CHAPTER 4

The Knowledge Level

A. Newell, Carnegie Mellon University

1. Introduction

This is the first presidential address of AAAI, the American Association for Artificial Intelligence. In the grand scheme of history, even the history of artificial intelligence (AI), this is surely a minor event. The field this scientific society represents has been thriving for quite some time. No doubt the society itself will make solid contributions to the health of our field. But it is too much to expect a presidential address to have a major impact.

So what is the role of the presidential address and what is the significance of the first one? I believe its role is to set a tone, to provide an emphasis. I think the role of the *first* address is to take a stand about what that tone and emphasis should be, to set expectations for future addresses and to communicate to my fellow presidents.

Only two foci are really possible for a presidential address: the state of the society or the state of the science. I believe the latter to be the correct focus. AAAI itself, its nature and its relationship to the larger society that surrounds it, are surely important.[1] However, our main business is to help AI become a science—albeit a science with a strong engineering flavor. Thus, though a president's address cannot be narrow or highly technical, it can certainly address a substantive issue. That is what I propose to do.

I wish to address the question of knowledge and representation. That is a little

*Presidential Address, American Association for Artificial Intelligence, AAAI80, Stanford University, 19 Aug 1980. Also published in the AI Magazine 2(2) 1981.

**This research was sponsored by the Defense Advanced Research Projects Agency (DOD), ARPA Order No.3597, monitored by the Air Force Avionics Laboratory Under Contract F33615-78-C-1551.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government.

[1] I have already provided some comments, as president, on such matters [25].

like a physicist wishing to address the question of radiation and matter. Such broad terms designate a whole arena of phenomena and a whole armada of questions. But comprehensive treatment is neither possible nor intended. Rather, such broad phrasing indicates an intent to deal with the subject in some basic way. Thus, the first task is to make clear the aspect of knowledge and representation of concern, namely, what is the nature of knowledge. The second task will be to outline an answer to this question. As the title indicates, I will propose the existence of something called the *knowledge level*. The third task will be to describe the knowledge level in as much detail as time and my own understanding permits. The final task will be to indicate some consequences of the existence of a knowledge level for various aspects of AI.

2. The Problem of Representation and Knowledge

2.1. The standard view

Two orthogonal and compatible basic views of the enterprise of AI serve our field, beyond all theoretical quibbles. The first is a geography of task areas. There is puzzle solving, theorem proving, game-playing induction, natural language, medical diagnosis, and on and on, with subdivisions of each major territory. AI, in this view, is an exploration, in breadth and in depth, of new territories of tasks with their new patterns of intellectual demands. The second view is the functional components that comprise an intelligent system. There is a perceptual system, a memory system, a processing system, a motor system, and so on. It is this second view that we need to consider to address the role of representation and knowledge.

Fig. 1. shows one version of the functional view, taken from [28], neither better nor worse than many others. An intelligent agent is embedded in a *task environment*; a *task statment* enters via a *perceptual* component and is encoded in an initial *representation*. Whence starts a cycle of activity in which a *recognition* occurs (as indicated by the *eyes*) of a method to use to attempt the problem. The method draws upon a memory of *general world knowledge*. In the course of such cycles, new methods and new representations may occur, as the agent attempts to solve the problem. The *goal structure*, a component we all believe to be important, does not receive its due in this figure, but no matter. Such a picture represents a convenient and stable decomposition of the *functions* to be performed by an intelligent agent, quite independent of particular implementations and anatomical arrangements. It also provides a convenient and stable decomposition of the entire scientific field into subfields. Scientists specialize in perception, or problem solving methods, or representation, etc.

It is clear to us all what *representation* is in this picture. It is the data structures that hold the problem and will be processed into a form that makes the solution available. Additionally, it is the data structures that hold the world

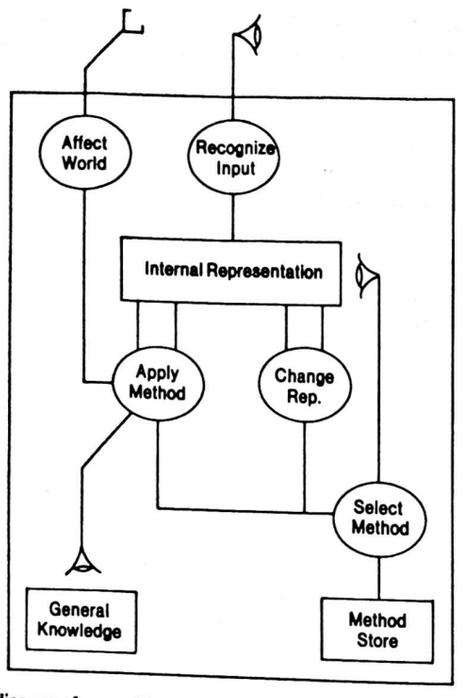

FIG. 1. Functional diagram of general intelligent agent (after [28]).

knowledge and will be processed to acquire parts of the solution or to obtain guidance in constructing it. The first data structures represent the problem, the second represent world knowledge.

A data structure by itself is impotent, of course. We have learned to take the representation to include the basic operations of reading and writing, of access and construction. Indeed, as we know, it is possible to take a pure process view of the representation and work entirely in terms of the inputs and outputs to the read and write processes, letting the data structure itself fade into a mythical story we tell ourselves to make the memory-dependent behavior of the read and write processes coherent.

We also understand, though not so transparently, why the representation represents. It is because of the totality of procedures that process the data structure. They transform it in ways consistent with an interpretation of the data structure as representing something. We often express this by saying that a data structure requires an *interpreter*, including in that term much more than just the basic read/write processes, namely, the whole of the active system that uses the data structure.

The term representation is used clearly (almost technically) in AI and

computer science. In contrast, the term *knowledge* is used informally, despite its prevalence in such phrases as knowledge engineering and knowledge sources. It seems mostly a way of referring to whatever it is that a representation has. If a system has (and can use) a data structure which can be said to represent something (an object, a procedure, . . . whatever), then the system itself can also be said to have knowledge, namely the knowledge embodied in that representation about that thing.

2.2. Why is there a problem?

This seems to be a reasonable picture, which is serving us well. Why then is there a problem? Let me assemble some contrary indicators from our current scene.

A first indicator comes from our continually giving to representation a somewhat magical role.[2] It is a cliche of AI that representation is the *real* issue we face. Though we have programs that search, it is said, we do not have programs that determine their own representations or invent new representations. There is of course some substance to such statments. What is indicative of underlying difficulties is our inclination to treat representation like a *homunculus*, as the locus of *real* intelligence.

A good example is our fascination with problems such as the *mutilated checkboard* problem [24]. The task is to cover a checkboard with two-square dominoes. This is easy enough to do with the regular board and clearly impossible to do if a single square is removed, say from the upper right corner. The problem is to do it on a (mutilated) board which has two squares removed, one from each of two opposite corners. This task also turns out to be impossible. The actual task, then, is to show the impossibility. This goes from apparently intractable combinatorially, if the task is represented as all ways of laying down dominoes, to transparently easy, if the task is represented as just the *numbers* of black and white squares that remain to be covered. Now, the crux for AI is that no one has been able to formulate in a reasonable way the problem of finding the good representation, so it can be tackled by an AI system. By implication—so goes this view—the capability to invent such appropriate representations requires intelligence of some new and different kind.

A second indicator is the great theorem-proving controversy of the late sixties and early seventies. Everyone in AI has some knowledge of it, no doubt, for its residue is still very much with us. It needs only brief recounting.

Early work in theorem proving programs for quantified logics culminated in 1965 with Alan Robinson's development of a machine-oriented formulation of first-order logic called *Resolution* [32]. There followed an immensely produc-

[2]Representation is not the only aspect of intelligent systems that has a magical quality; learning is another. But that is a different story for a different time.

tive period of exploration of resolution-based theorem-proving. This was fueled, not only by technical advances, which occurred rapidly and on a broad front [15], but also by the view that we had a general purpose reasoning engine in hand and that doing logic (and doing it well) was a foundation stone of all intelligent action. Within about five years, however, it became clear that this basic engine was not going to be powerful enough to prove theorems that are hard on a human scale, or to move beyond logic to mathematics, or to serve other sorts of problem solving, such as robot planning.

A reaction set in, whose slogan was "uniform procedures will not work". This reaction itself had an immensely positive outcome in driving forward the development of the second generation of AI languages: Planner, Microplanner, QA4, Conniver, POP2, etc. [6]. These unified some of the basic mechanisms in problem solving—goals, search, pattern matching, and global data bases—into a programming language framework, with its attendant gains of involution.

However, this reaction also had a negative residue, which still exists today, well after these new AI languages have come and mostly gone, leaving their own lessons. The residue in its most stereotyped form is that logic is a bad thing for AI. The stereotype is not usually met with in pure form, of course. But the mat of opinion is woven from a series of strands that amount to as much: Uniform proof techniques have been proven grossly inadequate; the failure of resolution theorem proving implicates logic generally; logic is permeated with a static view; and logic does not permit control. Any doubts about the reality of this residual reaction can be stilled by reading Pat Hayes's attempt to counteract it in [12].

A third indicator is the recent SIGART "Special issue of knowledge representation" [7]. This consisted of the answers (plus analysis) to an elaborate questionnaire developed by Ron Brachman of BBN and Brian Smith of MIT, which was sent to the AI research community working on knowledge representation. In practice, this meant work in natural language, semantic nets, logical formalisms for representing knowledge, and in the third generation of programming and representation systems, such as AIMDS, KRL, and KL-ONE. The questionnaire not only covered the basic demography of the projects and systems, but also the position of the respondent (and his system) on many critical issues of representation—quantification, quotation, self-description, evaluation vs. reference-finding, and so on.

The responses were massive, thoughtful and thorough, which was impressive given that the questionnaire took well over an hour just to read, and that answers were of the order of ten single-spaced pages. A substantial fraction of the field received coverage in the 80 odd returns, since many of them represented entire projects. Although the questionnaire left much to be desired in terms of the precision of its questions, the *Special Issue* still provides an extremely interesting glimpse of how AI sees the issues of knowledge representation.

