CHAPTER

11

Artificial Intelligence

In this chapter we explore the branch of computer science known as artificial intelligence. Although this field is relatively young, it has produced some astonishing results such as expert chess players, computers that appear to learn and reason, and machines that coordinate their activities to achieve a common goal such as winning a soccer game. In artificial intelligence, today's science fiction might well be tomorrow's reality.

Artificial Intelligence is the field of computer science that seeks to build autonomous machines—machines that can carry out complex tasks without human intervention. This goal requires that machines be able to perceive and reason. Such capabilities fall within the category of commonsense activities that, although natural for the human mind, are proving difficult for machines. The result is that work in the field continues to be challenging. In this chapter we explore some of the topics in this vast area of research.

11.1 Intelligence and Machines

The field of artificial intelligence is quite large and merges with other subjects such as psychology, neurology, mathematics, linguistics, and electrical and mechanical engineering. To focus our thoughts, then, we begin by considering the concept of an agent and the types of intelligent behavior that an agent might exhibit. Indeed, much of the research in artificial intelligence can be categorized in terms of an agent's behavior.

Intelligent Agents

An **agent** is a "device" that responds to stimuli from its environment. It is natural to envision an agent as an individual machine such as a robot, although an agent may take other forms such as an autonomous airplane, a character in an interactive video game, or a process communicating with other processes over the Internet (perhaps as a client, a server, or a peer). Most agents have sensors by which they receive data from their environments and actuators by which they can affect their environments. Examples of sensors include microphones, cameras, range sensors, and air or soil sampling devices. Examples of actuators include wheels, legs, wings, grippers, and speech synthesizers.

Much of the research in artificial intelligence can be characterized in the context of building agents that behave intelligently, meaning that the actions of the agent's actuators must be rational responses to the data received though its sensors. In turn, we can classify this research by considering different levels of these responses.

The simplest response is a reflex action, which is merely a predetermined response to the input data. Higher levels of response are required to obtain more "intelligent" behavior. For example, we might empower an agent with knowledge of its environment and require that the agent adjust its actions accordingly. The process of throwing a baseball is largely a reflex action but determining how and where to throw the ball requires knowledge of the current environment. (There is one out with runners on first and third.) How such real-world knowledge can be stored, updated, accessed, and ultimately applied in the decision-making process continues to be a challenging problem in artificial intelligence.

Another level of response is required if we want the agent to seek a goal such as winning a game of chess or maneuvering through a crowded passageway. Such goal-directed behavior requires that the agent's response, or sequence of responses, be the result of deliberately forming a plan of action or selecting the best action among the current options.

In some cases an agent's responses improve over time as the agent learns. This could take the form of developing **procedural knowledge** (learning "how") or storing **declarative knowledge** (learning "what"). Learning procedural

knowledge usually involves a trial and error process by which an agent learns appropriate actions by being punished for poor actions and rewarded for good ones. Following this approach, agents have been developed that, over time, improve their abilities in competitive games such as checkers and chess. Learning declarative knowledge usually takes the form of expanding or altering the "facts" in an agent's store of knowledge. For example, a baseball player must repeatedly adjust his or her database of knowledge (there is still just one out, but now runners are on first and second) from which rational responses to future events are determined.

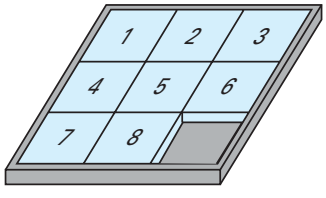
To produce rational responses to stimuli, an agent must "understand" the stimuli received by its sensors. That is, an agent must be able to extract information from the data produced by its sensors, or in other words, an agent must be able to perceive. In some cases this is a straightforward process. Signals obtained from a gyroscope are easily encoded in forms compatible with calculations for determining responses. But in other cases extracting information from input data is difficult. Examples include understanding speech and images. Likewise, agents must be able to formulate their responses in terms compatible with their actuators. This might be a straightforward process or it might require an agent to formulate responses as complete spoken sentences—meaning that the agent must generate speech. In turn, such topics as image processing and analysis, natural language understanding, and speech generation are important areas of research.

The agent attributes that we have identified here represent past as well as current areas of research. Of course, they are not totally independent of each other. We would like to develop agents that possess all of them, producing agents that understand the data received from their environments and develop new response patterns through a learning process whose goal is to maximize the agent's abilities. However, by isolating various types of rational behavior and pursuing them independently, researchers gain a toehold that can later be combined with progress in other areas to produce more intelligent agents.

We close this subsection by introducing an agent that will provide a context for our discussion in Sections 11.2 and 11.3. The agent is designed to solve the eight-puzzle, which consists of eight square tiles labeled 1 through 8 mounted in a frame capable of holding a total of nine such tiles in three rows and three columns (Figure 11.1). Among the tiles in the frame is a vacancy into which any of the adjacent tiles can be pushed, allowing the tiles in the frame to be scrambled. The problem posed is to move the tiles in a scrambled puzzle back to their initial positions (Figure 11.1).

Our agent takes the form of a box equipped with a gripper, a video camera, and a finger with a rubber end so that it does not slip when pushing something

Figure 11.1 The eight-puzzle in its solved configuration



(Figure 11.2). When the agent is first turned on, its gripper begins to open and close as if asking for the puzzle. When we place a scrambled eight-puzzle in the gripper, the gripper closes on the puzzle. After a short time the machine's finger lowers and begins pushing the tiles around in the frame until they are back in their original positions. At this point the machine releases the puzzle and turns itself off.

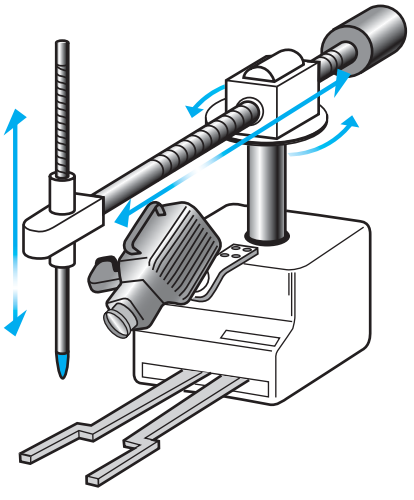
This puzzle-solving machine exhibits two of the agent attributes that we have identified. First, it must be able to perceive in the sense that it must extract the current puzzle state from the image it receives from its camera. We will address issues of understanding images in Section 11.2. Second, it must develop and implement a plan for obtaining a goal. We will address these issues in Section 11.3.

Research Methodologies

To appreciate the field of artificial intelligence, it is helpful to understand that it is being pursued along two paths. One is the engineering track in which researchers are trying to develop systems that exhibit intelligent behavior. The other is a theoretical track in which researchers are trying to develop a computational understanding of animal—especially human—intelligence. This dichotomy is clarified by considering the manner in which the two tracks are pursued. The engineering approach leads to a performance-oriented methodology because the underlying goal is to produce a product that meets certain performance goals. The theoretical approach leads to a simulation-oriented methodology because the underlying goal is to expand our understanding of intelligence and thus the emphasis is on the underlying process rather than the exterior performance.

As an example, consider the fields of natural language processing and linguistics. These fields are closely related and benefit from research in each other, yet the underlying goals are different. Linguists are interested in learning how humans process language and thus tend toward more theoretical

Figure 11.2 Our puzzle-solving machine



pursuits. Researchers in the field of natural language processing are interested in developing machines that can manipulate natural language and therefore lean in the engineering direction. Thus, linguists operate in simulation-oriented mode—building systems whose goals are to test theories. In contrast, researchers in natural language processing operate in performance-oriented mode—building systems to perform tasks. Systems produced in this latter mode (such as document translators and systems by which machines respond to verbal commands) rely heavily on knowledge gained by linguists but often apply "shortcuts" that happen to work in the restricted environment of the particular system.

As an elementary example, consider the task of developing a shell for an operating system that receives instructions from the outside world through verbal English commands. In this case, the shell (an agent) does not need to worry about the entire English language. More precisely, the shell does not need to distinguish between the various meanings of the word *copy.* (Is it a noun or a verb? Should it carry the connotation of plagiarism?) Instead, the shell needs merely to distinguish the word *copy* from other commands such as *rename* and *delete.* Thus the shell could perform its task just by matching its inputs to predetermined audio patterns. The performance of such a system may be satisfactory to an engineer but the way it is obtained would not be aesthetically pleasing to a theoretician.

The Turing Test

In the past the **Turing test** (proposed by Alan Turing in 1950) has served as a benchmark in measuring progress in the field of artificial intelligence. Today the significance of the Turing test has faded although it remains an important part of the artificial intelligence folklore. Turing's proposal was to allow a human, whom we call the interrogator, to communicate with a test subject by means of a typewriter system without being told whether the test subject was a human or a machine. In this environment, a machine would be declared to behave intelligently if the interrogator was not able to distinguish it from a human. Turing predicted that by the year 2000 machines would have a 30 percent chance of passing a five-minute Turing test—a conjecture that turned out to be surprisingly accurate.

The Origins of Artificial Intelligence

The quest to build machines that mimic human behavior has a long history, but many would agree that the modern field of artificial intelligence had its origins in 1950. This was the year that Alan Turing published the article "Computing Machinery and Intelligence" in which he proposed that machines could be programmed to exhibit intelligent behavior. The name of the field—*artificial intelligence*—was coined a few years later in the now legendary proposal written by John McCarthy who suggested that a "study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College" to explore "the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it."

One reason that the Turing test is no longer considered to be a meaningful measure of intelligence is that an eerie appearance of intelligence can be **produced** with relative ease. A well-known example arose as a result of the program DOCTOR (a version of the more general system called ELIZA) developed by Joseph Weizenbaum in the mid-1960s. This interactive program was designed to project the image of a Rogerian analyst conducting a psychological interview; the computer played the role of the analyst while the user played the patient. Internally, all that DOCTOR did was restructure the statements made by the patient according to some well-defined rules and direct them back to the patient. For example, in response to the statement "I am tired today," DOCTOR might have replied with "Why do you think you're tired today?" If DOCTOR was unable to recognize the sentence structure, it merely responded with something like "Go on" or "That's very interesting."

Weizenbaum's purpose in developing DOCTOR dealt with the study of natural language communication. The subject of psychotherapy merely provided an environment in which the program could "communicate." To Weizenbaum's dismay, however, several psychologists proposed using the program for actual psychotherapy. (The Rogerian thesis is that the patient, not the analyst, should lead the discussion during the therapeutic session, and thus, they argued, a computer could possibly conduct a discussion as well as a therapist could.) Moreover, DOCTOR projected the image of comprehension so strongly that many who "communicated" with it became subservient to the machine's question-and-answer dialogue. In a sense, DOCTOR passed the Turing test. The result was that ethical, as well as technical, issues were raised, and Weizenbaum became an advocate for maintaining human dignity in a world of advancing technology.

More recent examples of Turing test "successes" include Internet viruses that carry on "intelligent" dialogs with a human victim in order to trick the human into dropping his or her malware guard. Moreover, phenomena similar to Turing tests occur in the context of computer games such as chess-playing programs. Although these programs select moves merely by applying brute-force techniques (similar to those we will discuss in Section 11.3), humans competing against the computer often experience the sensation that the machine possesses creativity and even a personality. Similar sensations occur in robotics where machines have been built with physical attributes that project intelligent characteristics. Examples include toy robot dogs that project adorable personalities merely by tilting their heads or lifting their ears in response to a sound.

Questions & Exercises

1. Identify several types of "intelligent" actions that might be made by an agent.
2. A plant placed in a dark room with a single light source grows toward the light. Is this an intelligent response? Does the plant possess intelligence? What, then, is your definition of intelligence?
3. Suppose a vending machine is designed to dispense various products depending on which button is pressed. Would you say that such a machine is "aware" of which button is pressed? What, then, is your definition of awareness?

4. If a machine passes the Turing test, would you agree that it is intelligent? If not, would you agree that it appears to be intelligent?
5. Suppose you used a chat room to chat with someone over the Internet (or used Instant Messenger) and carried on a meaningful coherent conversation for ten minutes. If later you found out that you had conversed with a machine, would you conclude that the machine was intelligent? Why or why not?

11.2 Perception

To respond intelligently to the input from its sensors, an agent must be able to understand that input. That is, the agent must be able to perceive. In this section we explore two areas of research in perception that have proven to be especially challenging—understanding images and language.

Understanding Images

Let us consider the problems posed by the puzzle-solving machine introduced in the previous section. The opening and closing of the gripper on the machine presents no serious obstacle, and the ability to detect the presence of the puzzle in the gripper during this process is straightforward because our application requires very little precision. Even the problem of focusing the camera on the puzzle can be handled simply by designing the gripper to position the puzzle at a particular predetermined position for viewing. Consequently, the first intelligent behavior required by the machine is the extraction of information through a visual medium.

It is important to realize that the problem faced by our machine when looking at the puzzle is not that of merely producing and storing an image. Technology has been able to do this for years as in the case of traditional photography and television systems. Instead, the problem is to understand the image in order to extract the current status of the puzzle (and perhaps later to monitor the movement of the tiles).

In the case of our puzzle-solving machine, the possible interpretations of the puzzle image are relatively limited. We can assume that what appears is always an image containing the digits 1 through 8 in a well-organized pattern. The problem is merely to extract the arrangement of these digits. For this, we imagine that the picture of the puzzle has been encoded in terms of bits in the computer's memory, with each bit representing the brightness level of a particular pixel. Assuming a uniform size of the image (the machine holds the puzzle at a predetermined location in front of the camera), our machine can detect which tile is in which position by comparing the different sections of the picture to prerecorded templates consisting of the bit patterns produced by the individual digits used in the puzzle. As matches are found, the condition of the puzzle is revealed.

This technique of recognizing images is one method used in optical character readers. It has the drawback, however, of requiring a certain degree of uniformity for the style, size, and orientation of the symbols being read. In particular, the bit pattern produced by a physically large character does not match the template for a smaller version of the same symbol, even though the shapes are the same. Moreover, you can imagine how the problems increase in difficulty when trying to process handwritten material.

Another approach to the problem of character recognition is based on matching the geometric characteristics rather than the exact appearance of the symbols. In such cases the digit 1 might be characterized as a single vertical line, 2 might be an opened curved line joined with a horizontal straight line across the bottom, and so on. This method of recognizing symbols involves two steps: the first is to extract the features from the image being processed, and the second is to compare the features to those of known symbols. As with the template-matching approach, this technique for recognizing characters is not foolproof. For instance, minor errors in the image can produce a set of entirely different geometric features, as in the case of distinguishing between an O and a C or, in the case of the eight-puzzle, a 3 and an 8.

We are fortunate in our puzzle application because we do not need to understand images of general three-dimensional scenes. Consider, for example, the advantage we have by being assured that the shapes to be recognized (the digits 1 through 8) are isolated in different parts of the picture rather than appearing as overlapping images, as is common in more general settings. In a general photograph, for instance, one is faced not only with the problem of recognizing an object from different angles but also with the fact that some portions of the object might be hidden from view.

