Chapter 1

Computational Intelligence and Knowledge

1.1 What Is Computational Intelligence?

Computational intelligence is the study of the design of intelligent agents. An **agent** is something that acts in an environment—it does something. Agents include worms, dogs, thermostats, airplanes, humans, organizations, and society. An **intelligent agent** is a system that acts intelligently: What it does is appropriate for its circumstances and its goal, it is flexible to changing environments and changing goals, it learns from experience, and it makes appropriate choices given perceptual limitations and finite computation.

The central scientific goal of computational intelligence is to understand the principles that make intelligent behavior possible, in natural or artificial systems. The main hypothesis is that reasoning is computation. The central engineering goal is to specify methods for the design of useful, intelligent artifacts.

Artificial or Computational Intelligence?

Artificial intelligence (AI) is the established name for the field we have defined as computational intelligence (CI), but the term "artificial intelligence" is a source of much confusion. Is artificial intelligence real intelligence? Perhaps not, just as an artificial pearl is a fake pearl, not a real pearl. "Synthetic intelligence" might be a better name, since, after all, a synthetic pearl may not be a natural pearl but it is a real pearl. However, since we claimed that the central scientific goal is to understand both natural

and artificial (or synthetic) systems, we prefer the name "computational intelligence." It also has the advantage of making the computational hypothesis explicit in the name.

The confusion about the field's name can, in part, be attributed to a confounding of the field's purpose with its methodology. The purpose is to understand how intelligent behavior is possible. The methodology is to design, build, and experiment with computational systems that perform tasks commonly viewed as intelligent. Building these artifacts is an essential activity since computational intelligence is, after all, an empirical science; but it shouldn't be confused with the scientific purpose.

Another reason for eschewing the adjective "artificial" is that it connotes simulated intelligence. Contrary to another common misunderstanding, the goal is not to simulate intelligence. The goal is to understand real (natural or synthetic) intelligent systems by synthesizing them. A simulation of an earthquake isn't an earthquake; however, we want to actually create intelligence, as you could imagine creating an earthquake. The misunderstanding comes about because most simulations are now carried out on computers. However, you shall see that the digital computer, the archetype of an interpreted automatic, formal, symbol-manipulation system, is a tool unlike any other: It can produce the real thing.

The obvious intelligent agent is the human being. Many of us feel that dogs are intelligent, but we wouldn't say that worms, insects, or bacteria are intelligent (Exercise 1.1). There is a class of intelligent agents that may be more intelligent than humans, and that is the class of *organizations*. Ant colonies are the prototypical example of organizations. Each individual ant may not be very intelligent, but an ant colony can act more intelligently than any individual ant. The colony can discover food and exploit it very effectively as well as adapt to changing circumstances. Similarly, companies can develop, manufacture, and distribute products where the sum of the skills required is much more than any individual could understand. Modern computers, from the low-level hardware to high-level software, are more complicated than can be understood by any human, yet they are manufactured daily by organizations of humans. Human *society* viewed as an agent is probably the most intelligent agent known. We take inspiration from both biological and organizational examples of intelligence.

Flying Machines and Thinking Machines

It is instructive to consider an analogy between the development of flying machines over the last few centuries and the development of thinking machines over the last few decades.

First note that there are several ways to understand flying. One is to dissect known flying animals and hypothesize their common structural features as necessary fundamental characteristics of any flying agent. With this method an examination of birds, bats, and insects would suggest that flying involves the flapping of wings made of some structure covered with feathers or a membrane. Furthermore, the hypothesis

could be verified by strapping feathers to one's arms, flapping, and jumping into the air, as Icarus did. You might even imagine that some enterprising researchers would claim that one need only add enough appropriately layered feather structure to achieve the desired flying competence, or that improved performance required more detailed modeling of birds such as adding a cloaca.

An alternate methodology is to try to understand the principles of flying without restricting ourselves to the natural occurrences of flying. This typically involves the construction of artifacts that embody the hypothesized principles, even if they do not behave like flying animals in any way except flying. This second method has provided both useful tools, airplanes, and a better understanding of the principles underlying flying, namely *aerodynamics*.

It is this difference which distinguishes computational intelligence from other cognitive science disciplines. CI researchers are interested in testing general hypotheses about the nature of intelligence by building machines which are intelligent and which don't simply mimic humans or organizations. This also offers an approach to the question "Can computers really think?" by considering the analogous question "Can airplanes really fly?"

Technological Models of Mind

Throughout human history, people have used technology to model themselves. Consider this Taoist parable taken from the book *Lieh Tzu*, attributed to Lieh Yu-Khou:

> "Who is that man accompanying you?" asked the king. "That, Sir," replied Yen Shih, "is my own handiwork. He can sing and he can act." The king stared at the figure in astonishment. It walked with rapid strides, moving its head up and down, so that anyone would have taken it for a live human being. The artificer touched its chin, and it began singing, perfectly in tune. He touched its hand and it began posturing, keeping perfect time …. The king, looking on with his favorite concubine and other beauties, could hardly persuade himself that it was not real. As the performance was drawing to an end, the robot winked its eye and made advances to the ladies in attendance, whereupon the king became incensed and would have had Yen Shih executed on the spot had not the latter, in mortal fear, instantly taken the robot to pieces to let him see what it really was. And, indeed, it turned out to be only a construction of leather, wood, glue and lacquer, variously colored white, black, red and blue. Examining it closely, the king found all the internal organs complete—liver, gall, heart, lungs, spleen, kidneys, stomach and intestines; and over these again, muscles, bones and limbs with their joints, skin, teeth and

> hair, all of them artificial. Not a part but was fashioned with the utmost nicety and skill; and when it was put together again, the figure presented the same appearance as when first brought in. The king tried the effect of taking away the heart, and found that the mouth could no longer speak; he took away the liver and the eyes could no longer see; he took away the kidneys and the legs lost their power of locomotion. The king was delighted.

