AI do not count as agents. Examples include most theorem provers and expert systems.

The environment that an agent occupies may be physical (in the case of robots inhabiting the physical world) or a software environment (in the case of a software agent inhabiting a computer operating system or network). In the case of a physically embodied agent, actions will be physical, such as moving objects around. The sensor input received by the agent will be comprised of video feeds and the like. In the case of a software agent, actions will be software commands (such as the UNIX `rm` command, which removes a file), and sensor input will be obtained by performing commands such as `ls` (which obtains a directory listing) [53].

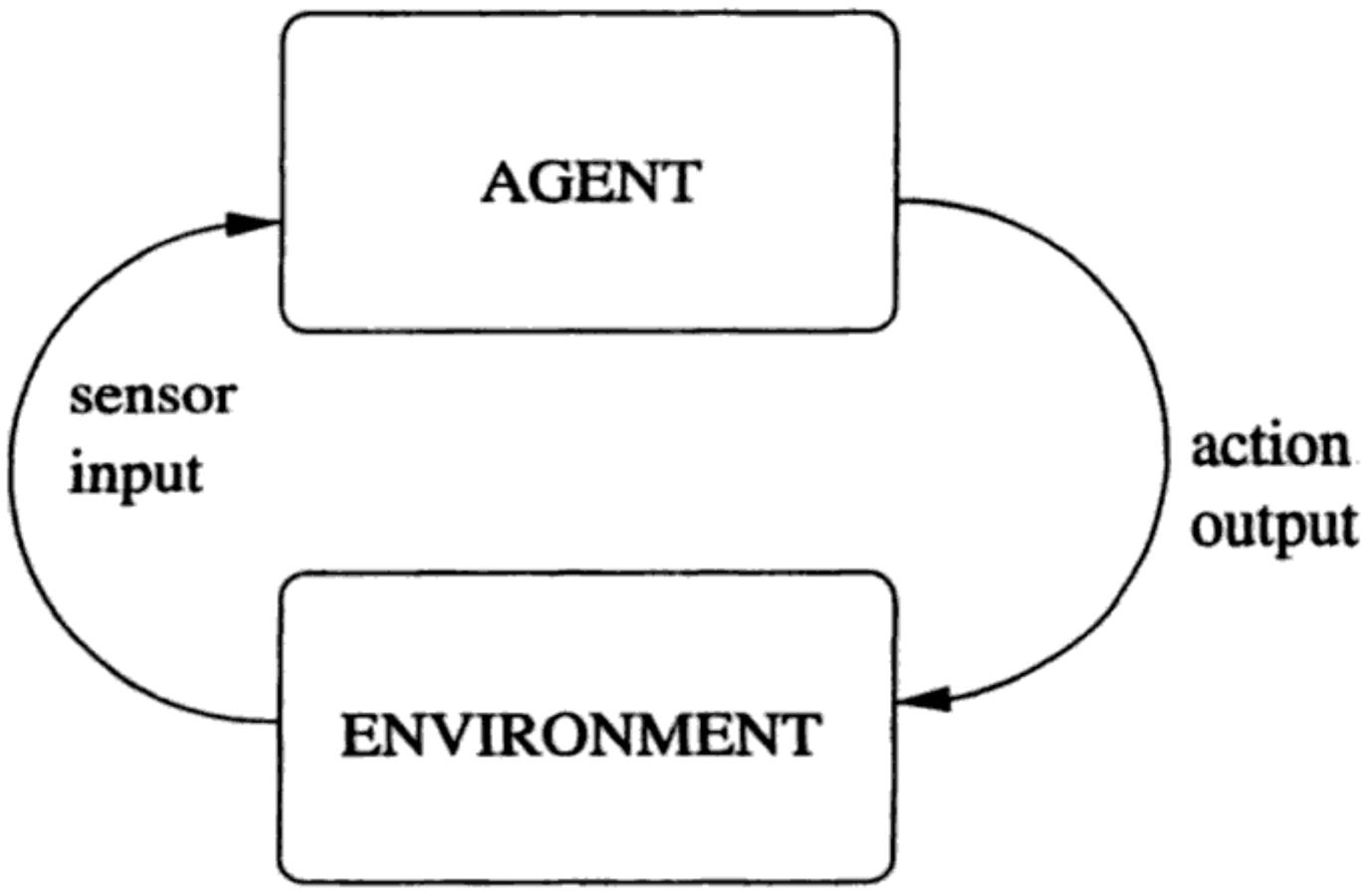


Figure 1.1
Agents and their environments. The agent takes sensory input from the environment, and produces as output actions that affect it.

Note that in almost all realistic applications, agents have at best *partial* control over their environment. Thus, while they can perform actions that change their environment, they cannot in general *completely* control it. It therefore makes sense to divide the environment, conceptually at least, into those parts that the agent *controls*, those parts it *partially controls*, and those parts over which it has *no control*. (If we regard an agent's internal state as being part of the environment, then this would be controlled by the agent.)

Apart from being situated in an environment, what other properties do we expect a rational agent to have? Wooldridge and Jennings [249] argue that agents should have the following properties:

- autonomy;
- proactiveness;
- reactivity; and
- social ability.

At its simplest, autonomy means nothing more than being able to operate independently. Thus, at the very least, an autonomous agent makes independent decisions — its decisions (and hence its actions) are under its own control, and are not driven by others. Autonomous agents are therefore the loci of decision making. However, autonomy is usually taken to have a slightly stronger meaning than this. We generally assume that an autonomous agent is one that has its own beliefs, desires, and intentions, which are not subservient to those

of other agents. This is not to say that these beliefs, desires, and intentions are necessarily *represented* within an agent — the point is that we regard a rational agent as "having its own agenda," which may or may not concur with that of other agents.

Proactiveness means being able to exhibit goal-directed behavior. If an agent has a particular goal or intention, then we expect the agent to try to *achieve* this intention. For example, if I have an intention to write a book, then it is rational of me to actually *act towards* this intention, by eventually putting pen to paper. Moreover, proactiveness means *exploiting serendipity*. For example, if I have a desire to become rich, and an opportunity to make a fortune arises, then it is only rational of me to try to exploit this opportunity, perhaps by generating a new goal. Proactiveness rules out entirely passive agents, who never try to do anything.