The main result was overwhelming diversity—a veritable jungle of opinions. There was no consensus on any question of substance. Brachman and Smith themselves highlight this throughout the issue, for it came as a major surprise to them. Many (but of course not all!) respondents themselves felt the same way. As one said, "Standard practice in the representation of knowledge is the scandal of AI".

What is so overwhelming about the diversity is that it defies characterization. The role of logic and theorem proving, just described above, are in evidence, but there is much else besides. There is no tidy space of underlying issues in which respondents, hence the field, can be plotted to reveal a pattern of concerns or issues. Not that Brachman and Smith could see. Not that this reader could see.

2.3. A formulation of the problem

These three items—mystification of the role of representation, the residue of the theorem-proving controversy, and the conflicting webwork of opinions on knowledge representation—are sufficient to indicate that our views on representation and knowledge are not in satisfactory shape. However, they hardly indicate a crisis, much less a scandal. At least not to me. Science easily inhabits periods of diversity; it tolerates bad lessons from the past in concert with good ones. The chief signal these three send is that we must redouble our efforts to bring some clarity to the area. Work on knowledge and representation should be a priority item on the agenda of our science.

No one should have any illusions that clarity and progress will be easy to achieve. The diversity that is represented in the SIGART Special Issue is highly articulate and often highly principled. Viewed from afar, any attempt to clarify the issues is simply one more entry into the cacophony—possibly treble, possibly bass, but in any case a note whose first effect will be to increase dissonance, not diminish it.

Actually, these indicators send an ambiguous signal. An alternative view of such situations in science is that effort is premature. Only muddling can happen for the next while—until more evidence accumulates or conceptions ripen elsewhere in AI to make evident patterns that now seem only one possibility among many. Work should be left to those already committed to the area; the rest of us should make progress where progress can clearly be made.

Still, though not compelled, I wish to have a go at this problem.

I wish to focus attention on the question: *What is knowledge*? In fact, knowledge gets very little play in the three indicators just presented. *Representation* occupies center stage, with *logic* in the main supporting role. I could claim that this is already the key—that the conception of knowledge is logically prior to that of representation, and until a clear conception of the former exists, the latter will remain confused. In fact, this is not so. Knowledge is simply one particular

entry point to the whole tangled knot. Ultimately, clarity will be attained on all these notions together. The path through which this is achieved will be grist for those interested in the history of science, but is unlikely to affect our final understanding.

To reiterate: What is the nature of knowledge? How is it related to representation? What is it that a system has, when it has knowledge? Are we simply dealing with redundant terminology, not unusual in natural language, which is better replaced by building on the notions of data structures, interpreters, models (in the strict sense used in logic), and the like? I think not. I think knowledge is a distinct notion, with its own part to play in the nature of intelligence.

2.4. The solution follows from practice

Before starting on matters of substance, I wish to make a methodological point. The solution I will propose follows from the practice of AI. Although the formulation I present may have some novelty, it should be basically familiar to you, for it arises from how we in AI treat knowledge in our work with intelligent systems. Thus, your reaction may (perhaps even should) be "But that is just the way I have been thinking about knowledge all along. What is this man giving me?" On the first part, you are right. This is indeed the way AI has come to use the concept of knowledge. However, this is not the way the rest of the world uses the concept. On the second part, what I am giving you is a directive that your practice represents an important source of knowledge about the nature of intelligent systems. It is to be taken seriously.

This point can use expansion. Every science develops its own ways of finding out about its subject matter. These get tidied up in meta-models about scientific activity, eg, the so called *scientific method* in the experimental sciences. But these are only models; in reality, there is immense diversity in how scientific progress is made.

For instance, in computer science many fundamental conceptual advances occur by (scientifically) uncontrolled experiments in our own style of computing.[3] Three excellent examples are the developments of time-sharing, packet switched networks, and locally-networked personal computing. These are major *conceptual* advances that have broadened our view of the nature of computing. Their primary validation is entirely informal. Scientific activity of a more traditional kind certainly takes place—theoretical development with careful controlled testing and evaluation of results. But it happens on the

[3]Computer science is not unique in having modes of progress that don't fit easily into the standard frames. In the heyday of paleontology, major conceptual advances occurred by stumbling across the bones of immense beasties. Neither controlled experimentation nor theoretical prediction played appreciable roles.

details, not on the main conceptions. Not everyone understands the necessary scientific role of such experiments in computational living, nor that standard experimental techniques cannot provide the same information. How else to explain, for example, the calls for controlled experimental validation that speech understanding will be useful to computer science? When that experiment of style is finally performed there will be no doubt at all. No standard experiment will be necessary. Indeed, none could have sufficed.

As an example related to the present paper, I have spent some effort recently in describing what Herb Simon and I have called the "*Physical symbol system hypothesis*" [26, 29]. This hypothesis identifies a class of systems as embodying the essential nature of symbols and as being the necessary and sufficient condition for a generally intelligent agent. Symbol systems turn out to be universal computational systems, viewed from a different angle. For my point here, the important feature of this hypothesis is that it grew out of the practice in AI—out of the development of list processing languages and Lisp, and out of the structure adopted in one AI program after another. We in AI were led to an adequate notion of a symbol by our practice. In the standard catechism of science, this is not how great ideas develop. Major ideas occur because great scientists discover (or invent) them, introducing them to the scientific community for testing and elaboration. But here, working scientists have evolved a new major scientific concept, under partial and alternative guises. Only gradually has it acquired its proper name.

The notions of knowledge and representation about to be presented also grow out of our practice. At least, so I assert. That does not give them immunity from criticism, for in listening for these lessons I may have a tin ear. But in so far as they are wanting, the solution lies in more practice and more attention to what emerges there as pragmatically successful. Of course, the message will be distorted by many things, e.g., peculiar twists in the evolution of computer science hardware and software, our own limitations of view, etc. But our practice remains a source of knowledge that cannot be obtained from anywhere else. Indeed, AI as a field is committed to it. If it is fundamentally flawed, that will just be too bad for us. Then, other paths will have to be found from elsewhere to discover the nature of intelligence.

3. The Knowledge Level

I am about to propose the existence of something called the *knowledge level*, within which knowledge is to be defined. To state this clearly, requires first reviewing the notion of computer systems levels.

3.1. Computer system levels

Fig. 2 shows the standard hierarchy, familiar to everyone in computer science. Conventionally, it starts at the bottom with the *device level*, then up to the

circuit level, then the *logic level*, with its two sublevels, *combinatorial and sequential circuits*, and the *register-transfer level*, then the *program level* (referred to also as the *symbolic level*) and finally, at the top, the *configuration level* (also called the *PMS* or *Processor-Memory-Switch level*). We have drawn the configuration level to one side, since it lies directly above both the symbol level and the register-transfer level.

The notion of levels occurs repeatedly throughout science and philosophy, with varying degrees of utility and precision. In computer science, the notion is quite precise and highly operational. Table 1 summarizes its essential attributes. A level consists of a *medium* that is to be processed, *components* that provide primitive processing, *laws of composition* that permit components to be assembled into *systems*, and *laws of behavior* that determine how system behavior depends on the component behavior and the structure of the system. There are many variant instantiations of a given level, e.g., many programming systems and machine languages and many register-transfer systems.[4]

Each level is defined in two ways. First, it can be defined autonomously, without reference to any other level. To an amazing degree, programmers need not know logic circuits, logic designers need not know electrical circuits, managers can operate at the configuration level with no knowledge of programming, and so forth. Second, each level can be reduced to the level below. Each aspect of a level—medium, components, laws of composition and behavior—can be defined in terms of systems at the level next below. The *architecture* is the name we give to the register-transfer level system that defines a symbol (programming) level, creating a machine language and making it run as described in the programmers manual for the machine. Neither of these two definitions of a level is the more fundamental. It is essential that they both exist and agree.

Some intricate relations exist between and within levels. Any instantiation of a level can be used to create *any* instantiation of the next higher level. Within each level, systems hierarchies are possible, as in the subroutine hierarchy at the programming level. Normally, these do not add anything special in terms of the computer system hierarchy itself. However, as we all know, at the program level it is possible to construct any instantiation within any other instantiation (mudulo some rigidity in encoding one data structure into another), as in creating new programming languages.

There is no need to spin out the details of each level. We live with them every day and they are the stuff of architecture textbooks [1], machine manuals and digital component catalogues, not research papers. However, it is noteworthy how radically the levels differ. The medium changes from electrons and magnetic domains at the device level, to current and voltage at the circuit level,

[4]Though currently dominated by electrical circuits, variant circuit level instantiations also exist, e.g., fluidic circuits.

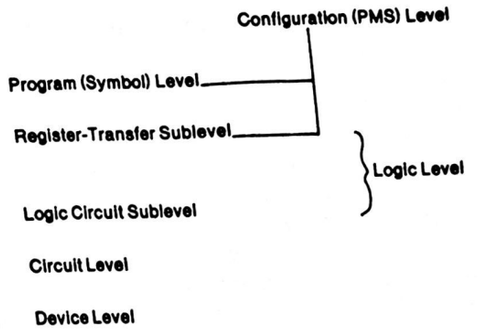

FIG. 2. Computer system levels.

to bits at the logic level (either single bits at the logic circuit level or bit vectors at the register-transfer level), to symbolic expressions at the symbol level, to amounts of data (measured in data bits) at the configuration level. System characteristics change from continuous to discrete processing, from parallel to serial operation, and so on.

Despite this variety, all levels share some common features. Four of these, though transparently obvious, are important to us.

Point 1. Specification of a system at a level always determines completely a definite behavior for the system at that level (given initial and boundary conditions).

Point 2. The behavior of the total system results from the local effects of each component of the system processing the medium at its inputs to produce its outputs.

Point 3. The immense variety of behavior is obtained by system structure, i.e., by the variety of ways of assembling a small number of component types (though perhaps a large number of instances of each type).

TABLE 1. Defining aspects of a computer system level

Aspects	Register-transfer level	Symbol level
Systems	Digital systems	Computers
Medium	Bit vectors	Symbols, expressions
Components	Registers	Memories
	Functional units	Operations
Composition laws	Transfer path	Designation, association
Behavior laws	Logical operations	Sequential interpretation

Point 4. The medium is realized by state-like properties of matter, which remain passive until changed by the components.