This story, dating from about the third century B.C., is one of the earliest written accounts of building intelligent agents, but the temples of early Egypt and Greece also bear witness to the universality of this activity. Each new technology has been exploited to build intelligent agents or models of mind. Clockwork, hydraulics, telephone switching systems, holograms, analog computers, and digital computers have all been proposed both as technological metaphors for intelligence and as mechanisms for modeling mind.

Parenthetically, we speculate that one reason for the king's delight was that he realized that functional equivalence doesn't necessarily entail structural equivalence. In order to produce the functionality of intelligent behavior it isn't necessary to reproduce the structural connections of the human body.

This raises the obvious question of whether the digital computer is just another technological metaphor, perhaps a fad soon to be superseded by yet another mechanism. In part, the answer must be empirical. We need to wait to see if we can get substantial results from this approach, but also to pursue alternate models to determine if they are more successful. We have reason to believe the answer to that question is "no." Some reasons are empirical: The results to date are impressive but not, of course, conclusive. There are other reasons. Consider the following two hypotheses. The first is called the **symbol-system hypothesis**:

> Reasoning is symbol manipulation.

The second hypothesis is called the **Church–Turing thesis**:

> Any symbol manipulation can be carried out on a Turing machine.

A Turing machine is an idealization of a digital computer with an unbounded amount of memory. These hypotheses imply that any symbol manipulation, and so any reasoning, can be carried out on a large enough deterministic computer.

There is no way you can prove these two hypothesis mathematically. All you can do is empirically test then by building reasoning systems. Why should you believe that they are true or even reasonable? The reason is that language, which provides one of the few windows to the mind, is inherently about transmission of symbols. Reasoning in terms of language has symbols as inputs and outputs, and so the function from inputs to outputs can be described symbolically, and presumably can be implemented in terms

of symbol manipulation. Also the intelligence that is manifest in an organization or in society is transmitted by language and other signals. Once you have expressed something in a language, reasoning about it is symbol manipulation. These hypotheses don't tell us how to implement arbitrary reasoning on a computer—this is CI's task. What it does tell us is that computation is an appropriate metaphor for reasoning.

This hypothesis doesn't imply that every detail of computation can be interpreted symbolically. Nor does it imply that every machine instruction in a computer or the function of every neuron in a brain can be interpreted symbolically. What it does mean is that there is a level of abstraction in which you can interpret reasoning as symbol manipulation, and that this level can explain an agent's actions in terms of its inputs.

Before you accept this hypothesis, it is important to consider how it may be wrong. An alternative is that action is some continuous function of the inputs to an agent such that the intermediate values don't necessarily correspond to anything meaningful. It is even possible that the functionality can't be interpreted symbolically, without resorting to using meaningless numbers. Alternative approaches are being pursued in both neural networks (page 408) and in building reactive robots (page 443) inspired by artificial insects.

Science and Engineering

As suggested by the flying analogy, there is tension between the science of CI, trying to understand the principles behind reasoning, and the engineering of CI, building programs to solve particular problems. This tension is an essential part of the discipline.

As CI is a science, its literature should manifest the scientific method, especially the creation and testing of refutable theories. Obvious questions are, "What are CI theories about?" and "How would I test one if I had one?" CI theories are about how interesting problems can be represented and solved by machine. Theories are supported empirically by constructing implementations, part of whose quality is judged by traditional computer science principles. You can't accomplish CI without specifying theories and building implementations; they are inextricably connected. Of course, not every researcher needs to do both, but both must be done. An experiment means nothing without a theory against which to evaluate it, and a theory without potentially confirming or refuting evidence is of little use. Ockham's Razor is our guide: Always prefer simple theories and implementations over the more complex.

With these thoughts in mind, you can quickly consider one of the most often considered questions that arises in the context of CI: "Is human behavior algorithmic?" You can dispense with this question and get on with your task by acknowledging that the answer to this question is unknown; it is part of cognitive science and CI's goal to find out.

Relationship to Other Disciplines

CI is a very young discipline. Other disciplines as diverse as philosophy, neurobiology, evolutionary biology, psychology, economics, political science, sociology, anthropology, control engineering, and many more have been studying intelligence much longer. We first discuss the relationship with philosophy, psychology, and other disciplines which study intelligence; then we discuss the relationship with computer science, which studies how to compute.

The science of of CI could be described as "synthetic psychology," "experimental philosophy," or "computational epistemology"—**Epistemology** is the study of knowledge. It can be seen as a way to study the old problem of the nature of knowledge and intelligence, but with a more powerful experimental tool than was previously available. Instead of being able to observe only the external behavior of intelligent systems, as philosophy, psychology, economics, and sociology have traditionally been able to do, we are able to experiment with executable models of intelligent behavior. Most importantly, such models are open to inspection, redesign, and experiment in a complete and rigorous way. In other words, you now have a way to construct the models that philosophers could only theorize about. You can experiment with these models, as opposed to just discussing their abstract properties. Our theories can be empirically grounded in implementation.

Just as the goal of aerodynamics isn't to synthesize birds, but to understand the phenomenon of flying by building flying machines, CI's ultimate goal isn't necessarily the full-scale simulation of human intelligence. The notion of psychological validity separates CI work into two categories: that which is concerned with mimicking human intelligence—often called *cognitive modeling*—and that which isn't.