The task of understanding general images is usually approached as a two-step process: (1) **image processing,** which refers to identifying characteristics of the image, and (2) **image analysis,** which refers to the process of understanding what these characteristics mean. We have already observed this dichotomy in the context of recognizing symbols by means of their geometric features. In that situation, we found image processing represented by the process of identifying the geometric features found in the image and image analysis represented by the process of identifying the meaning of those features.

Image processing entails numerous topics. One is edge enhancement, which is the process of applying mathematical techniques to clarify the boundaries between regions in an image. In a sense, edge enhancement is an attempt to convert a photograph into a line drawing. Another activity in image analysis is known as

Strong AI Versus Weak AI

The conjecture that machines can be programmed to exhibit intelligent behavior is known as **weak AI** and is accepted, to varying degrees, by a wide audience today. However, the conjecture that machines can be programmed to possess intelligence and, in fact, consciousness, which is known as **strong AI,** is widely debated. Opponents of strong AI argue that a machine is inherently different from a human and thus can never feel love, tell right from wrong, and think about itself in the same way that a human does. However, proponents of strong AI argue that the human mind is constructed from small components that individually are not human and are not conscious but, when combined, are. Why, they argue, would the same phenomenon not be possible with machines?

The problem in resolving the strong AI debate is that such attributes as intelligence and consciousness are internal characteristics that cannot be identified directly. As Alan Turing pointed out, we credit other humans with intelligence because they behave intelligently—even though we cannot observe their internal mental states. Are we, then, prepared to grant the same latitude to a machine if it exhibits the external characteristics of consciousness? Why or why not?

region finding. This is the process of identifying those areas in an image that have common properties such as brightness, color, or texture. Such a region probably represents a section of the image that belongs to a single object. (It is the ability to recognize regions that allows computers to add color to old-fashioned black and white motion pictures.) Still another activity within the scope of image processing is smoothing, which is the process of removing flaws in the image. Smoothing keeps errors in the image from confusing the other image-processing steps, but too much smoothing can cause the loss of important information as well.

Smoothing, edge enhancement, and region finding are all steps toward identifying the various components in an image. Image analysis is the process of determining what these components represent and ultimately what the image means. Here one faces such problems as recognizing partially obstructed objects from different perspectives. One approach to image analysis is to start with an assumption about what the image might be and then try to associate the components in the image with the objects whose presence is conjectured. This appears to be an approach applied by humans. For instance, we sometimes find it hard to recognize an unexpected object in a setting in which our vision is blurred, but once we have a clue to what the object might be, we can easily identify it.

The problems associated with general image analysis are enormous, and much research in the area remains to be done. Indeed, image analysis is one of the fields that demonstrates how tasks that are performed quickly and apparently easily by the human mind continue to challenge the capabilities of machines.

Language Processing

Another perception problem that has proven challenging is that of understanding language. The success obtained in translating formal high-level programming languages into machine language (Section 6.4) led early researchers to believe that the ability to program computers to understand natural language was only a few years away. Indeed, the ability to translate programs gives the illusion that the machine actually understands the language being translated. (Recall from Section 6.1 the story told by Grace Hopper about managers who thought she was teaching computers to understand German.)

What these researchers failed to understand was the depth to which formal programming languages differ from natural languages such as English, German, and Latin. Programming languages are constructed from well-designed primitives so that each statement has only one grammatical structure and only one meaning. In contrast, a statement in a natural language can have multiple meanings depending on its context or even the manner in which it is communicated. Thus, to understand natural language, humans rely heavily on additional knowledge.

For example, the sentences

> Norman Rockwell painted people.

and

> Cinderella had a ball.

have multiple meanings that cannot be distinguished by parsing or translating each word independently. Instead, to understand these sentences requires the ability to comprehend the context in which the statement is made. In other instances the true meaning of a sentence is not the same as its literal translation. For example,

> Do you know what time it is?

often means "Please tell me what time it is," or if the speaker has been waiting for a long time, it might mean "You are very late."

To unravel the meaning of a statement in a natural language therefore requires several levels of analysis. The first of these is **syntactic analysis,** whose major component is parsing. It is here that the subject of the sentence

Mary gave John a birthday card.

is recognized as *Mary* while the subject of

John got a birthday card from Mary.

is found to be *John.*

Another level of analysis is called **semantic analysis.** In contrast to the parsing process, which merely identifies the grammatical role of each word, semantic analysis is charged with the task of identifying the semantic role of each word in the statement. Semantic analysis seeks to identify such things as the action described, the agent of that action (which might or might not be the subject of the sentence), and the object of the action. It is through semantic analysis that the sentences "Mary gave John a birthday card" and "John got a birthday card from Mary" would be recognized as saying the same thing.

A third level of analysis is **contextual analysis.** It is at this level that the context of the sentence is brought into the understanding process. For example, it is easy to identify the grammatical role of each word in the sentence

The bat fell to the ground.

We can even perform semantic analysis by identifying the action involved as *falling,* the agent as *bat,* and so on. But it is not until we consider the context of the statement that the meaning of the statement becomes clear. In particular, it has a different meaning in the context of a baseball game than it does in the context of cave exploration. Moreover, it is at the contextual level that the true meaning of the question "Do you know what time it is?" would finally be revealed.

We should note that the various levels of analysis—syntactic, semantic, and contextual—are not necessarily independent. The subject of the sentence

Stampeding cattle can be dangerous.

is the noun *cattle* (modified by the adjective *stampeding*) if we envision the cattle stampeding on their own. But the subject is the gerund *stampeding* (with object *cattle*) in the context of a troublemaker whose entertainment consists of starting stampedes. Thus the sentence has more than one grammatical structure—which one is correct depends on the context.

Another area of research in natural language processing concerns an entire document rather than individual sentences. Here the problems of concern fall into two categories: **information retrieval** and **information extraction.** Information retrieval refers to the task of identifying documents that relate to the topic at hand. An example is the problem faced by users of the World Wide Web as they try to find the sites that relate to a particular topic. The current state of the art is to search sites for key words, but this often produces an avalanche of false leads and can overlook an important site because it deals with "automobiles" instead of "cars." What is needed is a search mechanism that understands the contents of the sites being considered. The difficulty of obtaining such understanding is the reason many are turning to techniques such as XML to produce a semantic Web, as introduced in Section 4.3.

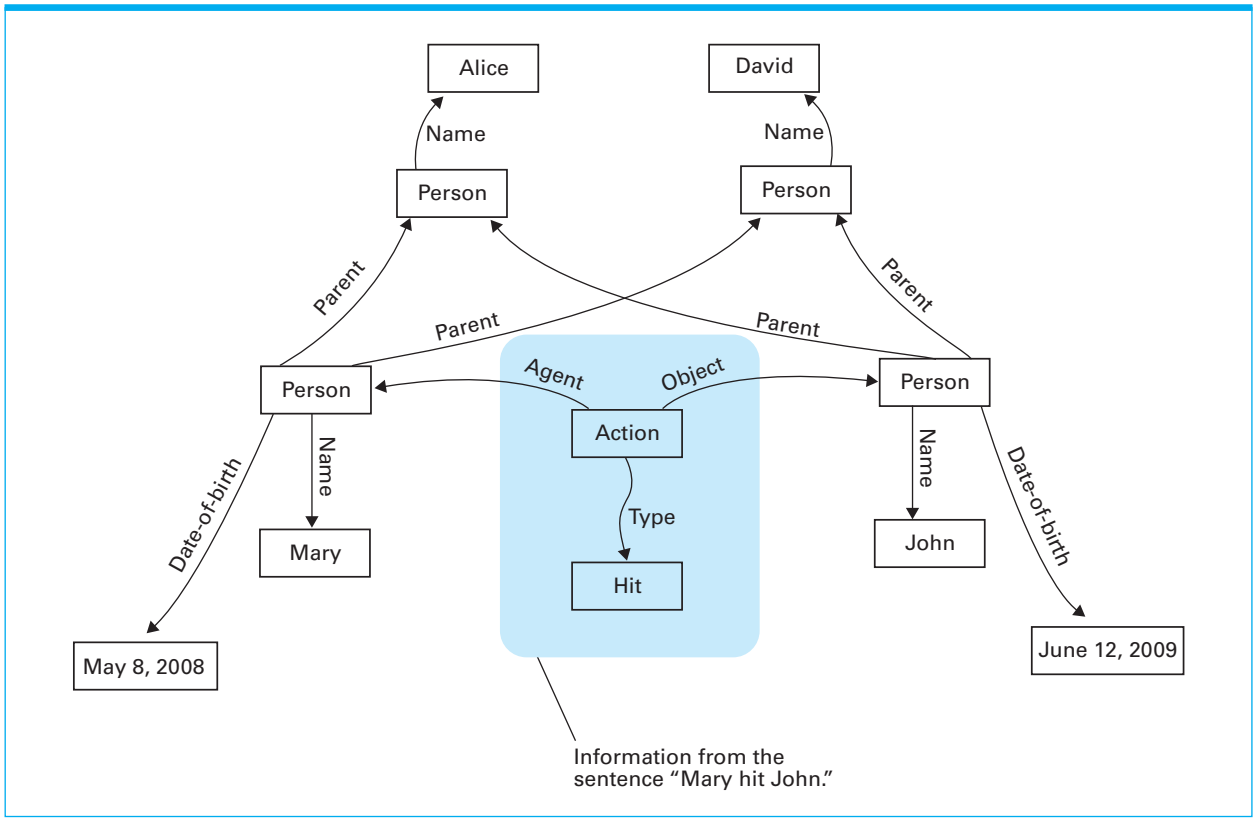
Information extraction refers to the task of extracting information from documents so that it takes a form that is useful in other applications. This might mean identifying the answer to a specific question or recording the information in a form from which questions can be answered at a later date. One such form is known as a **frame,** which is essentially a template in which specifics are recorded. For example, consider a system for reading a newspaper. The system might make use of a variety of frames, one for each type of article that might appear in a newspaper. If the system identifies an article as reporting on a burglary, it would proceed by trying to fill in the slots in the burglary frame. This frame would probably request such items as the address of the burglary, the time and date of the burglary, the items taken, and so on. In contrast, if the system identifies an article as reporting on a natural disaster, it would fill in the natural disaster frame, which would lead the system toward identifying the type of disaster, amount of damage, and so on.

Another form in which information extractors record information is known as a **semantic net.** This is essentially a large linked data structure in which pointers are used to indicate associations among the data items. Figure 11.3 shows part of a semantic net in which the information obtained from the sentence

Mary hit John.

has been highlighted.

Figure 11.3 A semantic net

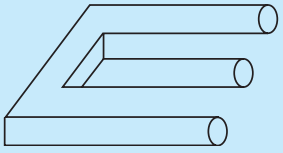


Artificial Intelligence in the Palm of Your Hand

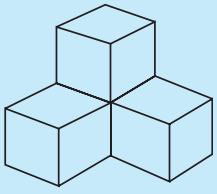
Artificial intelligence techniques are increasingly showing up in smartphone applications. For example, Google has developed Google Goggles, a smartphone application providing a visual search engine. Just take a picture of a book, landmark, or sign using a smartphone's camera and Goggles will perform image processing, image analysis, and text recognition, and then initiate a Web search to identify the object. If you are an English speaker visiting in France, you can take a picture of a sign, menu, or other text and have it translated to English. Beyond Goggles, Google is actively working on voice-to-voice language translation. Soon you will be able to speak English into your phone and have your words spoken in Spanish, Chinese, or another language. Smartphones will undoubtedly get smarter as AI continues to be utilized in innovative ways.

Questions & Exercises

1. How do the requirements of a video system on a robot differ if the robot itself uses them to control its activities rather than relaying them to a human who controls the robot remotely?
2. What tells you that the following drawing is nonsense? How can this insight be programmed into a machine?



3. How many blocks are in the stack represented next? How could a machine be programmed to answer such questions accurately?



4. How do you know that the two statements "Nothing is better than complete happiness" and "A bowl of cold soup is better than nothing" do not imply that "A bowl of cold soup is better than complete happiness"? How can your ability to make this differentiation be transferred to a machine?
5. Identify the ambiguities involved in translating the sentence "They are racing horses."
6. Compare the results of parsing the following two sentences. Then, explain how the sentences differ semantically.

   The farmer built the fence in the field.

   The farmer built the fence in the winter.

7. Based on the semantic net in Figure 11.3, what is the family relationship between Mary and John?

11.3 Reasoning

Let us now use the puzzle-solving machine introduced in Section 11.1 to explore techniques for developing agents with elementary reasoning abilities.

Production Systems

Once our puzzle-solving machine has deciphered the positions of the tiles from the visual image, its task becomes that of figuring out what moves are required to solve the puzzle. An approach to this problem that might come to mind is to preprogram the machine with solutions to all possible arrangements of the tiles. Then the machine's task would merely be to select and execute the proper program. However, the eight-puzzle has over 100,000 configurations, so the idea of providing an explicit solution for each is not inviting. Thus, our goal is to program the machine so that it can construct solutions to the eight-puzzle on its own. That is, the machine must be programmed to perform elementary reasoning activities.

The development of reasoning abilities within a machine has been a topic of research for many years. One of the results of this research is the recognition that there is a large class of reasoning problems with common characteristics. These common characteristics are isolated in an abstract entity known as a **production system,** which consists of three main components:

1. *A collection of states.* Each **state** is a situation that might occur in the application environment. The beginning state is called the **start** (or initial) **state;** the desired state (or states) is called the **goal state.** (In our case, a state is a configuration of the eight-puzzle; the start state is the configuration of the puzzle when it is handed to the machine; the goal state is the configuration of the solved puzzle as shown in Figure 11.1.)
2. *A collection of productions (rules or moves).* A **production** is an operation that can be performed in the application environment to move from one state to another. Each production might be associated with preconditions; that is, conditions might exist that must be present in the environment before a production can be applied. (Productions in our case are the movements of tiles. Each movement of a tile has the precondition that the vacancy must be next to the tile in question.)
3. *A control system.* The **control system** consists of the logic that solves the problem of moving from the start state to the goal state. At each step in the process the control system must decide which of those productions whose preconditions are satisfied should be applied next. (Given a particular state in our eight-puzzle example, there would be several tiles next to the vacancy and therefore several applicable productions. The control system must decide which tile to move.)