Computer systems levels are not simply levels of abstraction. That a system has a description at a given level does not necessarily imply it has a description at higher levels. There is no way to abstract from an arbitrary electronic circuit to obtain a logic-level system. There is no way to abstract from an arbitrary register-transfer system to obtain a symbol-level system. This contrasts with many types of abstraction which can be uniformly applied, and thus have a certain optional character (as in abstracting away from the visual appearance of objects to their masses). Each computer system level is a *specialization* of the class of systems capable of being described at the next lower level. Thus, it is a priori open whether a given level has any physical realizations.

In fact, computer systems at all levels are realizable, reflecting indirectly the structure of the physical world. But more holds than this. Computer systems levels are realized by *technologies*. The notion of a technology has not received the conceptual attention it deserves. But roughly, given a specification of a particular system at a level, it is possible to construct by *routine* means a physical system that realizes that specification. Thus, systems can be obtained to specification within limits of time and cost. It is not possible to invent arbitrarily additional computer system levels that nestle between existing levels. Potential levels do not become technologies, just by being thought up. Nature has a say in whether a technology can exist.

Computer system levels are *approximations*. All of the above notions are realized in the real world only to various degrees. Errors at lower levels propagate to higher ones, producing behavior that is not explicable within the higher level itself. Technologies are imperfect, with constraints that limit the size and complexity of systems that can actually be fabricated. These constraints are often captured in design rules (e.g., fan-out limits, stack-depth limits, etc), which transform system design from routine to problem solving. If the complexities become too great, the means of system creation no longer constitute a technology, but an arena of creative invention.

We live quite confortably with imperfect system levels, especially at the extremes of the hierarchy. At the bottom, the device level is not complete, being used only to devise components at the circuit level. Likewise, at the top, the configuration level is incomplete, not providing a full set of behavioral laws. In fact, it is more nearly a pure level of abstraction than a true system level. This accounts for both symbol level and register transfer level systems having configuration (PMS) level abstractions (see [2] for a PMS approach to the register-transfer level).

These levels provide ways of describing computer systems; they do not provide ways of describing their environments. This may seem somewhat unsatisfactory, because a level does not then provide a general closed description of an entire universe, which is what we generally expect (and get) from a level of scientific description in physics or chemistry. However, the situation is understandable enough. System design and analysis requires only that the interface between the environment and the system (i.e., the inner side of the transducers) be adequately described in terms of each level, e.g., as electrical signals, bits, symbols or whatever. Almost never does the universe of system plus environment have to be modeled in toto, with the structure and dynamics of the environment described in the same terms as the system itself. Indeed, in general no such description of the environment in the terms of a given computer level exists. For instance, no register-transfer level description exists of the airplane in which an airborne computer resides. Computer system levels describe the internal structure of a particular class of systems, not the structure of a total world.

To sum up, computer system levels are a reflection of the nature of the physical world. They are not just a point of view that exists solely in the eye of the beholder. This reality comes from computer system levels being genuine specializations, rather than being just abstractions that can be applied uniformly.

3.2. A new level

I now propose that there does exist yet another system level, which I will call the *knowledge level*. It is a true systems level, in the sense we have just reviewed. The thrust of this paper is that distinguishing this level leads to a simple and satisfactory view of knowledge and representation. It dissolves some of the difficulties and confusions we have about this aspect of artificial intelligence.

A quick overview of the knowledge level, with an indication of some of its immediate consequences, is useful before entering into details.

The system at the knowledge level is the *agent*. The components at the knowledge level are *goals*, *actions*, and *bodies*. Thus, an agent is composed of a set of actions, a set of goals and a body. The medium at the knowledge level is *knowledge* (as might be suspected). Thus, the agent processes its knowledge to determine the actions to take. Finally, the behavior law is the *principle of rationality*: Actions are selected to attain the agent's goals.

To treat a system at the knowledge level is to treat it as having some knowledge and some goals, and believing it will do whatever is within its power to attain its goals, in so far as its knowledge indicates. For example:

- "She knows where this restaurant is and said she'd meet me here. I don't know why she hasn't arrived."
- "Sure, he'll fix it. He knows about cars."
- "If you know that 2+2 = 4, why did you write 5?"

The knowledge level sits in the hierarchy of systems levels immediately above the symbol level, as Fig. 3 shows. Its components (actions, goals, body)

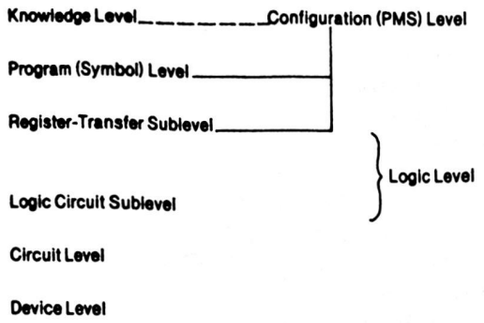

FIG. 3. New version of computer system levels.

and its medium (knowledge) can be defined in terms of systems at the symbol level, just as any level can be defined by systems at the level one below. The knowledge level has been placed side by side with the configuration level. The gross anatomical description of a knowledge-level system is simply (and only) that the agent has as parts bodies of knowledge, goals and actions. They are all connected together in that they enter into the determination of what actions to take. This structure carries essentially no information; the specification at the knowledge level is provided entirely by the content of the knowledge and the goals, not by any structural way they are connected together. In effect, the knowledge level can be taken to have a degenerate configuration level.

As is true of any level, although the knowledge level can be constructed from the level below (i.e., the symbol level), it also has an autonomous formulation as an independent level. Thus, knowledge can be defined independent of the symbol level, but can also be reduced to symbol systems.

As the Fig. 3 makes evident, the knowledge level is a separate level from the symbol level. Relative to the existing view of the computer systems hierarchy, the symbol level has been split in two, one aspect remaining as the symbol level proper and another aspect becoming the knowledge level. The description of a system as exhibiting intelligent behavior still requires both levels, as we shall see. Intelligent systems are not to be described exclusively in terms of the knowledge level.

To repeat the final remark of the prior section: Computer system levels really exist, as much as anything exists. They are not just a point of view. Thus, to claim that the knowledge level exists is to make a scientific claim, which can range from dead wrong to slightly askew, in the manner of all scientific claims. Thus, the matter needs to be put as a hypothesis:

The Knowledge Level Hypothesis. *There exists a distinct computer systems level, lying immediately above the symbol level, which is characterized by knowledge as the medium and the principle of rationality as the law of behavior.*

Some preliminary feeling for the nature of knowledge according to this hypothesis can be gained from the following remarks.

- Knowledge is intimately linked with rationality. Systems of which rationality can be posited can be said to have knowledge. It is unclear in what sense other systems can be said to have knowledge.
- Knowledge is a competence-like notion, being a potential for generating action.[5]
- The knowledge level is an approximation. Nothing guarantees how much of a system's behavior can be viewed as occurring at the knowledge level. Although extremely useful, the approximation is quite imperfect, not just in degree but in scope.
- Representations exist at the symbol level, being systems (data structures and processes) that realize a body of knowledge at the knowledge level.
- Knowledge serves as the specification of what a symbol structure should be able to do.
- Logics are simply one class of representations among many, though uniquely fitted to the analysis of knowledge and representation.

4. The Details of the Knowledge Level

We begin by defining the knowledge level autonomously, i.e., independently of lower system levels. Table 1 lists what is required, though we will take up the various aspects in a different order: first, the structure, consisting of the system, components and laws for composing systems; second, the laws of behavior (the law of rationality); and third, the medium (knowledge). After this we will describe how the knowledge level reduces to the symbol level.

Against the background of the common features of the familiar computer system levels listed earlier (Section 3.1), there will be four surprises in how the knowledge level is defined. These will stretch the notion of system level somewhat, but will not break it.

4.1. The structure of the knowledge level

An agent (the system at the knowledge level), has an extremely simple structure, so simple there is no need even to picture it.

First, the agent has some *physical body* with which it can act in the environment (and be acted upon). We talk about the body as if it consists of a set of *actions*, but that is only for simplicity. It can be an arbitrary physical system with arbitrary modes of interaction with its environment. Though this

[5]How almost interchangeable the two notions might be can be seen from a quotation from Chomsky [9, p. 315]: "In the past I have tried to avoid, or perhaps evade the problem of explicating the notion 'knowledge of language' by using an invented technical term, namely the term 'competence' in place of 'knowledge'".

body can be arbitrarily complex, its complexity lies *external* to the system described at the knowledge level, which simply has the power to evoke the behavior of this physical system.

Second, the agent has a *body* of knowledge. This body is like a memory. Whatever the agent knows at some time, it continues to know. Actions can add knowledge to the existing body of knowledge. However, in terms of structure, a body of knowledge is extremely simple compared to a memory, as defined at lower computer system levels. There are no structural constraints to the knowledge in a body, either in capacity (i.e., the amount of knowledge) or in how the knowledge is held in the body. Indeed, there is no notion of how knowledge is held (*encoding* is a notion at the symbol level, not knowledge level). Also, there are no well-defined structural properties associated with access and augmentation. Thus, it seems preferable to avoid calling the body of knowledge a memory. In fact, referring to a 'body of knowledge', rather than just to 'knowledge', is hardly more than a manner of speaking, since this body has no function except to be the physical component which has the knowledge.

Third, and finally, the agent has a *set of goals*. A goal is a body of knowledge of a state of affairs in the environment. Goals are structurally distinguished from the main body of knowledge. This permits them to enter into the behavior of the agent in a distinct way, namely, that which the organism strives to realize. But, except for this distinction, goal components are structurally identical to bodies of knowledge. Relationships exist between goals, of course, but these are not realized in the structure of the system, but in knowledge.

There are no laws of composition for building a knowledge level system out of these components. An agent always has just these components. They all enter directly into the laws of behavior. There is no way to build up complex agents from them.

This complete absence of significant structure in the agent is the first surprise, running counter to the common feature at all levels that variety of behavior is realized by variety of system structure (Section 3.1, Point 3). This is not fortuitous, but is an essential feature of the knowledge level. The focus for determining the behavior of the system rests with the knowledge, i.e., with the *content* of what is known. The internal structure of the system is defined exactly so that nothing need be known about it to predict the agent's behavior. The behavior is to depend only what the agent knows, what it wants and what means it has for interacting physically with the environment.

