27 AI: PRESENT AND FUTURE

In which we take stock of where we are and where we are going, this being a good thing to do before continuing.

In Part I, we proposed a unified view of AI as rational agent design. We showed that the design problem depends on the percepts and actions available to the agent, the goals that the agent's behavior should satisfy, and the nature of the environment. A variety of different agent designs are possible, ranging from reflex agents to fully deliberative, knowledge-based agents. Moreover, the components of these designs can have a number of different instantiations—for example, logical, probabilistic, or "neural." The intervening chapters presented the principles by which these components operate.

For all the agent designs and components, there has been tremendous progress both in our scientific understanding and in our technological capabilities. In this chapter, we stand back from the details and ask, *"Will all this progress lead to a general-purpose intelligent agent that can perform well in a wide variety of environments?* Section 27.1 looks at the components of an intelligent agent to assess what's known and what's missing. Section 27.2 does the same for the overall agent architecture. Section 27.3 asks whether "rational agent design" is the right goal in the first place. (The answer is, "Not really, but it's OK for now.") Finally, Section 27.4 examines the consequences of success in our endeavors.

27.1 AGENT COMPONENTS

Chapter 2 presented several agent designs and their components. To focus our discussion here, we will look at the utility-based agent, which we show again in Figure 27.1. This is the most general of our agent designs; we will also consider its extension with learning capabilities, as depicted in Figure 2.15.

Interaction with the environment through sensors and actuators: For much of the history of AI, this has been a glaring weak point. With a few honorable exceptions, AI systems were built in such a way that humans had to supply the inputs and interpret the outputs, while robotic systems focused on low-level tasks in which high-level reasoning and planning were largely absent. This was due in part to the great expense and engineering effort required

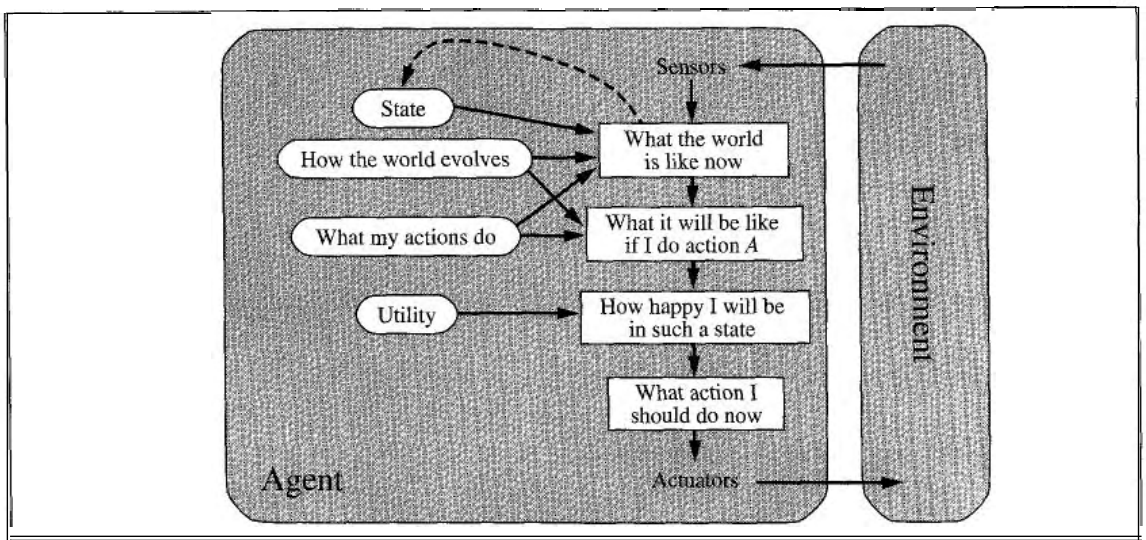

Figure 27.1 A model-based, utility-based agent, as first presented in Figure 2.14.