Note that the task assigned to our puzzle-solving machine can be formulated in the context of a production system. In this setting the control system takes the form

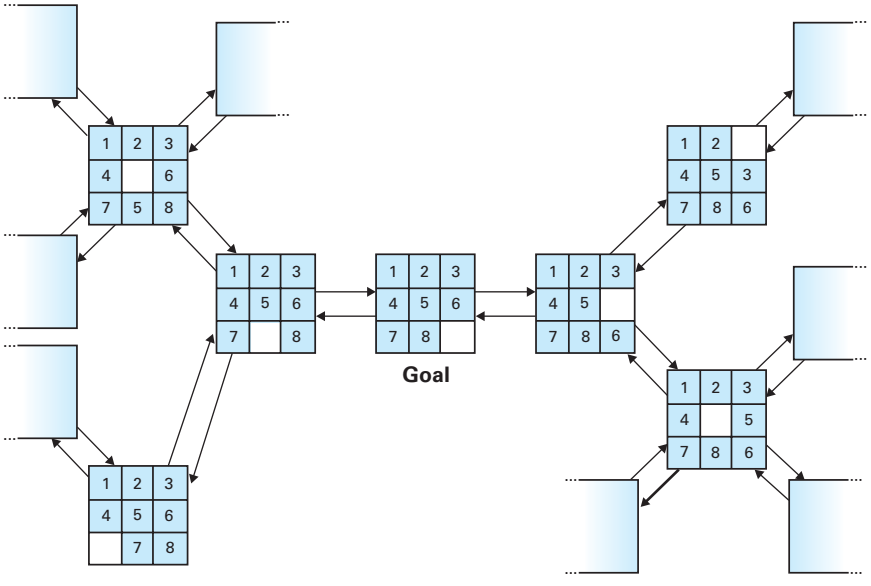
of a program. This program inspects the current state of the eight-puzzle, identifies a sequence of productions that leads to the goal state, and executes this sequence. It is therefore our task to design a control system for solving the eight-puzzle.

An important concept in the development of a control system is that of a **problem space,** which is the collection of all the states, productions, and preconditions in a production system. A problem space is often conceptualized in the form of a **state graph.** Here the term *graph* refers to a structure that mathematicians would call a **directed graph,** meaning a collection of locations called **nodes** connected by arrows. A state graph consists of a collection of nodes representing the states in the system connected by arrows representing the productions that shift the system from one state to another. Two nodes are connected by an arrow in the state graph if and only if there is a production that transforms the system from the state at the origin of the arrow to the state at the destination of the arrow.

We should emphasize that just as the number of possible states prevented us from explicitly providing preprogrammed solutions to the eight-puzzle, the problem of magnitude prevents us from explicitly representing the entire state graph. A state graph is therefore a way of conceptualizing the problem at hand but not something that we would consider drawing in its entirety. Nonetheless, you might find it helpful to consider (and possibly extend) the portion of the state graph for the eight-puzzle displayed in Figure 11.4.

When viewed in terms of the state graph, the problem faced by the control system becomes that of finding a sequence of arrows that leads from the start state to the goal state. After all, this sequence of arrows represents a sequence of productions that solves the original problem. Thus, regardless of the application, the task of the control system can be viewed as that of finding a path through a state graph. This universal view of control systems is the prize that we obtain by analyzing problems requiring reasoning in terms of production systems. If a problem can be characterized in terms of a production system, then its solution can be formulated in terms of searching for a path.

Figure 11.4 A small portion of the eight-puzzle's state graph

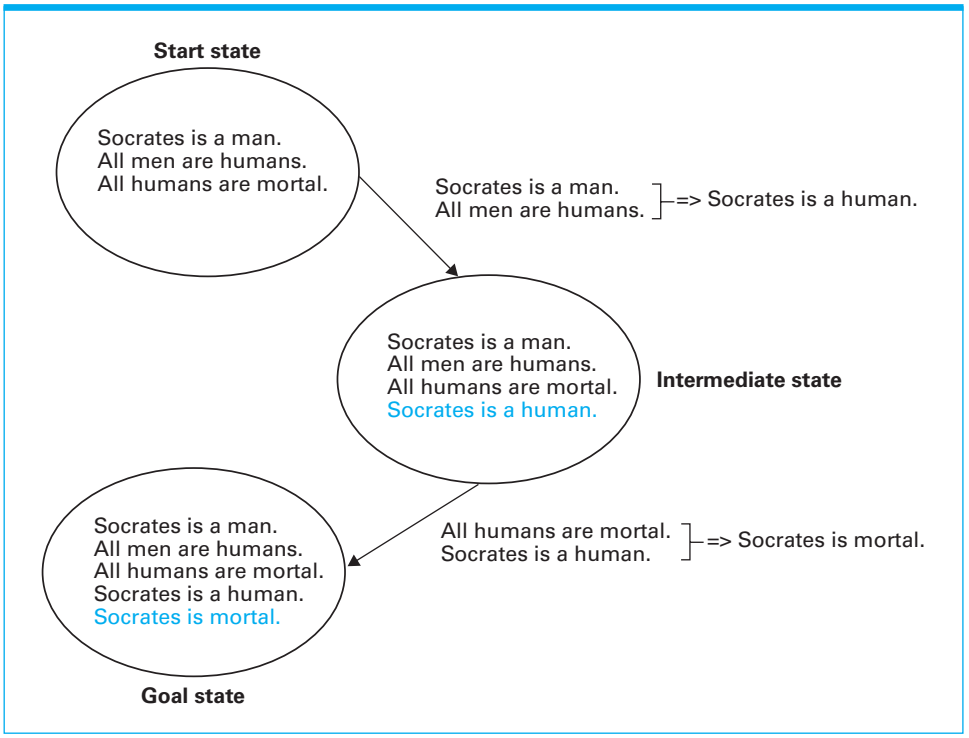


To emphasize this point, let us consider how other tasks can be framed in terms of production systems and thus performed in the context of control systems finding paths through state graphs. One of the classic problems in artificial intelligence is playing games such as chess. These games involve moderate complexity in a well-defined context and hence provide an ideal environment for testing theories. In chess the states of the underlying production system are the possible board configurations, the productions are the moves of the pieces, and the control system is embodied in the players (human or otherwise). The start node of the state graph represents the board with the pieces in their initial positions. Branching from this node are arrows leading to those board configurations that can be reached after the first move in a game; branching from each of those configurations are arrows to those configurations that are reachable by the next move; and so on. With this formulation, we can imagine a game of chess as consisting of two players, each trying to find a path through a large state graph to a goal node of his or her own choosing.

Perhaps a less obvious example of a production system is the problem of drawing logical conclusions from given facts. The productions in this context are the rules of logic, called **inference rules,** that allow new statements to be formed from old ones. For example, the statements "All super heroes are noble" and "Superman is a super hero" can be combined to produce "Superman is noble." States in such a system consist of collections of statements known to be true at particular points in the deduction process: The start state is the collection of basic statements (often called axioms) from which conclusions are to be drawn, and a goal state is any collection of statements that contain the proposed conclusion.

As an example, Figure 11.5 shows the portion of a state graph that might be traversed when the conclusion "Socrates is mortal" is drawn from the collection of

Figure 11.5 Deductive reasoning in the context of a production system



statements "Socrates is a man," "All men are humans," and "All humans are mortal." There we see the body of knowledge shifting from one state to another as the reasoning process applies appropriate productions to generate additional statements.

Today, such reasoning systems, often implemented in logic programming languages (Section 6.7), are the backbone of most **expert systems,** which are software packages designed to simulate the cause-and-effect reasoning that human experts would follow if confronted with the same situations. Medical expert systems, for example, are used to assist in diagnosing ailments or developing treatments.

Search Trees

We have seen that, in the context of a production system, a control system's job involves searching the state graph to find a path from the start node to a goal. A simple method of performing this search is to traverse each of the arrows leading from the start state and in each case record the destination state, then traverse the arrows leaving these new states and again record the results, and so on. The search for a goal spreads out from the start state like a drop of dye in water. This process continues until one of the new states is a goal, at which point a solution has been found, and the control system needs merely to apply the productions along the discovered path from the start state to the goal.

The effect of this strategy is to build a tree, called a **search tree,** that consists of the part of the state graph that has been investigated by the control system. The root node of the search tree is the start state, and the children of each node are those states reachable from the parent by applying one production. Each arc between nodes in a search tree represents the application of a single production, and each path from the root to a leaf represents a path between the corresponding states in the state graph.

The search tree that would be produced in solving the eight-puzzle from the configuration shown in Figure 11.6 is illustrated in Figure 11.7. The leftmost branch of this tree represents an attempt to solve the problem by first moving the 6 tile up, the center branch represents the approach of moving the 2 tile to the right, and the rightmost branch represents moving the 5 tile down. Furthermore, the search tree shows that if we do begin by moving the 6 tile up, then the only production allowable next is to move the 8 tile to the right. (Actually, at that point we could also move the 6 tile down but that would merely reverse the previous production and thus be an extraneous move.)

The goal state occurs in the last level of the search tree of Figure 11.7. Since this indicates that a solution has been found, the control system can terminate its search procedure and begin constructing the instruction sequence that will be used to solve the puzzle in the external environment. This turns out to be the simple process of walking up the search tree from the location of the goal node

Figure 11.6 An unsolved eight-puzzle

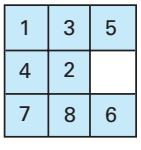
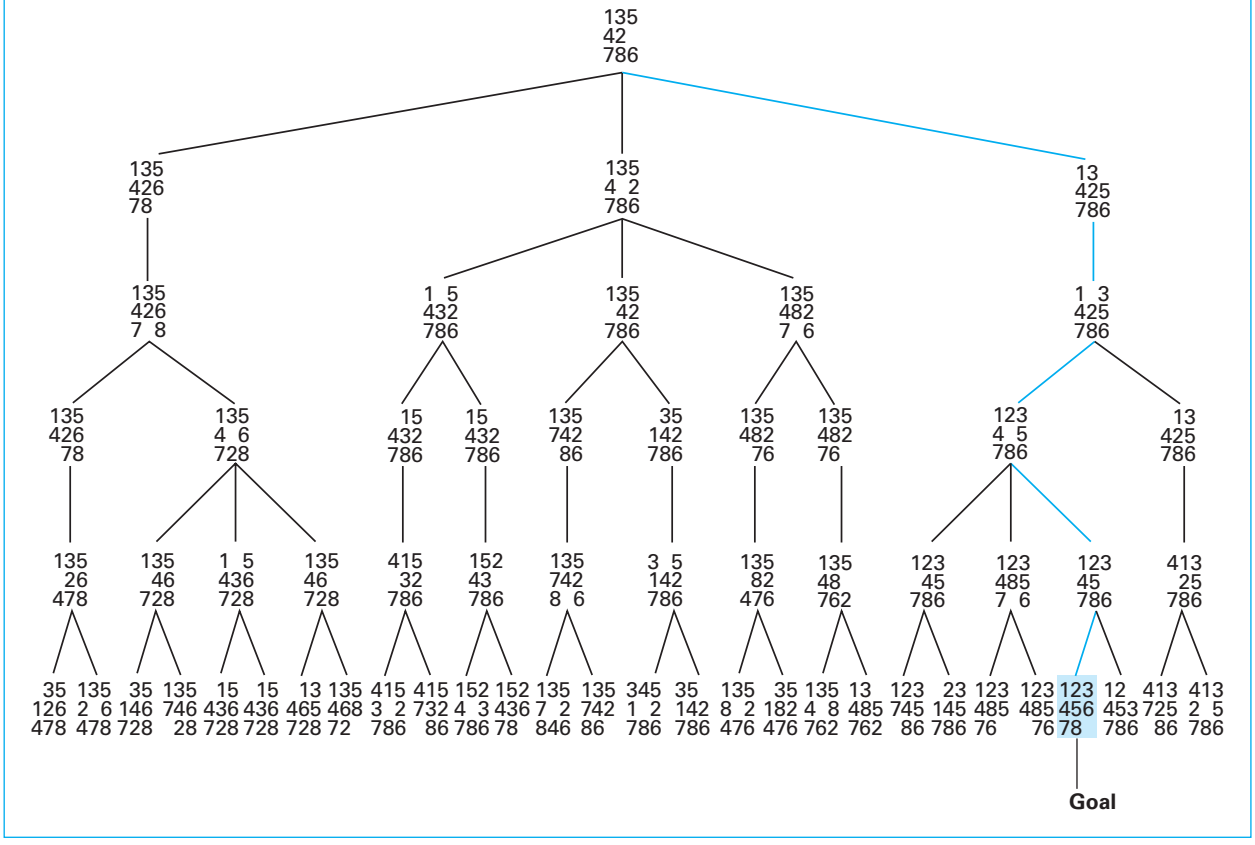


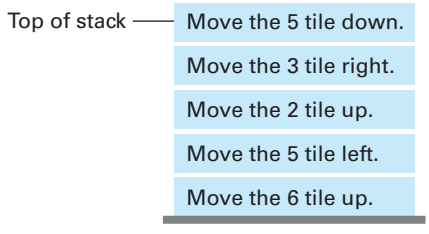
Figure 11.7 A sample search tree



while pushing the productions represented by the tree arcs on a stack as they are encountered. Applying this technique to the search tree in Figure 11.7 produces the stack of productions in Figure 11.8. The control system can now solve the puzzle in the outside world by executing the instructions as they are popped from this stack.

There is one more observation that we should make. Recall that the trees we discussed in Chapter 8 use a pointer system that points *down* the tree, thereby allowing us to move from a parent node to its children. In the case of a search

Figure 11.8 Productions stacked for later execution



tree, however, the control system must be able to move from a child to its parent as it moves *up* the tree from the goal state to the start state. Such trees are constructed with their pointer systems pointing up rather than down. That is, each child node contains a pointer to its parent rather than the parent nodes containing pointers to their children. (In some applications, both sets of pointers are used to allow movement within the tree in both directions).

Heuristics

For our example in Figure 11.7, we chose a starting configuration that produces a manageable search tree. In contrast, the search tree generated in an attempt to solve a more complex problem could grow much larger. In a game of chess, there are twenty possible first moves so the root node of the search tree in such a case would have twenty children rather than the three in the case of our example. Moreover, a game of chess can easily consist of thirty to thirty-five pairs of moves. Even in the case of the eight-puzzle, the search tree can become quite large if the goal is not quickly reached. As a result, developing a full search tree can become as impractical as representing the entire state graph.

One strategy for countering this problem is to change the order in which the search tree is constructed. Rather than building it in a **breadth-first** manner (meaning that the tree is constructed layer by layer), we can pursue the more promising paths to greater depths and consider the other options only if these original choices turn out to be false leads. This results in a **depth-first** construction of the search tree, meaning that the tree is constructed by building vertical paths rather than horizontal layers. More precisely, this approach is often called a **best-first** construction in recognition of the fact that the vertical path chosen for pursuit is the one that appears to offer the best potential.

The best-first approach is similar to the strategy that we as humans would apply when faced with the eight-puzzle. We would rarely pursue several options at the same time, as modeled by the breadth-first approach. Instead, we probably would select the option that appeared most promising and follow it first. Note that we said *appeared* most promising. We rarely know for sure which option is best at a particular point. We merely follow our intuition, which may, of course, lead us astray. Nonetheless, the use of such intuitive information seems to give humans an advantage over the brute-force methods in which each option was given equal attention, and it would therefore seem prudent to apply intuitive methods in automated control systems.

To this end, we need a way of identifying which of several states appears to be the most promising. Our approach is to use a **heuristic,** which in our case is a quantitative value associated with each state that attempts to measure the "distance" from that state to the nearest goal. In a sense, our heuristic is a measure of projected cost. Given a choice between two states, the one with the smaller heuristic value is the one from which a goal can apparently be reached with the least cost. This state, therefore, would represent the direction we should pursue.