4.2. The principle of rationality

The behavioral law that governs an agent, and permits prediction of its behavior, is the rational principle that knowledge will be used in the service of goals.[6] This can be formulated more precisely as follows:

Principle of rationality. *If an agent has knowledge that one of its actions will lead to one of its goals, then the agent will select that action.*

This principle asserts a connection between knowledge and goals, on the one hand, and the selection of actions on the other, without specification of any mechanism through which this connection is made. It connects all the components of the agent together directly.

This direct determination of behavior by a global principle is the second surprise, running counter to the common feature at all levels that behavior is determined bottom-up through the local processing of components (Section 3.1, Point 2). Such global principles are not incompatible with systems whose behavior is also describable by mechanistic causal laws, as testified by various global principles in physics, e.g., Fermat's principle of least time in geometrical optics, or the principle of least effort in mechanics. The principles in physics are usually optimization (i.e., extremum) principles. However, the principle of rationality does not have built into it any notion of optimal or best, only the automatic connection of actions to goals according to knowledge.

Under certain simple conditions, the principle as stated permits the calculation of a system's trajectory, given the requisite initial and boundary conditions (i.e., goals, actions, initial knowledge, and acquired knowledge as determined by actions). However, the principle is not sufficient to determine the behavior in many situations. Some of these situations can be covered by adding *auxiliary* principles. Thus, we can think of an *extended principle of rationality*, building out from the *central* or *main* principle, given above.

To formulate additional principles, the phrase *selecting an action* is taken to mean that the action becomes a member of a candidate set of actions, the *selected set*, rather than being the action that actually occurs. When all principles are taken into account, if only one action remains in the selected set, that action actually taken is determined; if several candidates remain, then the action actually taken is limited to these possibilities. An action can actually be taken only if it is physically possible, given the situation of the agent's body and the resources available. Such limits to action will affect the actions selected only if the agent has knowledge of them.

The main principle is silent about what happens if the principle applies for more than one action for a given goal. This can be covered by the following auxiliary principle.

[6] This principle is not intended to be directly responsive to all the extensive philosophical discussion on rationality, e.g., the notion that rationality implies the ability for an agent to give *reasons* for what it does.

Equipotence of acceptable actions. *For given knowledge, if action A_1 and action A_2 both lead to goal G, then both actions are selected.*[7]

This principle simply asserts that all ways of attaining the goal are equally acceptable from the standpoint of the goal itself. There is no implicit optimality principle that selects among such candidates.

The main principle is also silent about what happens if the principle applies to several goals in a given situation. A simple auxiliary principle is the following.

Preference of joint goal satisfaction. *For given knowledge, if goal G_1 has the set of selected actions $\{A_{1,i}\}$ and goal G_2 has the set of selected actions $\{A_{2,j}\}$, then the effective set of selected actions is the intersection of $\{A_{1,i}\}$ and $\{A_{2,j}\}$.*

It is better to achieve both of two goals than either alone. This principle determines behavior in many otherwise ambiguous situations. If the agent has general goals of minimizing effort, minimizing cost, or doing things in a simple way, these general goals select out a specific action from a set of otherwise equipotent task-specific actions.

However, this principle of joint satisfaction still goes only a little ways further towards obtaining a principle that will determine behavior in all situations. What if the intersection of selected action sets is null? What if there are several mutually exclusive actions leading to several goals? What if the attainment of two goals is mutually exclusive, no matter through what actions attained? These types of situations too can be dealt with by extending the concept of a goal to include *goal preferences*, that is, specific preferences for one state of the affairs over another that are not grounded in knowledge of how these states differentially aid in reaching some common superordinate goal.

Even this extended principle of rationality does not cover all situations. The central principle refers to an action leading to a goal. In the real world it is often not clear whether an action will attain a specific goal. The difficulty is not just one of the possibility of error. The actual outcome may be truly probabilistic, or the action may be only the first step in a sequence that depends on other agents' moves, etc. Again, extensions exist to deal with uncertainty and risk, such as adopting expected value calculations and principles of minimizing maximum loss.

In proposing each of these solutions, I am not inventing anything. Rather, this growing extension of rational behavior moves along a well-explored path, created mostly by modern day game theory, econometrics and decision theory

[7]For simplicity, in this principle and others, no explicit mention is made of the agent whose goals, knowledge and actions are under discussion.

[16,39]. It need not be retraced here. The exhibition of the first few elementary extensions can serve to indicate the total development to be taken in search of a principle of rationality that always determines the action to be taken.

Complete retracing is not necessary because the path does not lead, even ultimately, to the desired principle. No such principle exists.[8] Given an agent in the real world, there is no guarantee at all that his knowledge will determine which actions realize which goals. Indeed, there is no guarantee even that the difficulty resides in the incompleteness of the agent's knowledge. There need not exist any state of knowledge that would determine the action.

The point can be brought home by recalling Frank Stockton's famous short story, "The lady or the tiger?" [38]. The upshot has the lover of a beautiful princess caught by the king. He now faces the traditional ordeal of judgment in this ancient and barbaric land: In the public arena he must choose to open one of two utterly indistinguishable doors. Behind one is a ferocious tiger and death; behind the other a lovely lady, life and marriage. Guilt or innocence is determined by fate. The princess alone, spurred by her love, finds out the secret of the doors on the night before the trial. To her consternation she finds that the lady behind the door is to be her chief rival in loveliness. On the judgment day, from her seat in the arena beside the king, she indicates by a barely perceptible nod which door her lover should open. He, in unhesitating faithfulness, goes directly to that door. The story ends with a question. Did the princess send her lover to the lady or the tiger?

Our knowledge-level model of the princess, even if it were to include her complete knowledge, would not tell us which she chose. But the failure of determinancy is not the model's. Nothing says that multiple goals need be compatible, nor that the incompatibility be resolvable by any higher principles. The dilemma belongs to the princess, not to the scientist attempting to formulate an adequate concept of the knowledge level. That she resolved it is clear, but that her behavior in doing so was describable at the knowledge level does not follow.

This failure to determine behavior uniquely is the third surprise, running counter to the common feature at all levels that a system is a determinate machine (Section 3.1, Point 1). A complete description of a system at the program, logic or circuit level yields the trajectory of the system's behavior over time, given initial and boundary conditions. This is taken to be one of its important properties, consistent with being the description of a deterministic (macro) physical system. Yet, *radical* incompleteness characterizes the knowl-

[8]An adequate critical recounting of this intellectual development is not possible here. Formal systems (utility fields) can be constructed that appear to have the right property. But they work, not by connecting actions with goals via knowledge of the task environment, but by positing of the agent a complete set of goals (actually, preferences) that directly specify all action selections over all combinations of task states (or probabilistic options over task states). Such a move actually abandons the enterprise.

edge level. Sometimes behavior can be predicted by the knowledge level description; often it cannot. The incompleteness is not just a failure in certain special situations or in some small departures. The term *radical* is used to indicate that entire ranges of behavior may not be describable at the knowledge level, but only in terms systems at a lower level (namely, the symbolic level). However, the necessity of accepting this incompleteness is an essential aspect of this level.

4.3. The nature of knowledge

The ground is now laid to provide the definition of knowledge. As formulated so far, knowledge is defined to be the medium at the knowledge level, something to be processed according to the principle of rationality to yield behavior. We wish to elevate this into a complete definition:

Knowledge. *Whatever can be ascribed to an agent, such that its behavior can be computed according to the principle of rationality.*

Knowledge is to be characterized entirely *functionally*, in terms of what it does, not *structurally*, in terms of physical objects with particular properties and relations. This still leaves open the requirement for a physical structure for knowledge that can fill the functional role. In fact, that key role is never filled directly. Instead, it is filled only indirectly and approximately by *symbol systems* at the next lower level. These are total systems, not just symbol structures. Thus, knowledge, though a medium, is embodied in no medium-like passive physical structure.

This failure to have the medium at the knowledge level be a state-like physical structure is the fourth surprise, running counter to the common feature at all levels of a passive medium (Section 3.1, Point 4). Again, it is an essential feature of the knowledge level, giving knowledge its special abstract and competence-like character. One can see on the blackboard a symbolic expression (say a list of theorem names). Though the actual seeing is a tad harder, yet there is still no difficulty seeing this same expression residing in a definite set of memory cells in a computer. The same is true at lower levels—the bit vectors at the register-transfer level—marked on the blackboard or residing in a register as voltages. Moreover, the medium at one level plus additional static structure defines the medium at the next level up. The bit plus its organization into registers provides the bit-vector; collections of bit vectors plus functional specialization to link fields, type fields, etc., defines the symbolic expression. All this fails at the knowledge level. The knowledge cannot so easily be seen, only imagined as the result of interpretive processes operating on symbolic expressions. Moreover, knowledge is not just a collection of symbolic expressions plus some static organization; it requires both processes and data structures.

The definition above may seem like a reverse, even perverse, way of defining knowledge. To understand it—to see how it works, why it is necessary, why it is useful, and why it is effective—we need to back off and examine the situation in which knowledge is used.

How it works

Fig. 4 shows the situation, which involves an *observer* and an *agent*. The observer treats the agent as a system at the knowledge level, i.e., ascribes knowledge and goals to it. Thus, the observer knows the agent's knowledge (K) and goals (G), along with his possible actions and his environment (these latter by direct observation, say). In consequence, the observer can make predictions of the agent's actions using the principle of rationality.

Assume the observer ascribes correctly, i.e., the agent behaves as if he has knowledge K and goals G. What the agent really has is a symbol system, S, that permits it to carry out the calculations of what actions it will take, because it has K and G (with the given actions in the given environment).

The observer is itself an agent, i.e., is describable at the knowledge level. There is no way in this theory of knowledge to have a system that is not an agent have knowledge or ascribe knowledge to another system. Hence, the observer has all this knowledge (i.e., K', consisting of knowledge K, knowledge that K is the agent's knowledge, knowledge G, knowledge that these are the agent's goals, knowledge of the agent's actions etc.). But what the observer really has, of course, is a symbol system, S', that lets it calculate actions on the basis of K, goals, etc., i.e., calculate what the agent would do if it had K and G.