Being reactive means being responsive to changes in the environment. As I noted above, in everyday life, our plans rarely run smoothly. They are frequently thwarted, both by the vagaries of nature and by other agents fouling them up. When we become aware that our plans have gone awry, we do not ignore this fact. We respond, by choosing an alternative course of action. Responses may be of the "stimulus → response" style, or they may involve further deliberation. For example, if I am driving my car and I realize I am about to crash into something, I will swerve to avoid it. It is unlikely that this kind of response involves much deliberation on my part; the behavior seems to be "hardwired." In contrast, if I am traveling to Heathrow Airport for a 9.30 AM flight, and I discover that the train I was about to catch is not running, I will actively deliberate in order to fix upon a course of action that will still enable me to catch my flight. This deliberation typically involves at least some reasoning about the alternatives available (catching a taxi or bus, for example). Crucially, however, I fix upon an alternative sufficiently quickly that it will deliver me to the airport in time — I will not deliberate for so long that it becomes impossible to catch the flight.

Designing a *purely* reactive system, which simply responds to environmental stimuli, is not hard. We can implement such a system as a lookup table, which simply maps environment states directly to actions. Similarly, developing a purely goal-driven system is not hard. (After all, this is ultimately what conventional computer programs are.) However, implementing a system that achieves an effective *balance* between goal-directed and reactive behavior turns out to be very hard. We elaborate on this problem in the next chapter.

Finally, let us say something about *social ability*. In one sense, social ability is trivial. After all, every day, millions of computers across the world routinely exchange information with both humans and other computers. But the ability to exchange bit streams is not really social ability. Consider that, in the human world, comparatively few of our goals can be achieved without interaction with other people and organizations, who cannot be assumed to share our goals. These agents are themselves autonomous, with their own agendas to pursue. To achieve our goals in such situations, we must *negotiate* and *cooperate* with these agents. We may be required to understand and reason about their beliefs and goals, and to perform actions (such as paying them money) that we would not otherwise choose to perform, in order to get them to cooperate with us, and achieve our goals. This type of social ability — the ability to interact (through negotiation and cooperation) with other self-interested agents — is much more complex and much less understood than simply the ability to exchange bit streams. In the later chapters of this book, I focus on such interactions in detail.

1.2 A Software Engineering Perspective

It is important to understand that the engineering of rational agents is not simply an abstract intellectual pursuit. It arises as a result of problems faced by software engineers across the world at the start of the new millennium.

Originally, software engineering concerned itself primarily with what are known as "functional" systems. A functional system is one that simply takes some input, performs some computation over this input, and eventually produces some output. Such systems may formally be viewed as functions $f : I \rightarrow O$ from a set I of inputs to a set O of outputs. The classic example of such a system is a compiler, which can be viewed as a mapping from a set I of legal source programs to a set O of corresponding object or machine code programs. Although the internal complexity of a functional system may be great (e.g., in the case of a compiler for a complex programming language such as ADA), functional programs are, in general, comparatively simple to engineer correctly and efficiently. Unfortunately, many of the computer systems that we desire to build are not functional in the sense we describe here. Rather than simply computing a function of some input and then terminating, such systems are *reactive*, in the following sense:[1]

1 There are at least three current usages of the term "reactive system" in computer science. The

constrains decision-making in the manner described, seems attractive.

The BDI model has been implemented several times. Originally, it was realized in IRMA, the Intelligent Resource-bounded Machine Architecture [22]. IRMA was intended as a more or less direct realization of Bratman's theory of practical reasoning. However, the best-known implementation is the Procedural Reasoning System (PRS) [78] and its many descendants [61, 183, 46, 100]. In the PRS, an agent has data structures that explicitly correspond to beliefs, desires, and intentions. A PRS agent's beliefs are directly represented in the form of PROLOG-like facts [34, p.3]. Desires and intentions in PRS are realized through the use of a *plan library*.[2] A plan library, as its name suggests, is a collection of plans. Each plan is a recipe that can be used by the agent to achieve some particular state of affairs. A plan in the PRS is characterized by a *body* and an *invocation condition*. The body of a plan is a course of action that can be used by the agent to achieve some particular state of affairs. The invocation condition of a plan defines the circumstances under which the agent should "consider" the plan. Control in the PRS proceeds by the agent continually updating its internal beliefs, and then looking to see which plans have invocation conditions that correspond to these beliefs. The set of plans made active in this way correspond to the *desires* of the agent. Each desire defines a possible course of action that the agent may follow. On each control cycle, the PRS picks one of these desires, and pushes it onto an execution stack, for subsequent execution. The execution stack contains desires that have been chosen by the agent, and thus corresponds to the agent's *intentions*.

The third and final component of the BDI model is the logical component. This logical component provides a family of tools that allow us to reason about BDI agents.

1.4 Reasoning about Belief-Desire-Intention Agents

Computer science is, as much as it is about anything, about developing theories and formalisms that allow us to reason about the behavior of computational systems. The theory of a system can be understood as the *semantics* of that system. An obvious question, therefore, is: What sort of theory can we use to give a semantics to BDI systems? The answer presented by this book is to use BDI logics to *axiomatize* properties of agents.

---

2 In this description of the PRS, I have modified the original terminology somewhat, to be more in line with contemporary usage; I have also simplified the control cycle of the PRS slightly.