A heuristic should have two characteristics. First, it should constitute a reasonable estimate of the amount of work remaining in the solution if the associated state were reached. This means that it can provide meaningful information when selecting among options—the better the estimate provided by the heuristic, the better will be the decisions that are based on the information. Second, the heuristic should be easy to compute. This means that its use has a chance of benefiting the search process rather than of becoming a burden. If computing the

Behavior-Based Intelligence

Early work in artificial intelligence approached the subject in the context of explicitly writing programs to simulate intelligence. However, many argue today that human intelligence is not based on the execution of complex programs but instead by simple stimulus-response functions that have evolved over generations. This theory of "intelligence" is known as behavior-based intelligence because "intelligent" stimulus-response functions appear to be the result of behaviors that caused certain individuals to survive and reproduce while others did not.

Behavior-based intelligence seems to answer several questions in the artificial intelligence community such as why machines based on the von Neumann architecture easily outperform humans in computational skills but struggle to exhibit common sense. Thus behavior-based intelligence promises to be a major influence in artificial intelligence research. As described in the text, behavior-based techniques have been applied in the field of artificial neural networks to teach neurons to behave in desired ways, in the field of genetic algorithms to provide an alternative to the more traditional programming process, and in robotics to improve the performance of machines through reactive strategies.

heuristic is extremely complicated, then we might as well spend our time conducting a breadth-first search.

A simple heuristic in the case of the eight-puzzle would be to estimate the "distance" to the goal by counting the number of tiles that are out of place—the conjecture being that a state in which four tiles are out of place is farther from the goal (and therefore less appealing) than a state in which only two tiles are out of place. However, this heuristic does not take into account how far out of position the tiles are. If the two tiles are far from their proper positions, many productions could be required to move them across the puzzle.

A slightly better heuristic, then, is to measure the distance each tile is from its destination and add these values to obtain a single quantity. A tile immediately adjacent to its final destination would be associated with a distance of one, whereas a tile whose corner touches the square of its final destination would be associated with a distance of two (because it must move at least one position vertically and another position horizontally). This heuristic is easy to compute and produces a rough estimate of the number of moves required to transform the puzzle from its current state to the goal. For instance, the heuristic value associated with the configuration in Figure 11.9 is seven (because tiles 2, 5, and 8 are

Figure 11.9 An unsolved eight-puzzle

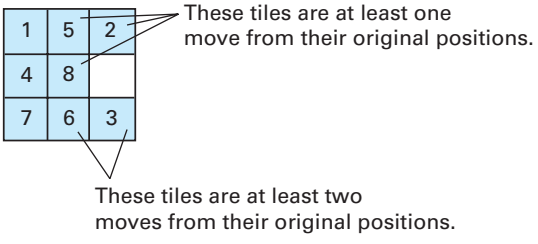


Figure 11.10 An algorithm for a control system using heuristics

```
Establish the start node of the state graph as the root of the
   search tree and record its heuristic value.
while (the goal node has not been reached) do
   [Select the leftmost leaf node with the smallest heuristic
      value of all leaf nodes.
    To this selected node attach as children those nodes that
      can be reached by a single production.
    Record the heuristic of each of these new nodes next
      to the node in the search tree
   ]
Traverse the search tree from the goal node up to the root,
   pushing the production associated with each arc traversed
   onto a stack.
Solve the original problem by executing the productions as they
   are popped off the stack.
```

each a distance of one from their final destinations while tiles 3 and 6 are each a distance of two from home). In fact, it actually takes seven moves to return this puzzle configuration to the solved configuration.

Now that we have a heuristic for the eight-puzzle, the next step is to incorporate it into our decision-making process. Recall that a human faced with a decision tends to select the option that appears closest to the goal. Thus our search procedure should consider the heuristic of each leaf node in the search tree and pursue the search from a leaf node associated with the smallest value. This is the strategy adopted in Figure 11.10, which presents an algorithm for developing a search tree and executing the solution obtained.

Let us apply this algorithm to the eight-puzzle, starting from the initial configuration in Figure 11.6. First, we establish this initial state as the root node and record its heuristic value, which is five. Then, the first pass through the body of the while statement instructs us to add the three nodes that can be reached from the initial state, as shown in Figure 11.11. Note that we have recorded the heuristic value of each leaf node in parentheses beneath the node.

The goal node has not been reached, so we again pass through the body of the while statement, this time extending our search from the leftmost node ("the

Figure 11.11 The beginnings of our heuristic search

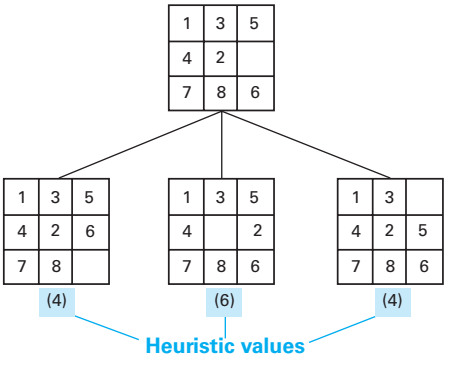
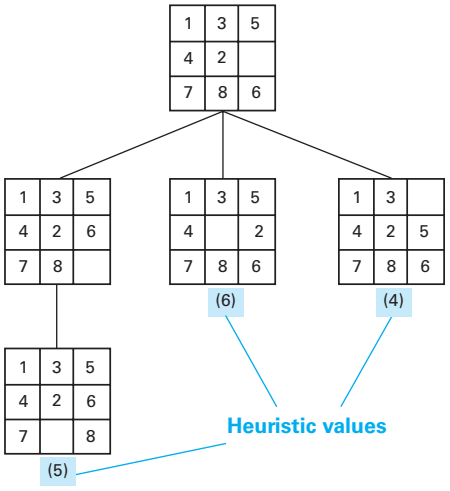


Figure 11.12 The search tree after two passes

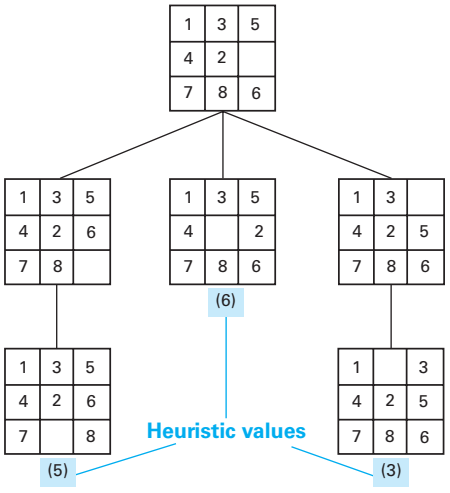


leftmost leaf node with the smallest heuristic value"). After this, the search tree has the form displayed in Figure 11.12.

The heuristic value of the leftmost leaf node is now five, indicating that this branch is perhaps not a good choice to pursue after all. The algorithm picks up on this and in the next pass through the while statement instructs us to expand the tree from the rightmost node (which now is the "leftmost leaf node with the smallest heuristic value"). Having been expanded in this fashion, the search tree appears as in Figure 11.13.

At this point the algorithm seems to be on the right track. Because the heuristic value of this last node is only three, the while statement instructs us to continue

Figure 11.13 The search tree after three passes



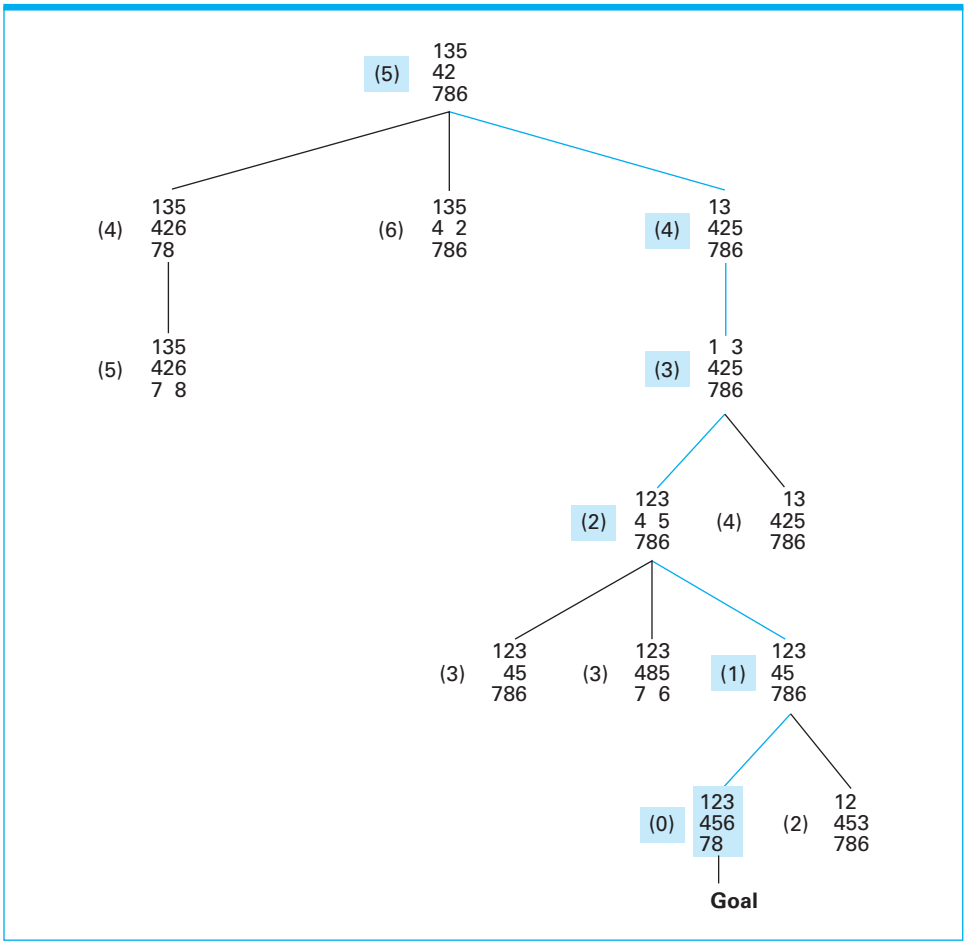
pursuing this path, and the search focuses toward the goal, producing the search tree appearing in Figure 11.14. Comparing this with the tree in Figure 11.7 shows that, even with the temporary wrong turn taken early on by the new algorithm, the use of heuristic information has greatly decreased the size of the search tree and produced a much more efficient process.

After reaching the goal state, the while statement terminates, and we move on to traverse the tree from the goal node up to the root, pushing the productions encountered onto a stack as we go. The resultant stack appears as depicted earlier, in Figure 11.8.

Finally, we are instructed to execute these productions as they are popped from the stack. At this point, we would observe the puzzle-solving machine lower its finger and begin to move the tiles.

One final comment regarding heuristic searching is in order. The algorithm we have proposed in this section, which is often called the best-fit algorithm, is not gauranteed to find be the best solution in all applications. For example, when searching for a path to a city using a Global Positioning System (GPS) in an automobile, one would like to find the shortest path rather than just any path. The **A* algorithm** (pronounced "A star algorithm") is a modified version of our best-fit

Figure 11.14 The complete search tree formed by our heuristic system



algorithm that finds an optimal solution. The major difference between the two algorithms is that, in addition to a hueristic value, the A* algorithm takes into account the "accumulated cost" incurred to reach each leaf node when selecting the next node to expand. (In the case of an automobile's GPS, this cost is the distance traveled that the GPS obtains from its internal database.) Thus, the A* algorithm bases its decisions on estimates of the cost of complete potential paths rather than merely on projections of remaining costs.

Questions & Exercises

1. What is the significance of production systems in artificial intelligence?
2. Draw a portion of the state graph for the eight-puzzle surrounding the node representing the following state:

4	1	3
	2	6
7	5	8

3. Using a breadth-first approach, draw the search tree that is constructed by a control system when solving the eight-puzzle from the following start state:

1	2	3
4	8	5
7	6	

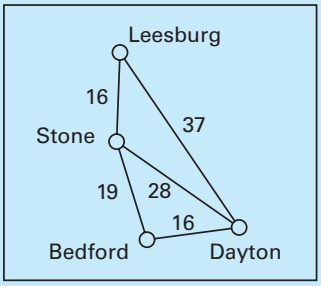
4. Use pencil, paper, and the breadth-first approach to try to construct the search tree that is produced in solving the eight-puzzle from the following start state. (You do not have to finish.) What problems do you encounter?

4	3	
2	1	8
7	6	5

5. What analogy can be drawn between our heuristic system for solving the eight-puzzle and a mountain climber who attempts to reach the peak by considering only the local terrain and always proceeding in the direction of steepest ascent?
6. Using the heuristic presented in this section, apply the best-fit algorithm of Figure 11.10 to the problem of solving the following eight-puzzle:

1	2	3
4		8
7	6	5

7. Refine our method of computing the heuristic value for a state of the eight-puzzle so that the search algorithm of Figure 11.10 does not make the wrong choice, as it did in the example in this section. Can you find an example in which your heuristic still causes the search to go astray?
8. Draw to the search tree produced by the best-fit algorithm (Figure 11.10) in finding the route from Leesburg to Bedford. Each node in the search tree will be a city on the map. Begin with a node for Leesburg. When expanding a node, add only the cities that are directly connected to the city being expanded. Record in each node the straight-line distance to Bedford and use this as the heuristic value. What is the solution found by the best-fit algorithm? Is the found solution the shortest route?



Straight-line distance to Bedford from

Dayton	16
Leesburg	34
Stone	19

9. The A* algorithm modifies the best-fist algorithm in two significant ways. First, it records the actual cost to reach a state. In the case of a route on a map, the actual cost is the distance traveled. Second, when selecting a node to expand, it chooses the node whose sum of the actual cost plus heuristic value is the smallest. Draw the search tree of Question 8 that would result from these two modifications. Record in each node the distance traveled to the city, the heuristic value to reach the goal, and their sum. What is the found solution? Is the found solution the shortest route?

11.4 Additional Areas of Research

In this section we explore issues of handling knowledge, learning, and dealing with very complex problems, which continue to challenge researchers in the field of artificial intelligence. These activities involve capabilities that appear to be easy for human minds but apparently tax the capabilities of machines. For now, much of the progress in developing "intelligent" agents has been achieved essentially by avoiding direct confrontation with these issues—perhaps by applying clever shortcuts or limiting the scope in which a problem arises.

Representing and Manipulating Knowledge

In our discussion of perception we saw that understanding images requires a significant amount of knowledge about the items in the image and that the meaning of a sentence might depend on its context. These are examples of the role played by the warehouse of knowledge, often called **real-world knowledge,** maintained

by human minds. Somehow, humans store massive amounts of information and draw from that information with remarkable efficiency. Giving machines this capability is a major challenge in artificial intelligence.

The underlying goal is to find ways to represent and store knowledge. This is complicated by the fact that, as we have already seen, knowledge occurs in both declarative and procedural forms. Thus, representing knowledge is not merely the representation of facts, but instead encompasses a much broader spectrum. Whether a single scheme for representing all forms of knowledge will ultimately be found is therefore questionable.