To emphasize the development of CI as a science of intelligence, we are concerned, in this book at least, not with psychological validity but with the more practical desire to create programs that solve real problems. Sometimes it will be important to have the computer to reason through a problem in a human-like fashion. This is especially important when a human requires an explanation of how the computer generated an answer. Some aspects of human cognition you usually do not want to duplicate, such as the human's poor arithmetic skills and propensity for error.

Computational intelligence is intimately linked with the discipline of computer science. While there are many non-computer scientists who are researching CI, much, if not most, CI (or AI) research is done within computer science departments. We believe this is appropriate, as the study of computation is central to CI. It is essential to understand algorithms, data structures, and combinatorial complexity in order to build intelligent machines. It is also surprising how much of computer science started as a spin off from AI, from timesharing to computer algebra systems.

There are other fields whose goal is to build machines that act intelligently. Two of these fields are control engineering and operations research. These start from different

points than CI, namely in the use of continuous mathematics. As building real agents involves both continuous control and CI-type reasoning, these disciplines should be seen as symbiotic with CI. A student of either discipline should understand the other. Moreover, the distinction between them is becoming less clear with many new theories combining different areas. Unfortunately there is too much material for this book to cover control engineering and operations research, even though many of the results, such as in search, have been studied in both the operations research and CI areas.

Finally, CI can be seen under the umbrella of *cognitive science*. Cognitive science links various disciplines that study cognition and reasoning, from psychology to linguistics to anthropology to neuroscience. CI distinguishes itself within cognitive science because it provides tools to build intelligence rather than just studying the external behavior of intelligent agents or dissecting the inner workings of intelligent systems.

1.2 Agents in the World

There are many interesting philosophical questions about the nature and substance of CI, but the bottom line is that, in order to understand how intelligent behavior might be algorithmic, you must attempt to program a computer to solve actual problems. It isn't enough to merely speculate that some particularly interesting behavior is algorithmic. You must develop a theory that explains how that behavior can be manifest in a machine, and then you must show the feasibility of that theory by constructing an implementation. We are interested in practical reasoning: reasoning in order to do something. Such a coupling of perception, reasoning, and acting comprises an *agent*. An agent could be, for example, a coupling of a computational engine with physical actuators and sensors, called a *robot*. It could be the coupling of an advice-giving computer—an *expert system*—with a human who provides the perceptual information and who carries out the task. An agent could be a program that acts in a purely computational environment—an *infobot*.

Figure 1.1 shows the inputs and outputs of an agent. At any time the agent has:

- Prior knowledge about the world
- Past experience that it can learn from
- Goals that it must try to achieve or values about what is important
- Observations about the current environment and itself

and it does some action. For each agent considered, we specify the forms of the inputs and the actions. The goal of this book is to consider what is in the black box so that the action is reasonable given the inputs.

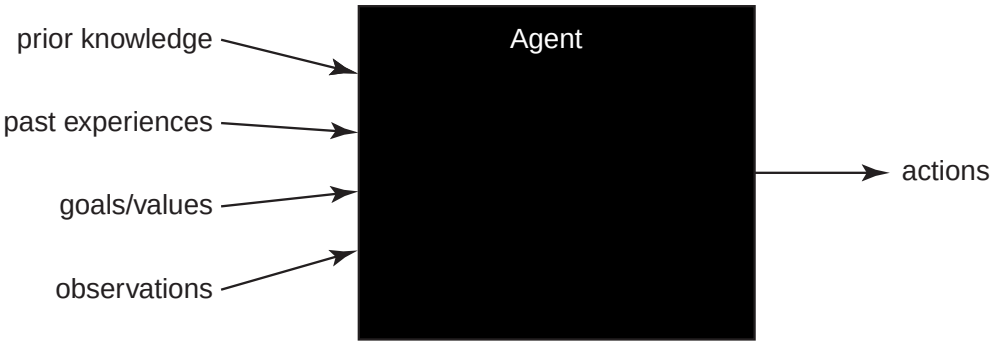

Figure 1.1: An agent as a black box

For our purpose, the world consists of an agent in an environment. The agent's environment may well include other agents. Each agent has some internal state that can encode beliefs about its environment and itself. It may have goals to achieve, ways to act in the environment to achieve those goals, and various means to modify its beliefs by reasoning, perception, and learning. This is an all-encompassing view of intelligent systems varying in complexity from a simple thermostat to a team of mobile robots to a diagnostic advising system whose perceptions and actions are mediated by human beings.

Success in building an intelligent agent naturally depends on the problem that one selects to investigate. Some problems are very well-suited to the use of computers, such as sorting a list of numbers. Others seem not to be, such as changing a baby's diaper or devising a good political strategy. We have chosen some problems that are representative of a range of applications of current CI techniques. We seek to demonstrate, by case study, CI's methodology with the goal that the methodology is transferable to various problems in which you may be interested. We establish a framework that places you, the reader, in a position to evaluate the current CI literature and anticipate the future; and, most importantly, we develop the concepts and tools necessary to allow you to build, test, and modify intelligent agents. Finally we must acknowledge there is still a huge gulf between the dream of computational intelligence and the current technology used in the practice of building what we now call intelligent agents. We believe we have many of the tools necessary to build intelligent agents, but we are certain we don't have all of them. We could, of course, be on the wrong track; it is this fallibility that makes CI science and makes the challenge of CI exciting.

1.3 Representation and Reasoning

Experience shows that the performance of tasks that seem to involve intelligence also seem to require a huge store of knowledge. A major thesis of this book is that CI is the study of knowledge. This raises the question which is part of our subject material, "What is knowledge?" Informally, knowledge is information about some domain or subject area, or about how to do something. Much of our effort will be devoted to formalizing and refining a common-sense notion of knowledge, with the motivation of developing both a theoretical and practical framework for representing and using knowledge.

