PART I

ARTIFICIAL INTELLIGENCE: ITS ROOTS AND SCOPE

Everything must have a beginning, to speak in Sanchean phrase; and that beginning must be linked to something that went before. Hindus give the world an elephant to support it, but they make the elephant stand upon a tortoise. Invention, it must be humbly admitted, does not consist in creating out of void, but out of chaos; the materials must, in the first place, be afforded. . . .

—MARY SHELLEY, Frankenstein

Artificial Intelligence: An Attempted Definition

Artificial intelligence (AI) may be defined as the branch of computer science that is concerned with the automation of intelligent behavior. This definition is particularly appropriate to this book in that it emphasizes our conviction that AI is a part of computer science and, as such, must be based on sound theoretical and applied principles of that field. These principles include the data structures used in knowledge representation, the algorithms needed to apply that knowledge, and the languages and programming techniques used in their implementation.

However, this definition suffers from the fact that intelligence itself is not very well defined or understood. Although most of us are certain that we know intelligent behavior when we see it, it is doubtful that anyone could come close to defining intelligence in a way that would be specific enough to help in the evaluation of a supposedly intelligent computer program, while still capturing the vitality and complexity of the human mind.

Thus the problem of defining artificial intelligence becomes one of defining intelligence itself: is intelligence a single faculty, or is it just a name for a collection of distinct and unrelated abilities? To what extent is intelligence learned as opposed to having an a priori existence? Exactly what does happen when learning occurs? What is creativity? What is intuition? Can intelligence be inferred from observable behavior, or does it require evidence of a particular internal mechanism? How is knowledge represented in the nerve tissue of a living being, and what lessons does this have for the design of intelligent machines? What is self-awareness; what role does it play in intelligence? Furthermore, is

it necessary to pattern an intelligent computer program after what is known about human intelligence, or is a strict "engineering" approach to the problem sufficient? Is it even possible to achieve intelligence on a computer, or does an intelligent entity require the richness of sensation and experience that might be found only in a biological existence?

These are unanswered questions, and all of them have helped to shape the problems and solution methodologies that constitute the core of modern AI. In fact, part of the appeal of artificial intelligence is that it offers a unique and powerful tool for exploring exactly these questions. AI offers a medium and a test-bed for theories of intelligence: such theories may be stated in the language of computer programs and consequently tested and verified through the execution of these programs on an actual computer.

For these reasons, our initial definition of artificial intelligence seems to fall short of unambiguously defining the field. If anything, it has only led to further questions and the paradoxical notion of a field of study whose major goals include its own definition. But this difficulty in arriving at a precise definition of AI is entirely appropriate. Artificial intelligence is still a young discipline, and its structure, concerns, and methods are less clearly defined than those of a more mature science such as physics.

Artificial intelligence has always been more concerned with expanding the capabilities of computer science than with defining its limits. Keeping this exploration grounded in sound theoretical principles is one of the challenges facing AI researchers in general and this book in particular.

Because of its scope and ambition, artificial intelligence defies simple definition. For the time being, we will simply define it as *the collection of problems and methodologies studied by artificial intelligence researchers.* This definition may seem silly and meaningless, but it makes an important point: artificial intelligence, like every science, is a human endeavor, and perhaps, is best understood in that context.

There are reasons that any science, AI included, concerns itself with a certain set of problems and develops a particular body of techniques for approaching these problems. In Chapter 1, a short history of artificial intelligence and the people and assumptions that have shaped it will explain why certain sets of questions have come to dominate the field and why the methods discussed in this book have been taken for their solution.

AI: EARLY HISTORY AND APPLICATIONS

All men by nature desire to know . . .

—ARISTOTLE, Opening sentence of the Metaphysics

Hear the rest, and you will marvel even more at the crafts and resources I have contrived. Greatest was this: in the former times if a man fell sick he had no defense against the sickness, neither healing food nor drink, nor unguent; but through the lack of drugs men wasted away, until I showed them the blending of mild simples wherewith they drive out all manner of diseases. . . .

It was I who made visible to men's eyes the flaming signs of the sky that were before dim. So much for these. Beneath the earth, man's hidden blessing, copper, iron, silver, and gold—will anyone claim to have discovered these before I did? No one, I am very sure, who wants to speak truly and to the purpose. One brief word will tell the whole story: all arts that mortals have come from Prometheus.

—AESCHYLUS, Prometheus Bound

1.1 From Eden to ENIAC: Attitudes toward Intelligence, Knowledge, and Human Artifice

Prometheus speaks of the fruits of his transgression against the gods of Olympus: his purpose was not merely to steal fire for the human race but also to enlighten humanity through the gift of intelligence or *nous*: the *rational mind.* This intelligence forms the foundation for all of human technology and ultimately all human civilization. The work of Aeschylus, the classical Greek dramatist, illustrates a deep and ancient awareness of the extraordinary power of knowledge. Artificial intelligence, in its very direct concern for Prometheus's gift, has been applied to all the areas of his legacy—medicine, psychology, biology, astronomy, geology—and many areas of scientific endeavor that Aeschylus could not have imagined.

Though Prometheus's action freed humanity from the sickness of ignorance, it also earned him the wrath of Zeus. Outraged over this theft of knowledge that previously belonged only to the gods of Olympus, Zeus commanded that Prometheus be chained to a barren rock to suffer the ravages of the elements for eternity. The notion that human efforts to gain knowledge constitute a transgression against the laws of God or nature is deeply ingrained in Western thought. It is the basis of the story of Eden and appears in the work of Dante and Milton. Both Shakespeare and the ancient Greek tragedians portrayed intellectual ambition as the cause of disaster. The belief that the desire for knowledge must ultimately lead to disaster has persisted throughout history, enduring the Renaissance, the Age of Enlightenment, and even the scientific and philosophical advances of the nineteenth and twentieth centuries. Thus, we should not be surprised that artificial intelligence inspires so much controversy in both academic and popular circles.

Indeed, rather than dispelling this ancient fear of the consequences of intellectual ambition, modern technology has only made those consequences seem likely, even imminent. The legends of Prometheus, Eve, and Faustus have been retold in the language of technological society. In her introduction to *Frankenstein*, subtitled, interestingly enough, *The Modern Prometheus*, Mary Shelley writes:

> Many and long were the conversations between Lord Byron and Shelley to which I was a devout and silent listener. During one of these, various philosophical doctrines were discussed, and among others the nature of the principle of life, and whether there was any probability of its ever being discovered and communicated. They talked of the experiments of Dr. Darwin (I speak not of what the doctor really did or said that he did, but, as more to my purpose, of what was then spoken of as having been done by him), who preserved a piece of vermicelli in a glass case till by some extraordinary means it began to move with a voluntary motion. Not thus, after all, would life be given. Perhaps a corpse would be reanimated; galvanism had given token of such things: perhaps the component parts of a creature might be manufactured, brought together, and endued with vital warmth (Butler 1998).

Mary Shelley shows us the extent to which scientific advances such as the work of Darwin and the discovery of electricity had convinced even nonscientists that the workings of nature were not divine secrets, but could be broken down and understood systematically. Frankenstein's monster is not the product of shamanistic incantations or unspeakable transactions with the underworld: it is assembled from separately "manufactured" components and infused with the vital force of electricity. Although nineteenth-century science was inadequate to realize the goal of understanding and creating a fully intelligent agent, it affirmed the notion that the mysteries of life and intellect might be brought into the light of scientific analysis.

1.1.1 A Brief History of the Foundations for AI

By the time Mary Shelley finally and perhaps irrevocably joined modern science with the Promethean myth, the philosophical foundations of modern work in artificial intelligence had been developing for several thousand years. Although the moral and cultural issues raised by artificial intelligence are both interesting and important, our introduction is more

properly concerned with AI's intellectual heritage. The logical starting point for such a history is the genius of Aristotle, or as Dante in the *Divine Comedy* refers to him, "the master of them that know". Aristotle wove together the insights, wonders, and fears of the early Greek tradition with the careful analysis and disciplined thought that were to become the standard for more modern science.

For Aristotle, the most fascinating aspect of nature was change. In his *Physics*, he defined his "philosophy of nature" as the "study of things that change". He distinguished between the *matter* and *form* of things: a sculpture is fashioned from the *material* bronze and has the *form* of a human. Change occurs when the bronze is molded to a new form. The matter/form distinction provides a philosophical basis for modern notions such as symbolic computing and data abstraction. In computing (even with numbers) we are manipulating patterns that are the forms of electromagnetic material, with the changes of form of this material representing aspects of the solution process. Abstracting the form from the medium of its representation not only allows these forms to be manipulated computationally but also provides the promise of a theory of data structures, the heart of modern computer science.

In his *Metaphysics*, beginning with the words "All men by nature desire to know", Aristotle developed a science of things that never change, including his cosmology and theology. More relevant to artificial intelligence, however, was Aristotle's epistemology or analysis of how humans "know" their world, discussed in his *Logic*. Aristotle referred to logic as the "instrument" (*organon*), because he felt that the study of thought itself was at the basis of all knowledge. In his *Logic*, he investigated whether certain propositions can be said to be "true" because they are related to other things that are known to be "true". Thus if we know that "all men are mortal" and that "Socrates is a man", then we can conclude that "Socrates is mortal". This argument is an example of what Aristotle referred to as a syllogism using the deductive form *modus ponens*. Although the formal axiomatization of reasoning needed another two thousand years for its full flowering in the works of Gottlob Frege, Bertrand Russell, Kurt Gödel, Alan Turing, Alfred Tarski, and others, its roots may be traced to Aristotle.

