In Chapter 11 we will look at how the techniques of cognitive neuroscience can be used to study the organization of the mind, focusing in particular on the strengths and limits of using imaging techniques to map the mind. In Chapter 12 we will work through a case study that brings the theoretical discussions about modularity to life. We will look at a debate that is very much at the forefront of contemporary cognitive science – the controversial question of whether there is a module responsible for reasoning about the mental states of others, or what many cognitive scientists have come to call the theory of mind module.

First, though, we will look at a way of thinking about the mind that brings the discussion of modularity in this chapter into contact with the discussion in earlier chapters of two competing ways of modeling information processing. In the next section we will look at hybrid mental architectures that have both symbolic components (as per the physical symbol system hypothesis) and subsymbolic components (as per the artificial neural networks approach).

10.4 Hybrid architectures

Up to now we have been thinking separately about the different aspects of mental architecture. We looked in detail in Chapters 6 through 9 at the two principal models of information storage and information processing – the symbolic paradigm associated with the physical symbol system hypothesis, and the distributed paradigm associated with artificial neural networks. In this chapter we have been looking at two different ways of thinking about the overall organization of the mind. We began with several different models of agent architectures and then went on to study both Fodor's sharp distinction between modular processing and central processing, and the massive modularity thesis associated with the evolutionary psychologists Leda Cosmides and John Tooby. In this section we will bring these two different aspects of mental architectures into contact. We will look at how the symbolic and distributed paradigms have been combined in a model of the overall organization of the mind – the ACT-R/PM cognitive architecture associated with the psychologist John R. Anderson and his research team at Carnegie Mellon University.

It may have occurred to you that the distinction between physical symbol systems and artificial neural networks is not all-or-nothing. As we saw when we looked at specific examples and models, symbolic and distributed information processing seem to be suited for different tasks and different types of problem-solving. The type of problems tackled by GOFAI physical symbol systems tend to be highly structured and sharply defined – playing checkers, for example, or constructing decision trees from databases. The type of problems for which artificial neural networks seem particularly well suited tend to be perceptual (distinguishing mines from rocks, for example, or modeling how infants represent unseen objects) and involve recognizing patterns (such as patterns in forming the past tense of English verbs).

The extreme version of the physical symbol system hypothesis holds that *all* information processing involves manipulating and transforming physical symbol structures.

It may be that Newell and Simon themselves had something like this in mind. There is a comparable version of the artificial neural networks approach, holding that physical symbol structures are completely redundant in modeling cognition – artificial neural networks are all we need. There seems to be room, though, for a more balanced approach that tries to incorporate both models of information processing. Anderson's ACT-R/PM cognitive architecture is a good example.

We have talked a lot about mental architectures in this book. Mental architectures, as we have been thinking about them, are theoretical in orientation – they incorporate theoretical models of information processing and how the mind is organized (whether it is modular, for example, and if so how). The notion of a cognitive architecture, as used by computer scientists and psychologists, is a more practical notion. A cognitive architecture is similar to a programming language. It gives researchers the tools to construct cognitive models using a common language and common toolkit.

One of the first cognitive architectures was actually developed by Allen Newell, working with John Laird and Paul Rosenbloom. It was originally called SOAR (for State Operator And Result). The current incarnation is known as Soar. Soar is very closely tied to the physical symbol system hypothesis. It is based on the means-end and heuristic search approaches to problem-solving that we looked at in Chapter 6. Soar is intended to be a unified model of cognition. It does not incorporate any elements corresponding to artificial neural networks. All knowledge is represented in the same way in the architecture, and manipulated in a rule-governed way.

The ACT-R/PM (Adaptive Control of Thought – Rational/Perceptual-Motor) cognitive architecture is the latest installment of a cognitive architecture that was first announced under the name ACT in 1976. It is a development of the ACT-R architecture, which itself develops the ACT* architecture. ACT-R/PM is less homogeneous than Soar. It counts as a hybrid architecture because it incorporates both symbolic and subsymbolic information processing. One of the things that makes ACT-R interesting from the perspective of this chapter is that it is a modular cognitive architecture. It has different modules performing different cognitive tasks and the type of information processing depends upon the type of task.

The ACT-R/PM architecture

The basic structure of ACT-R/PM is illustrated in Figure 10.6. As the diagram shows, the architecture has two layers – a perceptual-motor layer and a cognitive layer. It is the addition of the perceptual-motor layer that distinguishes ACT-R/PM from its predecessor ACT-R. Each layer contains a number of different modules.