Thus, as the figure shows, each agent (the observer and the observed) has

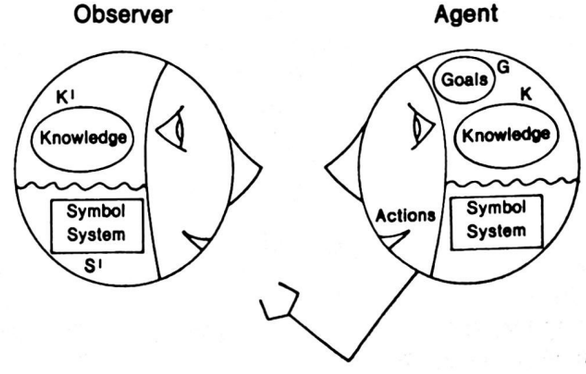

FIG. 4. The situation in which knowledge is used.

knowledge by virtue of a symbol system that provides the ability to act as if it had the knowledge. The total system (i.e., the dyad of the observing and observed agents) runs without there being any physical structure that is the knowledge.

Why it is necessary

Even granting the scheme of Fig. 4 to be possible, why cannot there simply be some physical structure that corresponds to knowledge? Why are not S and S' simply the knowledge? The diagram seems similar to what could be drawn between any two adjacent system levels, and in these other cases corresponding structures do exist at each level. For example, the medium at the symbol level (the symbolic expressions) corresponds to the medium at the register-transfer level (the bit vectors). The higher medium is simply a specialized use of the lower medium, but both are physical structures. The same holds when we descend from the register-transfer level (bit vectors) to the logic circuit level (single bits); the relationship is simply one of aggregation. Descending to the circuit level again involves specialization in which some feature of the circuit level medium (e.g. voltage) is taken to define the bit (e.g. a high and a low voltage level).

The answer in a nutshell is that knowledge of the world cannot be captured in a finite structure. The world is too rich and agents have too great a capability for responding.[9] Knowledge is about the world. Insofar as an agent can select actions based on some truth about the world, any candidate structure for knowledge must contain that truth. Thus knowledge as a structure must contain at least as much variety as the set of all truths (i.e. propositions) that the agent can respond to.

A representation of any fragment of the world (whether abstract or concrete), reveals immediately that the knowledge is not finite. Consider our old friend, the chess position. How many propositions can be stated about the position? For a given agent, only propositions need by considered that could materially affect an action of the agent relative to its goal. But given a reasonably intelligent agent who desires to win, the list of aspects of the position is unbounded. Can the Queen take the Pawn? Can it take the Pawn next move? Can it if the Bishop moves? Can it if the Rook is ever taken? Is a pin possible that will unblock the Rook attacker before a King side attack can be mounted? And so on.

A seemingly appropriate objection is: (1) the agent is finite so can't actually have an unbounded anything; and (2) the chess position is also just a finite structure. However, the objection fails, because it is not possible to state from afar (i.e. from the observer's viewpoint) a bound on the set of propositions about the position that will be available. Of course, if the observer had a model

[9] Agents with finite knowledge are certainly possible, but would be extraordinarily limited.

of the agent at the symbolic process level, then a (possibly accurate) prediction might be made. But the knowledge level is exactly the level that abstracts away from symbolic processes. Indeed, the way the observer determines what the agent might know is to consider what he (the observer) can find out. And this he cannot determine in advance.

The situation here is not really strange. The underlying phenomenon is the generative ability of computational systems, which involves an active process working on an initially given data structure. Knowledge is the posited extensive form of all that can be obtained potentially from the process. This potential is unbounded when the details of processing are unknown and the gap is closed by assuming (from the principle of rationality) the processing to be whatever makes the correct selection. Of course, there are limits to this processing, namely, that it cannot go beyond the knowledge given.

What the computational system generates are selections of actions for goals, conditioned on states of the world. Each such basic means-ends relation may be taken as an *element* of knowledge. To have the knowledge available in extension would be to have all these possible knowledge elements for all the goals, actions and states of the world discriminable to the agent at the given moment. The knowledge could then be thought of as a giant table full of these knowledge elements, but it would have to be an infinite table. Consequently, this knowledge (i.e., these elements) can only be created dynamically in time. If generated by some simple procedure, only relatively uninteresting knowledge can be found. Interesting knowledge requires generating only what is relevant to the task at hand, i.e. generating intelligently.

Why it is useful

The knowledge level permits predicting and understanding behavior without having an operational model of the processing that is actually being done by the agent. The utility of such a level would seem clear, given the widespread need in life's affairs for distal prediction, and also the paucity of knowledge about the internal workings of humans. (And animals, some of which are usefully described at the knowledge level.) The utility is also clear in designing AI systems, where the internal mechanisms are still to be specified. To the extent that AI systems successfully approximate rational agents, it is also useful for predicting and understanding them. Indeed, the usefulness extends beyond AI systems to all computer programs.

Prediction of behavior is not possible without knowing something about the agent. However, what is known is not the processing structure, but the agent's knowledge of its external environment, which is also accessible to the observer directly (to some extent). The agent's goals must also be known, of course, and these certainly are internal to the agent. But they are relatively stable characteristics that can be inferred from behavior and (for human adults) can sometimes be conveyed by language. One way of viewing the knowledge level is as the

attempt to build as good a model of an agent's behavior as possible based on information external to the agent, hence permitting distal prediction. This standpoint makes understandable why such a level might exist even though it is radically incomplete. If such incompleteness is the best that can be done, it must be tolerated.

Why it is effective

The knowledge level, being indirect and abstract, might be thought to be excessively complex, even abstruse. It could hardly have arisen naturally. On the contrary, given the situation of Fig. 4, it arises as day follows night. The knowledge attributed by the observer to the agent is knowledge about the external world. If the observer takes to itself the role of the other (i.e. the agent), assuming the agent's goals and attending to the common external environment, then the actions it determines for itself will be those that the agent should take. Its own computational mechanisms (i.e. its symbolic system) will produce the predictions that flow from attributing the appropriate knowledge to the agent.

This scheme works for the observer without requiring the development of any explicit theory of the agent or the construction of any computational model. To be more accurate, all that is needed is a single general purpose computational mechanism; namely, creating an *embedding context* that posits the agent's goals and symbolizes (rather than executes) the resulting actions.[10] Thus, *simulation* turns out to be a central mechanism that enables the knowledge level and makes it useful.

4.4. Solutions to the representation of knowledge

The principle of rationality provides, in effect, a general functional equation for knowledge. The problem for agents is to find systems at the symbol level that are solutions to this functional equation, and hence can serve as representations of knowledge. That, of course, is also our own problem, as scientists of the intelligent. If we wish to study the knowledge of agents we must use representations of that knowledge. Little progress can be made given only abstract characterizations. The solutions that agents have found for their own purposes are also potentially useful solutions for scientific purposes, quite independently of their interest, *because* they are used by agents whose intelligent processing we wish to study.

Knowledge, in the principle of rationality, is defined entirely in terms of the

[10]However, obtaining this is still not a completely trivial cognitive accomplishment, as indicated by the emphasis on egocentrism in the early work of Piaget [31] and the significance of *taking the role of the other* in the work of the social philosopher, George H. Mead [21], who is generally credited with originating the phrase.

environment of the agent, for it is the environment that is the object of the agent's goals, and whose features therefore bear on the way actions can attain goals. This is true even if the agent's goals have to do with the agent itself as a physical system. Therefore, the solutions are ways to say things about the environment, not ways to say things about reasoning, internal information processing states, and the like. (However, control over internal processing does require symbols that designate internal processing states and structures.)

Logics are obvious candidates. They are, exactly, refined means for saying things about environments. Logics certainly provide solutions to the functional equation. One can find many situations in which the agent's knowledge can be characterized by an expression in a logic, and from which one can go through in mechanical detail all the steps implied in the principle, deriving ultimately the actions to take and linking them up via a direct semantics so the actions actually occur. Examples abound. They do not even have to be limited to mathematics and logic, given the work in AI to use predicate logic (e.g., resolution) as the symbol level structures for robot tasks of spatial manipulation and movement, as well as for many other sorts of tasks [30].

A logic is just a representation of knowledge. It is not the knowledge itself, but a structure at the symbol level. If we are given a set of logical expressions, say $\{L_i\}$, of which we are willing to say that the agent "knows $\{L_i\}$", then the knowledge K that we ascribe to the agent is:

> The agent knows all that can be inferred from the conjunction of $\{L_i\}$.

This statment simply expresses for logic what has been set out more generally above. There exists a symbol system in the agent that is able to bring any inference from $\{L_i\}$ to bear to select the actions of the agent as appropriate (i.e., in the services of the agent's goals). If this symbol system uses the clauses themselves as a representation, then presumably the active processes would consist of a theorem prover on the logic, along with sundry heuristic devices to aid the agent in arriving at the implications in time to perform the requisite action selections.

This statement should bring to prominence an important question, which, if not already at the forefront of concern should be.

> Given that a human cannot know all the implications of an (arbitrary) set of axioms, how can such a formulation of knowledge be either correct or useful?

Philosophy has many times explicitly confronted the proposition that knowledge is the logical closure of a set of axioms. It has seemed so obvious. Yet the proposal has always come to grief on the rocky question above. It is trivial to generate counterexamples that show a person cannot possibly know all that is implied. In a field where counterexamples are a primary method for making

progress, this has proved fatal and the proposal has little standing.[11] Yet, the theory of knowledge being presented here embraces that the knowledge to be associated with a conjunction of logical expressions is its logical closure. How can that be?

The answer is straightforward. The knowledge level is only an *approximation*, and a relatively poor one on many occasions—we called it radically incomplete. It is poor for prodicting whether a person remembers a telephone number just looked up. It is poor for predicting what a person knows given a new set of mathematical axioms with only a short time to study them. And so on, through whole meadows of counterexamples. Equally, it is a good approximation in many other cases. It is good for predicting that a person can find his way to the bedroom of his own house, for predicting that a person who knows arithmetic will be able to add a column of numbers. And so on, through much of what is called common sense knowledge.