A fundamental problem in developing such a theory of rational agency is to give an account of the relationships that exist between an agent's mental states. In particular, a complete agent theory would explain how an agent's mental states lead it to select and perform rational actions, thus realizing the mapping from perception to action illustrated in Figure 1.1. To give a flavor of the issues involved in developing such a theory, consider the relationship between belief and intention. Suppose I intend to bring about some state of affairs φ. What does the fact that I intend φ imply about my beliefs with respect to φ? Well, clearly, it would make no sense for me to intend φ if I believe that φ is already true. If my book is written, then there is no point in me intending to write it. This property — that intending φ implies you do not already believe φ — is a *constraint* that should be satisfied by a rational agent. We can capture this constraint in a formula schema of $\mathcal{LORA}$, the logic I use throughout this book to represent the properties of agents.

$$(\mathsf{Int}\ i\ \varphi) \Rightarrow \neg(\mathsf{Bel}\ i\ \varphi) \tag{1.1}$$

Here, $(\mathsf{Int}\ i\ \varphi)$ is a construction that means "agent i intends φ." It is not hard to guess that $(\mathsf{Bel}\ i\ \varphi)$ means "agent i believes φ." A "complete" theory of rational agency would be comprised of a number of formula schemas such as (1.1), which would together axiomatize the properties of rational agency.

Unfortunately, giving anything like a complete account of the relationships between an agent's mental states is extremely difficult. Part of the problem is that the objects of study — beliefs, desires, and the like — are rather different to phenomena that we observe in the physical world. In attempting to develop formal theories of such notions, we are forced to rely very much on our intuitions about them. As a consequence, such theories are hard to validate (or invalidate) in the way that good scientific theories should be. Fortunately, we have powerful tools available to help in our investigation. Mathematical logic allows us to represent our theories in a transparent, readable form, and examine (through the medium of formal proof) the predictions that our theories make, in order to see whether or not their consequences make sense.

$\mathcal{LORA}$ stands for "$\mathcal{L}$ogic $\mathcal{O}$f $\mathcal{R}$ational $\mathcal{A}$gents." $\mathcal{LORA}$ is a BDI logic: originally developed by Anand Rao and Michael Georgeff [187, 186, 188, 189, 184, 185], BDI logics allow us to represent the beliefs, desires, and intentions of agents. In addition to the BDI component, $\mathcal{LORA}$ contains a *temporal* component, which allows us to represent the dynamics of agents and their environments — how they change over time. Finally, $\mathcal{LORA}$ contains an *action* component, which allows us to represent the actions that agents

perform, and the effects that these actions have.

1.5 Frequently Asked Questions (FAQ)

Certain questions *are* frequently asked about the work presented in this book, and so it seems only appropriate that I should include a FAQ.

Why Not Decision Theory?

Decision theory is a mathematical theory of rational decision making. Decision theory defines a rational agent as one that *maximizes expected utility*. The basic ideas of decision theory are very simple. Assume we have some agent, and that Ac is the set of all possible actions available to it. The performance of an action by the agent may result in a number of possible outcomes, where the set of all outcomes is $\Omega = \{\omega, \omega', \ldots\}$. Let the probability of outcome $\omega \in \Omega$ given that the agent performs action $\alpha \in Ac$ be denoted by $P(\omega \mid \alpha)$, and finally, let the *utility* of an outcome $\omega \in \Omega$ for the agent be given by a function $U : \Omega \rightarrow \mathbb{R}$. If $U(\omega) > U(\omega')$, then the agent prefers outcome ω over outcome ω'.

The *expected utility* of an action $\alpha \in Ac$ is denoted $EU(\alpha)$. Thus $EU(\alpha)$ represents the utility that an agent could expect to obtain by performing action α.

$$EU(\alpha) = \sum_{\omega \in \Omega} U(\omega) P(\omega \mid \alpha) \tag{1.2}$$

A rational agent is then one that chooses to perform an action α that maximizes $EU(\ldots)$. It is straightforward to define the behavior of such an agent. Let the function f_{opt} take as input a set of possible actions, a set of outcomes, a probability distribution over possible outcomes given the performance of actions, and a utility function, and let this function return an action. Then we can define the desired behavior of f_{opt} as follows.

$$f_{opt}(Ac, \Omega, P, U) = \arg \max_{\alpha \in Ac} \sum_{\omega \in \Omega} U(\omega) P(\omega \mid \alpha) \tag{1.3}$$

As Russell and Norvig point out, equations (1.2) and (1.3) could be seen in one sense as defining the whole of artificial intelligence [198, p.472]. It seems that in order to build a rational agent, all we need to do is implement the function f_{opt}. But while decision theory is a *normative* theory of rational action (it tells us what an agent *should in principle do*) it has nothing to say about how we might efficiently *implement* the function f_{opt}. The problem is that f_{opt} seems to require

an unconstrained search over the space of all actions and their outcomes. Such a search is prohibitively expensive, particularly in the case where an agent needs to consider sequences of actions (i.e., plans).

Another problem is that it is hard in practice to obtain the probability distribution P or utility function U. If we consider a domain defined by m attributes, each of which may take n values, then the set Ω will contain m^n different outcomes. Simply enumerating these outcomes is likely to be impractical in many circumstances. Even if it is possible to enumerate the outcomes, actually obtaining the required probabilities and utilities for them is notoriously hard [198, pp.475–480]. See the "Notes and Further Reading" section at the end of this chapter for references to how decision theory is currently being used in artificial intelligence.

Why Not Game Theory?

Game theory is a close relative of decision theory, which studies interactions between agents where each agent is accorded a utility function as described above [14]. The tools and techniques of game theory have found many applications in computational multiagent systems research, particularly when applied to such problems as negotiation [195, 199]. Game theory shares with decision theory many concepts, in particular the concept of a rational agent as one that acts to maximize expected utility. However, game theory also shares many of the problems of decision theory. While it tells us what a rational agent should do in principle, it does not give us many clues as to how to implement such decision makers efficiently. In addition, like decision theory, game theory is *quantitative* in nature.

It is important to note that all this should not be taken to imply that game theory is in any sense wrong, or that it is of no use. On the contrary, game theory is an extremely powerful analytic tool, which will certainly be applied more widely in years to come. But it does have limitations.

Why Logic?

There are some in the AI research community who believe that logic is (to put it crudely) the work of the devil, and that the effort devoted to such problems as logical knowledge representation and theorem proving over the years has been, at best, a waste of time. At least a brief justification for the use of logic therefore seems necessary.

First, by fixing on a structured, well-defined artificial language (as opposed to unstructured, ill-defined natural language), it is possible to investigate the question of *what* can be expressed in a rigorous, mathematical way (see, for example, Emerson and Halpern [50], where the expressive power of a number of temporal logics are compared formally). Another major advantage is that any ambiguity can be removed (see, e.g., proofs of the unique readability of propositional logic and first-order predicate logic [52, pp.39–43]).