Renaissance thought, building on the Greek tradition, initiated the evolution of a different and powerful way of thinking about humanity and its relation to the natural world. Science began to replace mysticism as a means of understanding nature. Clocks and, eventually, factory schedules superseded the rhythms of nature for thousands of city dwellers. Most of the modern social and physical sciences found their origin in the notion that processes, whether natural or artificial, could be mathematically analyzed and understood. In particular, scientists and philosophers realized that thought itself, the way that knowledge was represented and manipulated in the human mind, was a difficult but essential subject for scientific study.

Perhaps the major event in the development of the modern world view was the Copernican revolution, the replacement of the ancient Earth-centered model of the universe with the idea that the Earth and other planets are actually in orbits around the sun. After centuries of an "obvious" order, in which the scientific explanation of the nature of the cosmos was consistent with the teachings of religion and common sense, a drastically different and not at all obvious model was proposed to explain the motions of heavenly bodies. For perhaps the first time, *our ideas about the world were seen as fundamentally*

distinct from that world's appearance. This split between the human mind and its surrounding reality, between ideas about things and things themselves, is essential to the modern study of the mind and its organization. This breach was widened by the writings of Galileo, whose scientific observations further contradicted the "obvious" truths about the natural world and whose development of mathematics as a tool for describing that world emphasized the distinction between the world and our ideas about it. It is out of this breach that the modern notion of the mind evolved: introspection became a common motif in literature, philosophers began to study epistemology and mathematics, and the systematic application of the scientific method rivaled the senses as tools for understanding the world.

In 1620, Francis Bacon's *Novum Organun* offered a set of search techniques for this emerging scientific methodology. Based on the Aristotelian and Platonic idea that the "form" of an entity was equivalent to the sum of its necessary and sufficient "features", Bacon articulated an algorithm for determining the essence of an entity. First, he made an organized collection of all instances of the entity, enumerating the features of each in a table. Then he collected a similar list of negative instances of the entity, focusing especially on near instances of the entity, that is, those that deviated from the "form" of the entity by single features. Then Bacon attempts—this step is not totally clear—to make a systematic list of all the features essential to the entity, that is, those that are common to all positive instances of the entity and missing from the negative instances.

It is interesting to see a form of Francis Bacon's approach to concept learning reflected in modern AI algorithms for Version Space Search, Chapter 10.2. An extension of Bacon's algorithms was also part of an AI program for discovery learning, suitably called *Bacon* (Langley et al. 1981). This program was able to induce many physical laws from collections of data related to the phenomena. It is also interesting to note that the question of whether a general purpose algorithm was possible for producing scientific proofs awaited the challenges of the early twentieth century mathematician Hilbert (his *Entscheidungsproblem*) and the response of the modern genius of Alan Turing (his *Turing Machine* and proofs of *computability* and the *haltying problem*); see Davis et al. (1976).

Although the first calculating machine, the abacus, was created by the Chinese in the twenty-sixth century BC, further mechanization of algebraic processes awaited the skills of the seventeenth century Europeans. In 1614, the Scots mathematician, John Napier, created logarithms, the mathematical transformations that allowed multiplication and the use of exponents to be reduced to addition and multiplication. Napier also created his *bones* that were used to represent overflow values for arithmetic operations. These bones were later used by Wilhelm Schickard (1592–1635), a German mathematician and clergyman of Tübingen, who in 1623 invented a *Calculating Clock* for performing addition and subtraction. This machine recorded the overflow from its calculations by the chiming of a clock.

Another famous calculating machine was the *Pascaline* that Blaise Pascal, the French philosopher and mathematician, created in 1642. Although the mechanisms of Schickard and Pascal were limited to addition and subtraction—including carries and borrows—they showed that processes that previously were thought to require human thought and skill could be fully automated. As Pascal later stated in his *Pensees* (1670), "The arithmetical machine produces effects which approach nearer to thought than all the actions of animals".

Pascal's successes with calculating machines inspired Gottfried Wilhelm von Leibniz in 1694 to complete a working machine that become known as the *Leibniz Wheel*. It integrated a moveable carriage and hand crank to drive wheels and cylinders that performed the more complex operations of multiplication and division. Leibniz was also fascinated by the possibility of an automated logic for proofs of propositions. Returning to Bacon's entity specification algorithm, where concepts were characterized as the collection of their necessary and sufficient features, Leibniz conjectured a machine that could calculate with these features to produce logically correct conclusions. Leibniz (1887) also envisioned a machine, reflecting modern ideas of deductive inference and proof, by which the production of scientific knowledge could become automated, a calculus for reasoning.

The seventeenth and eighteenth centuries also saw a great deal of discussion of epistemological issues; perhaps the most influential was the work of René Descartes, a central figure in the development of the modern concepts of thought and theories of mind. In his *Meditations*, Descartes (1680) attempted to find a basis for reality purely through introspection. Systematically rejecting the input of his senses as untrustworthy, Descartes was forced to doubt even the existence of the physical world and was left with only the reality of thought; even his own existence had to be justified in terms of thought: "Cogito ergo sum" (I think, therefore I am). After he established his own existence purely as a thinking entity, Descartes inferred the existence of God as an essential creator and ultimately reasserted the reality of the physical universe as the necessary creation of a benign God.

We can make two observations here: first, the schism between the mind and the physical world had become so complete that the process of thinking could be discussed in isolation from any specific sensory input or worldly subject matter; second, the connection between mind and the physical world was so tenuous that it required the intervention of a benign God to support reliable knowledge of the physical world! This view of the duality between the mind and the physical world underlies all of Descartes's thought, including his development of analytic geometry. How else could he have unified such a seemingly worldly branch of mathematics as geometry with such an abstract mathematical framework as algebra?

Why have we included this mind/body discussion in a book on artificial intelligence? There are two consequences of this analysis essential to the AI enterprise:

1. By attempting to separate the mind from the physical world, Descartes and related thinkers established that the structure of ideas about the world was not necessarily the same as the structure of their subject matter. This underlies the methodology of AI, along with the fields of epistemology, psychology, much of higher mathematics, and most of modern literature: mental processes have an existence of their own, obey their own laws, and can be studied in and of themselves.

2. Once the mind and the body are separated, philosophers found it necessary to find a way to reconnect the two, because interaction between Descartes mental, *res cogitans*, and physical, *res extensa*, is essential for human existence.

Although millions of words have been written on this *mind–body problem*, and numerous solutions proposed, no one has successfully explained the obvious interactions between mental states and physical actions while affirming a fundamental difference

between them. The most widely accepted response to this problem, and the one that provides an essential foundation for the study of AI, holds that the mind and the body are not fundamentally different entities at all. On this view, mental processes are indeed achieved by physical systems such as brains (or computers). Mental processes, like physical processes, can ultimately be characterized through formal mathematics. Or, as acknowledged in his *Leviathan* by the 17th century English philosopher Thomas Hobbes (1651), "By ratiocination, I mean computation".

1.1.2 AI and the Rationalist and Empiricist Traditions

Modern research issues in artificial intelligence, as in other scientific disciplines, are formed and evolve through a combination of historical, social, and cultural pressures. Two of the most prominent pressures for the evolution of AI are the empiricist and rationalist traditions in philosophy.

The rationalist tradition, as seen in the previous section, had an early proponent in Plato, and was continued on through the writings of Pascal, Descartes, and Leibniz. For the rationalist, the external world is reconstructed through the clear and distinct ideas of a mathematics. A criticism of this dualistic approach is the forced disengagement of representational systems from their field of reference. The issue is whether the meaning attributed to a representation can be defined independent of its application conditions. If the world is different from our beliefs about the world, can our created concepts and symbols still have meaning?

Many AI programs have very much of this rationalist flavor. Early robot planners, for example, would describe their application domain or "world" as sets of predicate calculus statements and then a "plan" for action would be created through proving theorems about this "world" (Fikes et al. 1972, see also Section 8.4). Newell and Simon's *Physical Symbol System Hypothesis* (Introduction to Part II and Chapter 17) is seen by many as the archetype of this approach in modern AI. Several critics have commented on this rationalist bias as part of the failure of AI at solving complex tasks such as understanding human languages (Searle 1980, Winograd and Flores 1986, Brooks 1991*a*).

Rather than affirming as "real" the world of clear and distinct ideas, empiricists continue to remind us that "nothing enters the mind except through the senses". This constraint leads to further questions of how the human can possibly perceive general concepts or the pure forms of Plato's cave (Plato 1961). Aristotle was an early empiricist, emphasizing in his *De Anima* the limitations of the human perceptual system. More modern empiricists, especially Hobbes, Locke, and Hume, emphasize that knowledge must be explained through an introspective but empirical psychology. They distinguish two types of mental phenomena, perceptions on one hand, and thought, memory, and imagination on the other. The Scots philosopher, David Hume, for example, distinguishes between *impressions* and *ideas*. Impressions are lively and vivid, reflecting the presence and existence of an external object and not subject to voluntary control. Ideas, on the other hand, are less vivid and detailed and more subject to the subject's voluntary control.

Given this distinction between impressions and ideas, how can knowledge arise? For Hobbes, Locke, and Hume the fundamental explanatory mechanism is *association.*

Particular perceptual properties are associated through repeated experience. This repeated association creates a disposition in the mind to associate the corresponding ideas. A fundamental property of this account is presented with Hume's skepticism. Hume's purely descriptive account of the origins of ideas cannot, he claims, support belief in causality. Even the use of logic and induction cannot be rationally supported in this radical empiricist epistemology.

In *An Inquiry Concerning Human Understanding* (1748), Hume's skepticism extended to the analysis of miracles. Although Hume didn't address the nature of miracles directly, he did question the testimony-based belief in the miraculous. This skepticism, of course, was seen as a direct threat by believers in the bible as well as many other purveyors of religious traditions. The Reverend Thomas Bayes was both a mathematician and a minister. One of his papers, called *Essay towards Solving a Problem in the Doctrine of Chances* (1763) addressed Hume's questions mathematically. Bayes' theorem demonstrates formally how, through learning the correlations of the effects of actions, we can determine the probability of their causes.