This *move* to appeal to approximation (as the philosophers are wont to call such proposals) seems weak, because declaring something an approximation seems a general purpose dodge, applicable to dissolving every difficulty, hence clearly dispelling none. However, an essential part of the current proposal is the existence of the second level of approximation, namely, the symbol level. We now have models of the symbol level that describe how information processing agents arrive at actions by means of search—search of problem spaces and search of global data bases—and how they map structures that are representations of given knowledge to structures that are representations of task-state knowledge in order to create representations of solutions. The discovery, development and elaboration of this second level of approximation to describing and predicting the behavior of an intelligent agent has been what AI has been all about in the quarter century of its existence. In sum, given a theory of the symbol level, we can finally see that the knowledge level is just about what seemed obvious all along.

Returning to the search for solutions to the functional equation expressed by the principle of rationality, logics are only one candidate. They are in no way priviledged.[12] There are many other systems (i.e., combinations of symbol structures and processes) that can yield useful solutions. To be useful an observer need only use it to make predictions according to the principle of rationality. If we consider the problem from the point of view of agents, rather

[11]For example, both the beautiful formal treatment of knowledge by Hintikka [13] and the work in AI by Moore [23] continually insist that knowing P and knowing that P implies Q need not lead to knowing Q.

[12]Though the contrary might seem the case. The principle of rationality might seem to presuppose logic, or at least its formulation might. Untangling this knot requires more care than can be spent here. Note only that it is we, the observer, who formulates the principle of rationality (in some representation). Agents only use it; indeed, only approximate it.

than of AI scientists, then the fundamental principle of the observer must be:

> To ascribe to an agent the structure S is to ascribe whatever the observer can know from structure S.

Theories, models, pictures, physical views, remembered scenes, linguistic texts and utterances, etc., etc.: all these are entirely appropriate structures for ascribing knowledge. They are appropriate because for an observer to *have* these structures is also for it to have means (i.e., the symbolic processes) for extracting knowledge from them.

Not only are logics not privileged, there are difficulties with them. One, already mentioned in connection with the resolution controversy, is processing inefficiency. Another is the problem of contradiction. From an inconsistent conjunction of propositions, any proposition follows. Further, in general, the contradiction cannot be detected or extirpated by any finite amount of effort. One response to this latter difficulty takes the form of developing new logics or logic-like representations, such as non-monotonic logics [5]. Another is to treat the logic as only an approximation, with a limited scope, embedding its use in a larger symbol processing system. In any event, the existence of difficulties does not distinguish logics from other candidate representations. It just makes them one of the crowd.

When we turn from the agents themselves and consider representations from the view-point of the AI scientist, many candidates become problems rather than solutions. If the data structures alone are known (the pictures, language expressions, and so on), and not the procedures used to generate the knowledge from them, they cannot be used in engineered AI systems or in theoretical analyses of knowledge. The difficulty follows simply from the fact that we do not know the entire representational system.[13] As a result, representations, such as natural language, speech and vision, become arenas for research, not tools to be used by the AI scientist to characterize the knowledge of agents under study. Here, logics have the virtue that the entire system of data structure and processes (i.e., rules of inference) has been externalized and is well understood.

The development of AI is the story of constructing many other systems that are not logics, but can be used as representations of knowledge. Furthermore, the development of mathematics, science and technology is in part the story of bringing representational structures to a degree of explicitness very close to what is needed by AI. Often only relatively modest effort has been needed to

[13]It is irrelevent that each of us, as agents rather than AI scientists, happens to embody some of the requisite procedures, so long as we, as AI scientists, cannot get them externalized appropriately.

extend such representations to be useful for AI systems. Good examples are algebraic mathematics and chemical notations.

4.5. Relation of the knowledge level to the symbol level

Making clear the nature of knowledge has already required discussing the central core of the reduction of the knowledge level to the symbol level. Hence the matter can be summarized briefly.

Table 2 lists the aspects of the knowledge level and shows to what each corresponds at the symbol level. Starting at the top, the agent corresponds to the total system at the symbol level. Next, the actions correspond to systems that include external transducers, both input and output. An arbitrary amount of programmed system can surround and integrate the operations that are the actual primitive transducers at the symbolic level.

As we have seen, knowledge, the medium at the knowledge level, corresponds at the symbol level to data structures plus the processes that extract from these structures the knowledge they contain. To 'extract knowledge' is to participate with other symbolic processes in executing actions, just to the extent that the knowledge leads to the selection of these actions at the knowledge level. The total body of knowledge corresponds, then, to the sum total of the memory structure devoted to such data and processes.

A goal is simply more knowledge, hence corresponds at the symbol level to data structures and processes, just as does any body of knowledge. Three sorts of knowledge are involved: knowledge of the desired state of affairs; knowledge that the state of affairs is desired; and knowledge of associated concerns, such as useful methods, prior attempts to attain the goals etc. It is of little moment whether these latter items are taken as part of the goal or as part of the body of knowledge of the world.

The principle of rationality corresponds at the symbol level to the processes (and associated data structures) that attempt to carry out problem solving to attain the agent's goals. There is more to the total system than just the separate symbol systems that correspond to the various bodies of knowledge. As

TABLE 2. Reduction of the knowledge level to the symbol level

Knowledge level	Symbol level
Agent	Total symbol system
Actions	Symbol systems with transducers
Knowledge	Symbol structure plus its processes
Goals	(Knowledge of goals)
Principle of rationality	Total problem solving process

repeatedly emphasized, the agent cannot generate at any instant all the knowledge that it has encoded in its symbol systems that correspond to its bodies of knowledge. It must generate and bring to bear the knowledge that, in fact, is relevant to its goals in the current environment.

At the knowledge level, the principle of rationality and knowledge present a seamless surface: a uniform principle to be applied uniformly to the content of what is known (i.e., to whatever is the case about the world). There is no reason to expect this to carry down seamlessly to the symbolic level, with (say) separate subsystems for each aspect and a uniform encoding of knowledge. Decomposition must occur, of course, but the separation into processes and data structures is entirely a creation of the symbolic level, which is governed by processing and encoding considerations that have no existence at the knowledge level. The interface between the problem solving processes and the knowledge extraction processes is as diverse as the potential ways of designing intelligent systems. A look at existing AI programs will give some idea of the diversity, though no doubt we still are only at the beginnings of exploration of potential mechanisms. In sum, the seamless surface at the knowledge level is most likely a pastiche of interlocked intricate structures when seen from below, much like the smooth skin of a baby when seen under a microscope.

The theory of the knowledge level provides a definition of *representation*, namely, a symbol system that encodes a body of knowledge. It does not provide a *theory* of representation, which properly exists only at the symbol level and which tells how to create representations with particular properties, how to analyse their efficiency, etc. It does suggest that a useful way of thinking about representation is according to the slogan equation

$$\text{Representation} = \text{Knowledge} + \text{Access}$$

The representation consists of a system for providing access to a body of knowledge, i.e., to the knowledge in a form that can be used to make selections of actions in the service of goals. The access function is not a simple generator, producing one knowledge element (i.e., means-end relation) after another. Rather, it is a system for delivering the knowledge encoded in a data structure that can be used by the larger system that represents the knowledge about goals, actions etc. Access is a computational process, hence has associated costs. Thus, a representation imposes a profile of computational costs on delivering different parts of the total knowledge encoded in the representation.

Mixed systems

The classic relationship between computer system levels is that, once a level is adopted, useful analysis and synthesis can proceed exclusively in terms of that level, with only side studies on how lower level behavior might show as errors or lower level constraints might condition the types of structures that are

efficient. The radical incompleteness of the knowledge level leads to a different relationship between it and the symbol level. Mixed systems are often considered, even becoming the norm on occasions.

One way this happens is in the distal prediction of human behavior, in which it often pays to mix a few processing notions along with the pure knowledge considerations. This is what is often called man-in-the-street psychology. We recognize that forgetting is possible, and so we do not assume that knowledge once obtained is forever. We know that inferences are available only if the person thinks it through, so we don't assume that knowing X means knowing all the remote consequences of X, though we have no good way of determining exactly what inferences will be known. We know that people can only do a little processing in a short time, or do less processing when under stress. Having only crude models of the processing at the symbol level, these mixed models are neither very tidy nor uniformly effective. The major tool is the use of self as simulator for the agent. But mixed models are often better than pure knowledge-level models.

Another important case of mixed models—especially for AI and computer science—is the use of the knowledge level to characterize components in a symbol-level description of a system. Memories are described as having a given body of knowledge and messages are described as transferring knowledge from one memory to another. This carries all the way to design philosophies that work in terms of a 'society of minds' [22] and to executive systems that oversee and analyse the operation of other internal processing. The utility of working with such mixed-level systems is evident for both design and analysis. For design, it permits specifications of the behavior of components to be given prior to specifying their internal structure. For analysis, it lets complex behavior be summarized in terms of the external environments of the components (which comprises the other internal parts of the system).

Describing a component at the knowledge level treats it as an intelligent agent. The danger in this is well known; it is called the problem of the *homunculus*. If an actual system is produced, all knowledge-level descriptions must ultimately be replaced by symbol-level descriptions and there is no problem. As such replacement proceeds, the internal structure of the components becomes simpler, thus moving further away from a structure that could possibly realize an intelligent agent. Thus, the interesting question is how the knowledge-level description can be a good approximation even though the subsystem being so described is quite simple. The answer turns on the limited nature of the goals and environments of such agents, whose specification is also under the control of the system designer.

5. Consequences and Relationships

We have set out the theory in sufficient outline to express its general nature. Here follows some discussion of its consequences, its relations to other aspects of AI, and its relations to conceptions in other fields.

5.1. The practice of AI

At the beginning I claimed that this theory of knowledge derived from our practice in AI. Some of its formal aspects clearly do not, especially, positing a distinct systems level, which splits apart the symbol and the knowledge level. Thus, it is worth exploring the matter of our practice briefly.

When we say, as we often do in explaining an action of a program, that "the program knows K" (e.g., "the theorem prover knows the distributive law"), we mean that there is some structure in the program that we view as holding K and also that this structure was involved in selecting the action in exactly the manner claimed by the principle of rationality, namely, the encoding of that knowledge is related to the goal the action is to serve, etc.