Humans require and use a lot of knowledge to carry out even the most simple common-sense tasks. Computers are very good at tasks which do not require much knowledge, such as simple arithmetic, symbolic differentiation, or sorting. They aren't, as yet, very good at many knowledge-intensive tasks at which humans excel, such as recognizing faces in a picture, medical diagnosis, understanding natural language, or legal argumentation. At the heart of this book is the design of computational systems that have knowledge about the world and that can act in the world based on that knowledge. The notion of knowledge is central to this book. The systems we want to develop should be able to acquire and use knowledge to solve the problems at hand. The main issues are how to acquire and represent knowledge about some domain and how to use that knowledge to answer questions and solve problems.

You will notice that we make a strong commitment to *logic* approach in this book. Our commitment is really to a precise specification of meaning rather than to any particular syntax. We have no great commitment to any particular syntax. Many different notations are possible. Sometimes we will write sentences, sometimes we will use diagrams. In order to represent anything, you have to commit to some notation, and the simpler the better. We use Prolog's syntax, not because we particularly like Prolog or its syntax, but because it is important for scholars of CI to get experience with using logic to solve problems, and Prolog is probably the most accessible system that allows you to do this.

Representation and Reasoning System

In order to use knowledge and reason with it, you need what we call a *representation and reasoning system* (RRS). A representation and reasoning system is composed of a language to communicate with a computer, a way to assign meaning to the language, and procedures to compute answers given input in the language. Intuitively, an RRS lets you tell the computer something in a language where you have some meaning associated with the sentences in the language, you can ask the computer questions,

and the computer will produce answers that you can interpret according to the meaning associated with the language.

At one extreme, the language could be a low-level programming language such as Fortran, C++, or Lisp. In these languages the meaning of the sentences, the programs, is purely in terms of the steps the computer will carry out to execute the program. What computation will be carried out given a program and some input, is straightforward to determine. How to map from an informal statement of a problem to a representation of the problem in these RRSs, programming, is a difficult task.

At the other extreme, the language could be a natural language, such as English, where the sentences can refer to the problem domain. In this case, the mapping from a problem to a representation is not very difficult: You need to describe the problem in English. However, what computation needs to be carried out in the computer in response to the input is much more difficult to determine.

In between these two extremes are the RRSs that we consider in this book. We want RRSs where the distance from a natural specification of the problem to the representation of the problem is not very far. We also want RRSs where the appropriate computation, given some input, can be effectively determined. We consider languages for the specification of problems, the meaning associated with such languages, and what computation is appropriate given input in the languages.

One simple example of a representation and reasoning system between these two extremes is a database system. In a database system, you can tell the computer facts about a domain and then ask queries to retrieve these facts. What makes a database system into a representation and reasoning system is the notion of *semantics*. Semantics allows us to debate the truth of information in a knowledge base and makes such information knowledge rather than just data. In most of the RRSs we are interested in, the form of the information is more flexible and the procedures for answering queries are more sophisticated than in a database. A database typically has table lookup; you can ask about what is in the database, not about what else must be true, or is likely to be true, about the domain.

Chapter 2 gives a more precise definition an RRS and a particular RRS that is both simple and yet very powerful. It is this RRS that we build upon throughout this book, eventually presenting RRSs that can reason about such things as time, typicality, uncertainty, and action.

Ontology and Conceptualization

An important and fundamental prerequisite to using an RRS is to decide how a task domain is to be described. This requires us to decide what kinds of things the domain consists of, and how they are to be related in order to express task domain problems. A major impediment to a general theory of CI is that there is no comprehensive theory of how to appropriately conceive and express task domains. Most of what we know about

this is based on experience in developing and refining representations for particular problems.

Despite this fundamental problem, we recognize the need for the following commitments.

- The world can be described in terms of **individuals** (things) and relationships among individuals. An **ontology** is a commitment to what exists in any particular task domain. This notion of relationship is meant to include propositions that are true or false independently of any individuals, properties of single individuals, as well as relationships between pairs or more individuals. This assumption that the world can be described in terms of things is the same that is made in logic and natural language. This isn't a strong assumption, as individuals can be anything nameable, whether concrete or abstract. For example, people, colors, emotions, numbers, and times can all be considered as individuals. What is a "thing" is a property of an observer as much as it is a property of the world. Different observers, or even the same observer with different goals, may divide up the world in different ways.
- For each task or domain, you need to identify specific individuals and relations that can be used to express what is true about the world under consideration. How you do so can profoundly affect your ability to solve problems in that domain.

For most of this book we assume that the human who is representing a domain decides on the ontology and the relationships. To get human-level computational intelligence it must be the agent itself that decides how to divide up the world, and which relationships to reason about. However, it is important for you to understand what knowledge is required for a task before you can expect to build a computer to learn or introspect about how to solve a problem. For this reason we concentrate on what it takes to solve a problem. It should not be thought that the problem of CI is solved. We have only just begun this endeavor.

1.4 Applications

Theories about representation and reasoning are only useful insofar as they provide the tools for the automation of problem solving tasks. CI's applications are diverse, including medical diagnosis, scheduling factory processes, robots for hazardous environments, chess playing, autonomous vehicles, natural language translation systems, and cooperative systems. Rather than treating each application separately, we abstract essential features of such applications to allow us to study principles behind intelligent reasoning and action.

This section outlines three application domains that will be developed in examples throughout the book. Although the particular examples presented are simple—for otherwise they wouldn't fit into the book—the application domains are representative of the sorts of domains in which CI techniques can be, and have been, used.