The associational account of knowledge plays a significant role in the development of AI representational structures and programs, for example, in memory organization with *semantic networks* and *MOPS* and work in natural language understanding (see Sections 7.0, 7.1, and Chapter 14). Associational accounts have important influences of machine learning, especially with connectionist networks (see Sections 10.6, 10.7, and Chapter 11). Associationism also plays an important role in cognitive psychology including the *schemas* of Bartlett and Piaget as well as the entire thrust of the behaviorist tradition (Luger 1994). Finally, with AI tools for stochastic analysis, including the *Bayesian belief network* (BBN) and its current extensions to first-order Turing-complete systems for stochastic modeling, associational theories have found a sound mathematical basis and mature expressive power. Bayesian tools are important for research including diagnostics, machine learning, and natural language understanding (see Chapter 5 and Section 9.3).

Immanuel Kant, a German philosopher trained in the rationalist tradition, was strongly influenced by the writing of Hume. As a result, he began the modern synthesis of these two traditions. Knowledge for Kant contains two collaborating energies, an a priori component coming from the subject's reason along with an a posteriori component coming from active experience. Experience is meaningful only through the contribution of the subject. Without an active organizing form proposed by the subject, the world would be nothing more than passing transitory sensations. Finally, at the level of judgement, Kant claims, passing images or representations are bound together by the active subject and taken as the diverse appearances of an identity, of an "object". Kant's realism began the modern enterprise of psychologists such as Bartlett, Brunner, and Piaget. Kant's work influences the modern AI enterprise of machine learning (Part IV) as well as the continuing development of a constructivist epistemology (see Chapter 17).

1.1.3 The Development of Formal Logic

Once thinking had come to be regarded as a form of computation, its formalization and eventual mechanization were obvious next steps. As noted in Section 1.1.1, Gottfried

Wilhelm von Leibniz, with his *Calculus Philosophicus*, introduced the first system of formal logic as well as proposing a machine for automating its tasks (Leibniz 1887). Furthermore, the steps and stages of this mechanical solution can be represented as movement through the states of a tree or graph. Leonhard Euler, in the eighteenth century, with his analysis of the "connectedness" of the bridges joining the riverbanks and islands of the city of Königsberg (see the introduction to Chapter 3), introduced the study of representations that can abstractly capture the structure of relationships in the world as well as the discrete steps within a computation (Euler 1735).

The formalization of graph theory also afforded the possibility of *state space search*, a major conceptual tool of artificial intelligence. We can use graphs to model the deeper structure of a problem. The nodes of a *state space graph* represent possible stages of a problem solution; the arcs of the graph represent inferences, moves in a game, or other steps in a problem solution. Solving the problem is a process of searching the state space graph for a path to a solution (Introduction to Part II and Chapter 3). By describing the entire space of problem solutions, state space graphs provide a powerful tool for measuring the structure and complexity of problems and analyzing the efficiency, correctness, and generality of solution strategies.

As one of the originators of the science of operations research, as well as the designer of the first programmable mechanical computing machines, Charles Babbage, a nineteenth century mathematician, may also be considered an early practitioner of artificial intelligence (Morrison and Morrison 1961). Babbage's *difference engine* was a special-purpose machine for computing the values of certain polynomial functions and was the forerunner of his *analytical engine*. The analytical engine, designed but not successfully constructed during his lifetime, was a general-purpose programmable computing machine that presaged many of the architectural assumptions underlying the modern computer.

In describing the analytical engine, Ada Lovelace (1961), Babbage's friend, supporter, and collaborator, said:

> We may say most aptly that the Analytical Engine weaves algebraical patterns just as the Jacquard loom weaves flowers and leaves. Here, it seems to us, resides much more of originality than the difference engine can be fairly entitled to claim.

Babbage's inspiration was his desire to apply the technology of his day to liberate humans from the drudgery of making arithmetic calculations. In this sentiment, as well as with his conception of computers as mechanical devices, Babbage was thinking in purely nineteenth century terms. His analytical engine, however, also included many modern notions, such as the separation of memory and processor, the *store* and the *mill* in Babbage's terms, the concept of a digital rather than analog machine, and programmability based on the execution of a series of operations encoded on punched pasteboard cards. The most striking feature of Ada Lovelace's description, and of Babbage's work in general, is its treatment of the "patterns" of algebraic relationships as entities that may be studied, characterized, and finally implemented and manipulated mechanically without concern for the particular values that are finally passed through the mill of the calculating machine. This is an example implementation of the "abstraction and manipulation of form" first described by Aristotle and Leibniz.

The goal of creating a formal language for thought also appears in the work of George Boole, another nineteenth-century mathematician whose work must be included in any discussion of the roots of artificial intelligence (Boole 1847, 1854). Although he made contributions to a number of areas of mathematics, his best known work was in the mathematical formalization of the laws of logic, an accomplishment that forms the very heart of modern computer science. Though the role of Boolean algebra in the design of logic circuitry is well known, Boole's own goals in developing his system seem closer to those of contemporary AI researchers. In the first chapter of *An Investigation of the Laws of Thought, on which are founded the Mathematical Theories of Logic and Probabilities*, Boole (1854) described his goals as

> to investigate the fundamental laws of those operations of the mind by which reasoning is performed: to give expression to them in the symbolical language of a Calculus, and upon this foundation to establish the science of logic and instruct its method; ...and finally to collect from the various elements of truth brought to view in the course of these inquiries some probable intimations concerning the nature and constitution of the human mind.

The greatness of Boole's accomplishment is in the extraordinary power and simplicity of the system he devised: three operations, "AND" (denoted by * or $\wedge$), "OR" (denoted by + or $\vee$), and "NOT" (denoted by $\neg$), formed the heart of his logical calculus. These operations have remained the basis for all subsequent developments in formal logic, including the design of modern computers. While keeping the meaning of these symbols nearly identical to the corresponding algebraic operations, Boole noted that "the Symbols of logic are further subject to a special law, to which the symbols of quantity, as such, are not subject". This law states that for any X, an element in the algebra, X*X=X (or that once something is known to be true, repetition cannot augment that knowledge). This led to the characteristic restriction of Boolean values to the only two numbers that may satisfy this equation: 1 and 0. The standard definitions of Boolean multiplication (AND) and addition (OR) follow from this insight.

Boole's system not only provided the basis of binary arithmetic but also demonstrated that an extremely simple formal system was adequate to capture the full power of logic. This assumption and the system Boole developed to demonstrate it form the basis of all modern efforts to formalize logic, from Russell and Whitehead's *Principia Mathematica* (Whitehead and Russell 1950), through the work of Turing and Gödel, up to modern automated reasoning systems.

Gottlob Frege, in his *Foundations of Arithmetic* (Frege 1879, 1884), created a mathematical specification language for describing the basis of arithmetic in a clear and precise fashion. With this language Frege formalized many of the issues first addressed by Aristotle's *Logic*. Frege's language, now called the *first-order predicate calculus*, offers a tool for describing the propositions and truth value assignments that make up the elements of mathematical reasoning and describes the axiomatic basis of "meaning" for these expressions. The formal system of the predicate calculus, which includes predicate symbols, a theory of functions, and quantified variables, was intended to be a language for describing mathematics and its philosophical foundations. It also plays a fundamental role in creating a theory of representation for artificial intelligence (Chapter 2). The first-order

predicate calculus offers the tools necessary for automating reasoning: a language for expressions, a theory for assumptions related to the meaning of expressions, and a logically sound calculus for inferring new true expressions.

Whitehead and Russell's (1950) work is particularly important to the foundations of AI, in that their stated goal was to derive the whole of mathematics through formal operations on a collection of axioms. Although many mathematical systems have been constructed from basic axioms, what is interesting is Russell and Whitehead's commitment to mathematics as a purely formal system. This meant that axioms and theorems would be treated solely as strings of characters: proofs would proceed solely through the application of well-defined rules for manipulating these strings. There would be no reliance on intuition or the meaning of theorems as a basis for proofs. Every step of a proof followed from the strict application of formal (syntactic) rules to either axioms or previously proven theorems, even where traditional proofs might regard such a step as "obvious". What "meaning" the theorems and axioms of the system might have in relation to the world would be independent of their logical derivations. This treatment of mathematical reasoning in purely formal (and hence mechanical) terms provided an essential basis for its automation on physical computers. The logical syntax and formal rules of inference developed by Russell and Whitehead are still a basis for automatic theorem-proving systems, presented in Chapter 13, as well as for the theoretical foundations of artificial intelligence.

Alfred Tarski is another mathematician whose work is essential to the foundations of AI. Tarski created a *theory of reference* wherein the *well-formed formulae* of Frege or Russell and Whitehead can be said to refer, in a precise fashion, to the physical world (Tarski 1944, 1956; see Chapter 2). This insight underlies most theories of formal semantics. In his paper *The Semantic Conception of Truth and the Foundation of Semantics*, Tarski describes his theory of reference and truth value relationships. Modern computer scientists, especially Scott, Strachey, Burstall (Burstall and Darlington 1977), and Plotkin have related this theory to programming languages and other specifications for computing.

Although in the eighteenth, nineteenth, and early twentieth centuries the formalization of science and mathematics created the intellectual prerequisite for the study of artificial intelligence, it was not until the twentieth century and the introduction of the digital computer that AI became a viable scientific discipline. By the end of the 1940s electronic digital computers had demonstrated their potential to provide the memory and processing power required by intelligent programs. It was now possible to implement formal reasoning systems on a computer and empirically test their sufficiency for exhibiting intelligence. An essential component of the science of artificial intelligence is this commitment to digital computers as the vehicle of choice for creating and testing theories of intelligence.