to get real robots to work at all. The situation has changed rapidly in recent years with the availability of ready-made programmable robots, such as the four-legged robots shown in Figure 25.4(b). These, in turn, have benefited from small, cheap, high-resolution CCD cameras and compact, reliable motor drives. MEMS (micro-electromechanical systems) technology has supplied miniaturized accelerometers and gyroscopes and is now producing actuators that will, for example, power an artificial flying insect. (It may also be possible to combine millions of MEMS actuators to produce very powerful macroscopic actuators.) For physical environments, then, AI systems no longer have a real excuse. Furthermore, an entirely new environment—the Internet—has become available.

Keeping track of the state of the world: This is one of the core capabilities required for an intelligent agent. It requires both perception and updating of internal representations. Chapter 7 described methods for keeping track of worlds described by propositional logic; Chapter 10 extended this to first-order logic; and Chapter 15 described filtering algorithms for tracking uncertain environments. These filtering tools are required when real (and therefore imperfect) perception is involved. Current filtering and perception algorithms can be combined to do a reasonable job of reporting low-level predicates such as "the cup is on the table" but we have some way to go before they can report that "Dr. Russell is having a cup of tea with Dr. Norvig." Another problem is that, although approximate filtering algorithms can handle quite large environments, they are still essentially propositional—like propositional logic, they do not represent objects and relations explicitly. Chapter 14 explained how probability and first-order logic can be combined to solve this problem; we expect that the application of these ideas for tracking complex environments will yield huge benefits. Incidentally, as soon as we start talking about objects in an uncertain environment, we encounter identity uncertainty—we don't know which object is which. This problem has been largely ignored in logic-based AI, where it has generally been assumed that percepts incorporate constant symbols that identify the objects.

IDENTITY UNCERTAINTY

Projecting, evaluating, and selecting future courses of action: The basic knowledge representation requirements here are the same as for keeping track of the world; the primary difficulty is coping with courses of action—such as having a conversation or a cup of tea—that consist eventually of thousands or millions of primitive steps for a real agent. It is only by imposing **hierarchical structure** on behavior that we humans cope at all. Some of the planning algorithms in Chapter 12 use hierarchical representations and first-order representations to handle problems of this scale; on the other hand, the algorithms given in Chapter 17 for decision making under uncertainty are essentially using the same ideas as the state-based search algorithms of Chapter **3.** There is clearly a great deal of work to do here, perhaps along the lines of recent developments in **hierarchical reinforcement learning.**

Utility as an expression of preferences: In principle, basing rational decisions on the maximization of expected utility is completely general and avoids many of the problems of purely goal-based approaches, such as conflicting goals and uncertain attainment. As yet, however, there has been very little work on constructing *realistic* utility functions—imagine, for example, the complex web of interacting preferences that must be understood by an agent operating as an office assistant for a human being. It has proven very difficult to decompose preferences over complex states in the same way that Bayes nets decompose beliefs over complex states. One reason may be that preferences over states are really *compiled* from preferences over state histories, which are described by **reward functions** (see Chapter 17). Even if the reward function is simple, the corresponding utility function may be very complex. This suggests that we take seriously the task of knowledge engineering for reward functions as a way of conveying to our agents what it is that we want them to do.

Learning: Chapters 18 to 20 described how learning in an agent can be formulated as inductive learning (supervised, unsupervised, or reinforcement-based) of the functions that constitute the various components of the agent. Very powerful logical and statistical techniques have been developed that can cope with quite large problems, often reaching or exceeding human capabilities in the identification of predictive patterns defined on a given vocabulary. On the other hand, machine learning has made very little progress on the important problem of constructing new representations at levels of abstraction higher than the input vocabulary. For example, how can an autonomous robot generate useful predicates such as *Office* and Cafe if they are not supplied to it by humans? Similar considerations apply to learning behavior—HavingACupOfTea is an important high-level action, but how does it get into an action library that initially contains much simpler actions such as *RaiseArm* and Swallow? Unless we understand such issues, we are faced with the daunting task of constructing large commonsense knowledge bases by hand.