Transparency is another advantage:

> By expressing the properties of agents, and multiagent systems as logical axioms and theorems in a language with clear semantics, the focal points of [the theory] are explicit. The theory is transparent; properties, interrelationships, and inferences are open to examination. This contrasts with the use of computer code, which requires implementational and control aspects within which the issues to be tested can often become confused. [68, p.88]

Finally, by adopting a logic-based approach, one makes available all the results and techniques of what is arguably the oldest, richest, most fundamental, and best-established branch of mathematics.

The final chapter in the main text of this book, chapter 9, considers in detail the role of logic in the theory and practice of multiagent systems.

Why Not First-Order Logic?

Those active in the area of logic for AI can be divided broadly into two camps: those who make use of classical (usually first-order) logic, and those who use non-classical logics (in particular, modal and temporal logics of the kind used in this book) of some sort. This book makes use of a logic called $\mathcal{LORA}$. From a technical point of view, $\mathcal{LORA}$ is a first-order branching time logic, containing modal connectives for representing the beliefs, desires, and intentions of agents, as well as a dynamic logic-style apparatus for representing and reasoning about the actions that agents perform.

Strong opinions are held in both camps on the relative merits of classical versus non-classical logics. Those in favor of classical logic point to a number of general arguments against modal logics:

1. First-order logic is sufficiently expressive that it can be used to encode almost any form of knowledge required.
2. While the satisfiability problem for first-order logic is only semi-decidable (see, e.g., [73, p.58]) the corresponding problem for modal and temporal logics tends to be even worse. Some multimodal logics are undecidable even in the

science will cover these prerequisites. If you have such a background, then you should be able to read the book and gain an understanding of the main ideas without delving into the formal details. Ultimately, all you need to know to understand most of the book is how to read formulae of $\mathcal{LORA}$. Chapter 3 explains how to read formulae of $\mathcal{LORA}$ without going into the details of the logic at all. If you prefer not to read the mathematically oriented sections of the book, then you should avoid chapter 4 completely, and all sections marked with an asterisk ("*") in the section header. The appendices survey the logical preliminaries required to understand the book.

Do I Need to Read the Proofs?

I do not enjoy reading proofs, or indeed writing them, but they are "a necessary evil" in the words of my editor at MIT Press. If you feel the same as I feel, then my advice is as follows. If you simply want to become familiar with the techniques used and the results in the book, then there is no reason to read any of the proofs at all. However, if you want to *use* $\mathcal{LORA}$ yourself, to build on any of the results presented in this book, or to gain a really deep understanding of the book, then you will simply have to bite the bullet. In fact, most of the proofs rely on a relatively small set of techniques, which will already be familiar to readers with a grounding in modal, temporal, and first-order logic.

How Do I Implement This Book?

$\mathcal{LORA}$ is *not* an executable logic, and there is currently no simple way of automating $\mathcal{LORA}$ so that it can be used directly as a knowledge representation formalism or a programming language. The possible roles that logics such as $\mathcal{LORA}$ might play in the engineering of agent systems are explored in detail in chapter 9.

Where Can I Find Out More?

I have done my best to make the material in this book as self-contained as possible, in that while it presupposes some basic knowledge of discrete maths and logic, all definitions and such are included in their entirety. There should therefore be no need to refer directly to other texts in order to understand the book. However, to help the reader find out more, every chapter (including this one) concludes with a section entitled "Notes and Further Reading," which gives historical notes, pointers to further reading, and open problems. The appendices also contain background material.

1.6 The Structure of This Book

The remainder of this book can be broadly divided into three parts:

- The first part (chapters 2 and 3) is background material. This part sets the scene, by introducing the basic concepts and formalism used throughout the book.
- The second part (chapters 4 and 5) develops the formal model that is used throughout the book, and shows how this framework can be used to capture some properties of individual rational agents.
- The third part (chapters 6, 7, and 8) shows how agents can be used to capture properties of multiagent, social systems.

In more detail:

- Chapter 2 ("The Belief-Desire-Intention Model") gives a detailed introduction to the BDI model. It begins by introducing Bratman's intention-based theory of practical reasoning, the foundation upon which the BDI model rests. After introducing this theory, I turn to the question of how BDI agents might be implemented. I consider a number of pseudo-code agent control loops, and discuss the extent to which they can be considered as capturing our pre-theoretic intuitions about beliefs, desires, and intentions.
- Chapter 3 ("An Introduction to $\mathcal{LORA}$") informally introduces $\mathcal{LORA}$. This chapter is intended for readers who do not have a strong background in logic. After this chapter, readers should be able to understand formulae of $\mathcal{LORA}$.
- Chapter 4 ("$\mathcal{LORA}$ Defined") gives a complete definition of the syntax and semantics of $\mathcal{LORA}$, and investigates some of its properties. This chapter presupposes some familiarity with quantified modal, temporal, and dynamic logics, but is otherwise entirely self-contained.
- Chapter 5 ("Properties of Rational Agents") investigates how $\mathcal{LORA}$ can be used to capture properties of rational agents. Examples of such properties include the possible interrelationships among beliefs, desires, and intentions, and degrees of realism. This chapter focuses in particular on the properties of *individual* agents.
- Chapter 6 ("Collective Mental States") changes the focus of the book from individual agents to groups of agents. It shows how $\mathcal{LORA}$ can be used to formalize collective mental states such as mutual beliefs, desires, and intentions,

but in addition, such notions as joint commitment.

• Chapter 7 ("Communication") shows how $\mathcal{LORA}$ can be used to define communicative (speech) acts between agents. Following a discussion of the history of speech acts in multiagent systems, I show how "request" and "inform" actions can be defined in $\mathcal{LORA}$, building on the prior work of Cohen and Levesque [36]. Roughly speaking, an inform action takes place when one agent attempts to get another agent to believe something; a request action takes place when one agent gets another agent to intend something. Using these two speech acts as primitives, I show how a range of other speech acts may be defined using $\mathcal{LORA}$.