The modules within each layer are generally able to communicate directly with each other. Communication between modules on different layers, on the other hand, only takes place via a number of *buffers*. A buffer is rather like a workspace. It contains the "sensory" input that is available for processing by the central cognitive modules. The cognitive modules can only access sensory information that is in the relevant buffer (visual information in the visual buffer, and so on).

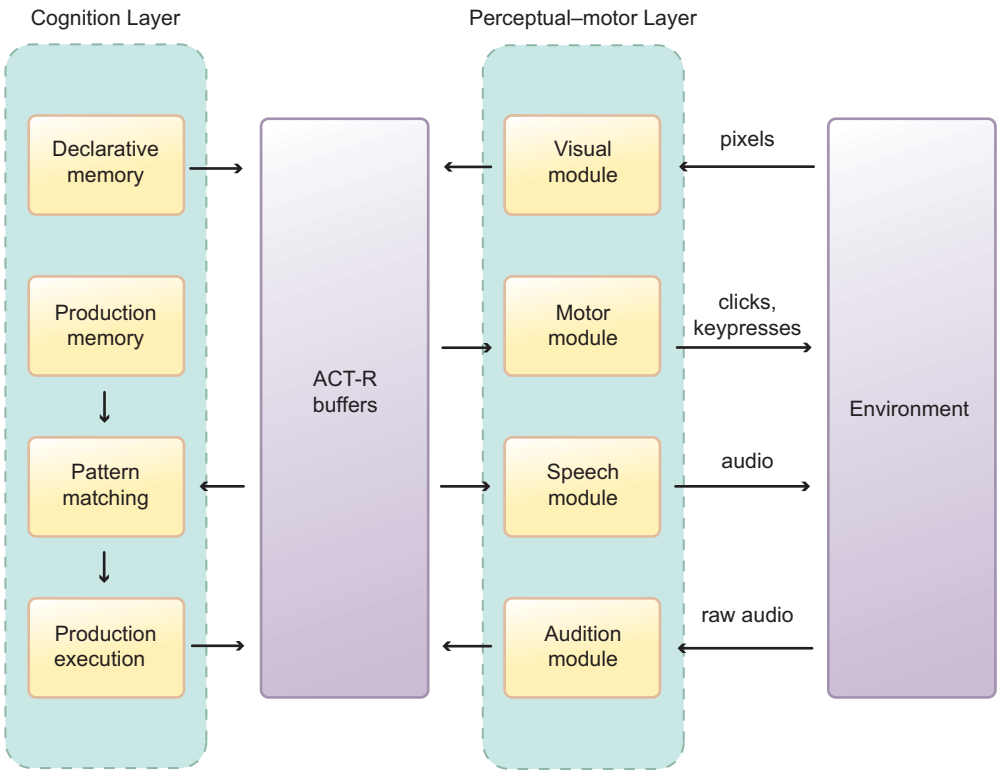

Figure 10.6 The ACT-R/PM cognitive architecture. The architecture has two layers – a cognitive layer and a perceptual–motor (PM) layer. By permission of Lorin Hochstein.

The cognition layer is built upon a basic distinction between two types of knowledge - declarative and procedural. In philosophy this is often labeled the distinction between *knowledge-that* (declarative) and *knowledge-how* (procedural) - between, for example, knowing that Paris is the capital of France and knowing how to speak French. The first type of knowledge involves the storage and recall of a very specific piece of information. The second is a much more general skill, one that is manifested in many different ways and in many different types of situations.

Declarative and procedural knowledge are both represented symbolically, but in different ways. Declarative knowledge is organized in terms of "chunks." A chunk is an organized set of elements. These elements may be derived from the perceptual systems, or they may be further chunks. The basic ideas behind chunking as a way of representing the content of declarative memory are directly related to the physical symbol system hypothesis. We can think of chunks as symbol structures (say the equation "7 + 6 = 13") built up in rule-governed ways from physical symbols (corresponding to "7," "6," "+," "1," and "3"). These chunks are stored in the declarative memory module. The chunks in declarative memory might encode objects in the environment. Or they might encode goals of the system.