27.2 AGENT ARCHITECTURES

It is natural to ask, "Which of the agent architectures in Chapter 2 should an agent use?" The answer is, "All of them!" We have seen that reflex responses are needed for situations in which time is of the essence, whereas knowledge-based deliberation allows the agent to

HYBRID ARCHITECTURE

plan ahead. A complete agent must be able to do both, using a **hybrid architecture.** One important property of hybrid architectures is that the boundaries between different decision components are not fixed. For example, **compilation** continually converts declarative information at the deliberative level into more efficient representations, eventually reaching the reflex level—see Figure 27.2. (This is the purpose of explanation-based learning, as discussed in Chapter 19.) Agent architectures such as SOAR (Laird *et* al., 1987) and THEO (Mitchell, 1990) have exactly this structure. Every time they solve a problem by explicit deliberation, they save away a generalized version of the solution for use by the reflex component. A less studied problem is the *reversal* of this process: when the environment changes, learned reflexes may no longer be appropriate and the agent must return to the deliberative level to produce new behaviors.

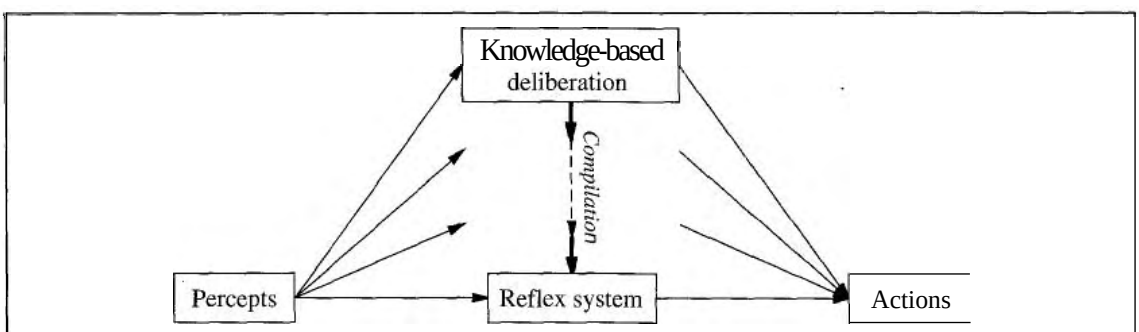

Figure 27.2 Compilation serves to convert deliberative decision making into more efficient, reflexive mechanisms.

Agents also need ways to control their own deliberations. They must be able to cease deliberating when action is demanded, and they must be able to use the time available for deliberation to execute the most profitable computations. For example, a taxi-driving agent that sees an accident ahead must decide in a split second either to brake or to take evasive action. It should also spend that split second thinking about the most important questions, such as whether the lanes to the left and right are clear and whether there is a large truck close behind, rather than worrying about wear and tear on the tires or where to pick up the next passenger. These issues are usually studied under the heading of **real-time** AI. As AI systems move into more complex domains, all problems will become real-time, because the agent will never have long enough to solve the decision problem exactly.

REAL-TIME AI

Clearly, there is a pressing need for methods that work in more general decision-making situations. Two promising techniques have emerged in recent years. The first involves the use of **anytime algorithms** (Dean and Boddy, 1988; Horvitz, 1'987). An anytime algorithm is an algorithm whose output quality improves gradually over time, so that it has a reasonable decision ready whenever it is interrupted. Such algorithms are controlled by a metalevel decision procedure that assesses whether further computation is worthwhile. Iterative deepening search in game playing provides a simple example of an anytime algorithm. More complex systems, composed of many such algorithms working together, can also be constructed (Zilberstein and Russell, 1996). The second technique is **decision-theoretic metareasoning**

ANYTIME ALGORITHMS

DECISION-THEORETIC METAREASONING