The three application domains are:

- *An autonomous delivery robot* that can roam around a building delivering packages and coffee to people in the building. This delivery agent needs to be able to, for example, find paths, allocate resources, receive requests from people, make decisions about priorities, and deliver packages without injuring people or itself.
- *A diagnostic assistant* that helps a human troubleshoot problems and suggests repairs or treatments to rectify the problems. One example is an electrician's assistant that can suggest what may be wrong in a house, such as a fuse blown, a light switch broken, or a light burned out given some symptoms of electrical problems. Another example is of a medical diagnostician that finds potential diseases, possible tests, and appropriate treatments based on knowledge of a particular medical domain and a patient's symptoms and history. This assistant needs to be able to explain its reasoning to the person who is carrying out the tests and repairs, as that person is ultimately responsible for what they do. The diagnostic assistant must add substantial value in order to be worth using.
- *An "infobot"* that can search for information on a computer system for naive users such as company managers or people off the street. In order to do this the infobot must find out, using the user's natural language, what information is requested, determine where to find out the information, and access the information from the appropriate sources. It then must report its findings in an appropriate format so that the human can understand the information found, including what they can infer from the lack of information.

These three domains will be used for the motivation for the examples in the book. In the next sections we discuss each application domain in detail.

The Autonomous Delivery Robot

Imagine a robot that has wheels and can pick up objects and put them down. It has sensing capabilities so that it can recognize the objects it needs to manipulate and so it can avoid obstacles. It can be given orders in natural language and obey them, making common sense assumptions about what to do when its goals conflict. Such a robot could be used in an office environment to deliver packages, mail, or coffee. It needs to be useful as well as safe.

In terms of the black box definition of an agent in Figure 1.1, the autonomous delivery robot has as inputs:

- Prior knowledge in terms of knowledge about its capabilities, what objects it may encounter and need to differentiate, and perhaps some prior knowledge about its environment, such as maps.
- Past experience about, for instance, which actions are useful in which situations, what objects are in the world, how its actions affect its position, and experience about previous requests for it to act.
- Goals in terms of what it needs to deliver and when, as well as values that specify tradeoffs such as when it must forgo one goal to pursue another or the tradeoff between acting quickly and acting safely.
- Observations about its environment from such input devices as cameras, sonar, sound, laser range finders, or keyboards for requests.

The robot's output is motor controls that specify where its wheels should turn, where its limbs should move, and what it should do with it grippers.

In order for this robot to be able to function, it has to be able to:

- Determine where individuals' offices are, where to get coffee, how to estimate the length of a trip, and so on. This involves being able to infer information from a database of facts about the domain. How to infer implicit information from a knowledge base is explored in Chapters 2 and 3.
- Find a path between different locations. It may want the shortest, the quickest, or the safest path. This involves searching as developed in Chapter 4.
- Be able to represent knowledge about the domain so that inference can be quick, so that knowledge can be easily acquired, and so that the appropriate knowledge is represented. Such issues are discussed in Chapter 5.
- Plan how to carry out multiple goals, even when they use the same resources, for example, when the robot's carrying capacity is limited. Planning is discussed in Chapter 8.
- Make default assumptions—for example, about where people will be or where coffee can be found. See Chapter 9.
- Make tradeoffs about plans even though there may be uncertainty about what is in the world and about the outcome of its actions. Such reasoning under uncertainty is discussed in Chapter 10.
- Learn about features of its domain, as well as learn about how its actions affect its position and its rewards. See Chapter 11.
- Sense the world, know where it is, steer around the corridors (avoiding people and other objects), and pick up and put down objects. See Chapter 12.

Figure 1.2 depicts a typical laboratory environment for a delivery robot. This environment consists of four laboratories and many offices arranged in a grid. We assume that the robot can only push doors, and the directions of the doors in the