More revealing, when we talk, as we often do during the design of a program, about a proposed data structure having or holding knowledge K (e.g., "this table holds the knowledge of co-articulation effects"), we imply that some processes must exist that takes that data structure as input and make selections of which we can say, "The program did action A because it knew K". Those processes may not be known to the system's designer yet, but the belief exists that they can be found. They may not be usable when found, because they take too much time or space. Such considerations do not affect whether "knowledge K is there", only whether it can be extracted usefully. Thus, our notion of knowledge has precisely a competence-like character. Indeed, one of its main uses is to let us talk about what *can* be done, before we have found out how to do it.

Most revealingly of all, perhaps, when we say, as we often do, that a program "can't do action A, because it doesn't have knowledge K", we mean that no amount of processing by the processes now in the program on the data structures now in the program can yield the selection of A. (E.g., "This chess program can't avoid the tie, because it doesn't know about repetition of positions".) Such a statment presupposes that the principle of rationality would lead to A given K, and no way to get A selected other than having K satisfies the principle of rationality. If in fact some rearrangement of the processing did lead to selecting A, then additional knowledge can be expected to have been imported, e.g., from the mind of the programmer who did the rearranging (though accident can confound expectation on occasion, of course).

The Hearsay II speech understanding system [11] provides a concrete example of how the concept of knowledge is used. Hearsay has helped to make widespread a notion of a system composed of numerous *sources of knowledge*, each of which is associated with a separate module of the program (called, naturally enough, a *knowledge source*), and all of which act cooperatively and concurrently to attain a solution. What makes this idea so attractive—indeed, seductive—is that such a system seems a close approximation to a system that operates purely at the knowledge level, i.e., purely in terms of simply having knowledge and bringing it to bear. It permits design by identifying first a

source of knowledge in the abstract—e.g., syntax, phonology, co-articulation, etc.—and then designing representations and processes that encode that knowledge and provide for its extraction against a common representation of the task in the blackboard (the working memory).

This theory of knowledge has not arisen sui generis from unarticulated practice. On the contrary, the fundamental insights on which the theory draws have been well articulated and are part of existing theoretical notions, not only in AI but well beyond, in psychology and the social sciences. That an adaptive organism, by the very act of adapting, conceals its internal structure from view and makes its behavior solely a function of the task environment, has been a major theme in the artificial sciences. In the work of my colleague, Herb Simon, it stretches back to the forties [36], with a concern for the nature of *administrative man* versus *economic man*. In our book on *Human Problem Solving* [28] we devoted an entire chapter to an analysis of the task environment, which turned on precisely this point. And Herb Simon devoted his recent talk at the formative meeting of the Cognitive Science Society to a review of this same topic [37], to which I refer you for a wider discussion and references. In sum, the present theory is to be seen as a refinement of this existing view of adaptive systems, not as a new theory.

5.2. Contributions to the knowledge level vs. the symbol level

By distinguishing sharply between the knowledge level and the symbol level the theory implies an equally sharp distinction between the knowledge required to solve a problem and the processing required to bring that knowledge to bear in real time and real space. Contributions to AI may be of either flavor, i.e., either to the knowledge level or to the symbol level. Both aspects always occur in particular studies, because experimentation always occurs in total AI systems. But looking at the major contribution to science, it is usually toward one pole or the other; only rarely is a piece of research innovative enough to make both types of contributions. For instance, the major thrust of the work on MYCIN [35] was fundamentally to the knowledge level in capturing the knowledge used by medical experts. The processing, an adaptation of well understood notions of backward chaining, played a much smaller role. Similarly, the SNAC procedure used by Berliner [3] to improve radically his Backgammon program was primarily a contribution to our understanding of the symbol level, since it discovered (and ameliorated) the effects of discontinuities in global evaluation functions patched together from many local ones. It did not add to our formulation of our knowledge about Backgammon.

This proposed separation immediately recalls the well-known distinction of John McCarthy and Pat Hayes [20] between *epistemological adequacy* and *heuristic adequacy*. Indeed, they would seem likely to be steadfast supporters of the theory presented here, perhaps even claiming much of it to be at most a refinement on their own position. I am not completely against such an interpretation, for I find considerable merit in their position. In fact, a recent essay by McCarthy on ascribing mental qualities to machines [18] makes many points similar to those of the present paper (though without embracing the notion of a new computer systems level).

However, all is not quite so simple. I once publicly put to John McCarthy [17] the proposition that the role of logic was as a tool for the analysis of knowledge, not for reasoning by intelligent agents, and he denied it flat out.

The matter is worth exploring briefly. It appears to be a prime plank in McCarthy's research program that the appropriate representation of knowledge is with a logic. The use of other forms plays little role in his analyses. Thus, the fundamental question of epistemological adequacy, namely, whether there exists an adequate explicit representation of some knowledge, is conflated with how to represent the knowledge in a logic. As observed earlier, there are many other forms in which knowledge can be represented, even setting entirely to one side forms whose semantics are not yet well enough understood scientifically, such as natural language and visual images.

Let us consider a simple example [17, p.987], shown in Fig. 5, one of several that McCarthy has used to epitomize various problems in representation within logic. This one is built to show difficulties in transparency of reference. In obvious formulations, having identified Mary's and Mike's telephone numbers, it is difficult to retain the distinction between what Pat knows and what is true in fact.

However, the difficulty simply does not exist if the situation is represented by an appropriate model. Let Pat and the program be modeled as agents who have knowledge, with the knowledge localized inside the agent and associated with definite data structures, with appropriate input and output actions, etc.—i.e., a simple version of an information processing system. Then, there is no difficulty in keeping knowledges straight, as well as what can be and cannot be inferred.

The example exhibits a couple of wrinkles, but they do not cause conceptual problems. The program must model itself, if it is to talk about what it doesn't know. That is, it must circumscribe the data structures that are the source of its knowledge about Pat. Once done, however, its problem of ascertaining its own

When program is told:
"Mary has the same telephone number as Mike."
"Pat knows Mike's telephone number."
"Pat dialed Mike's telephone number."

Program should assert:
"Pat dialed Mary's telephone number."
"I do not know if Pat knows Mary's telephone number."

FIG. 5. Example from McCarthy [17].

knowledge is no different from its problem of ascertaining Pat's knowledge. Also, one of its utterances depends on whether from its representation of the knowledge of Pat it can infer some other knowledge. But the difficulties of deciding whether a fact cannot be inferred from a given finite base, seem no different from deciding any issue of failure to infer from given premises.

To be sure, my treatment here is a little cavalier, given existing analyses. It has been pointed out that difficulties emerge in the model-based solution, if it is known only that either Pat knows Mike's telephone number or his address; and that even worse difficulties ensue from knowing that Pat doesn't know Mike's telephone number (e.g., see [23, Chapter 2]). Without pretending to an adequate discussion, it does not seem to me that the difficulties here are other than those in dealing with knowing only that a box is painted red or blue inside, or knowing that your hen will not lay a golden egg. In both cases, the representation of this knowledge by the observer must be in terms of descriptions of the model, not of an instance of the model. But knowledge does not pose special difficulties.[14]

As one more example of differences between contributions to the knowledge level and the symbol level, consider a well-known albeit somewhat controversial case, namely, Schank's work on *conceptual dependency structures* [33]. I believe its main contribution to AI has been at the knowledge level. That such a view is not completely obvious, can be seen by the contrary stance of Pat Hayes in his already mentioned piece, "In defense of logic" [12]. Though not much at variance from the present paper in its basic theme, his paper exhibits Fig. 6, classifies it as *pretend-it's-English*, and argues that there is no effective way to know its meaning.

On the contrary, I claim that conceptual dependency structures made a real contribution to our understanding at the knowledge level. The content of this contribution lies in the model indicated in Fig. 7, taken rather directly from [34]. The major claim of conceptual dependency is that the simplest causal model is adequate to give first approximation semantics of a certain fragment of natural language. This model, though really only sketched in the figure, is not in itself very problematical, though as with all formalization it is an important act to reduce it to a finite apparatus. There is a world of states filled with objects that have attributes and whose dynamics occur through actions. Some objects, called actors, have a mentality, which is to say they have representations in a long term memory and are capable of mental acts. The elementary dynamics of this world, in terms of what can produce or inhibit what, are indicated in the figure.

[14]Indeed, two additional recent attempts on this problem, though cast as logical systems, seem essentially to adopt a model view. One is by McCarthy himself [19], who introduces the *concept* of a number as distinct from the number. Concepts seem just to be a way to get data structures to talk about. The other attempt [14] uses expressions in two languages, again with what seems the same effect.

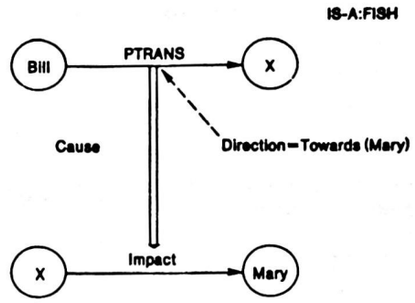

FIG. 6. A conceptual dependency diagram from Pat Hayes [12].

The claim in full is that if sentences are mapped into a model of this sort in the obvious way, an interpretation of these sentences is obtained that captures a good deal of its meaning. The program Margie [33] provided the initial demonstration, using paraphrase and inference as devices to provide explicit evidence. The continued use of these mechanisms as part of the many programs that have followed Margie has added to the evidence implicitly, as it gradually has become part of the practice in AI.

Providing a simple model such as this constitutes a contribution to the knowledge level—to how to encode knowledge of the world in a representation. It is a quite general contribution, expressed in a way that makes it adaptable to a wide variety of intelligent systems.[15] On the other hand, this work made relatively little contribution to the symbol level, i.e., to our notions of symbolic processing. The techniques that were used were essentially state of the art. This can be seen in the relative lack of emphasis or discussion of the internal mechanics of the program. For many of us, the meaning of conceptual dependency seemed undefined without a process that created conceptual dependency structures from sentences. Yet, when this was finally forthcoming (in Margie), there was nothing there except a large AI program containing the usual sorts of things, e.g. various ad hoc mechanisms within a familiar framework (a parser, etc.). What was missed was that the program was simply the *implementation* of the model in the obvious, i.e. straightforward, way. The program was not supposed to add significantly to the specification of the mapping. There would have been trouble if additions had been required, just as a computer program for partial differential equations is not supposed to add to the mathematics.