Digital computers are not merely a vehicle for testing theories of intelligence. Their architecture also suggests a specific paradigm for such theories: intelligence is a form of information processing. The notion of search as a problem-solving methodology, for example, owes more to the sequential nature of computer operation than it does to any biological model of intelligence. Most AI programs represent knowledge in some formal language that is then manipulated by algorithms, honoring the separation of data and program fundamental to the von Neumann style of computing. Formal logic has emerged as an important representational tool for AI research, just as graph theory plays an

indispensable role in the analysis of problem spaces as well as providing a basis for semantic networks and similar models of semantic meaning. These techniques and formalisms are discussed in detail throughout the body of this text; we mention them here to emphasize the symbiotic relationship between the digital computer and the theoretical underpinnings of artificial intelligence.

We often forget that the tools we create for our own purposes tend to shape our conception of the world through their structure and limitations. Although seemingly restrictive, this interaction is an essential aspect of the evolution of human knowledge: a tool (and scientific theories are ultimately only tools) is developed to solve a particular problem. As it is used and refined, the tool itself seems to suggest other applications, leading to new questions and, ultimately, the development of new tools.

1.1.4 The Turing Test

One of the earliest papers to address the question of machine intelligence specifically in relation to the modern digital computer was written in 1950 by the British mathematician Alan Turing. *Computing Machinery and Intelligence* (Turing 1950) remains timely in both its assessment of the arguments against the possibility of creating an intelligent computing machine and its answers to those arguments. Turing, known mainly for his contributions to the theory of computability, considered the question of whether or not a machine could actually be made to think. Noting that the fundamental ambiguities in the question itself (what is thinking? what is a machine?) precluded any rational answer, he proposed that the question of intelligence be replaced by a more clearly defined empirical test.

The *Turing test* measures the performance of an allegedly intelligent machine against that of a human being, arguably the best and only standard for intelligent behavior. The test, which Turing called the *imitation game*, places the machine and a human counterpart in rooms apart from a second human being, referred to as the *interrogator* (Figure 1.1). The interrogator is not able to see or speak directly to either of them, does not know which entity is actually the machine, and may communicate with them solely by use of a textual device such as a terminal. The interrogator is asked to distinguish the computer from the human being solely on the basis of their answers to questions asked over this device. If the interrogator cannot distinguish the machine from the human, then, Turing argues, the machine may be assumed to be intelligent.

By isolating the interrogator from both the machine and the other human participant, the test ensures that the interrogator will not be biased by the appearance of the machine or any mechanical property of its voice. The interrogator is free, however, to ask any questions, no matter how devious or indirect, in an effort to uncover the computer's identity. For example, the interrogator may ask both subjects to perform a rather involved arithmetic calculation, assuming that the computer will be more likely to get it correct than the human; to counter this strategy, the computer will need to know when it should fail to get a correct answer to such problems in order to seem like a human. To discover the human's identity on the basis of emotional nature, the interrogator may ask both subjects to respond to a poem or work of art; this strategy will require that the computer have knowledge concerning the emotional makeup of human beings.

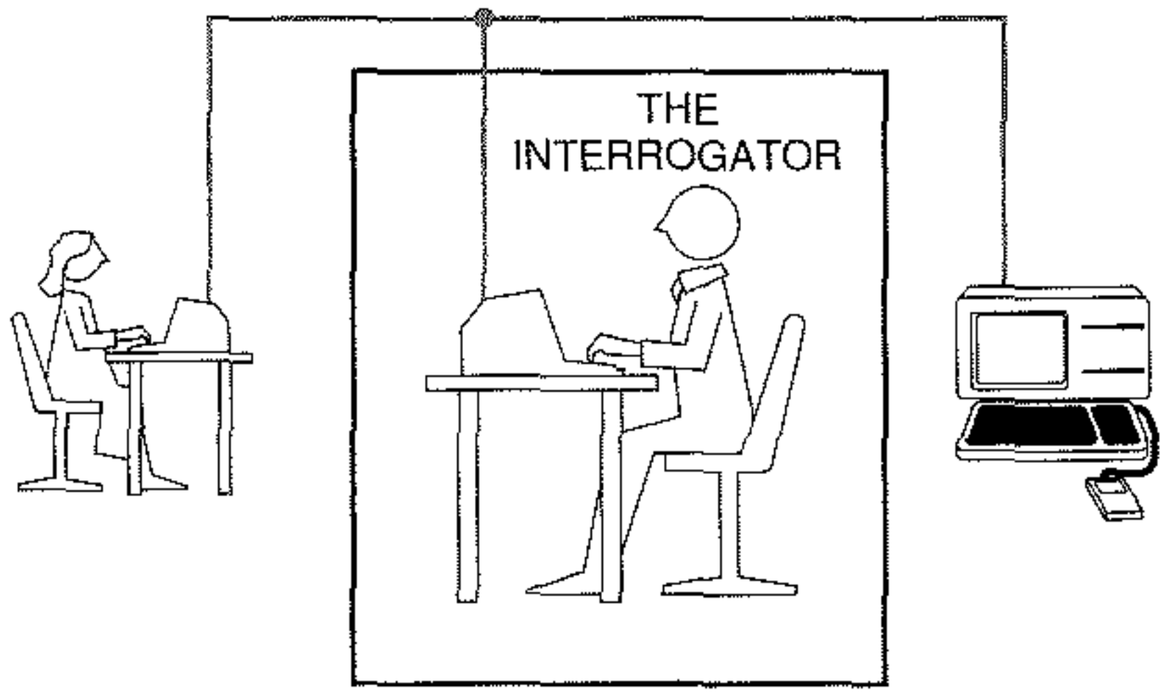

Figure 1.1 The Turing test.

The important features of Turing's test are:

1. It attempts to give an objective notion of intelligence, i.e., the behavior of a known intelligent being in response to a particular set of questions. This provides a standard for determining intelligence that avoids the inevitable debates over its "true" nature.
2. It prevents us from being sidetracked by such confusing and currently unanswerable questions as whether or not the computer uses the appropriate internal processes or whether or not the machine is actually conscious of its actions.
3. It eliminates any bias in favor of living organisms by forcing the interrogator to focus solely on the content of the answers to questions.

Because of these advantages, the Turing test provides a basis for many of the schemes actually used to evaluate modern AI programs. A program that has potentially achieved intelligence in some area of expertise may be evaluated by comparing its performance on a given set of problems to that of a human expert. This evaluation technique is just a variation of the Turing test: a group of humans are asked to blindly compare the performance of a computer and a human being on a particular set of problems. As we will see, this methodology has become an essential tool in both the development and verification of modern expert systems.

The Turing test, in spite of its intuitive appeal, is vulnerable to a number of justifiable criticisms. One of the most important of these is aimed at its bias toward purely symbolic problem-solving tasks. It does not test abilities requiring perceptual skill or manual dexterity, even though these are important components of human intelligence. Conversely, it is sometimes suggested that the Turing test needlessly constrains machine intelligence to fit a human mold. Perhaps machine intelligence is simply different from human intelligence and trying to evaluate it in human terms is a fundamental mistake. Do we really wish a machine would do mathematics as slowly and inaccurately as a human? Shouldn't an intelligent machine capitalize on its own assets, such as a large, fast, reliable memory,

rather than trying to emulate human cognition? In fact, a number of modern AI practitioners (e.g., Ford and Hayes 1995) see responding to the full challenge of Turing's test as a mistake and a major distraction to the more important work at hand: developing general theories to explain the mechanisms of intelligence in humans and machines and applying those theories to the development of tools to solve specific, practical problems. Although we agree with the Ford and Hayes concerns in the large, we still see Turing's test as an important component in the verification and validation of modern AI software.

Turing also addressed the very feasibility of constructing an intelligent program on a digital computer. By thinking in terms of a specific model of computation (an electronic discrete state computing machine), he made some well-founded conjectures concerning the storage capacity, program complexity, and basic design philosophy required for such a system. Finally, he addressed a number of moral, philosophical, and scientific objections to the possibility of constructing such a program in terms of an actual technology. The reader is referred to Turing's article for a perceptive and still relevant summary of the debate over the possibility of intelligent machines.

Two of the objections cited by Turing are worth considering further. *Lady Lovelace's Objection*, first stated by Ada Lovelace, argues that computers can only do as they are told and consequently cannot perform original (hence, intelligent) actions. This objection has become a reassuring if somewhat dubious part of contemporary technological folklore. Expert systems (Section 1.2.3 and Chapter 8), especially in the area of diagnostic reasoning, have reached conclusions unanticipated by their designers. Indeed, a number of researchers feel that human creativity can be expressed in a computer program.

The other related objection, the *Argument from Informality of Behavior*, asserts the impossibility of creating a set of rules that will tell an individual exactly what to do under every possible set of circumstances. Certainly, the flexibility that enables a biological intelligence to respond to an almost infinite range of situations in a reasonable if not necessarily optimal fashion is a hallmark of intelligent behavior. While it is true that the control structure used in most traditional computer programs does not demonstrate great flexibility or originality, it is not true that all programs must be written in this fashion. Indeed, much of the work in AI over the past 25 years has been to develop programming languages and models such as production systems, object-based systems, network representations, and others discussed in this book that attempt to overcome this deficiency.

Many modern AI programs consist of a collection of modular components, or rules of behavior, that do not execute in a rigid order but rather are invoked as needed in response to the structure of a particular problem instance. Pattern matchers allow general rules to apply over a range of instances. These systems have an extreme flexibility that enables relatively small programs to exhibit a vast range of possible behaviors in response to differing problems and situations.