• Chapter 8 ("Cooperation") shows how $\mathcal{LORA}$ can be used to define a model of cooperative problem solving. This model is an adaptation of the theory of cooperation introduced in [247, 248, 250]. It treats cooperative problem solving as a four-stage process, beginning when one agent recognizes the potential for cooperation with respect to one of its actions, which is followed by the agent attempting to solicit assistance from a team of agents, which then attempt to agree to a plan of (joint) action to achieve the goal, which finally is executed by the group.

• Chapter 9 ("Logic and Agent Theory") attempts to put the book into perspective. It discusses the role that logical theories of agency can or should play in the development of agent systems. Adopting a software engineering perspective, where such theories are treated as *specifications* that an idealized agent would satisfy, it investigates the extent to which they can be used in the implementation or verification of practical agent systems.

In addition, the book contains two appendices:

• Appendix A contains a summary of the notation used in the main text of the book.

• Appendix B contains a short, self-contained introduction to modal and temporal logics, for readers unfamiliar with this material.

1.7 Notes and Further Reading

There are now many introductions to intelligent agents and multiagent systems. Ferber [56] is an undergraduate textbook, although as its name suggests, this volume focussed on multiagent aspects rather than on the theory and practice of individual agents. A first-rate collection of articles introducing agent

and multiagent systems is Weiß [233]; in particular, chapters 1, 2, 3, and 8 would provide an excellent introduction to the current volume. Three collections of research articles provide a comprehensive introduction to the field of autonomous rational agents and multiagent systems: Bond and Gasser's 1988 collection, *Readings in Distributed Artificial Intelligence*, introduces almost all the basic problems in the multiagent systems field, and although some of the papers it contains are now rather dated, it remains essential reading [18]; the *Readings in Planning* collection, from Allen, Hendler, and Tate may not appear to be the most obvious place to start for an introduction to agents, but the basic issue it addresses — deciding what action to perform — is central to the study of agents [2]; finally, Huhns and Singh's more recent collection sets itself the ambitious goal of providing a survey of the whole of the agent field, and succeeds in this respect very well [102]. For a general introduction to the theory and practice of intelligent agents, see Wooldridge and Jennings [249], which focuses primarily on the theory of agents, but also contains an extensive review of agent architectures and programming languages. For a collection of articles on the applications of agent technology, see [109]. A comprehensive roadmap of agent technology was published as [108].

Some authors view the whole of the artificial intelligence endeavor as one of constructing rational agents: Russell and Norvig's enormously successful, encyclopedic introductory AI textbook is the best-known example [198].

This question of "what is an agent" is one that continues to generate some debate, particularly on unmoderated email lists and news forums. Unfortunately, much of the debate is not as informed as one might wish. For an interesting collection of answers to the question, see [164]. The notion of a *software agent*, which inhabits a software environment, was introduced by Oren Etzioni [53]. Another good discussion can be found in Kaelbling [112].

The relevance of decision theory and game or economic theory to artificial intelligence was recognized in the very earliest days of the discipline; the work of Herb Simon is perhaps best known in this regard (see, e.g., [208]). Decision theoretic approaches to the planning problem are currently the focus of considerable attention within the artificial intelligence community — see [16] for an overview. Rao and Georgeff, who developed the basic BDI logic framework upon which $\mathcal{LORA}$ is based, investigated the relationship between the BDI model and classical decision theory in [185, pp.297–200]. They showed how the "choice and chance" trees employed in decision theory could be mapped into semantic models for their BDI logics.

Following the seminal work of Jeffrey Rosenschein and colleagues on the application of game-theoretic techniques to negotiation [195], game theory is now widely applied in multiagent systems research. A comprehensive overview is provided in [199].

Finally, logic has been widely used in artificial intelligence since John McCarthy's early work on the "advice taker" system [149]; see [73, 198] for detailed introductions to the use of logic in artificial intelligence, and see [103] for a general discussion on the role of logic in artificial intelligence. See also [167] and Birnbaum's response to this for some more strident views [15]. For discussions on the relative merits of first-order logic versus the alternatives (modal or temporal logics), see [191] or [70].

Although he is a strong advocate of logic in artificial intelligence, John McCarthy has reservations about the utility of *modal* logic; a position paper on the topic, entitled *Modality, Si! Modal Logic, No!* appeared as [150]. Joseph Halpern responded to McCarthy's statement in the online journal *Electronic Transactions on Artificial Intelligence* in late 1999; at the time of writing, the online debate was continuing.

of deciding which career to aim for is deliberation. Once one has fixed upon a career, there are further choices to be made; in particular, how to bring about this career. Suppose that after deliberation, you choose to pursue a career as an academic. The next step is to decide *how to achieve* this state of affairs. This process is means-ends reasoning. The end result of means-ends reasoning is a *plan* or *recipe* of some kind for achieving the chosen state of affairs. For the career example, a plan might involve first applying to an appropriate university for a Ph.D. place, and so on. After obtaining a plan, an agent will typically then attempt to carry out (or *execute*) the plan, in order to bring about the chosen state of affairs. If all goes well (the plan is sound, and the agent's environment cooperates sufficiently), then after the plan has been executed, the chosen state of affairs will be achieved.

Thus described, practical reasoning seems a straightforward process, and in an ideal world, it would be. But there are several complications. The first is that deliberation and means-ends reasoning are *computational* processes. In all real agents (and in particular, artificial agents), such computational processes will take place under *resource bounds*. By this I mean that an agent will only have a fixed amount of memory and a fixed processor available to carry out its computations. Together, these resource bounds impose a limit on the size of computations that can be carried out in any given amount of time. No real agent will be able to carry out arbitrarily large computations in a finite amount of time. Since almost any real environment will also operate in the presence of *time constraints* of some kind, this means that means-ends reasoning and deliberation must be carried out in a fixed, finite number of processor cycles, with a fixed, finite amount of memory space. From this discussion, we can see that resource bounds have two important implications:

- Computation is a valuable resource for agents situated in real-time environments. The ability to perform well will be determined at least in part by the ability to make efficient use of available computational resources. In other words, an agent must *control* its reasoning effectively if it is to perform well.