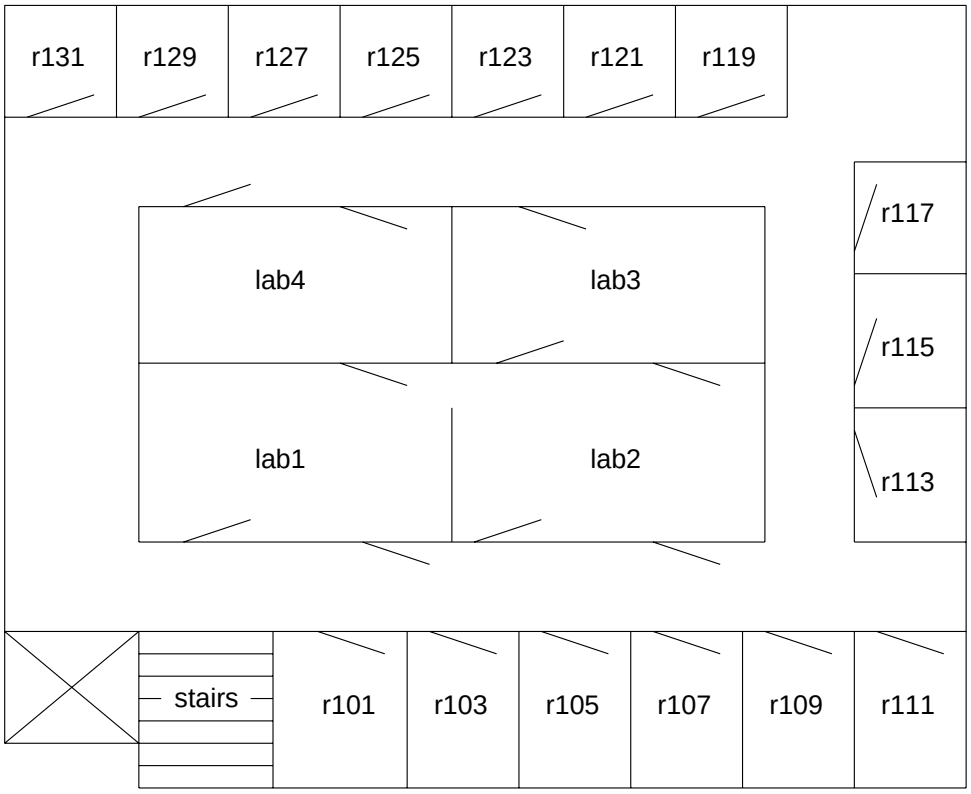

Figure 1.2: An environment for the delivery robot

diagram reflect the directions where the robot can travel. We also assume that rooms need keys, and that keys can be obtained from various sources. The robot needs to deliver parcels and letters from room to room. The environment also contains a stairway that can be hazardous to the robot.

The Diagnostic Assistant

A diagnostic assistant is intended to advise a human about some particular artifact, such as a medical patient, the electrical system in a house, or an automobile, when symptoms are manifest. It should advise about potential underlying faults or diseases, what tests to carry out, and what treatment to prescribe. In order to give such advice the assistant needs to have some model of the system, knowledge of potential causes, available tests, available treatments, and observations about a particular artifact. As-

sisting a human involves making sure that the system provides added value, is easy for a human to use, and isn't more trouble than it is worth. It must be able to justify why the suggested diagnoses or actions are appropriate. Humans are, and should be, suspicious of computer systems that are impenetrable. When humans are responsible for what they do, even if it is based on a computer system's advice, they need to have reasonable justifications for the suggested actions.

In terms of the black box definition of an agent in Figure 1.1, the diagnostic assistant has as inputs:

- Prior knowledge such as what's normal and what's abnormal about how switches and lights work, how diseases or malfunctions manifest themselves, what information tests provide, and the side effects of repairs or treatments.
- Past experience in terms of data of previous cases that include the effects of repairs or treatments, the prevalence of faults or diseases, the prevalence of symptoms for these faults or diseases, and the accuracy of tests.
- Goals of fixing the device and tradeoffs such as between fixing or replacing different components, or whether a patient prefers to live longer if it means they will be less coherent.
- Observations of symptoms of a device or patient.

The output of the diagnostic assistant is in terms of recommendations of treatments and tests, along with rationales for its recommendations.

In order for the diagnostic assistant to be useful it must be able to:

- Derive the effects of faults and interventions (Chapter 3).
- Search through the space of possible faults or disease complexes (Chapter 4).
- Explain its reasoning to the human who is using it (Chapter 6).
- Derive possible causes for symptoms; rule out other causes based on the symptoms (Chapter 7).
- Plan courses of tests and treatments to address the problems (Chapter 8).
- Hypothesize problems and use default knowledge that may not always be true (Chapter 9).
- Reason about the uncertainties about the artifact given only partial information about the state of the artifact, the uncertainty about the effects of the treatments, and the tradeoffs between the alternate courses of action (Chapter 10).
- Learn about what symptoms are associated with the faults or diseases, the effects of treatments, and the accuracy of tests (Chapter 11).

Figure 1.3 shows a depiction of the electrical distribution in a house. In this house, power comes into the house through circuit breakers, and then it goes to power outlets or to lights through light switches. For example, light l_1 is on if there is power coming

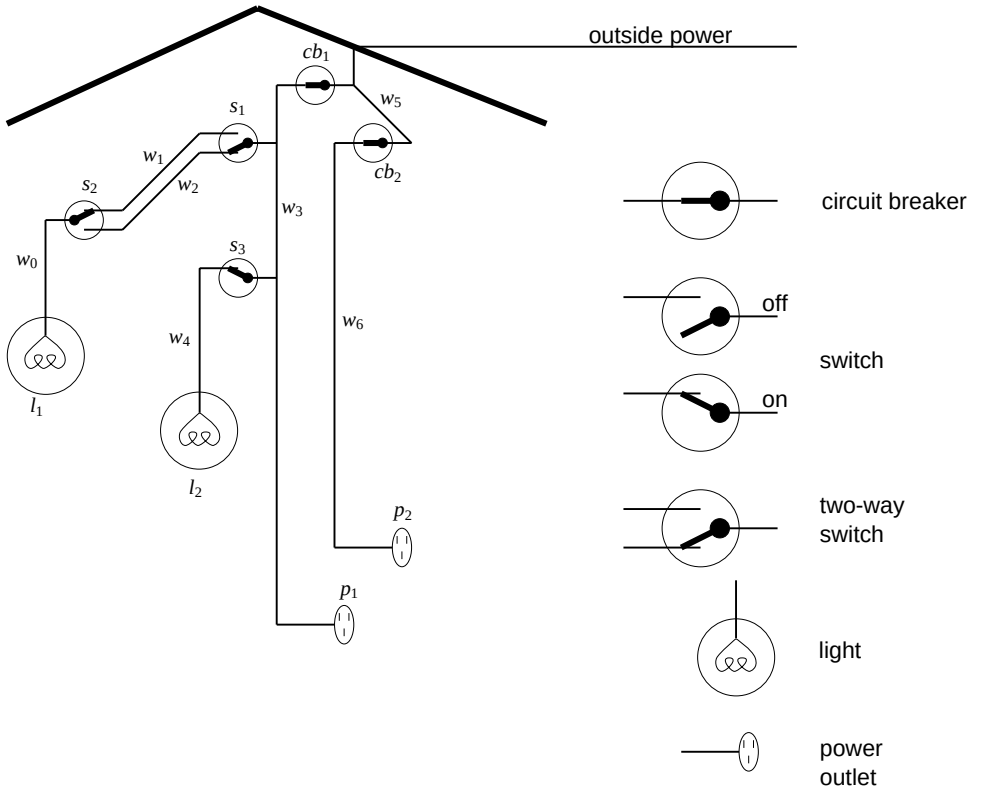

Figure 1.3: An electrical environment for the diagnostic assistant

into the house, if circuit breaker cb_1 is *on*, and if switches s_1 and s_2 are either both up or both down. This is the sort of model that a normal householder may have of the electrical power in the house which they could use to determine what is wrong given evidence about the position of the switches and which lights are on and which are off. The diagnostic assistant is there to help the householder or an electrician to troubleshoot electrical problems.