ACT-R/PM represents procedural knowledge in terms of *production rules*. Production rules are also known as Condition-Action Rules. As this alternative name suggests, production rules identify specific actions for the system to perform, depending upon which condition it finds itself in. When a production rule fires (as the jargon has it) in a given condition, it can perform one of a range of actions. It can retrieve a chunk from declarative memory, for example. Or it can modify that chunk – updating its representation of the environment, for example, or modifying a goal. It can also modify its environment. In this case the action really is an action – it sends a command to the motor module. And of course production rules can be nested within each other, so that the output of a given production rule serves as a condition triggering the firing of another production rule. This allows complex abilities (such as multiplication) to be modeled as sets of production rules.

So far there is nothing hybrid about ACT-R/PM. The way declarative and procedural knowledge is encoded and manipulated in the architecture is entirely in line with the physical symbol system hypothesis. And in fact the same holds for the perceptual and motor modules. Here too information is encoded in the form of physical symbols. The perceptual and motor modules are designed on the basis of the EPIC (Executive Process/Interactive Control) architecture developed by David Kieras and David Meyer. EPIC falls squarely within the physical symbol system approach.

ACT-R/PM as a hybrid architecture

What makes ACT-R/PM a hybrid architecture is that this symbolic, modular architecture is run on a subsymbolic base. In order to appreciate what is going on here, take another look at Figure 10.6. In many ways the overall organization looks very Fodorean. There are input modules and output modules. These modules are all encapsulated. They communicate only via the buffer systems. And yet there is something missing. There is no system responsible for what Fodor would call central processing. But nor, on the other hand, is ACT-R/PM massively modular. It does not have dedicated, domain-specific modules.

So, a natural question to ask of ACT-R/PM is: How does it decide what to do? If a given production rule or set of production rules is active, then there is no difficulty. The system follows the "instructions" provided by the production rules – it performs the actions triggered by the conditions in which it finds itself. But how does it decide which production rules to apply? ACT-R/PM is designed to operate serially. At any given moment, only one production rule can be active. But most of the time there are many different production rules that could be active. Only one of them is selected. In a Fodorean architecture, this job would be done by some type of central processing system that operates symbolically. In ACT-R/PM, in contrast, the process of selection takes place subsymbolically. This is what makes it a hybrid architecture.

The job of selecting which production rule is to be active at a given moment is performed by the pattern-matching module. This module controls which production rule gains access to the buffer. It does this by working out which production rule has the highest utility at the moment of selection. The concept of utility is directly derived from

the theory of rational choice (as developed, for example, in statistics, decision theory, and economics) – this is why the "R" in ACT-R/PM stands for "rational."

Utility can be understood in many different ways, but the basic idea is that the production rule with the highest utility is the rule whose activation will best benefit the cognitive system. The notion of benefit here is understood with reference to the system's goals – or rather, to the system's current goal. The utility of a particular production rule is determined by two things. The first is how likely the system is to achieve its current goal if the production rule is activated. The second is the cost of activating the production rule.

So, the pattern-matching module essentially carries out a form of cost–benefit analysis in order to determine which production rule should gain access to the buffer. The entire process takes place without any overseeing central system. It is a type of "winner-take-all" system. All the work is done by the equations that continually update the cost and utility functions. Once the numbers are in, the outcome is determined.

The designers of ACT-R/PM describe these calculations as *subsymbolic*. This is a very important concept that is also standardly used to describe how artificial neural networks operate. Each production rule is purely symbolic. Production rules are built up in rule-governed ways from basic constituent symbols exactly as the physical symbol system hypothesis requires. The compositional structure of production rules determines how the production rule behaves once it is activated, but it does not play a part in determining whether or not the rule is activated. For that we need to turn to the numbers that represent the production rule's utility. These numbers are subsymbolic because they do not reflect the symbolic structure of the production rule.

ACT-R/PM has other subsymbolic dimensions. The architecture also uses subsymbolic equations to model the accessibility of information in declarative memory. It is a very basic fact about cognition that memories are not all created equal. Some are easier to access than others. Cognitive psychologists studying memory have discovered all sorts of different effects when they study how memories are accessed and retrieved. An architecture such as ACT-R/PM has to find a way of modeling this type of variability.

The accessibility and retrievability of memories in ACT-R/PM is modeled subsymbolically. Recall that the basic units of declarative memory are chunks – as opposed to the production rules that are the basic units of procedural memory. Each chunk has associated with it a particular activation level. This activation level can be represented numerically. The higher the activation level, the easier it is to retrieve the chunk from storage.

The activation levels of chunks in declarative memory are determined by equations. These equations are rather similar to the equations governing the utilities of production rules. There are two basic components determining a chunk's overall activation level. The first component has to do with how useful the chunk has been in the past. Usefulness is understood in terms of utility, which in turn is understood in terms of how the chunk has contributed to realizing the system's goals. The second component has to do with how relevant the chunk is to the current situation and context.