- Agents cannot deliberate indefinitely. They must clearly *stop* deliberating at some point, having chosen some state of affairs, and commit to achieving this state of affairs. It may well be that the state of affairs it has fixed upon is not optimal — further deliberation may have led it to fix upon an another state of affairs.

We refer to the states of affairs that an agent has chosen and committed to as its *intentions*. The BDI model of agency is ultimately one that recognizes the primacy of intentions in practical reasoning, and it is therefore worth discussing the roles that they play in more detail.

2.2 Intentions in Practical Reasoning

First, notice that it is possible to distinguish several different types of intention. In ordinary speech, we use the term "intention" to characterize both *actions* and *states of mind*. To adapt an example from Bratman [20, p.1], I might intentionally push someone under a train, and push them with the intention of killing them. Intention is here used to characterize an action — the action of pushing someone under a train. Alternatively, I might have the intention this morning of pushing someone under a train this afternoon. Here, intention is used to characterize my state of mind. In this book, when I talk about intentions, I mean intentions as states of mind. In particular, I mean *future-directed intentions* — intentions that an agent has towards some future state of affairs.

The most obvious role of intentions is that they are *pro-attitudes* [21, p.23]. By this, I mean that they tend to lead to action. Suppose I have an intention to write a book. If I truly have such an intention, then you would expect me to make a *reasonable attempt* to achieve it. This would usually involve, at the very least, me initiating some plan of action that I believed would satisfy the intention. In this sense, intentions tend to play a primary role in the production of action. As time passes, and my intention about the future becomes my intention about the present, then it plays a direct role in the production of action. Of course, having an intention does not necessarily lead to action. For example, I can have an intention now to attend a conference later in the year. I can be utterly sincere in this intention, and yet if I learn of some event that must take precedence over the conference, I may never even get as far as considering travel arrangements.

Bratman notes that intentions play a much stronger role in influencing action than other pro-attitudes, such as mere desires:

> My desire to play basketball this afternoon is merely a potential influencer of my conduct this afternoon. It must vie with my other relevant desires [...] before it is settled what I will do. In contrast, once I intend to play basketball this afternoon, the matter is settled: I normally need not continue to weigh the pros and cons. When the afternoon arrives, I will normally just proceed to execute my intentions. [21, p.22]

The second main property of intentions is that they *persist*. If I adopt an intention to become an academic, then I should persist with this intention and attempt to achieve it. For if I immediately drop my intentions without devoting any resources to achieving them, then I will not be acting rationally. Indeed, you might be inclined to say that I never really had intentions in the first place.

Of course, I should not persist with my intention for too long — if it becomes clear to me that I will never become an academic, then it is only rational to drop my intention to do so. Similarly, if the reason for having an intention goes away, then it would be rational for me to drop the intention. For example, if I adopted the intention to become an academic because I believed it would be an easy life, but then discover that this is not the case (e.g., I might be expected to actually teach!), then the justification for the intention is no longer present, and I should drop the intention.

If I initially fail to achieve an intention, then you would expect me to *try again* — you would not expect me to simply give up. For example, if my first application for a Ph.D. program is rejected, then you might expect me to apply to alternative universities.

The third main property of intentions is that once I have adopted an intention, the very fact of having this intention will constrain my future practical reasoning. For example, while I hold some particular intention, I will not subsequently entertain options that are *inconsistent* with that intention. Intending to write a book, for example, would preclude the option of partying every night: the two are mutually exclusive. This is in fact a highly desirable property from the point of view of implementing rational agents, because in providing a "filter of admissibility," intentions can be seen to constrain the space of possible intentions that an agent needs to consider.

Finally, intentions are closely related to beliefs about the future. For example, if I intend to become an academic, then I should believe that, assuming some certain background conditions are satisfied, I will indeed become an academic. For if I truly believe that I will never be an academic, it would be nonsensical of me to have an intention to become one. Thus if I intend to become an academic, I should at least believe that there is a good chance I will indeed become one. However, there is what appears at first sight to be a paradox here. While I might believe that I will indeed succeed in achieving my intention, if I am rational, then I must also recognize the possibility that I can *fail* to bring it about — that there is some circumstance under which my intention is not satisfied.

From this discussion, we can identify the following closely related situations:

• Having an intention to bring about φ, while believing that you will not bring about φ is called *intention-belief inconsistency*, and is not rational (see, e.g., [20, pp.37–38]).

• Having an intention to achieve φ without believing that φ will be the case is *intention-belief incompleteness*, and is an acceptable property of rational agents (see, e.g., [20, p.38]).

The distinction between these two cases is known as the *asymmetry thesis* [20, pp.37–41].

Summarizing, we can see that intentions play the following important roles in practical reasoning:

• *Intentions drive means-ends reasoning.*
If I have formed an intention, then I will attempt to achieve the intention, which involves, among other things, deciding *how* to achieve it. Moreover, if one particular course of action fails to achieve an intention, then I will typically attempt others.

• *Intentions persist.*
I will not usually give up on my intentions without good reason — they will persist, typically until I believe I have successfully achieved them, I believe I cannot achieve them, or I believe the reason for the intention is no longer present.

• *Intentions constrain future deliberation.*
I will not entertain options that are inconsistent with my current intentions.

• *Intentions influence beliefs upon which future practical reasoning is based.*
If I adopt an intention, then I can plan for the future on the assumption that I will achieve the intention. For if I intend to achieve some state of affairs while simultaneously believing that I will not achieve it, then I am being irrational.

Notice from this discussion that intentions *interact* with an agent's beliefs and other mental states. For example, having an intention to φ implies that I do not believe φ is impossible, and moreover that I believe given the right circumstances, φ will be achieved. However, satisfactorily capturing the interaction between intention and belief turns out to be surprisingly hard: the way in which intentions interact with other mental states is considered in chapter 5.

```
Algorithm: Agent Control Loop Version 1
1.  while true
2.       observe the world;
3.       update internal world model;
4.       deliberate about what intention to achieve next;
5.       use means-ends reasoning to get a plan for the intention;
6.       execute the plan
7.  end while
```

Figure 2.1
A basic agent control loop.