Whether these systems can ultimately be made to exhibit the flexibility shown by a living organism is still the subject of much debate. Nobel laureate Herbert Simon has argued that much of the originality and variability of behavior shown by living creatures is due to the richness of their environment rather than the complexity of their own internal programs. In *The Sciences of the Artificial*, Simon (1981) describes an ant progressing circuitously along an uneven and cluttered stretch of ground. Although the ant's path seems quite complex, Simon argues that the ant's goal is very simple: to return to its colony as

quickly as possible. The twists and turns in its path are caused by the obstacles it encounters on its way. Simon concludes that

> An ant, viewed as a behaving system, is quite simple. The apparent complexity of its behavior over time is largely a reflection of the complexity of the environment in which it finds itself.

This idea, if ultimately proved to apply to organisms of higher intelligence as well as to such simple creatures as insects, constitutes a powerful argument that such systems are relatively simple and, consequently, comprehensible. It is interesting to note that if one applies this idea to humans, it becomes a strong argument for the importance of culture in the forming of intelligence. Rather than growing in the dark like mushrooms, intelligence seems to depend on an interaction with a suitably rich environment. Culture is just as important in creating humans as human beings are in creating culture. Rather than denigrating our intellects, this idea emphasizes the miraculous richness and coherence of the cultures that have formed out of the lives of separate human beings. In fact, the idea that intelligence emerges from the interactions of individual elements of a society is one of the insights supporting the approach to AI technology presented in the next section.

1.1.5 Biological and Social Models of Intelligence: Agents Theories

So far, we have approached the problem of building intelligent machines from the viewpoint of mathematics, with the implicit belief of logical reasoning as paradigmatic of intelligence itself, as well as with a commitment to "objective" foundations for logical reasoning. This way of looking at knowledge, language, and thought reflects the rationalist tradition of western philosophy, as it evolved through Plato, Galileo, Descartes, Leibniz, and many of the other philosophers discussed earlier in this chapter. It also reflects the underlying assumptions of the Turing test, particularly its emphasis on symbolic reasoning as a test of intelligence, and the belief that a straightforward comparison with human behavior was adequate to confirming machine intelligence.

The reliance on logic as a way of representing knowledge and on logical inference as the primary mechanism for intelligent reasoning are so dominant in Western philosophy that their "truth" often seems obvious and unassailable. It is no surprise, then, that approaches based on these assumptions have dominated the science of artificial intelligence from its inception through to the present day.

The latter half of the twentieth century has, however, seen numerous challenges to rationalist philosophy. Various forms of philosophical relativism question the objective basis of language, science, society, and thought itself. Ludwig Wittgenstein's later philosophy (Wittgenstein 1953), has forced us to reconsider the basis on meaning in both natural and formal languages. The work of Godel, Nagel and Newman (1958), and Turing has cast doubt on the very foundations of mathematics itself. Post-modern thought has changed our understanding of meaning and value in the arts and society. Artificial intelligence has not been immune to these criticisms; indeed, the difficulties that AI has encountered in achieving its goals are often taken as evidence of the failure of the rationalist viewpoint (Winograd and Flores 1986, Lakoff and Johnson 1999).

Two philosophical traditions, that of Wittgenstein (1953) as well as that of Husserl (1970, 1972) and Heidegger (1962), are central to this reappraisal of the Western philosophical tradition. In his later work, Wittgenstein questioned many of the assumptions of the rationalist tradition, including the foundations of language, science, and knowledge. Natural language was a major focus of Wittgenstein's analysis: he challenged the notion that human language derived its meaning from any sort of objective foundation.

For Wittgenstein, as well as the speech act theory developed by Austin (1962) and his followers (Grice 1975, Searle 1969), the meaning of any utterance depends on its being situated in a human, cultural context. Our understanding of the meaning of the word "chair", for example, is dependent on having a physical body that conforms to a sitting posture and the cultural conventions for using chairs. When, for example, is a large, flat rock a chair? Why is it odd to refer to the throne of England as a chair? What is the difference between a human being's understanding of a chair and that of a dog or cat, incapable of sitting in the human sense? Based on his attacks on the foundations of meaning, Wittgenstein argued that we should view the use of language in terms of choices made and actions taken in a shifting cultural context. Wittgenstein even extended his criticisms to science and mathematics, arguing that they are just as much social constructs as is language use.

Husserl (1970, 1972), the father of phenomenology, was committed to abstractions as rooted in the concrete *Lebenswelt* or *life-world*: a rationalist model was very much secondary to the concrete world that supported it. For Husserl, as well as for his student Heidegger (1962), and their proponent Merleau-Ponty (1962), intelligence was not knowing what was true, but rather knowing how to cope in a world that was constantly changing and evolving. Gadamer (1976) also contributed to this tradition. For the existentialist/phenomenologist, intelligence is seen as survival in the world, rather than as a set of logical propositions about the world (combined with some inferencing scheme).

Many authors, for example Dreyfus and Dreyfus (1985) and Winograd and Flores (1986), have drawn on Wittgenstein's and the Husserl/Heidegger work in their criticisms of AI. Although many AI practitioners continue developing the rational/logical agenda, also known as *GOFAI*, or *Good Old Fashioned AI*, a growing number of researchers in the field have incorporated these criticisms into new and exciting models of intelligence. In keeping with Wittgenstein's emphasis on the anthropological and cultural roots of knowledge, they have turned to social, sometimes referred to as *situated*, models of intelligent behavior for their inspiration.

As an example of an alternative to a logic-based approach, research in connectionist learning (Section 1.2.9 and Chapter 11) de-emphasizes logic and the functioning of the rational mind in an effort to achieve intelligence by modeling the architecture of the physical brain. Neural models of intelligence emphasize the brain's ability to adapt to the world in which it is situated by modifying the relationships between individual neurons. Rather than representing knowledge in explicit logical sentences, they capture it implicitly, as a property of patterns of relationships.

Another biologically based model of intelligence takes its inspiration from the processes by which entire species adapt to their surroundings. Work in artificial life and genetic algorithms (Chapter 12) applies the principles of biological evolution to the problems of finding solutions to difficult problems. These programs do not solve problems

by reasoning logically about them; rather, they spawn populations of competing candidate solutions and drive them to evolve ever better solutions through a process patterned after biological evolution: poor candidate solutions tend to die out, while those that show the promise for solving a problem survive and reproduce by constructing new solutions out of components of their successful parents.

Social systems provide another metaphor for intelligence in that they exhibit global behaviors that enable them to solve problems that would confound any of their individual members. For example, although no individual could accurately predict the number of loaves of bread to be consumed in New York City on a given day, the entire system of New York bakeries does an excellent job of keeping the city stocked with bread, and doing so with minimal waste. The stock market does an excellent job of setting the relative values of hundreds of companies, even though each individual investor has only limited knowledge of a few companies. A final example comes from modern science. Individuals located in universities, industry, or government environments focus on common problems. With conferences and journals as the main communication media, problems important to society at large are attacked and solved by individual agents working semi-independently, although progress in many instances is also driven by funding agencies.

These examples share two themes: first, the view of intelligence as rooted in culture and society and, as a consequence, emergent. The second theme is that intelligence is reflected by the collective behaviors of large numbers of very simple interacting, semi-autonomous individuals, or agents. Whether these agents are neural cells, individual members of a species, or a single person in a society, their interactions produce intelligence.

What are the main themes supporting an agent-oriented and emergent view of intelligence? They include:

1. Agents are autonomous or semi-autonomous. That is, each agent has certain responsibilities in problem solving with little or no knowledge of either what other agents do or how they do it. Each agent does its own independent piece of the problem solving and either produces a result itself (does something) or reports results back to others in the community (communicating agent).

2. Agents are "situated." Each agent is sensitive to its own surrounding environment and (usually) has no knowledge of the full domain of all agents. Thus, an agent's knowledge is limited to the tasks to hand: "the-file-I'm-processing" or "the-wall-next-to-me" with no knowledge of the total range of files or physical constraints in the problem solving task.

3. Agents are interactional. That is, they form a collection of individuals that cooperate on a particular task. In this sense they may be seen as a "society" and, as with human society, knowledge, skills, and responsibilities, even when seen as collective, are distributed across the population of individuals.

4. The society of agents is structured. In most views of agent-oriented problem solving, each individual, although having its own unique environment and skill set, will coordinate with other agents in the overall problem solving. Thus, a final solution will not only be seen as collective, but also as cooperative.

5. Finally, the phenomenon of intelligence in this environment is "emergent." Although individual agents are seen as possessing sets of skills and responsibilities, the overall cooperative result can be viewed as greater than the sum of its individual contributors. Intelligence is seen as a phenomenon resident in and emerging from a society and not just a property of an individual agent.

Based on these observations, we define an agent as an element of a society that can perceive (often limited) aspects of its environment and affect that environment either directly or through cooperation with other agents. Most intelligent solutions require a variety of agents. These include rote agents, that simply capture and communicate pieces of information, coordination agents that can support the interactions between other agents, search agents that can examine multiple pieces of information and return some chosen bit of it, learning agents that can examine collections of information and form concepts or generalizations, and decision agents that can both dispatch tasks and come to conclusions in the light of limited information and processing. Going back to an older definition of intelligence, agents can be seen as the mechanisms supporting decision making in the context of limited processing resources.

The main requisites for designing and building such a society are:

1. structures for the representation of information,
2. strategies for the search through alternative solutions, and
3. the creation of architectures that can support the interaction of agents.

The remaining chapters of our book, especially Section 7.4, include prescriptions for the construction of support tools for this society of agents, as well as many examples of agent-based problem solving.