The Infobot

An **infobot** is like a robot, but instead of interacting with a physical environment, it interacts with an information environment. Its task is to extract information from a network of diverse information sources such as the Internet or a multimedia encyclopedia. The infobot must determine what information is needed from a query in a formal language, from a sophisticated user, or from a natural language query from a

casual user such as a manager or person off the street. It must determine where the information may be obtained, retrieve the information, and present it in a meaningful way to the user.

In terms of the black box definition of an agent in Figure 1.1, the infobot has as inputs:

- Prior knowledge about the meaning of words, the types of information sources, and how to access information.
- Past experience about where information can be obtained, the relative speed of various servers, and information about the preferences of the user.
- Goals in terms of what information it needs to find out and tradeoffs about how much expense should be involved to get the information and the tradeoff between the volume and quality of information.
- Observations about what information is at the current sites, what links are available, and the load on various connections.

The output of the infobot is information presented so that the user can understand what is there and the significance of missing information.

The infobot needs to be able to:

- Derive information that is only implicit in the knowledge base(s), as well as interact in natural language (Chapter 3).
- Search through a variety of knowledge bases looking for relevant information (Chapter 4).
- Find good representations of knowledge so that answers can be computed efficiently (Chapter 5).
- Explain how an answer was derived or why some information was unavailable (Chapter 6).
- Make conclusions about lack of knowledge, determine conflicting knowledge, and be able to conclude disjunctive knowledge (Chapter 7).
- Use default reasoning about where to obtain different information (Chapter 9).
- Make tradeoffs between cheap but unreliable information sources and more expensive but more comprehensive information sources (Chapter 10).
- Learn about what knowledge is available where, and what information the user is interested in (Chapter 11).

We consider two different infobots: the "unibot" and the "webbot." The **unibot** interacts with a database of information about courses, scheduling, degree requirements, and grades. The **webbot** interacts with the World Wide Web, finding information that may be of use to a user. One of the most interesting aspects of an infobot is that it ought to be able to volunteer information that users don't know exists, and so can't be expected to ask for even though they may be interested.

Common Features

These three examples have common features. At one level of abstraction, they each have four tasks:

Modeling the environment The robot needs to be able to model the physical environment, its own capabilities, and the mechanics of delivering parcels. The diagnostic assistant needs to be able to model the general course of diseases or faults, know how normal artifacts work, know how treatments work, and know what information tests provide. The infobot needs to be able to model how information can be obtained, what are legal answers to questions, and what information is actually needed, based on a request.

Evidential reasoning or perception This is what control theorists call "system identification" and what doctors call "diagnosis." Given some observations about the world, the task is to determine what is really in the world. This is most evident in the diagnostic assistant, where the system is given symptoms (observations) and has to determine the underlying faults or diseases. The delivery robot must try to determine where it is and what else is in its environment based on limited sensing information such as touch, sonar, or vision. The infobot has to determine where information is available, given only partial information about the contents of information sources.

Action Given a model of the world and a goal, the task is to determine what should be done. For the delivery robot, this means that it must actually do something, such as rove around the corridors and deliver things. For the diagnostic assistant, the actions are treatments and tests. It isn't enough to theorize about what may be wrong, but a diagnostician must make tests and has to consider what it will do based on the outcome of tests. It isn't necessary to test if the same treatment will be carried out no matter what the test's outcome, such as replacing a board on a computer or giving a patient an antibiotic. The actions of the infobot are computational, such as, consulting a knowledge base in order to extract some information.

Learning from past experience This includes learning what the particular environment is like, the building the delivery robot is in, the particular patient being diagnosed, or the communication bottlenecks of a computer network; learning general information, how the robot sensors actually work, how well particular diseases respond to particular treatments, or how fast different types of computer connections are; and learning how to solve problems more efficiently.

These tasks cut across all application domains. It is essentially the study of these four tasks that we consider in this book. These four tasks interact. It is difficult to study one without the others. We have decided that the most sensible organization is to build

the tools needed from the bottom up and to show how the tools can be used for each task and, through these tasks, demonstrate the limitations of the tools. We believe this organization will help in understanding the commonalities across different domains and in understanding the interaction among the different tasks.

1.5 Overview

Our quest for a unified view of CI is based on the fundamental nature of the concepts of representation and reasoning. We seek to present these techniques as an evolution of ideas used to solve progressively more difficult problems.