The problem, however, is not just to represent and store knowledge. The knowledge must also be readily accessible, and achieving this accessibility is a challenge. Semantic nets, as introduced in Section 11.2, are often used as a means of knowledge representation and storage, but extracting information from them can be problematic. For example, the significance of the statement "Mary hit John" depends on the relative ages of Mary and John. (Are the ages 2 and 30 or vice versa?) This information would be stored in the complete semantic net suggested by Figure 11.3, but extracting such information during contextual analysis could require a significant amount of searching through the net.

Yet another problem dealing with accessing knowledge is identifying knowledge that is implicitly, instead of explicitly, related to the task at hand. Rather than answering the question "Did Arthur win the race?" with a blunt "No," we want a system that might answer with "No, he came down with the flu and was not able to compete." In the next section we will explore the concept of associative memory, which is one area of research that is attempting to solve this related information problem. However, the task is not merely to retrieve related information. We need systems that can distinguish between related information and relevant information. For example, an answer such as "No, he was born in January and his sister's name is Lisa" would not be considered a worthy response to the previous question, even though the information reported is in some way related.

Another approach to developing better knowledge extraction systems has been to insert various forms of reasoning into the extraction process, resulting in what is called **meta-reasoning**—meaning reasoning about reasoning. An example, originally used in the context of database searches, is to apply the **closed-world assumption,** which is the assumption that a statement is false unless it can be explicitly derived from the information available. For example, it is the closed-world assumption that allows a database to conclude that Nicole Smith does not subscribe to a particular magazine even though the database does not contain any information at all about Nicole. The process is to observe that Nicole Smith is not on the subscription list and then apply the closed-world assumption to conclude that Nicole Smith does not subscribe.

On the surface the closed-world assumption appears trivial, but it has consequences that demonstrate how apparently innocent meta-reasoning techniques can have subtle, undesirable effects. Suppose, for example, that the only knowledge we have is the single statement

Mickey is a mouse OR Donald is a duck.

From this statement alone we cannot conclude that Mickey is in fact a mouse. Thus the closed-world assumption forces us to conclude that the statement

Mickey is a mouse.

is false. In a similar manner, the closed-world assumption forces us to conclude that the statement

Donald is a duck.

is false. Thus, the closed-world assumption has led us to the contradictory conclusion that although at least one of the statements must be true, both are false. Understanding the consequences of such innocent-looking meta-reasoning techniques is a goal of research in the fields of both artificial intelligence and database, and it also underlines the complexities involved in the development of intelligent systems.

Finally, there is the problem, known as the **frame problem,** of keeping stored knowledge up to date in a changing environment. If an intelligent agent is going to use its knowledge to determine its behavior, then that knowledge must be current. But the amount of knowledge required to support intelligent behavior can be enormous, and maintaining that knowledge in a changing environment can be a massive undertaking. A complicating factor is that changes in an environment often alter other items of information indirectly and accounting for such indirect consequences is difficult. For example, if a flower vase is knocked over and broken, your knowledge of the situation no longer contains the fact that water is in the vase, even though spilling the water was only indirectly involved with breaking the vase. Thus, to solve the frame problem not only requires the ability to store and retrieve massive amounts of information in an efficient manner, but it also demands that the storage system properly react to indirect consequences.

Learning

In addition to representing and manipulating knowledge, we would like to give intelligent agents the ability to acquire new knowledge. We can always "teach" a computer-based agent by writing and installing a new program or explicitly adding to its stored data, but we would like intelligent agents to be able to learn on their own. We want agents to adapt to changing environments and to perform tasks for which we cannot easily write programs in advance. A robot designed for household chores will be faced with new furniture, new appliances, new pets, and even new owners. An autonomous, self-driving car must adapt to variations in the boundary lines on roads. Game playing agents should be able to develop and apply new strategies.

One way of classifying approaches to computer learning is by the level of human intervention required. At the first level is learning by **imitation,** in which a person directly demonstrates the steps in a task (perhaps by carrying out a sequence of computer operations or by physically moving a robot through a sequence of motions) and the computer simply records the steps. This form of learning has been used for years in application programs such as spreadsheets and word processors, where frequently occurring sequences of commands are recorded and later replayed by a single request. Note that learning by imitation places little responsibility on the agent.

At the next level is learning by **supervised training.** In supervised training a person identifies the correct response for a series of examples and then the agent generalizes from those examples to develop an algorithm that applies to new cases. The series of examples is called the **training set.** Typical applications of supervised training include learning to recognize a person's handwriting or voice, learning to distinguish between junk and welcome email, and learning how to identify a disease from a set of symptoms.

A third level is learning by **reinforcement.** In learning by reinforcement, the agent is given a general rule to judge for itself when it has succeeded or failed at a task during trial and error. Learning by reinforcement is good for learning how to play a game like chess or checkers, as success or failure is easy to define. In contrast to supervised training, learning by reinforcement allows the agent to act autonomously as it learns to improve its behavior over time.

Learning remains a challenging field of research since no general, universal principle has been found that covers all possible learning activities. However, there are numerous examples of progress. One is ALVINN (Autonomous Land Vehicle in a Neural Net), a system developed at Carnegie Mellon University to learn to steer a van with an on-board computer using a video camera for input. The approach used was supervised training. ALVINN collected data from a human driver and used that data to adjust its own steering decisions. As it learned, it would predict where to steer, check its prediction against the human driver's data, and then modify its parameters to come closer to the human's steering choice. ALVINN succeeded well enough that it could steer the van at seventy miles an hour, leading to additional research that has produced control systems that have successfully driven at highway speeds in traffic.

Finally, we should recognize a phenomenon that is closely related to learning: discovery. The distinction is that learning is "target based" whereas discovery is not. The term *discovery* has a connotation of the unexpected that is not present in learning. We might set out to learn a foreign language or how to drive a car, but we might discover that those tasks are more difficult than we expected. An explorer might discover a large lake, whereas the goal was merely to learn what was there.

Developing agents with the ability to discover efficiently requires that the agent be able to identify potentially fruitful "trains of thought." Here, discovery relies heavily on the ability to reason and the use of heuristics. Moreover, many potential applications of discovery require that an agent be able to distinguish meaningful results from insignificant ones. A data mining agent, for example, should not report every trivial relationship it finds.

Knowledge in Logic Programming

An important concern in representing and storing knowledge is that it be done in a way that is compatible with the system that must access the knowledge. It is in this context that logic programming (see Section 6.7) often proves beneficial. In such systems knowledge is represented by "logic" statements such as

Dumbo is an elephant.

and

X is an elephant implies X is gray.

Such statements can be represented using notational systems that are readily accessible to the application of inference rules. In turn, sequences of deductive reasoning, such as we saw in Figure 11.5, can be implemented in a straightforward manner. Thus, in logic programming the representation and storage of knowledge are well integrated with the knowledge extraction and application process. One might say that logic programming systems provide a "seamless" boundary between stored knowledge and its application.

Examples of success in computer discovery systems include Bacon, named after the philosopher Sir Francis Bacon, that has discovered (or maybe we should say "rediscovered") Ohm's law of electricity, Kepler's third law of planetary motion, and the conservation of momentum. Perhaps more persuasive is the system AUTOCLASS that, using infrared spectral data, has discovered new classes of stars that were previously unknown in astronomy—a true scientific discovery by a computer.

Genetic Algorithms

The A* algorithm (introduced in the previous section) will find the optimal solution to many search problems; however there are some problems that are too complex to be solved (execution exceeds available memory or cannot be completed within a reasonable amount of time) by such seach techniques. For these problems, a solution can sometimes be discovered through an evolutionary process involving many generations of trial solutions. This strategy is the foundation for what is called **genetic algorithms.** In essence, genetic algorithms will discover a solution by random behavior combined with a simulation of reproductive theory and the evolutionary process of natural selection.

A genetic algorithm begins by generating a random pool of trial solutions. Each solution is just a guess. (In the case of the eight-puzzle, a trial solution can be a random sequence of tile movements.) Each trial solution is called a **chromosome** and each component of the chromosome is a called a **gene** (a single tile movement in the case of the eight-puzzle).

Since each initial chromosome is a random guess, it is very unlikely that it will represent a solution to the problem at hand. Thus, the genetic algorithm proceeds to generate a new pool of chromosomes whereby each chromosome is an offspring (child) of two chromosomes (parents) of the previous pool. The parents are randomly selected from the pool giving a probabilistic preference to those chromosomes that appear to provide the best chance of leading to a solution, thereby emulating the evolutionary principle of survival of the fittest. (Determining which chromosomes are the best candidates for parenthood is perhaps the most problematic step in the generic algrithm process.) Each offspring is a random combination of genes from the parents. In addition, a resulting offspring may occasionally be mutated in some random way (switch two moves). Hopefully, by repeating this process over and over, better and better trial solutions will evolve until a very good one, if not the best, is discovered. Unfortunately, there is no assurance that the genetic algorithm will ultimately find a solution, yet research has demonstrated that genetic algorithms can be effective in solving a surprisingly wide range of complex problems.

When applied to the task of program development, the genetic algorithm approach is known as **evolutionary programming.** Here the goal is to develop programs by allowing them to evolve rather than by explicitly writing them. Researchers have applied evolutionary programming techniques to the program-development process using functional programming languages. The approach has been to start with a collection of programs that contain a rich variety of functions. The functions in this starting collection form the "gene pool" from which future generations of programs will be constructed. One then allows the evolutionary process to run for many generations, hoping that by producing each generation from the best performers in the previous generation, a solution to the target problem will evolve.

Questions & Exercises

1. What is meant by the term *real-world knowledge,* and what is its significance in artificial intelligence?
2. A database about magazine subscribers typically contains a list of subscribers to each magazine but does not contain a list of those who do not subscribe. How, then, does such a database determine that a person does not subscribe to a particular magazine?
3. Summarize the frame problem.
4. Identify three ways of training a computer. Which one does not involve direct human intervention?
5. How do evolutionary techniques differ from more traditional problem-solving techniques?

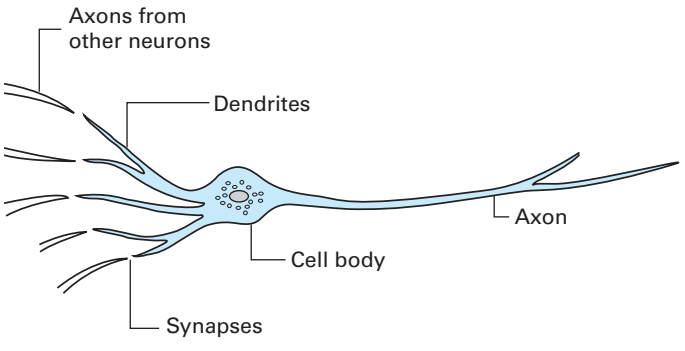
11.5 Artificial Neural Networks

With all the progress that has been made in artificial intelligence, many problems in the field continue to tax the abilities of computers using traditional algorithmic approaches. Sequences of instructions do not seem capable of perceiving and reasoning at levels comparable to those of the human mind. For this reason, many researchers are turning to approaches that leverage phenomena observed in nature. One such approach is genetic algorithms presented in the previous section. Another approach is the artificial neural network.

Basic Properties

Artificial neural networks provide a computer processing model that mimics networks of neurons in living biological systems. A biological neuron is a single cell with input tentacles called dendrites and an output tentacle called the axon (Figure 11.15). The signals transmitted via a cell's axon reflect whether the cell is in an inhibited or excited state. This state is determined by the combination of signals received by the cell's dendrites. These dendrites pick up signals from the axons of other cells across small gaps known as synapses. Research suggests that

Figure 11.15 A neuron in a living biological system



the conductivity across a single synapse is controlled by the chemical composition of the synapse. That is, whether the particular input signal will have an exciting or inhibiting effect on the neuron is determined by the chemical composition of the synapse. Thus it is believed that a biological neural network learns by adjusting these chemical connections between neurons.

A neuron in an artificial neural network is a software unit that mimics this basic understanding of a biological neuron. It produces an output of 1 or 0, depending on whether its effective input exceeds a given value, which is called the neuron's **threshold** value. This effective input is a weighted sum of the actual inputs, as represented in Figure 11.16. In this figure, a neuron is represented with an oval and connections between neurons are represented with arrows. The values obtained from the axons of other neurons (denoted by v_1, v_2, and v_3) are used as inputs to the depicted neuron. In addition to these values, each connection is associated with a **weight** (denoted by w_1, w_2, and w_3). The neuron receiving these input values multiplies each by the associated weight for the connection and then adds these products to form the effective input ($v_1w_1 + v_2w_2 + v_3w_3$). If this sum exceeds the neuron's threshold value, the neuron produces an output of 1 (simulating an excited state); otherwise the neuron produces a 0 as its output (simulating an inhibited state).

Following the lead of Figure 11.16, we adopt the convention of representing neurons as circles. Where each input connects to a neuron, we record the weight associated with that input. Finally, we write the neuron's threshold value in the middle of the circle. As an example, Figure 11.17 represents a neuron with a threshold value of 1.5 and weights of -2, 3, and -1 associated with each of its input connections. Therefore if the neuron receives the inputs 1, 1, and 0, its effective input is $(1)(-2) + (1)(3) + (0)(-1) = 1$, and thus its output is 0. But, if the neuron receives 0, 1, and 1, its effective input is $(0)(-2) + (1)(3) + (1)(-1) = 2$, which exceeds the threshold value. The neuron's output will thus be 1.

The fact that a weight can be positive or negative means that the corresponding input can have either an inhibiting or exciting effect on the receiving neuron. (If the weight is negative, then a 1 at that input position reduces the weighted sum and thus tends to hold the effective input below the threshold value. In contrast, a positive weight causes the associated input to have an increasing effect on the weighted sum and thus increase the chances of that sum exceeding the threshold value.) Moreover, the actual size of the weight controls the degree to which the corresponding input is allowed to inhibit or excite the receiving neuron. Consequently, by adjusting the values of the weights throughout an artificial neural network, we can program the network to respond to different inputs in a predetermined manner.