2.3 Implementing Rational Agents

Based on the discussion above, let us consider how we might go about building a BDI agent. The strategy I use is to introduce progressively more complex agent designs, and for each of these designs, to investigate the type of behavior that such an agent would exhibit, compared to the desired behavior of a rational agent as discussed above. This then motivates the introduction of a refined agent design, and so on.

The first agent design is shown in Figure 2.1. The agent continually executes a cycle of observing the world, deciding what intention to achieve next, determining a plan of some kind to achieve this intention, and then executing this plan.

The first point to note about this loop is that we will not be concerned with stages (2) or (3). Observing the world and updating an internal model of it are important processes, worthy of study in their own right; but they lie outside the scope of this volume. Instead, we are concerned primarily with stages (4) to (6). Readers interested in understanding the processes of observing and updating can find some key references in the "further reading" section at the end of this chapter.

One of the most important issues raised by this simple agent control loop follows from the fact that the deliberation and means-ends reasoning processes are not instantaneous. They have a *time cost* associated with them. Suppose that the agent starts deliberating at time t_0, begins means-ends reasoning at t_1, and subsequently begins executing the newly formed plan at time t_2. The time taken to deliberate is thus

$$t_{deliberate} = t_1 - t_0$$

planning are a major research topic in their own right — see [2] for a detailed survey. For our purposes, a simple model of plans will suffice. A plan is viewed as a tuple consisting of:

- a *pre-condition*, which defines the circumstances under which a plan is applicable — for example, one of the pre-conditions of the plan to catch a taxi to the airport is that I have sufficient funds to pay for it;
- a *post-condition*, which defines what states of affairs the plan achieves — for example, post-conditions of my plan to catch a taxi to the airport include me having less money, and me now being located at the airport;
- a *body*, which is the actual "recipe" part of the plan — for our purposes, the body is simply a sequence of actions.

We will use π (with decorations: $\pi', \pi_1, \ldots$) to denote plans, and let *Plan* be the set of all plans (over some set of actions *Ac*). We will make use of a number of auxiliary definitions for manipulating plans (some of these will not actually be required until later in this chapter):

- if π is a plan, then we write $pre(\pi)$ to denote the pre-condition of π, $post(\pi)$ to denote the post-condition of π, and $body(\pi)$ to denote the body of π;
- if π is plan, then we write $empty(\pi)$ to mean that plan π is the empty sequence (thus $empty(\ldots)$ is a boolean-valued function);
- $execute(\ldots)$ is a procedure that takes as input a single plan and executes it without stopping — executing a plan simply means executing each action in the plan body in turn;
- if π is a plan then by $hd(\pi)$ we mean the plan made up of the first action in the plan body of π; for example, if the body of π is $\alpha_1, \ldots, \alpha_n$, then the body of $hd(\pi)$ contains only the action α_1;
- if π is a plan then by $tail(\pi)$ we mean the plan made up of all but the first action in the plan body of π; for example, if the body of π is $\alpha_1, \alpha_2, \ldots, \alpha_n$, then the body of $tail(\pi)$ contains actions $\alpha_2, \ldots, \alpha_n$;
- if π is a plan, $I \subseteq Int$ is a set of intentions, and $B \subseteq Bel$ is a set of beliefs, then we write $sound(\pi, I, B)$ to mean that π is a sound plan for achieving I given beliefs B. (We will not discuss or attempt to define what makes a plan sound here — the classic paper on the subject is Lifschitz [136].)

We can now define the components of an agent's control loop. An agent's belief update process is formally modeled as a *belief revision function*. Such a belief

revision function has the following signature.

$$brf : \wp(Bel) \times Per \rightarrow \wp(Bel)$$

In other words, on the basis of the current beliefs and current percept, the belief revision function determines a new set of beliefs. (As noted above, in this book we are *not* concerned with how belief revision might work; see [71].)

The agent's deliberation process is given by a function

$$deliberate : \wp(Bel) \rightarrow \wp(Int)$$

which takes a set of beliefs and returns a set of intentions — those selected by the agent to achieve, on the basis of its beliefs.

An agent's means-ends reasoning is represented by a function

$$plan : \wp(Bel) \times \wp(Int) \rightarrow Plan$$

which on the basis of an agent's current beliefs and current intentions, determines a plan to achieve the intention. Note that there is nothing in the definition of the $plan(\ldots)$ function which requires an agent to engage in *plan generation* — constructing a plan from scratch [2]. In most BDI systems, the $plan(\ldots)$ function is implemented by giving the agent a *plan library* [78]. A plan library is a pre-assembled collection of plans, which an agent designer gives to an agent. Finding a plan to achieve an intention then simply involves a single pass through the plan library to find a plan that, when executed, will have the intention as a post-condition, and will be sound given the agent's current beliefs. In implemented BDI agents, pre- and post-conditions are often represented as (lists of) atoms of first-order logic, and beliefs and intentions as ground atoms of first-order logic. Finding a plan to achieve an intention then reduces to finding a plan whose pre-condition unifies with the agent's beliefs, and whose post-condition unifies with the intention.

The agent control loop is now as shown in Figure 2.2. This algorithm highlights some limitations of this simple approach to agent control. In particular, steps (4) to (7) inclusive implicitly assume that the environment has not changed since it was observed at stage (3). Assuming that the time taken to actually execute the plan dominates the time taken to revise beliefs, deliberate, or plan, then the crucial concern is that the environment might change while the plan is being executed. The problem is that the agent remains *committed* to the intention it forms at step (5) until it has executed the plan in step (7). If the environment changes after step (3), then the assumptions upon which this plan

```
Algorithm: Agent Control Loop Version 2
1.  B := B0;      /* B0 are initial beliefs */
2.  while true do
3.       get next percept ρ;
4.       B := brf(B, ρ);
5.       I := deliberate(B);
6.       π := plan(B, I);
7.       execute(π)
8.  end while
```

Figure 2.2
A first refinement of the agent control loop.

depends may well be invalid by the time the plan is actually executed.