Chapter 2 starts with a simple representation and reasoning system, where we assume that the agents have complete knowledge about the world and that the world never changes. Subsequent chapters discuss the removal of such constraints in terms of their effect on representation and reasoning. In Chapter 3, we give specific examples of using a definite knowledge encoding for various useful applications. In Chapter 4, we show how many reasoning problems can be understood as search problems. We review some standard approaches to search-based problem solving, including various kinds of informed and uninformed search. Chapter 5 discusses knowledge representation issues and explains how they are manifest in the ideas developed in the first three chapters. Chapter 6 provides further details about knowledge-based systems and presents an overview of a system architecture of a typical expert system, including tools to enable an expert system to explain its reasoning. Chapter 7 removes the assumptions about definite knowledge, by allowing disjunctive and negative knowledge, culminating in full first-order predicate calculus and some aspects of modal logic. Chapter 8 removes the assumption that the world is static. To represent a dynamic world requires some notion of time or state change, which, in turn, introduces us to the planning problem. Chapter 9 discusses hypothetical reasoning and its application to default reasoning, diagnostic reasoning, and recognition. Chapter 10 introduces reasoning under uncertainty, representations for uncertain knowledge, and decision making under uncertainty. Chapter 11, which discusses learning, shows how previous experience can be used by an agent. Chapter 12 shows how the reasoning capabilities can be put together to build agents that perceive and interact in an environment.

Three appendices provide supplementary material including a glossary of terms used in CI, a Prolog tutorial, and implementations of various system components presented in the main text.

1.6 References and Further Reading

The ideas in this chapter have been derived from many sources. Here we will try to acknowledge those that are explicitly attributable to other authors. Most of the other ideas are part of AI folklore. To try to attribute them to anyone would be impossible.

Minsky (1986) presents a theory of intelligence as emergent from a "society" of unintelligent agents. Haugeland (1997) contains a good collection of articles on the philosophy behind computational intelligence.

Turing (1950) proposes an objective method for answering the question "Can machines think?" in terms of what is now known as the *Turing test*.

The symbol-system hypothesis is due to Newell & Simon (1976). See also Simon (1996) who discusses the role of symbol systems in a multi-disciplinary context. The distinctions between real, synthetic and artificial intelligence are discussed by Haugeland (1985), who also provides useful introductory material on interpreted, automatic formal symbol systems, and the Church–Turing thesis. For a critique of the symbol-system hypothesis see Winograd (1990). Wegner (1997) argues that computers that interact with the world may be more powerful than Turing machines, thus that the Church–Turing thesis is in fact false.

The Taoist story is from Needham's classic study of science and technology in China (Ronan, 1986).

For discussions on the foundations of AI and the breadth of research in AI see Kirsh (1991a), Bobrow (1993), and the papers in the corresponding volumes, as well as Schank (1990) and Simon (1995). The importance of knowledge in AI is discussed in Lenat & Feigenbaum (1991) and Smith (1991).

For overviews of cognitive science and the role that AI and other disciplines play in that field see Gardner (1985), Posner (1989), and Stillings, Feinstein, Garfield, Rissland, Rosenbaum, Weisler & Baker-Ward (1987).

A number of AI texts are valuable as reference books complementary to this book, providing a different perspective on AI. See classic books by Nilsson (1971; 1980), Genesereth & Nilsson (1987), Charniak & McDermott (1985) and more recent books including Ginsberg (1993), Russell & Norvig (1995) and Dean, Allen & Aloimonos (1995).

The Encyclopedia of Artificial Intelligence (Shapiro, 1992) is an encyclopedic reference on AI written by leaders in the field. There are a number of collections of classic research papers. The general collections of most interest to readers of this book include Webber & Nilsson (1981) and Brachman & Levesque (1985). More specific collections are given in the appropriate chapters.

There are many journals that provide in-depth research contributions and conferences where the most up-to-date research can be found. These include the journals

Artificial Intelligence, *Journal of Artificial Intelligence Research*, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *Computational Intelligence*, *International Journal of Intelligent Systems*, and *New Generation Computing*, as well as more specialized journals such as *Neural Computation*, *Computational Linguistics*, *Machine Learning*, *Journal of Automated Reasoning*, *Journal of Approximate Reasoning*, *IEEE Transactions on Robotics and Automation*, and the *Logic Programming Journal*. *AI Magazine*, published by the *American Association for Artificial Intelligence (AAAI)*, often has excellent overview articles and descriptions of particular applications. There are many conferences on Artificial Intelligence. Those of most interest to a general audience are the biennial *International Joint Conference on Artificial Intelligence (IJCAI)*, the *European Conference on AI (ECAI)*, the *Pacific Rim International Conference on AI (PRICAI)*, and various national conferences, especially the *American Association for Artificial Intelligence National Conference on AI*, and innumerable specialized conferences and workshops.

1.7 Exercises

Exercise 1.1 For each of the following, give five reasons why:

(a) A dog is more intelligent than a worm.

(b) A human is more intelligent that a dog.

(c) An organization is more intelligent than an individual human.

Based on these, give a definition of what "more intelligent" may mean.

Exercise 1.2 Give as many disciplines as you can whose aim is to study intelligent behavior of some sort. For each discipline find out where the behavior is manifest and what tools are used to study it. Be as liberal as you can as to what defines intelligent behavior.

Exercise 1.3 Choose a particular world, for example, what is on some part of your desk at the current time.

i) Get someone to list all of the things that exist in this world (or try it yourself as a thought experiment).

ii) Try to think of twenty things that they missed. Make these as different from each other as possible. For example, the ball at the tip of the right-most ball-point pen on the desk, or the spring in the stapler, or the third word on page 21 of a particular book on the desk.

iii) Try to find a thing that can't be described using natural language.

iv) Choose a particular task, such as making the desk tidy, and try to write down all of the things in the world at a level of description that is relevant to this task.

Based on this exercise, discuss the following statements:

(a) What exists in a world is a property of the observer.

(b) You need general constructors to describe individuals, rather than expecting each individual to have a separate name.

(c) What individuals exist is a property of the task as well as the world.

(d) To describe the individuals in a domain, you need what is essentially a dictionary of a huge number of words and ways to combine them to describe individuals, and this should be able to be done independently of any particular domain.