Our preliminary discussion of the possibility of a theory of automated intelligence is in no way intended to overstate the progress made to date or minimize the work that lies ahead. As we emphasize throughout this text, it is important to be aware of our limitations and to be honest about our successes. For example, there have been only limited results with programs that in any interesting sense can be said to "learn." Our accomplishments in modeling the semantic complexities of a natural language such as English have also been very modest. Even fundamental issues such as organizing knowledge or fully managing the complexity and correctness of very large computer programs (such as large knowledge bases) require considerable further research. Knowledge-based systems, though they have achieved marketable engineering successes, still have many limitations in the quality and generality of their reasoning. These include their inability to perform *commonsense reasoning* or to exhibit knowledge of rudimentary physical reality, such as how things change over time.

But we must maintain a reasonable perspective. It is easy to overlook the accomplishments of artificial intelligence when honestly facing the work that remains. In the next section, we establish this perspective through an overview of several areas of artificial intelligence research and development.

1.2 Overview of AI Application Areas

The Analytical Engine has no pretensions whatever to originate anything. It can do whatever we know how to order it to perform.

—ADA BYRON, *Countess of Lovelace*

I'm sorry Dave; I can't let you do that.

—HAL 9000 in *2001: A Space Odyssey* by Arthur C. Clarke

We now return to our goal of defining artificial intelligence through an examination of the ambitions and accomplishments of workers in the field. The two most fundamental concerns of AI researchers are *knowledge representation* and *search*. The first of these addresses the problem of capturing in a language, i.e., one suitable for computer manipulation, the full range of knowledge required for intelligent behavior. Chapter 2 introduces predicate calculus as a language for describing the properties and relationships among objects in problem domains that require qualitative reasoning rather than arithmetic calculations for their solutions. Later, Part III discusses the tools that artificial intelligence has developed for representing the ambiguities and complexities of areas such as commonsense reasoning and natural language understanding. Chapters 15 and 16 demonstrate the use of LISP and PROLOG to implement these representations.

Search is a problem-solving technique that systematically explores a space of *problem states*, i.e., successive and alternative stages in the problem-solving process. Examples of problem states might include the different board configurations in a game or intermediate steps in a reasoning process. This space of alternative solutions is then searched to find an answer. Newell and Simon (1976) have argued that this is the essential basis of human problem solving. Indeed, when a chess player examines the effects of different moves or a doctor considers a number of alternative diagnoses, they are searching among alternatives. The implications of this model and techniques for its implementation are discussed in Chapters 3, 4, 6, and 17.

Like most sciences, AI is decomposed into a number of subdisciplines that, while sharing an essential approach to problem solving, have concerned themselves with different applications. In this section we outline several of these major application areas and their contributions to artificial intelligence as a whole.

1.2.1 Game Playing

Much of the early research in state space search was done using common board games such as checkers, chess, and the 15-puzzle. In addition to their inherent intellectual appeal, board games have certain properties that made them ideal subjects for this early work. Most games are played using a well-defined set of rules: this makes it easy to generate the search space and frees the researcher from many of the ambiguities and complexities

inherent in less structured problems. The board configurations used in playing these games are easily represented on a computer, requiring none of the complex formalisms needed to capture the semantic subtleties of more complex problem domains. As games can be easily played, testing a game-playing program presents no financial or ethical burden. State space search, the paradigm underlying most game-playing research, is presented in Chapters 3 and 4.

Games can generate extremely large search spaces. These are large and complex enough to require powerful techniques for determining what alternatives to explore in the problem space. These techniques are called *heuristics* and constitute a major area of AI research. A heuristic is a useful but potentially fallible problem-solving strategy, such as checking to make sure that an unresponsive appliance is plugged in before assuming that it is broken or to castle in order to try and protect your king from capture in a chess game. Much of what we commonly call intelligence seems to reside in the heuristics used by humans to solve problems.

Because most of us have some experience with these simple games, it is possible to devise and test the effectiveness of our own heuristics. We do not need to find and consult an expert in some esoteric problem area such as medicine or mathematics (chess is an obvious exception to this rule). For these reasons, games provide a rich domain for the study of heuristic search. Chapter 4 introduces heuristics using these simple games; Chapter 8 extends their application to expert systems. Game-playing programs, in spite of their simplicity, offer their own challenges, including an opponent whose moves may not be deterministically anticipated, Chapters 5 and 8. This presence of the opponent further complicates program design by adding an element of unpredictability and the need to consider psychological as well as tactical factors in game strategy.

1.2.2 Automated Reasoning and Theorem Proving

We could argue that automatic theorem proving is the oldest branch of artificial intelligence, tracing its roots back through Newell and Simon's Logic Theorist (Newell and Simon 1963*a*) and General Problem Solver (Newell and Simon 1963*b*), through Russell and Whitehead's efforts to treat all of mathematics as the purely formal derivation of theorems from basic axioms, to its origins in the writings of Babbage and Leibniz. In any case, it has certainly been one of the most fruitful branches of the field. Theorem-proving research was responsible for much of the early work in formalizing search algorithms and developing formal representation languages such as the predicate calculus (Chapter 2) and the logic programming language PROLOG (Chapter 15).

Most of the appeal of automated theorem proving lies in the rigor and generality of logic. Because it is a formal system, logic lends itself to automation. A wide variety of problems can be attacked by representing the problem description and relevant background information as logical axioms and treating problem instances as theorems to be proved. This insight is the basis of work in automatic theorem proving and mathematical reasoning systems (Chapter 13).

Unfortunately, early efforts at writing theorem provers failed to develop a system that could consistently solve complicated problems. This was due to the ability of any

reasonably complex logical system to generate an infinite number of provable theorems: without powerful techniques (heuristics) to guide their search, automated theorem provers proved large numbers of irrelevant theorems before stumbling onto the correct one. In response to this inefficiency, many argue that purely formal, syntactic methods of guiding search are inherently incapable of handling such a huge space and that the only alternative is to rely on the informal, *ad hoc* strategies that humans seem to use in solving problems. This is the approach underlying the development of expert systems (Chapter 8), and it has proved to be a fruitful one.

Still, the appeal of reasoning based in formal mathematical logic is too strong to ignore. Many important problems such as the design and verification of logic circuits, verification of the correctness of computer programs, and control of complex systems seem to respond to such an approach. In addition, the theorem-proving community has enjoyed success in devising powerful solution heuristics that rely solely on an evaluation of the syntactic form of a logical expression, and as a result, reducing the complexity of the search space without resorting to the *ad hoc* techniques used by most human problem solvers.

Another reason for the continued interest in automatic theorem provers is the realization that such a system does not have to be capable of independently solving extremely complex problems without human assistance. Many modern theorem provers function as intelligent assistants, letting humans perform the more demanding tasks of decomposing a large problem into subproblems and devising heuristics for searching the space of possible proofs. The theorem prover then performs the simpler but still demanding task of proving lemmas, verifying smaller conjectures, and completing the formal aspects of a proof outlined by its human associate (Boyer and Moore 1979, Bundy 1988, Veroff 1997).

1.2.3 Expert Systems

One major insight gained from early work in problem solving was the importance of domain-specific knowledge. A doctor, for example, is not effective at diagnosing illness solely because she possesses some innate general problem-solving skill; she is effective because she knows a lot about medicine. Similarly, a geologist is effective at discovering mineral deposits because he is able to apply a good deal of theoretical and empirical knowledge about geology to the problem at hand. Expert knowledge is a combination of a theoretical understanding of the problem and a collection of heuristic problem-solving rules that experience has shown to be effective in the domain. Expert systems are constructed by obtaining this knowledge from a human expert and coding it into a form that a computer may apply to similar problems.

This reliance on the knowledge of a human domain expert for the system's problem solving strategies is a major feature of expert systems. Although some programs are written in which the designer is also the source of the domain knowledge, it is far more typical to see such programs growing out of a collaboration between a domain expert such as a doctor, chemist, geologist, or engineer and a separate artificial intelligence specialist. The domain expert provides the necessary knowledge of the problem domain through a general discussion of her problem-solving methods and by demonstrating those skills on a carefully chosen set of sample problems. The AI specialist, or *knowledge engineer*, as

expert systems designers are often known, is responsible for implementing this knowledge in a program that is both effective and seemingly intelligent in its behavior. Once such a program has been written, it is necessary to refine its expertise through a process of giving it example problems to solve, letting the domain expert criticize its behavior, and making any required changes or modifications to the program's knowledge. This process is repeated until the program has achieved the desired level of performance.

One of the earliest systems to exploit domain-specific knowledge in problem solving was DENDRAL, developed at Stanford in the late 1960s (Lindsay et al. 1980). DENDRAL was designed to infer the structure of organic molecules from their chemical formulas and mass spectrographic information about the chemical bonds present in the molecules. Because organic molecules tend to be very large, the number of possible structures for these molecules tends to be huge. DENDRAL addresses the problem of this large search space by applying the heuristic knowledge of expert chemists to the structure elucidation problem. DENDRAL's methods proved remarkably effective, routinely finding the correct structure out of millions of possibilities after only a few trials. The approach has proved so successful that descendants of the system are used in chemical and pharmaceutical laboratories throughout the world.

Whereas DENDRAL was one of the first programs to effectively use domain-specific knowledge to achieve expert level problem-solving performance, MYCIN established the methodology of contemporary expert systems (Buchanan and Shortliffe 1984). MYCIN uses expert medical knowledge to diagnose and prescribe treatment for spinal meningitis and bacterial infections of the blood.

MYCIN, developed at Stanford in the mid-1970s, was one of the first programs to address the problems of reasoning with uncertain or incomplete information. MYCIN provided clear and logical explanations of its reasoning, used a control structure appropriate to the specific problem domain, and identified criteria to reliably evaluate its performance. Many of the expert system development techniques currently in use were first developed in the MYCIN project (Chapter 8).