Figure 11.16 The activities within a neuron

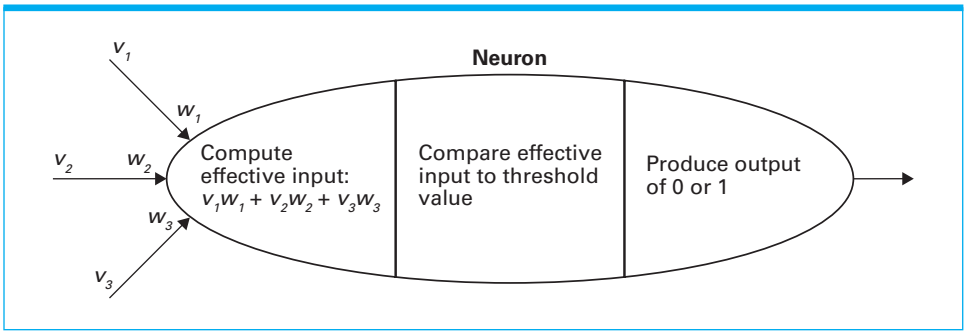
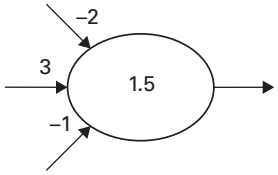


Figure 11.17 Representation of a neuron



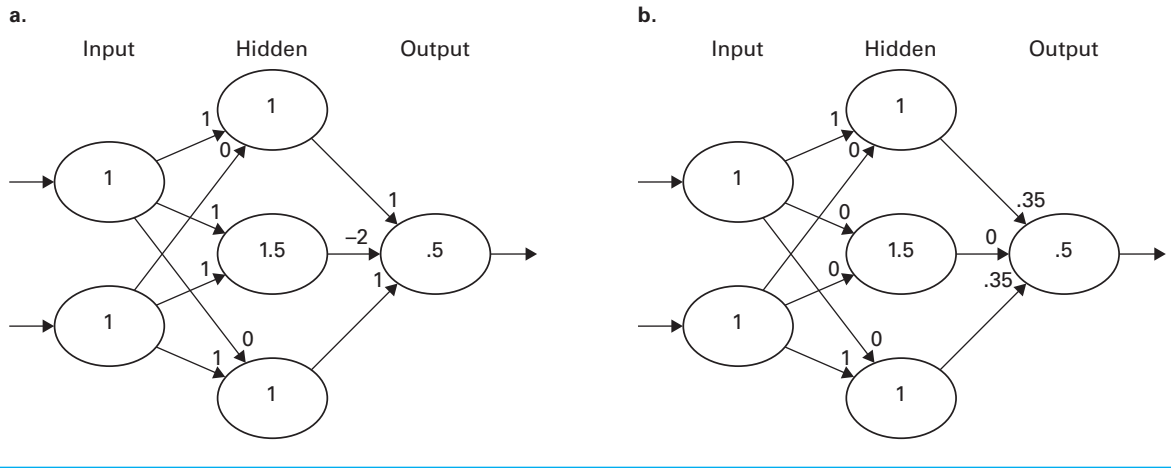
Artificial neural networks are typically arranged in a topology of several layers. The input neurons are in the first layer and the output neurons are in the last. Additional layers of neurons (called hidden layers) may be included between the input and output layers. Each neuron of one layer is interconnected with every neuron in the subsequent layer. As an example, the simple network presented in Figure 11.18a is programmed to produce an output of 1 if its two inputs differ and an output of 0 otherwise. If, however, we change the weights to those shown in Figure 11.18b, we obtain a network that responds with a 1 if both of its inputs are 1s and with a 0 otherwise.

We should note that the network configuration in Figure 11.18 is far more simplistic than an actual biological network. A human brain contains approximately 10^{11} neurons with about 10^4 synapses per neuron. Indeed, the dendrites of a biological neuron are so numerous that they appear more like a fibrous mesh than the individual tentacles represented in Figure 11.15.

Training Artificial Neural Networks

An important feature of artificial neural networks is that they are not programmed in the traditional sense but instead are trained. That is, a programmer does not determine the values of the weights needed to solve a particular problem and then "plug" those values into the network. Instead, an artificial neural

Figure 11.18 A neural network with two different programs



network learns the proper weight values via supervised training (Section 11.4) involving a repetitive process in which inputs from the training set are applied to the network and then the weights are adjusted by small increments so that the network's performance approaches the desired behavior.

It is interesting to note how genetic algorithm techniques have been applied to the task of training artificial neural networks. In particular, to train a neural network, a number of sets of weights for the network can be randomly generated—each set of which will serve as a chromosome for the genetic algorithm. Then, in a step-by-step process, the network can be assigned the weights represented by each chromosone and tested over a variety of inputs. The chromosones producing fewest errors during this testing process can then be given a greater probabilty of being selected as parents for the next generation. In numerous experiments this appoarch has ultimately led to a successful set of weights.

Let us consider an example in which training an artificial neural network to solve a problem has been successful and perhaps more productive than trying to provide a solution by means of traditional programming techniques. The problem is one that might be faced by a robot when trying to understand its environment via the information it receives from its video camera. Suppose, for example, that the robot must distinguish between the walls of a room, which are white, and the floor, which is black. At first glance, this would appear to be an easy task: Simply classify the white pixels as part of a wall and the black pixels at part of the floor. However, as the robot looks in different directions or moves around in the room, various lighting conditions can cause the wall to appear gray in some cases whereas in other cases the floor may appear gray. Thus, the robot needs to learn to distinguish between walls and floor under a wide variety of lighting conditions.

To accomplish this, we could build an artificial neural network whose inputs consist of values indicating the color characteristics of an individual pixel in the image as well as a value indicating the overall brightness of the entire image. We could then train the network by providing it with numerous examples of pixels representing parts of walls and floors under various lighting conditions.

The results of training an artificial neural network using these techniques are represented in Figure 11.19. The first column represents the original images; the next depicts the robot's interpretation. Note that although the walls in the top original are rather dark, the robot has correctly identified most of the associated pixels as white wall pixels, yet the floor in the lower image has still been correctly identifed. (The ball in the images was part of a more extensive experiment.) You will also notice that the robot's image processing system is not perfect. The neural network has mistakenly identified some of the wall pixels as floor pixels (and some of the floor pixels as wall pixels). These are examples of realities that often must be accommodated in the application of a theory. In this case, the errors can be corrected by programming the robot to ignore individual floor pixels that appear among a multitude of wall pixels (and vice versa).

Beyond simple learning problems (such as the classification of pixels), artificial neural networks have been used to learn sophisticated intelligent behavior, as testified by the ALVINN project cited in the previous section. Indeed, ALVINN was an artificial neural network whose composition was surprisingly simple (Figure 11.20). Its input was obtained from a 30 by 32 array of sensors,

each of which observed a unique portion of the video image of the road ahead and reported its findings to each of four neurons on a hidden layer. (Thus, each of these four neurons had 960 inputs.) The output of each of these four neurons was connected to each of thirty output neurons, whose outputs indicated the direction to steer. Excited neurons at one end of the thirty neuron row indicated a sharp turn to the left, while excited neurons at the other end indicated a sharp turn to the right.

ALVINN was trained by "watching" a human drive while it made its own steering decisions, comparing its decisions to those of the human, and making slight modifications to its weights to bring its decisions closer to those of the human. There was, however, an interesting side issue. Although ALVINN learned to steer following this simple technique, ALVINN did not learn how to recover from mistakes. Thus, the data collected from the human was artificially enriched to include recovery situations as well. (One approach to this recovery training that was initially considered was to have the human swerve the vehicle so that ALVINN could watch the human recover and thus learn how to recover on its own. But unless ALVINN was disabled while the human performed the initial swerve procedure, ALVINN learned to swerve as well as to recover—an obviously undesirable trait.)

Associative Memory

The human mind has the amazing ability to retrieve information that is associated with a current topic of consideration. When we experience certain smells, we might readily recall memories of our childhood. The sound of a friend's voice might conjure an image of the person or perhaps memories of good times. Certain music might generate thoughts of particular holiday seasons. These are examples of **associative memory**—the retrieval of information that is associated with, or related to, the information at hand.

To construct machines with associative memory has been a goal of research for many years. One approach is to apply techniques of artificial neural networks. For instance, consider a network consisting of many neurons that are interconnected to form a web with no inputs or outputs. (In some designs, called Hopfield networks, the output of each neuron is connected as inputs to each of the other neurons; in other cases the output of a neuron may be connected only to its immediate neighbors.) In such a system, the excited neurons will tend to excite other neurons, whereas the inhibited neurons will tend to inhibit others. In turn, the entire system may be in a constant state of change, or it may be that the system will find its way to a stable configuration where the excited neurons remain excited and the inhibited neurons remain inhibited. If we start the network in a nonstable configuration that is close to a stable one, we would expect it to wander to that stable configuration. In a sense, when given a part of a stable configuration, the network might be able to complete the configuration.

Now suppose that we represent an excited state by 1 and an inhibited state by 0 so that the condition of the entire network at any time can be envisioned as a configuration of 0s and 1s. Then, if we set the network to a bit pattern that is close to a stable pattern, we could expect the network to shift to the stable pattern. In other words, the network might find the stable bit pattern that is close to

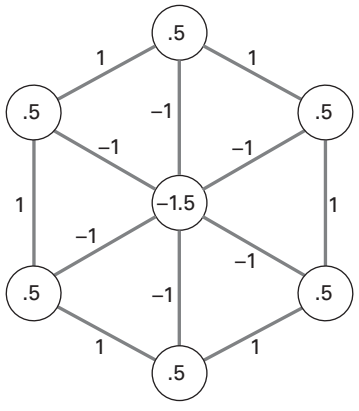
the pattern it was given. Thus if some of the bits are used to encode smells and others are used to encode childhood memories, then initializing the smell bits according to a certain stable configuration could cause the remaining bits to find their way to the associated childhood memory.

Now consider the artificial neural network shown in Figure 11.21. Following the conventions used to depict artificial neural networks, each circle in the figure represents a neuron whose threshold value is recorded inside the circle. Instead of arrows, the lines connecting the circles represent two-way connections between the corresponding neurons. That is, a line connecting two neurons indicates that the output of each neuron is connected as an input to the other. Thus the output of the center neuron is connected as an input to each of the neurons around the perimeter, and the output of each of the neurons around the perimeter is connected as an input to the center neuron as well as an input to each of its immediate neighbors on the perimeter. Two connected neurons associate the same weight with each other's output. This common weight is recorded next to the line connecting the neurons. Thus the neuron at the top of the diagram associates a weight of -1 with the input it receives from the center neuron and a weight of 1 with the inputs it receives from its two neighbors on the perimeter. Likewise, the center neuron associates a weight of -1 with each of the values it receives from the neurons around the perimeter.

The network operates in discrete steps in which all neurons respond to their inputs in a synchronized manner. To determine the next configuration of the network from its current configuration, we determine the effective inputs of each neuron throughout the network and then allow all the neurons to respond to their inputs at the same time. The effect is that the entire network follows a coordinated sequence of compute effective inputs, respond to inputs, compute effective inputs, respond to inputs, etc.

Consider the sequence of events that would occur if we initialized the network with its two rightmost neurons inhibited and the other neurons excited

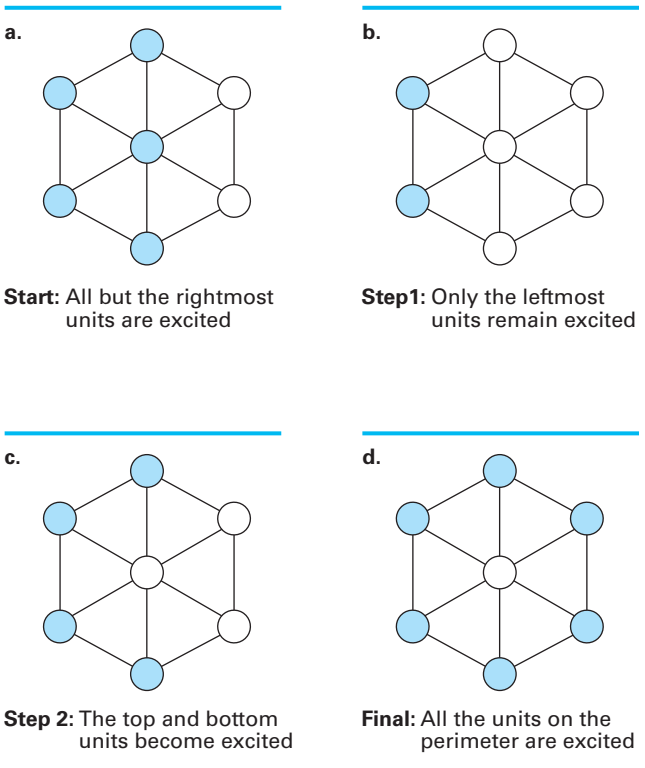
Figure 11.21 An artificial neural network implementing an associative memory



(Figure 11.22a). The two leftmost neurons would have effective inputs of 1, so they would remain excited. But, their neighbors on the perimeter would have effective inputs of 0, so they would become inhibited. Likewise, the center neuron would have an effective input of −4, so it would become inhibited. Thus the entire network would shift to the configuration shown in Figure 11.22b in which only the two leftmost neurons are excited. Since the center neuron would now be inhibited, the excited conditions of the leftmost neurons would cause the top and bottom neurons to become excited again. Meanwhile, the center neuron would remain inhibited since it would have an effective input of −2. Thus the network would shift to the configuration in Figure 11.22c, which would then lead to the configuration in Figure 11.22d. (You might wish to confirm that a blinking phenomenon would occur if the network were initialized with only the upper four neurons excited. The top neuron would remain excited while its two neighbors on the perimeter and the center neuron would alternate between being excited and inhibited.)

Finally, observe that the network has two stable configurations: one in which the center neuron is excited and the others are inhibited, and another configuration in which the center neuron is inhibited and the others are excited. If we initialize the network with the center neuron excited and no more than two of the other neurons excited, the network will wander to the former stable configuration. If we initialize the network with at least four adjacent neurons on the perimeter in their excited states, the network will wander to the latter configuration. Thus we

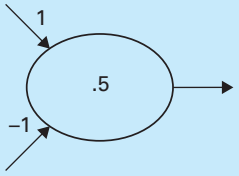
Figure 11.22 The steps leading to a stable configuration



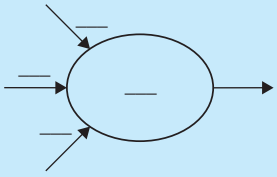
could say that the network associates the former stable configuration with initial patterns in which its center neuron and fewer than three of its perimeter neurons are excited, and associates the latter stable configuration with initial patterns in which four or more of its perimeter neurons are excited. In short, the network represents an elementary associative memory.

Questions & Exercises

1. What is the output of the following neuron when both its inputs are 1s? What about the input patterns 0, 0; 0, 1; and 1, 0?



2. Adjust the weights and threshold value of the following neuron so that its output is 1 if and only if at least two of its inputs are 1s.



3. Identify a problem that might occur in training an artificial neural network.
4. To which stable configuration will the network in Figure 11.22 wander if it is initialized with all its neurons inhibited?