2.4 The Deliberation Process

So far, we have glossed over the problem of exactly how an agent might go about deliberating. In this section, we consider the process in more detail. It is not hard to see that in real life, deliberation typically begins by trying to understand what the *options* available to you are. Returning to the career choice example introduced above, if you gain a good first degree, then one option is that of becoming an academic; if you fail to obtain a good degree, this option is not available to you. Another option is entering industry. After deciding what the options are, you must *choose between them*, and *commit* to some. These chosen options then become intentions.

From this discussion, we can see that the *deliberate* function as discussed above can be decomposed into two distinct functional components:

- *option generation* — in which the agent generates a set of possible alternatives; and
- *filtering* — in which the agent chooses between competing alternatives, and commits to achieving them.

We represent option generation via a function, *options*, which takes the agent's current beliefs and current intentions, and from them determines a set of options that we will hereafter refer to as *desires*. The intuitive interpretation of a desire is that, in an "ideal world," an agent would like *all* its desires achieved. In any moderately realistic scenario, however, an agent will not be able to

```
Algorithm: Agent Control Loop Version 3
1.
2.   B := B0;        /* B0 are initial beliefs */
3.   I := I0;        /* I0 are initial intentions */
4.   while true do
5.        get next percept ρ;
6.        B := brf(B, ρ);
7.        D := options(B, I);
8.        I := filter(B, D, I);
9.        π := plan(B, I);
10.       execute(π)
11.  end while
```

Figure 2.3
Refining the deliberation process into option generation and filtering.

achieve all its desires. This is because desires are often mutually exclusive. For example, in the OASIS air-traffic control system, which was implemented using a BDI architecture, an agent was tasked with the problem of finding the optimal sequence in which to land aircraft at an airport [138]. The option generation process in OASIS might generate the options of landing two different aircraft on the same runway at the same time. Clearly, such options are mutually exclusive: it would be undesirable to land both aircraft simultaneously.

Formally, the signature of the option generation function *options* is as follows.

$$options : \wp(Bel) \times \wp(Int) \rightarrow \wp(Des)$$

In order to select between competing options, an agent uses a *filter* function. Intuitively, the filter function must simply select the "best" option for the agent to commit to. We represent the filter process through a function *filter*, with a signature as follows.

$$filter : \wp(Bel) \times \wp(Des) \times \wp(Int) \rightarrow \wp(Int)$$

The agent control loop incorporating explicit deliberation and means-ends reasoning is shown in Figure 2.3. Notice that desires are an *input to* the filter process, whereas intentions are an *output from* it.

2.5 Commitment Strategies

When an option successfully passes through the *filter* function and is hence chosen by the agent as an intention, we say that the agent has made a *commitment* to that option. Commitment implies *temporal persistence* — an intention, once adopted, should not immediately evaporate. A critical issue is just *how* committed an agent should be to its intentions. That is, how long should an intention persist? Under what circumstances should an intention vanish?

To motivate the discussion further, consider the following scenario:

> Some time in the not-so-distant future, you are having trouble with your new household robot. You say "Willie, bring me a beer." The robot replies "OK boss." Twenty minutes later, you screech "Willie, why didn't you bring me that beer?" It answers "Well, I intended to get you the beer, but I decided to do something else." Miffed, you send the wise guy back to the manufacturer, complaining about a lack of commitment. After retrofitting, Willie is returned, marked "Model C: The Committed Assistant." Again, you ask Willie to bring you a beer. Again, it accedes, replying "Sure thing." Then you ask: "What kind of beer did you buy?" It answers: "Genessee." You say "Never mind." One minute later, Willie trundles over with a Genessee in its gripper. This time, you angrily return Willie for overcommitment. After still more tinkering, the manufacturer sends Willie back, promising no more problems with its commitments. So, being a somewhat trusting customer, you accept the rascal back into your household, but as a test, you ask it to bring you your last beer. Willie again accedes, saying "Yes, Sir." (Its attitude problem seems to have been fixed.) The robot gets the beer and starts towards you. As it approaches, it lifts its arm, wheels around, deliberately smashes the bottle, and trundles off. Back at the plant, when interrogated by customer service as to why it had abandoned its commitments, the robot replies that according to its specifications, it kept its commitments as long as required — commitments must be dropped when fulfilled or impossible to achieve. By smashing the bottle, the commitment became unachievable. [35, pp.213–214]

The mechanism an agent uses to determine when and how to drop intentions is known as a *commitment strategy*. The following three commitment strategies are commonly discussed in the literature of rational agents [187]:

- *Blind commitment*

  A blindly committed agent will continue to maintain an intention until it believes the intention has actually been achieved. Blind commitment is also sometimes referred to as *fanatical* commitment.

- *Single-minded commitment*

  A single-minded agent will continue to maintain an intention until it believes that either the intention has been achieved, or else that it is no longer possible

REASONING ABOUT RATIONAL AGENTS

Michael Wooldridge

One goal of modern computer science is to engineer computer programs that can act as autonomous, rational agents, software that can independently make good decisions about what actions to perform on our behalf and can execute those actions. Applications range from small programs that intelligently search the Web buying and selling goods via electronic commerce, to autonomous space probes. This book focuses on the belief-desire-intention (BDI) model of rational agents, which recognizes the primacy of beliefs, desires, and intentions in rational action. The BDI model has three distinct strengths: an underlying philosophy based on practical reasoning in humans, a software architecture that is implementable in real systems, and a family of logics that support a formal theory of rational agency.

The book introduces a BDI logic called LORA (Logic of Rational Agents). In addition to the BDI component, LORA contains a temporal component, which allows one to represent the dynamics of how agents and their environments change over time, and an action component, which allows one to represent the actions that agents perform and the effects of the actions. The book shows how LORA can be used to capture many components of a theory of rational agency, including such notions as communication and cooperation.

Michael Wooldridge is Professor of Computer Science and Head of the Agents Research Group at the University of Liverpool.

The MIT Press
Massachusetts Institute of Technology
Cambridge, Massachusetts 02142
http://mitpress.mit.edu

0-262-23213-8