Other classic expert systems include the PROSPECTOR program for determining the probable location and type of ore deposits based on geological information about a site (Duda et al. 1979*a*, 1979*b*), the INTERNIST program for performing diagnosis in the area of internal medicine, the Dipmeter Advisor for interpreting the results of oil well drilling logs (Smith and Baker 1983), and XCON for configuring VAX computers. XCON was developed in 1981, and at one time every VAX sold by Digital Equipment Corporation was configured by that software. Numerous other expert systems are currently solving problems in areas such as medicine, education, business, design, and science (Waterman 1986, Durkin 1994).

It is interesting to note that most expert systems have been written for relatively specialized, expert level domains. These domains are generally well studied and have clearly defined problem-solving strategies. Problems that depend on a more loosely defined notion of "common sense" are much more difficult to solve by these means. In spite of the promise of expert systems, it would be a mistake to overestimate the ability of this technology. Current deficiencies include:

1. Difficulty in capturing "deep" knowledge of the problem domain. MYCIN, for example, lacks any real knowledge of human physiology. It does not know what blood does or the function of the spinal cord. Folklore has it that once, when selecting a drug for treatment of meningitis, MYCIN asked whether the patient was pregnant, even though it had been told that the patient was male. Whether this actually occurred or not, it does illustrate the potential narrowness of knowledge in expert systems.

2. Lack of robustness and flexibility. If humans are presented with a problem instance that they cannot solve immediately, they can generally return to an examination of first principles and come up with some strategy for attacking the problem. Expert systems generally lack this ability.

3. Inability to provide deep explanations. Because expert systems lack deep knowledge of their problem domains, their explanations are generally restricted to a description of the steps they took in finding a solution. For example, they often cannot tell "why" a certain approach was taken.

4. Difficulties in verification. Though the correctness of any large computer system is difficult to prove, expert systems are particularly difficult to verify. This is a serious problem, as expert systems technology is being applied to critical applications such as air traffic control, nuclear reactor operations, and weapons systems.

5. Little learning from experience. Current expert systems are handcrafted; once the system is completed, its performance will not improve without further attention from its programmers, leading to doubts about the intelligence of such systems.

In spite of these limitations, expert systems have proved their value in a number of important applications. It is hoped that these limitations will only encourage the student to pursue this important branch of computer science. Expert systems are a major topic in this text and are discussed in Chapters 7 and 8.

1.2.4 Natural Language Understanding and Semantics

One of the long-standing goals of artificial intelligence is the creation of programs that are capable of understanding and generating human language. Not only does the ability to use and understand natural language seem to be a fundamental aspect of human intelligence, but also its successful automation would have an incredible impact on the usability and effectiveness of computers themselves. Much effort has been put into writing programs that understand natural language. Although these programs have achieved success within restricted contexts, systems that can use natural language with the flexibility and generality that characterize human speech are beyond current methodologies.

Understanding natural language involves much more than parsing sentences into their individual parts of speech and looking those words up in a dictionary. Real understanding depends on extensive background knowledge about the domain of discourse and the

idioms used in that domain as well as an ability to apply general contextual knowledge to resolve the omissions and ambiguities that are a normal part of human speech.

Consider, for example, the difficulties in carrying on a conversation about baseball with an individual who understands English but knows nothing about the rules, players, or history of the game. Could this person possibly understand the meaning of the sentence: "With none down in the top of the ninth and the go-ahead run at second, the manager called his relief from the bull pen"? Even though all of the words in the sentence may be individually understood, this sentence would be gibberish to even the most intelligent non-baseball fan.

The task of collecting and organizing this background knowledge in such a way that it may be applied to language comprehension forms the major problem in automating natural language understanding. Responding to this need, researchers have developed many of the techniques for structuring semantic meaning used throughout artificial intelligence (Chapters 7 and 14).

Because of the tremendous amounts of knowledge required for understanding natural language, most work is done in well-understood, specialized problem areas. One of the earliest programs to exploit this "micro world" methodology was Winograd's SHRDLU, a natural language system that could "converse" about a simple configuration of blocks of different shapes and colors (Winograd 1973). SHRDLU could answer queries such as "what color block is on the blue cube?" as well as plan actions such as "move the red pyramid onto the green brick". Problems of this sort, involving the description and manipulation of simple arrangements of blocks, have appeared with surprising frequency in AI research and are known as "blocks world" problems.

In spite of SHRDLU's success in conversing about arrangements of blocks, its methods did not generalize from that domain. The representational techniques used in the program were too simple to capture the semantic organization of richer and more complex domains in a useful way. Much of the current work in natural language understanding is devoted to finding representational formalisms that are general enough to be used in a wide range of applications yet adapt themselves well to the specific structure of a given domain. A number of different techniques (many of which are extensions or modifications of *semantic networks*) are explored for this purpose and used in the development of programs that can understand natural language in constrained but interesting knowledge domains. Finally, in current research (Marcus 1980, Manning and Schutze 1999, Jurafsky and Martin 2000) stochastic models, describing how words and language structures "occur" in use, are employed to characterize both syntax and semantics. Full computational understanding of language, however, remains beyond the current state of the art.

1.2.5 Modeling Human Performance

Although much of the above discussion uses human intelligence as a reference point in considering artificial intelligence, it does not follow that programs should pattern themselves after the organization of the human mind. Indeed, many AI programs are engineered to solve some useful problem without regard for their similarities to human mental architecture. Even expert systems, while deriving much of their knowledge from

human experts, do not really attempt to simulate human internal mental problem solving processes. If performance is the only criterion by which a system will be judged, there may be little reason to attempt to simulate human problem-solving methods; in fact, programs that take nonhuman approaches to solving problems are often more successful than their human counterparts. Still, the design of systems that explicitly model aspects of human performance is a fertile area of research in both artificial intelligence and psychology.

Human performance modeling, in addition to providing AI with much of its basic methodology, has proved to be a powerful tool for formulating and testing theories of human cognition. The problem-solving methodologies developed by computer scientists have given psychologists a new metaphor for exploring the human mind. Rather than casting theories of cognition in the vague language used in early research or abandoning the problem of describing the inner workings of the human mind entirely (as suggested by the behaviorists), many psychologists have adopted the language and theory of computer science to formulate models of human intelligence. Not only do these techniques provide a new vocabulary for describing human intelligence, but also computer implementations of these theories offer psychologists an opportunity to empirically test, critique, and refine their ideas (Luger 1994). Further discussion of the relationship between artificial and human intelligence is found throughout this book and is summarized in Chapter 17.

1.2.6 Planning and Robotics

Research in planning began as an effort to design robots that could perform their tasks with some degree of flexibility and responsiveness to the outside world. Briefly, planning assumes a robot that is capable of performing certain atomic actions. It attempts to find a sequence of those actions that will accomplish some higher-level task, such as moving across an obstacle-filled room.

Planning is a difficult problem for a number of reasons, not the least of which is the size of the space of possible sequences of moves. Even an extremely simple robot is capable of generating a vast number of potential move sequences. Imagine, for example, a robot that can move forward, backward, right, or left, and consider how many different ways that robot can possibly move around a room. Assume also that there are obstacles in the room and that the robot must select a path that moves around them in some efficient fashion. Writing a program that can intelligently discover the best path under these circumstances, without being overwhelmed by the huge number of possibilities, requires sophisticated techniques for representing spatial knowledge and controlling search through possible environments.

One method that human beings use in planning is *hierarchical problem decomposition*. If you are planning a trip from Albuquerque to London, you will generally treat the problems of arranging a flight, getting to the airport, making airline connections, and finding ground transportation in London separately, even though they are all part of a bigger overall plan. Each of these may be further decomposed into smaller subproblems such as finding a map of the city, negotiating the subway system, and finding a decent pub. Not only does this approach effectively restrict the size of the space that must be searched, but also allows saving of frequently used subplans for future use.

While humans plan effortlessly, creating a computer program that can do the same is a difficult challenge. A seemingly simple task such as breaking a problem into independent subproblems actually requires sophisticated heuristics and extensive knowledge about the planning domain. Determining what subplans should be saved and how they may be generalized for future use is an equally difficult problem.

A robot that blindly performs a sequence of actions without responding to changes in its environment or being able to detect and correct errors in its own plan could hardly be considered intelligent. Often, a robot will have to formulate a plan based on incomplete information and correct its behavior as it executes the plan. A robot may not have adequate sensors to locate all obstacles in the way of a projected path. Such a robot must begin moving through the room based on what it has "perceived" and correct its path as other obstacles are detected. Organizing plans in a fashion that allows response to changing environmental conditions is a major problem for planning (Lewis and Luger 2000).

Finally, robotics was one of the research areas in AI that produced many of the insights supporting agent-oriented problem solving (Section 1.1.5). Frustrated by both the complexities of maintaining the large representational space as well as the design of adequate search algorithms for traditional planning, researchers, including Agre and Chapman (1987) and Brooks (1991*a*), restated the larger problem in terms of the interaction of multiple semi-autonomous agents. Each agent was responsible for its own portion of the problem task and through their coordination the larger solution would emerge.

Planning research now extends well beyond the domains of robotics, to include the coordination of any complex set of tasks and goals. Modern planners are applied to agents (Nilsson 1994) as well as to control of particle beam accelerators (Klein et al. 1999, 2000).

1.2 Languages and Environments for AI

Some of the most important by-products of artificial intelligence research have been advances in programming languages and software development environments. For a number of reasons, including the size of many AI application programs, the importance of prototyping methodology, the tendency of search algorithms to generate huge spaces, and the difficulty of predicting the behavior of heuristically driven programs, AI programmers have been forced to develop a powerful set of programming methodologies.

Programming environments include knowledge-structuring techniques such as object-oriented programming. High-level languages, such as LISP and PROLOG (Part VIII), which support modular development, help manage program size and complexity. Trace packages allow a programmer to reconstruct the execution of a complex algorithm and make it possible to unravel the complexities of heuristic search. Without such tools and techniques, it is doubtful that many significant AI systems could have been built.