11.6 Robotics

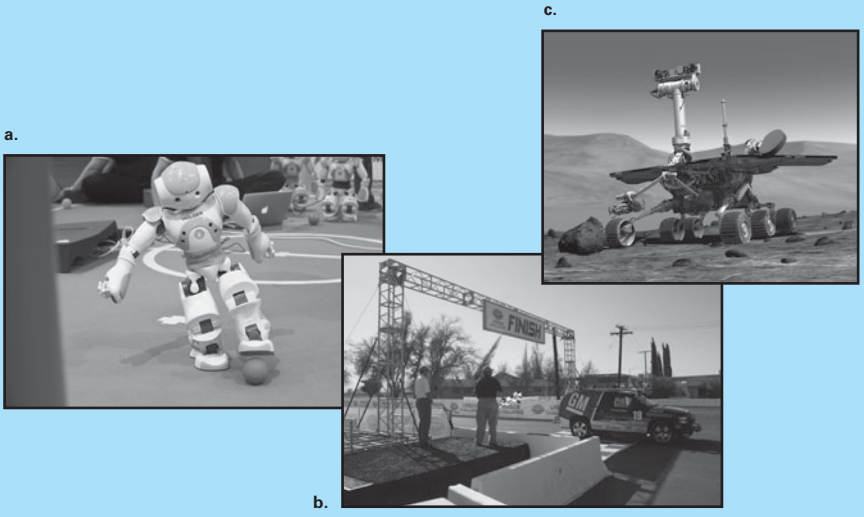
Robotics is the study of physical, autonomous agents that behave intelligently. As with all agents, robots must be able to perceive, reason, and act in their environment. Research in robotics thereby encompasses all areas of artificial intelligence as well as drawing heavily from mechanical and electrical engineering.

To interact with the world, robots need mechanisms to manipulate objects and to move about. In the early days of robotics, the field was closely allied with the development of manipulators, most often mechanical arms with elbows, wrists, and hands or tools. Research dealt not only with how such devices could be maneuvered but also with how knowledge of their location and orientation could be maintained and applied. (You are able to close your eyes and still touch your nose with your finger because your brain maintains a record of where your nose and finger are.) Over time robots' arms have become more dexterous to where, with a sense of touch based on force feedback, they can handle eggs and paper cups successfully.

Recently, the development of faster, lighter weight computers has lead to greater research in mobile robots that can move about. Achieving this mobility

Robots Making History

a. A soccer robot kicks a ball during the RoboCup German Open 2010 on April 15, 2010 in Magdeburg, eastern Germany (© Jens Schlueter/SFP/gettyimages/Newscom). **b.** Tartan Racing's "Boss"—winner of the Urban Challenge, a contest sponsored by DARPA to have vehicles drive themselves through an urban environment (© DARPA). **c.** One of NASA's Rovers—a robot geologist exploring the surface of Mars (Courtesy NASA/JPL-Caltech).



has led to an abundance of creative designs. Researchers in robot locomotion have developed robots that swim like fish, fly like dragonflies, hop like grasshoppers, and crawl like snakes.

Wheeled robots are very popular since they are relatively easy to design and build, but they are limited in the type of terrain they can traverse. Overcoming this restriction, using combinations of wheels or tracks to climb stairs or roll over rocks, is the goal of current research. As an example, the NASA Mars rovers used specially designed wheels to move on rocky soil.

Legged robots offer greater mobility but are significantly more complex. For instance, two-legged robots, designed to walk as humans, must constantly monitor and adjust their stance or they will fall. However, such difficulties can be overcome, as exemplified by the two-legged humanoid robot named Asimo, developed by Honda, that can walk up stairs and even run.

Despite great advances in manipulators and locomotion, most robots are still not very autonomous. Industrial robot arms are typically rigidly programmed for each task and work without sensors, assuming parts will be given to them in exact positions. Other mobile robots such as the NASA Mars rovers and military unmanned aerial vehicles (UAVs) rely on human operators for their intelligence.

Overcoming this dependency on humans is a major goal of current research. One question deals with what an autonomous robot needs to know about its environment and to what degree it needs to plan its actions in advance. One

approach is to build robots that maintain detailed records of their environments, containing an inventory of objects and their locations with which they develop precise plans of action. Research in this direction depends heavily on progress in knowledge representation and storage as well as improved reasoning and plan-development techniques.

An alternative approach is to develop reactive robots that, rather than maintaining complex records and expending great efforts in constructing detailed plans of action, merely apply simple rules for interacting with the world to guide their behavior moment by moment. Proponents of reactive robotics argue that when planning a long trip by car, humans do not make all-encompassing, detailed plans in advance. Instead, they merely select the major roads, leaving such details as where to eat, what exits to take, and how to handle detours for later consideration. Likewise, a reactive robot that needs to navigate a crowded hallway or to go from one building to another does not develop a highly detailed plan in advance, but instead applies simple rules to avoid each obstacle as it is encountered. This is the approach taken by the best-selling robot in history, the iRobot Roomba vacuum cleaner, which moves about a floor in a reactive mode without bothering to remember the details of furniture and other obstacles. After all, the family pet will probably not be in the same place next time.

Of course, no single approach will likely prove the best for all situations. Truly autonomous robots will most likely use multiple levels of reasoning and planning, applying high-level techniques to set and achieve major goals and lower-level reactive systems to achieve minor sub-goals. An example of such multilevel reasoning is found in the Robocup competition—an international competition of robot soccer teams—that serves as a forum for research toward developing a team of robots that can beat world-class human soccer teams by the year 2050. Here the emphasis is not just to build mobile robots that can "kick" a ball but to design a team of robots that cooperate with each other to obtain a common goal. These robots not only have to move and to reason about their actions, but they have to reason about the actions of their teammates and their opponents.

Another example of research in robotics is the field known as evolutionary robotics in which theories of evolution are applied to develop schemes for both low-level reactive rules and high-level reasoning. Here we find the survival-of-the-fittest theory being used to develop devices that over multiple generations acquire their own means of balance or mobility. Much of the research in this area distinguishes between a robot's internal control system (largely software) and the physical structure of its body. For example, the control system for a swimming tadpole robot was transferred to a similar robot with legs. Then evolutionary techniques were applied within the control system to obtain a robot that crawled. In other instances, evolutionary techniques have been applied to a robot's physical body to discover positions for sensors that are optimal for performing a particular task. More challenging research seeks ways to evolve software control systems simultaneously with physical body structures.

To list all the impressive results from research in robotics would be an overwhelming task. Our current robots are far from the powerful robots in fictional movies and novels, but they have achieved impressive successes on specific tasks. We have robots that can drive in traffic, behave like pet dogs, and guide weapons to their targets. However, while relishing in these successes, we should note that the affection we feel for an artificial pet dog and the awesome power of smart weapons raise social and ethical questions that challenge society. Our future is what we make it.

Questions & Exercises

1. In what way does the reactive approach to robot behavior differ from the more traditional "plan-based" behavior?
2. What are some current topics of research in the field of robotics?
3. What are two levels at which evolutionary theories are being applied to robot development?

11.7 Considering the Consequences

Without a doubt, advances being made in artificial intelligence have the potential of benefiting humankind, and it is easy to become caught up in the enthusiasm generated by the potential benefits. However, there are also potential perils lurking in the future whose ramifications could be as devastating as their counterparts are beneficial. The distinction is often merely one's point of view or perhaps one's position in society—one person's gain might be another's loss. It is fitting then that we take a moment to look at advancing technology from alternative perspectives.

Some view the advancement of technology as a gift to humanity—a means of freeing humans from boring, mundane tasks and opening the door to more enjoyable lifestyles. But others see this same phenomenon as a curse that robs citizens of employment and channels wealth toward those with power. This, in fact, was a message of the devoted humanitarian Mahatma Gandhi of India. He repeatedly argued that India would be better served by replacing large textile mills with spinning wheels placed in the homes of the peasants. In this way, he claimed, centralized mass production that employed only a few would be replaced by a distributed mass production system that would benefit multitudes.

History is full of revolutions with roots in the disproportionate distribution of wealth and privilege. If today's advancing technology is allowed to entrench such discrepancies, catastrophic consequences could result.

But the consequences of building increasingly intelligent machines is more subtle—more fundamental—than those dealing with power struggles between different segments of society. The issues strike at the very heart of humanity's self-image. In the nineteenth century, society was appalled by Charles Darwin's theory of evolution and the thought that humans might have evolved from lesser life forms. How then will society react if faced with the onslaught of machines whose mental capabilities challenge those of humans?

In the past, technology has developed slowly, allowing time for our self-image to be preserved by readjusting our concept of intelligence. Our ancient ancestors would have interpreted the mechanical devices of the nineteenth century as having supernatural intelligence, but today we do not credit these machines with any intelligence at all. But how will humanity react if machines truly challenge the intelligence of humans, or, more likely, if the capabilities of machines begin to advance faster than our ability to adapt?

We might get a clue to humanity's potential reaction to machines that challenge our intellect by considering society's response to IQ tests in the middle of the twentieth century. These tests were considered to identify a child's level of

intelligence. Children in the United States were often classified by their performances on these tests and channeled into educational programs accordingly. In turn, educational opportunities were opened to those children who performed well on these tests, whereas children who performed poorly were directed toward remedial programs of study. In short, when given a scale on which to measure an individual's intelligence, society tended to disregard the capabilities of those who found themselves on the lower end of the scale. How then would society handle the situation if the "intellectual" capabilities of machines became comparable, or even appeared to be comparable, with those of humans? Would society discard those whose abilities were seen as "inferior" to those of machines? If so, what would be the consequences for those members of society? Should a person's dignity be subject to how he or she compares to a machine?

We have already begun to see the intellectual powers of humans challenged by machines in specific fields. Machines are now capable of beating experts in chess; computerized expert systems are capable of giving medical advice; and simple programs managing investment portfolios often outperform investment professionals. How do such systems affect the self-image of the individuals involved? How will an individual's self-esteem be affected as that individual is outperformed by machines in more and more areas?

Many argue that the intelligence possessed by machines will always be inherently different from that of humans since humans are biological and machines are not. Thus, they argue, machines will never reproduce a human's decision-making process. Machines might reach the same decisions as humans but those decisions would not be made on the same basis as those made by humans. To what extent, then, are there different kinds of intelligence, and would it be ethical for society to follow paths proposed by nonhuman intelligence?

In his book, *Computer Power and Human Reason,* Joseph Weizenbaum argues against the unchecked application of artificial intelligence as follows:

> Computers can make judicial decisions, computers can make psychiatric judgments. They can flip coins in much more sophisticated ways than can the most patient human being. The point is that they *ought* not be given such tasks. They might even be able to arrive at "correct" decisions in some cases—but always and necessarily on bases no human being should be willing to accept.
>
> There have been many debates on "Computers and Mind." What I conclude here is that the relevant issues are neither technological nor even mathematical; they are ethical. They cannot be settled by asking questions beginning with "can." The limits of the applicability of computers are ultimately statable only in terms of oughts. What emerges as the most elementary insight is that, since we do not now have any ways of making computers wise, we ought not now to give computers tasks that demand wisdom.

You might argue that much of this section borders on science fiction rather than computer science. It was not too long ago, however, that many dismissed the question "What will happen if computers take over society?" with the same it-will-never-happen attitude. But in many respects, that day has now arrived. If a computerized database erroneously reports that you have a bad credit rating, a criminal record, or an overdrawn checking account, is it the computer's statement or your claim of innocence that will prevail? If a malfunctioning

navigational system indicates that a fog-covered runway is in the wrong place, where will the aircraft land? If a machine is used to predict the public's reaction to various political decisions, which decision does a politician make? How many times has a clerk been unable to help you because "the computer is down"? Who (or what), then, is in charge? Have we not already surrendered society to machines?

Questions & Exercises

1. How much of today's population would survive if the machines developed over the last one hundred years were removed? What about the last fifty years? What about twenty years? Where would the survivors be located?
2. To what extent is your life controlled by machines? Who controls the machines that affect your life?
3. Where do you get the information on which you base your daily decisions? What about your major decisions? What confidence do you have in the accuracy of that information? Why?

Chapter Review Problems

(Asterisked problems are associated with optional sections.)

1. As demonstrated in Section 11.2, humans might use a question for a purpose other than asking. Another example is "Do you know that your tire is flat?" which is used to inform rather than to ask. Give examples of questions used to reassure, to warn, and to criticize.
2. Analyze a soda dispensing machine as an agent. What are its sensors? What are its actuators? What level of response (reflex, knowledge based, goal based) does it exhibit?
3. Identify each of the following responses as being reflex, knowledge based, or goal based. Justify your answers.
   a. A computer program translating text from German to English
   b. A thermostat turning on the furnace when the temperature in a house drops below the current setting
   c. A pilot landing a plane safely on a runway
4. If a researcher uses computer models for studying the memorization capabilities of the human mind, do the programs developed for the machine necessarily memorize to the best of the machine's abilities? Explain.
5. Give some examples of declarative knowledge. Give some examples of procedural knowledge.

*6. In the context of object-oriented programming, what parts of an object are used to store declarative knowledge? What parts are used to store procedural knowledge?

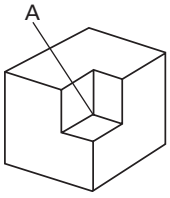
7. Which of the following activities do you expect to be performance oriented and which are simulation oriented?
   a. The design of an automated shuttle system (often used at airports between terminals)
   b. The design of a model predicting the path of a hurricane
   c. The design of a Web search database used to derive and maintain indices for documents stored on the World Wide Web
   d. The design of a model of a nation's economy for testing theories
   e. The design of a program for monitoring a patient's vital signs

8. Today, some telephone calls to businesses are handled by automated answering systems that use speech and voice recognition to converse with the caller. Do these systems pass the Turing test? Explain your answer.

9. Identify a small set of geometric properties that can be used to distinguish between the symbols F, E, L, and T.

***10.** Describe the similarities between the technique of identifying characteristics by comparing them to templates and the error-correcting codes discussed in Chapter 1.

11. Describe two interpretations of the following line drawing based on whether the "corner" marked A is convex or concave:



12. Compare the roles of the prepositional phrases in the following two sentences (which differ by only one word). How could a machine be programmed to make such distinctions?

The pigpen was built by the barn.
The pigpen was built by the farmer.

13. How do the results of parsing the following two sentences differ? How do the results of semantic analysis differ?

An awesome sunset was seen by Andrea.
Andrea saw an awesome sunset.

14. How do the results of parsing the following two sentences differ? How do the results of semantic analysis differ?

If X < 10 then subtract 1 from X else add 1 from X.
If X > 10 then add 1 to X else subtract 1 from X.

15. In the text we briefly discussed the problems of understanding natural languages as opposed to formal programming languages. As an example of the complexities involved in the case of natural languages, identify situations in which the question "Do you know what time it is?" has different meanings.

16. Changes in the context of a sentence can change the significance of the sentence as well as its meaning. In the context of Figure 11.3, how would the significance of the sentence "Mary hit John" change if the birth dates were in the late 2000s? What if one were in the 1980s and the other in the late 2000s?

17. Draw a semantic net representing the information in the following paragraph.

Donna threw the ball to Jack, who hit it into center field. The center fielder tried to catch it, but it bounced off the wall instead.

18. Sometimes the ability to answer a question depends as much on knowing the limits of knowledge as it does on the facts themselves. For example, suppose databases A and B both contain a complete list of employees who belong to the company's health insurance program, but only database A is aware that the list is complete. What could database A conclude about a member who was not on its list that database B could not?