Again, we can draw the same basic contrast here between symbolic and subsymbolic dimensions. Chunks themselves are symbolic ways of carrying information. They are

TABLE 10.2 Comparing the symbolic and subsymbolic dimensions of knowledge representation in the hybrid ACT-R/PM architecture

	PERFORMANCE MECHANISMS		LEARNING MECHANISMS	
	SYMBOLIC	SUBSYMBOLIC	SYMBOLIC	SUBSYMBOLIC
Declarative chunks	Knowledge usually facts) that can be directly verbalized	Relative activation of declarative chunks affects retrieval	Adding new declarative chunks to the set	Changing activation of declarative chunks and changing strength of links between chunks
Production rules	Knowledge for taking particular actions in particular situations	Relative utility of production rules affects choice	Adding new production rules to the set	Changing utility of production rules

built up from basic symbols and the way they function within the architecture is determined by this symbolic structure. But they cannot do anything while they are stored in the declarative memory module. In order to function within the architecture they need to be retrieved from storage and placed in the buffer. This process is governed by the subsymbolic equations that fix each chunk's activation level as a function of its past usefulness and current relevance. These equations are subsymbolic because they are completely independent of the chunk's internal symbolic structure. Table 10.2 summarizes the relation between the symbolic and subsymbolic dimensions of ACT-R/PM.

What ACT-R/PM reveals, therefore, is the possibility of an approach to thinking about the overall organization of the mind that combines elements of the two different approaches to information processing that we have been considering - the symbolic approach associated with the physical symbol system hypothesis, on the one hand, and the subsymbolic approach associated with the artificial neural network approach, on the other. Knowledge is represented in ACT-R/PM in the form of physical symbol structures – either as chunks of declarative knowledge, or as production rules in procedural memory. Once these items of knowledge reach the buffer, and so become available for general processing within the system, they operate purely symbolically. But the processes that govern when and how they reach the buffer are subsymbolic.

It is true that the designers of ACT-R/PM do not see these types of subsymbolic processing as being implemented by artificial neural networks – artificial neural networks do not have a monopoly on subsymbolic information processing. So, the distinction between symbolic and subsymbolic information processing in this architecture does not map precisely onto the distinction between physical symbol systems and artificial neural networks. But there are (at least) two very important lessons to be learnt from ACT-R/PM.

The first lesson is the very close connection between debates about the organization of the mind and debates about the nature of information processing. Thinking properly

about the modular organization of the mind requires thinking about how the different modules might execute their information-processing tasks. The second lesson follows directly on from this. Different parts of a mental architecture might exploit different models of information processing. Some tasks lend themselves to a symbolic approach. Others to a subsymbolic approach. The debate between models of information processing is not all-or-nothing.

Summary

This chapter has focused on the third of the questions that a mental architecture has to answer: How is the mind organized so that it can function as an information processor? We began by looking at three different architectures for intelligent agents in AI, in order to see what distinguishes the organization of cognitive agents from that of simple reflex agents. Cognitive agents are standardly modeled in terms of quasi-autonomous information-processing systems, which raises the question of how those systems should be understood. Pursuing this question we looked at Jerry Fodor's analysis of modular information-processing systems and explored his reasons for thinking that cognitive science is best suited to explaining modular systems, as opposed to non-modular, central information-processing systems. We then examined an alternative proposed by massive modularity theorists, who hold that all information processing is modular. Finally we turned to the hybrid architecture ACT-R/PM, which brings the discussion of modularity into contact with the discussion of information processing in Part III. ACT-R/PM is a modular system that combines the symbolic approach associated with the physical symbol system hypothesis and the subsymbolic neural networks approach.

Checklist

Computer scientists building intelligent agents distinguish different types of agent architectures

(1) Simple reflex agents have condition-action rules (production rules) that directly link sensory and effector systems.
(2) Simple reflex agents are not cognitive systems, unlike goal-based agents and learning agents.
(3) Goal-based agents and learning agents are built up from sub-systems that perform specific information-processing tasks.
(4) This general approach to agent architecture raises theoretical questions explored in discussions of modularity.

Fodor's modularity thesis

(1) The thesis is built on a rejection of horizontal faculty psychology (the idea that the mind is organized in terms of faculties such as memory and attention that can process any type of information).