Many of these techniques are now standard tools for software engineering and have little relationship to the core of AI theory. Others, such as object-oriented programming, are of significant theoretical and practical interest. Finally, many AI algorithms are also now built in more traditional computing languages, such as C++ and Java.

The languages developed for artificial intelligence programming are intimately bound to the theoretical structure of the field. We cover both LISP and PROLOG in this book and

prefer to remain apart from religious debates over their relative merits. Rather, we adhere to the adage "a good worker knows all her tools." The language chapters (15 and 16) discuss the advantages of each language for specific programming tasks.

1.2.8 Machine Learning

Learning has remained a challenging area for AI. The importance of learning, however, is beyond question, particularly as this ability is one of the most important components of intelligent behavior. An expert system may perform extensive and costly computations to solve a problem. Unlike a human being, however, if it is given the same or a similar problem a second time, it usually does not remember the solution. It performs the same sequence of computations again. This is true the second, third, fourth, and every time it solves that problem—hardly the behavior of an intelligent problem solver. The obvious solution to this problem is for programs to learn on their own, either from experience, analogy, examples, or by being "told" what to do.

Although learning is a difficult area, there are several programs that suggest that it is not impossible. One striking program is AM, the *Automated Mathematician*, designed to discover mathematical laws (Lenat 1977, 1982). Initially given the concepts and axioms of set theory, AM was able to induce such important mathematical concepts as cardinality, integer arithmetic, and many of the results of number theory. AM conjectured new theorems by modifying its current knowledge base and used heuristics to pursue the "best" of a number of possible alternative theorems. More recently, Cotton et al. (2000) designed a program that automatically invents "interesting" integer sequences.

Early influential work includes Winston's research on the induction of structural concepts such as "arch" from a set of examples in the blocks world (Winston 1975*a*). The ID3 algorithm has proved successful in learning general patterns from examples (Quinlan 1986*a*). *Meta-DENDRAL* learns rules for interpreting mass spectrographic data in organic chemistry from examples of data on compounds of known structure. *Teiresias*, an intelligent "front end" for expert systems, converts high-level advice into new rules for its knowledge base (Davis 1982). *Hacker* devises plans for performing blocks world manipulations through an iterative process of devising a plan, testing it, and correcting any flaws discovered in the candidate plan (Sussman 1975). Work in explanation-based learning has shown the effectiveness of prior knowledge in learning (Mitchell et al. 1986, DeJong and Mooney 1986). There are also now many important biological and sociological models of learning; we review these in the connectionist and emergent learning chapters (11 and 12).

The success of machine learning programs suggests the existence of a set of general learning principles that will allow the construction of programs with the ability to learn in realistic domains. We present several approaches to learning in Part IV.

1.2.9 Alternative Representations: Neural Nets and Genetic Algorithms

Most of the techniques presented in this AI book use explicitly represented knowledge and carefully designed search algorithms to implement intelligence. A very different approach

seeks to build intelligent programs using models that parallel the structure of neurons in the human brain or the evolving patterns found in genetic algorithms and artificial life.

A simple schematic of a neuron (Figure 1.2) consists of a cell body that has a number of branched protrusions, called *dendrites*, and a single branch called the *axon*. Dendrites receive signals from other neurons. When these combined impulses exceed a certain threshold, the neuron fires and an impulse, or *spike*, passes down the axon. Branches at the end of the axon form *synapses* with the dendrites of other neurons. The synapse is the point of contact between neurons; synapses may be either *excitatory* or *inhibitory*, either adding to the total of signals reaching the neuron or subtracting from that total.

This description of a neuron is excessively simple, but it captures those features that are relevant to neural models of computation. In particular, each computational unit computes some function of its inputs and passes the result along to connected units in the network: the final results are produced by the parallel and distributed processing of this network of neural connections and threshold weights.

Neural architectures are appealing mechanisms for implementing intelligence for a number of reasons. Traditional AI programs can be brittle and overly sensitive to noise. Human intelligence is much more flexible and good at interpreting noisy input, such as a face in a darkened room or a conversation at a noisy party. Neural architectures, because they capture knowledge in a large number of fine-grained units distributed about a network, seem to have more potential for partially matching noisy and incomplete data.

With genetic algorithms and artificial life we evolve new problem solutions from components of previous solutions. The genetic operators, such as crossover and mutation, work much like their genetic equivalents in the natural world, work to produce, for each new generation, ever better potential problem solutions. Artificial life produces its new generation as a function of the "quality" of its neighbors in previous generations.

Both neural architectures and genetic algorithms provide a natural model for parallelism, because each neuron or segment of a solution is an independent unit. Hillis (1985) has commented on the fact that humans get faster at a task as they acquire more knowledge, while computers tend to slow down. This slowdown is due to the cost of sequentially searching a knowledge base; a massively parallel architecture like the human brain would not suffer from this problem. Finally, something is intrinsically appealing about approaching the problems of intelligence from a neural or genetic point of view. After all, the evolved brain achieves intelligence and it does so using a neural architecture. We present neural networks, genetic algorithms, and artificial life, in Chapters 11 and 12.

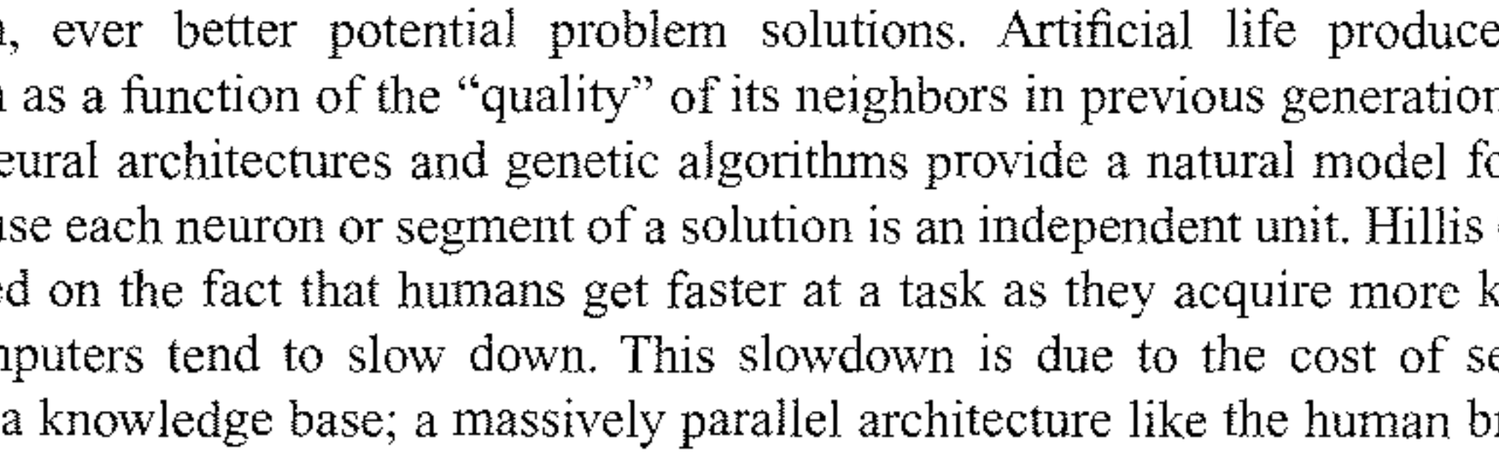

Figure 1.2 A simplified diagram of a neuron, from Crick and Asanuma (1986).

1.2.10 AI and Philosophy

In Section 1.1 we presented the philosophical, mathematical, and sociological roots of artificial intelligence. It is important to realize that modern AI is not just a product of this rich intellectual tradition but also contributes to it.

For example, the questions that Turing posed about intelligent programs reflect back on our understanding of intelligence itself. What is intelligence, and how is it described? What is the nature of knowledge? Can knowledge be represented? How does knowledge in an application area relate to problem-solving skill in that domain? How does *knowing what* is true, Aristotle's *theoria*, relate to *knowing how* to perform, his *praxis*?

Answers proposed to these questions make up an important part of what AI researchers and designers do. In the scientific sense, AI programs can be viewed as experiments. A design is made concrete in a program and the program is run as an experiment. The program designers observe the results and then redesign and rerun the experiment. In this manner we can determine whether our representations and algorithms are sufficient models of intelligent behavior. Newell and Simon (1976) proposed this approach to scientific understanding in their 1976 Turing Award lecture (Part VII). Newell and Simon (1976) also propose a stronger model for intelligence with their physical symbol system hypothesis: *the necessary and sufficient condition for a physical system to exhibit intelligence is that it be a physical symbol system*. We take up in Part VII what this hypothesis means in practice as well as how it has been criticized by many modern thinkers.

A number of AI application areas also open up deep philosophical issues. In what sense can we say that a computer can understand natural language expressions? To produce or understand a language requires interpretation of symbols. It is not sufficient to be able to say that a string of symbols is well formed. A mechanism for understanding must be able to impute meaning or interpret symbols in context. What is meaning? What is interpretation? In what sense does interpretation require responsibility?

Similar philosophical issues emerge from many AI application areas, whether they be building expert systems to cooperate with human problem solvers, designing computer vision systems, or designing algorithms for machine learning. We look at many of these issues as they come up in the chapters of this book and address the general issue of relevance to philosophy again in Part VII.

1.3 Artificial Intelligence—A Summary

We have attempted to define artificial intelligence through discussion of its major areas of research and application. This survey reveals a young and promising field of study whose primary concern is finding an effective way to understand and apply intelligent problem solving, planning, and communication skills to a wide range of practical problems. In spite of the variety of problems addressed in artificial intelligence research, a number of important features emerge that seem common to all divisions of the field; these include:

1. The use of computers to do reasoning, pattern recognition, learning, or some other form of inference.