19. Give an example in which the closed-world assumption leads to a contradiction.

20. Give two examples where the closed-world assumption is commonly used.

21. In the context of a production system, what is the difference between a state graph and a search tree?

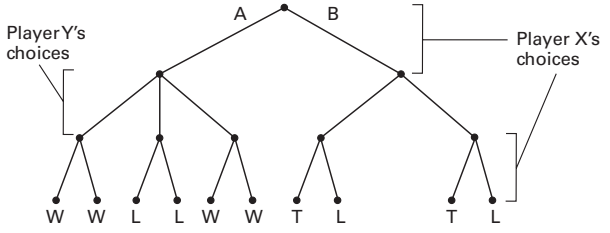
22. Analyze the task of solving the Rubik's cube in terms of a production system. (What are the states, the productions, and so on?)

23. a. Suppose a search tree is a binary tree and reaching the goal requires eight productions. What is the largest number of nodes that could be in the tree when the goal state is reached if the tree is constructed with a breadth-first manner?

b. Explain how the total number of nodes considered during the search could be reduced by conducting two searches at the same time—one beginning at the initial state while the other searches backward from the goal—until the two meet. (Assume that the search tree recording the states found in the backward search is also a binary tree and that both searches progress at the same rate.)

24. In the text we mentioned that a production system is often used as a technique for drawing conclusions from known facts. The states of the system are the facts known to be true at each stage of the reasoning process, and the productions are the rules of logic for manipulating the known facts. Identify some rules of logic that allow the conclusion "John is tall" to be obtained from the facts that "John is a basketball player," "Basketball players are not short," and "John is either short or tall."

25. The following tree represents possible moves in a competitive game, showing that player X currently has a choice between move A and move B. Following the move of player X, player Y is allowed to select a move, and then player X is allowed to select the last move of the game. The leaf nodes of the tree are labeled W, L, or T, depending on whether that ending represents a win, loss, or tie for player X. Should player X select move A or move B? Why? How does selecting a "production" in a competitive atmosphere differ from a one-person game such as the eight-puzzle?



26. Analyze the game of checkers as a production system and describe a heuristic that could be used to determine which of two states is closer to the goal. How would the control system in this setting differ from that of a one-person game such as the eight-puzzle?

27. By considering the manipulation rules of algebra as productions, problems involving the simplification of algebraic expressions can be solved in the context of a production system. Identify a set of algebraic productions that allow the equation $3/(2x - 1) = 6/(3x + 1)$ to be reduced to the form $x = 3$. What are some rules of thumb (that is, heuristic rules) used when performing such algebraic simplifications?

28. Draw the search tree that is generated by a breadth-first search in an attempt to solve the eight-puzzle from the following start state without using the assistance of any heuristic information.

	1	3
4	2	5
7	8	6

29. Draw the search tree that is generated by the best-fit algorithm of Figure 11.10 in an attempt to solve the eight-puzzle from the start state in Problem 28 if the number of tiles out of place is used as a heuristic.

30. Draw the search tree that is generated by the best-fit algorithm of Figure 11.10 in an attempt to solve the eight-puzzle from the following start state, assuming the heuristic used is the same as that developed in Section 11.3.

1	2	3
5	7	6
4		8

31. When solving the eight-puzzle, why would the number of tiles out of place not be as good a heuristic as the one used in Section 11.3?

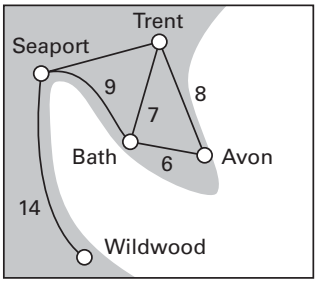
32. What is the distinction between the technique of deciding which half of the list to consider when performing a binary search (Section 5.5) and deciding which branch to pursue when performing a heuristic search?

33. Note that if a state in the state graph of a production system has an extremely low heuristic value in comparison to the other states and if there is a production from that state to itself, the algorithm in Figure 11.10 can get caught in the loop of considering that state over and over again. Show that if the cost of executing any production in the system is at least one, then by computing the projected cost to be the sum of the heuristic value plus the cost of reaching the state along the path being traversed, this endless looping process will be avoided.

34. What heuristic do you use when searching for a route between two cities on a large road map?

35. Draw to four levels the search tree produced by the best-fit algorithm of Figure 11.10 in finding the route from Trent to Wildwood. Each node in the search tree will be a city on the map. Begin with a node for Trent. When expanding a

node, add only the cities that are directly connected to the city being expanded. Record in each node the straight-line distance to Wildwood and use this as the heuristic value. Does the best-fit algorithm have a defect in its processing? If so, what correction is needed?



Straight line distance to Wildwood from

Avon	10
Bath	8
Trent	15
Seaport	13

36. The A* algorithm modifies the best-fist algorithm in two significant ways. First, it records the actual cost to reach a state. In the case of a route on a map, the actual cost is the distance traveled. Second, when selecting a node to expand, it chooses the node whose sum of the actual cost plus heuristic value is the smallest. Draw the search tree of Problem 35 that would result from these two modifications. Record in each node the distance traveled to the city, the heuristic value to reach the goal, and their sum. What is the found path from Dearborn to Wildwood?

37. List two properties that a heuristic should have if it is to be useful in a production system.

38. Suppose you have two buckets. One has a capacity of exactly three liters; the other has a capacity of five liters. You can pour water from one bucket to another, empty a bucket, or fill a bucket at any time. Your problem is to place exactly four liters of water in the five-liter bucket. Describe how this problem could be framed as a production system.

39. Suppose your job is to supervise the loading of two trucks, each of which can carry at most fourteen tons. The cargo is a variety of crates whose total weight is twenty-eight tons but whose individual weights vary from crate to crate. The weight of each crate is marked on its side. What heuristic would you use for dividing the crates between the two trucks?

40. Which of the following are examples of meta-reasoning?

a. He has been gone long so he must have gone far.

b. Since I usually make the wrong decision and the last two decisions I made were correct, I will reverse my next decision.

c. I am getting tired so I am probably not thinking clearly.

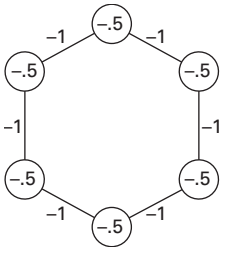
d. I am getting tired so I think I will take a nap.

41. Describe how a human's ability to solve the frame problem helps the human find lost articles.

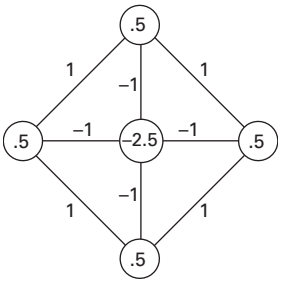
42. a. In what sense is learning by imitation similar to learning by supervised training?

b. In what sense is learning by imitation different from learning by supervised training?

43. The following diagram represents an artificial neural network for an associative memory as discussed in Section 11.5. What pattern does it associate with any pattern in which only two neurons that are separated by a single neuron are excited? What will happen if the network is initialized with all its neurons inhibited?



44. The following diagram represents an artificial neural network for an associative memory as discussed in Section 11.5. What stable configuration does it associate with any initial pattern in which at least three of the neurons on the perimeter are excited and the center neuron is inhibited? What would happen if it were given an initial pattern in which only two neurons that are opposite each other on the perimeter were excited?



45. Design an artificial neural network for an associative memory (as discussed in Section 11.5)

consisting of a rectangular array of neurons that tries to move toward stable patterns in which a single vertical column of neurons is excited.

46. Adjust the weights and threshold values in the artificial neural network in Figure 11.18 so that its output is 1 when both inputs are the same (both 0 or both 1) and 0 when the inputs are different (one being 0 while the other is 1).

47. Draw a diagram similar to Figure 11.5 representing the process of simplifying the algebraic expression $7x + 3 = 3x - 5$ to the expression $x = -2$.

48. Expand your answer to the previous problem to show other paths that a control system might pursue when attempting to solve the problem.

49. Draw a diagram similar to Figure 11.5 representing the reasoning process involved when concluding that "Polly can fly" from the initial facts "Polly is a parrot," "A parrot is a bird," and "All birds can fly."

50. In contrast to the statement in the preceding problem, some birds, such as an ostrich or a robin with a broken wing, cannot fly. However, it would not seem reasonable to construct a deductive reasoning system in which all the exceptions to the statement "All birds can fly" are explicitly listed. How then do we as humans decide whether a particular bird can or cannot fly?

51. Explain how the meaning of the sentence "I read the new tax law" depends on the context.

52. Describe how the problem of traveling from one city to another could be framed as a production system. What are the states? What are the productions?

53. Suppose you must perform three tasks, A, B, and C, that can be performed in any order (but not simultaneously). Describe how this problem can be framed as a production system and draw its state graph.

54. How does the state graph in the previous problem change if task C must be performed before task B?

55. a. If the notation (i, j), where i and j are positive integers, is used to mean "if the entry in the ith position in the list is greater than the entry in the jth position, interchange the two entries," which of the following two sequences does a better job of sorting a list of length three?
(1, 3) (3, 2)
(1, 2) (2, 3) (1, 2)

b. Note that by representing sequences of interchanges in this manner, sequences can be broken into sub-sequences that can then be reconnected to form new sequences. Use this approach to describe a genetic algorithm for developing a program that sorts lists of length ten.

56. Suppose each member in a group of robots is to be equipped with a pair of sensors. Each sensor can detect an object directly in front of it within a range of two meters. Each robot is shaped like a round trash can and can move in any direction. Design a sequence of experiments to determine where the sensors should be placed to produce a robot that successfully pushes a basketball in a straight line. How does your sequence of experiments compare to an evolutionary system?

57. Do you tend to make decisions in a reactive or plan-based mode? Does your answer depend on whether you are deciding on what to have for lunch or making a career decision?

Social Issues

The following questions are intended as a guide to the ethical/social/legal issues associated with the field of computing. The goal is not merely to answer these questions. You should also consider why you answered as you did and whether your justifications are consistent from one question to the next.

1. To what extent should researchers in nuclear power, genetic engineering, and artificial intelligence be held responsible for the way the results of their work

are used? Is a scientist responsible for the knowledge revealed by his or her research? What if the resulting knowledge was an unexpected consequence?

2. How would you distinguish between intelligence and simulated intelligence? Do you believe there is a difference?
3. Suppose a computerized medical expert system gains a reputation within the medical community for giving sound advice. To what extent should a physician allow that system to alter his or her decisions regarding the treatment of patients? If the physician applies a treatment contrary to that proposed by the expert system and the system turns out to be right, is the physician guilty of malpractice? In general, if an expert system becomes well-known within a field, to what degree could it hamper, rather than enhance, the ability of human experts when making their own judgments?
4. Many would argue that a computer's actions are merely consequences of how it was programmed, and thus a computer cannot possess free will. In turn, a computer should not be held responsible for its actions. Is a human's mind a computer? Are humans preprogrammed at birth? Are humans programmed by their environments? Are humans responsible for their actions?
5. Are there avenues that science should not pursue even though it might be capable of doing so? For instance, if it becomes possible to construct a machine with perception and reasoning skills comparable to those of humans, would the construction of such a machine be appropriate? What issues could the existence of such a machine raise? What are some of the issues being raised today by advancements in other scientific fields?
6. History abounds with instances in which the work of scientists and artists was affected by the political, religious, or other social influences of their times. In what ways are such issues affecting current scientific efforts? What about computer science in particular?
7. Many cultures today take at least some responsibility toward helping to retrain those whose jobs have been made redundant by advancing technology. What should/can society do as technology makes more and more of our capabilities redundant?
8. Suppose you receive a computer-generated bill for $0.00. What should you do? Suppose you do nothing and 30 days later you receive a second notice of $0.00 due in your account. What should you do? Suppose you do nothing and 30 days later you receive another notice of $0.00 due in your account along with a note stating that, unless the bill is paid promptly, legal action will be taken. Who is in charge?
9. Are there times when you associate personalities with your personal computer? Are there times when it seems vindictive or stubborn? Do you ever get mad at your computer? What is the difference between being mad *at* your computer and being mad *as a result of* your computer? Does your computer ever get mad at you? Do you have similar relationships with other objects such as cars, televisions, and ball-point pens?
10. On the basis of your answers to Question 9, to what extent are humans willing to associate an entity's behavior with the presence of intelligence and awareness? To what extent should humans make such associations? Is it possible for an intelligent entity to reveal its intelligence in some way other than its behavior?

11. Many feel that the ability to pass the Turing test does not imply that a machine is intelligent. One argument is that intelligent behavior does not, in itself, imply intelligence. Yet the theory of evolution is based on the survival of the fittest, which is a behavior-based test. Does the theory of evolution imply that intelligent behavior is a predecessor to intelligence? Would the ability to pass the Turing test imply that machines were on their way to becoming intelligent?
12. Medical treatment has advanced to the point that numerous parts of the human body can now be replaced with artificial parts or parts from human donors. It is conceivable that this might someday include parts of the brain. What ethical problems would such capabilities raise? If a patient's neurons were replaced one at a time with artificial neurons, would the patient remain the same person? Would the patient ever notice a difference? Would the patient remain human?
13. A GPS in an automobile provides a friendly voice notifying the driver of upcoming turns and other actions. In the event the driver makes a mistake, it will automatically make adjustments and provide directions to get back on route without undue emotion. Do you feel that a GPS reduces a driver's stress when driving to a new destination? In what ways does a GPS contribute to stress?
14. Suppose your smartphone provided voice-to-voice language translation, would you feel comfortable using this feature? Would you trust it to convey the correct meaning? Would you have any concerns?

Additional Reading

Banzhaf, W., P. Nordin, R. E. Deller, and F. D. Francone. *Genetic Programming: An Introduction.* San Francisco, CA: Morgan Kaufmann, 1998.

Lu, J. and J. Wu. *Multi-Agent Robotic Systems*. Boca Raton, FL: CRC Press, 2001.

Luger, G. F. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving,* 5th ed. Boston, MA: Addison-Wesley, 2005.

Mitchell, M. *An Introduction to Genetic Algorithms.* Cambridge, MA: MIT Press, 1998.

Negnevitsky, M. *Artificial Intelligence: A Guide to Intelligent Systems,* 2nd ed. Boston, MA: Addison-Wesley, 2005.

Nilsson, N. *Artificial Intelligence: A New Synthesis.* San Francisco, CA: Morgan Kaufmann, 1998.

Nolfi, S. and D. Floreano. *Evolutionary Robotics*. Cambridge, MA: MIT Press, 2000.

Rumelhart, D. E. and J. L. McClelland. *Parallel Distributed Processing.* Cambridge, MA: MIT Press, 1986.

Russell, S. and P. Norvig. *Artificial Intelligence: A Modern Approach,* 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 2009.

Shapiro, L. G. and G. C. Stockman. *Computer Vision*. Englewood Cliffs, NJ: Prentice-Hall, 2001.

Shieber, S. *The Turing Test.* Cambridge, MA: MIT Press, 2004.

Weizenbaum, J. *Computer Power and Human Reason.* New York: W. H. Freeman, 1979.