[15]This interpretation of conceptual dependency in terms of a model is my own; Schank and Abelson [34] prefer to cast it as a *causal syntax*. This latter may mildly obscure its true nature, for it seems to beg for a *causal semantics* as well.

World: states, actions, objects, attributes
Actors: objects with mentality (central processor plus long-term memory)

↑	Cause
↑r	An act results in a state.
↑E	A state enables an act.
↑I	A state or act initiates a mental state.
↑R	A mental act is the reason for a physical act.
↑dE	A state disables an act.

FIG. 7. Conceptual dependency model (after [34]).

5.3. Laying to rest the predicate calculus phobia

Let us review the theorem proving controversy of the sixties. From all that has been said earlier, it is clear that the residue can be swept aside. Logic is the appropriate tool for analyzing the knowledge level, though it is often not the preferred representation to use for a given domain. Indeed, given a representation—e.g., a semantic net, a data structure for a chess position, a symbolic structure for some abstract problem space, a program, or whatever—to determine exactly what knowledge is in the representation and to characterize it requires the use of logic. Whatever the detailed differences represented in my discussion in the last section of the Pat and Mike example, the types of analysis being performed by McCarthy, Moore and Konolige (to mention only the names that arose there) are exactly appropriate. Just as talking of *programmerless* programming violates truth in packaging, so does talking of a *non-logical analysis* of knowledge.

Logic is of course a representation (actually, a family of them), and is therefore a candidate (along with its theorem proving engine) for the representation to be used by an intelligent agent. Its use in such a role depends strongly on computational considerations, for the agent must gain access to very large amounts of encoded knowledge, swiftly and reliably. The lessons of the sixties taught us some things about the limitations of using logics for this role. However, these lessons do not touch the role of logic as the essential language of analysis at the knowledge level.

Let me apply this view to Nilsson's new textbook in AI [30]. Now, I am an admirer of Nilsson's book [27].[16] It is the first attempt to transform textbooks in AI into the mold of basic texts in science and engineering. Nilsson's book uses the first order predicate calculus as the lingua franca for presenting and discussing representation throughout the book. The theory developed here says that is just right; I think it is an important position for the book to have adopted. However, the book also introduces logic in intimate connection with

[16]Especially so, because Nils and I both set out at the same time to write textbooks of AI. His is now in print and I am silent about mine.

resolution theorem proving, thus asserting to the reader that logic is to be seen as a representation for problem solving systems. That seems to me just wrong. Logic as a representation for problem solving rates only a minor theme in our textbooks. One consequence of the present theory of knowledge will be to help assign logic its proper role in AI.

5.4. The relationship with philosophy

The present theory bears some close and curious relationships with philosophy, though only a few preliminary remarks are possible here. The nature of mind and the nature of knowledge have been classical concerns in philosophy, forming major continents of its geography. There has been increasing contact between philosophy and AI in the last decade, focussed primarily in the area of knowledge representation and natural language. Indeed, the respondents to the Brachman and Smith questionnaire, reflecting exactly this group, opined that philosophy was more relevent to AI than psychology.

Philosophy's concern with knowledge centers on the issue of certainty. When can knowledge be trusted? Does a person have privileged access to his subjective awareness, so his knowledge of it is infallible? This is ensconced in the distinction in philosophy between knowledge and belief, as indicated in the slogan phrase, *knowledge is justified true belief*. AI, taking all knowledge to be errorful, has seen fit to call all such systems *knowledge systems*. It uses the term belief only informally when the lack of veridicality is paramount, as in political belief systems. From philosophy's standpoint, AI deals only in belief systems. Thus, the present theory of knowledge, sharing as it does AI's view of general indifference to the problems of absolute certainty, is simply inattentive to some central philosophical concerns.

An important connection appears to be with the notion of *intentional* systems. Starting in the work of Brentano [8], the notion was of a class of systems capable of having desires, expectations, etc.—all things that were *about* the external world. The major function of the formulation was to provide a way of distinguishing the physical from the mental, i.e. of providing a characterization of the mental. A key result of this analysis was to open an unbridgeable gap between the physical and the mental. Viewing a system as physical precluded being able to ascribe intentionality to it. The enterprise seems at opposite poles from work in AI, which is devoted precisely to realizing mental functions in physical systems.

In the hands of Daniel Dennett [10], a philosopher who has concerned himself rather deeply with AI, the doctrine of intentional systems has taken a form that corresponds closely to the notion of the knowledge level, as developed here. He takes pains to lay out an *intentional stance*, and to relate it to what he calls the *subpersonal stance*. His notion of *stance* is a system level, but ascribed entirely to the observer, i.e. to he who takes the stance. The

subpersonal stance corresponds to the symbolic or programming level, being illustrated repeatedly by Dennett with the gross flow diagrams of AI programs. The intentional stance corresponds to the knowledge level. In particular, Dennett takes the important step of jettisoning the major result-cum-assumption of the original doctrine, to wit, that the intentional is unalterably separated from the physical (i.e. subpersonal).

Dennett's formulation differs in many details from the present one. It does not mention knowledge at all, but focuses on intentions. It does not provide (in the papers I have seen) a technical analysis of intentions—one understands this class of systems more by intent than by characterization. It does not deal with the details of the reduction. It does not, as noted, assign reality to the different system levels, but keeps them in the eye of the beholder. Withal, there is little doubt that both Dennett and myself are reaching for the characterization of exactly the same class of systems. In particular, the role of rationality is central to both, and in the same way. A detailed examination of the relation of Dennett's theory with the present one is in order, though it cannot be accomplished here.

However, it should at least be noted that the knowledge level does not itself explain the notion of *aboutness*; rather, it assumes it. The explanation occurs partly at the symbol level, in terms of what it means for symbols to designate external situations, and partly at lower levels, in terms of the mechanisms that permit designation to actually occur [26].

5.5. A generative space of rational systems

My talk on physical symbol systems to the La Jolla Cognitive Science Conference last year [26], employed a frame story that decomposed the attempt to understand the nature of mind into a large number of constraints—universal flexibility of response, use of symbols, use of language, development, real time response, and so on. The importance of physical symbol systems was underscored by its being a single class of systems that embodied two distinct constraints, universality of functional response and symbolic behavior, and was intimately tied to a third, goal-directed behavior, as indicated by the experience in AI.

An additional indicator that AI is on the right track to understanding mind came from the notion of a *generative* class of systems, in analogy with the use of the term in *generate and test*. Designing a system is a problem precisely because there is in general no way simply to generate a system with specified properties. Always the designer must back off to some class of encompassing systems that can be generated, and then test (intelligently) whether generated candidate systems have desirable properties. The game, as we all know, is to embody as many constraints as possible in the generator, leaving as few as possible to be dealt with by testing.

Now, the remarkable property about universal, symbolic systems is that they

are generative in this sense. We have fashioned a technology that lets us take for granted that whatever system we construct will be universal and will have full symbolic capability. Anyone who works in Lisp, or other similar systems, gets these constraints satisfied automatically. Effort is then devoted to contriving designs to satisfy additional constraints—real time, or learning, or whatnot.

For most constraints we do not have generative classes of systems—for real-time, for development, for goal-directedness, etc. There is no way to explore spaces of systems that automatically satisfy these constraints, looking for instances with additional important properties. An interesting question is whether the present theory offers some hope of building a generative class of rational goal-directed systems. It would perforce also need to be universal and symbolic, but that can be taken for granted.

It seems to me possible to glimpse what such a class might be like, though the idea is fairly speculative. First, implicit in all that has gone before, the class of rational goal-directed systems is the class of systems that has a knowledge level. Second, though systems only approximate a knowledge level description, they are rational systems precisely to the extent they do. Thus, the design form for all intelligent systems is in terms of the body of knowledge that they contain and the approximation they provide to being a system describable at the knowledge level. If the technology of symbol systems can be developed in this factored form, then it may be possible to remain always within the domain of rational systems, while exploring variants that meet the additional constraints of real time, learnability, and so forth.

6. Conclusion

I have presented a theory of the nature of knowledge and representation. Knowledge is the medium of a systems level that resides immediately above the symbol level. A representation is the structure at the symbol level that realizes knowledge, i.e., it is the reduction of knowledge to the next lower computer systems level. The nature of the approximation is such that the representation at the symbol level can be seen as knowledge plus the access structure to that knowledge.

This new level fits into the existing concept of computer systems levels. However, it has several surprising features:

(1) a complete absence of structure, as characterized at the configuration level;

(2) no specification of processing mechanisms, only a global principle to be satisfied by the system behavior;

(3) a radical degree of approximation that does not guarantee a deterministic machine; and

(4) a medium that is not realized in physical space in a passive way, but only in an active process of selection.

Both little and much flow from this theory. This notion of knowledge and

representation corresponds to how we in AI already use these terms in our (evolving) everyday scientific practice. Also, it is a refinement of some fundamental features of adaptive systems that have been well articulated and it has nothing that is incompatible with foundation work in logic. To this extent not much change will occur, especially in the short run. We already have assimilated these notions and use them instinctively. In this respect, my role in this paper is merely that of reporter.

However, as I emphasized at the beginning, I take this close association with current practice as a source of strength, not an indication that the theory is not worthwhile, because it is not novel enough. Observing our own practice—that is, seeing what the computer implicitly tells us about the nature of intelligence as we struggle to synthesize intelligent systems—is a fundamental source of scientific knowledge for us. It must be used wisely and with acumen, but no other source of knowledge comes close to it in value.

Making the theory explicit will have many consequences. I have tried to point out some of these, ranging from fundamental issues to how some of my colleagues should do their business. Reiteration of that entire list would take too long. Let me just emphasize those that seem most important, in my current view.

- Knowledge is that which makes the principle of rationality work as a law of behavior. Thus, knowledge and rationality are intimately tied together.
- Splitting what was a single level (symbol) into two (knowledge plus symbol) has immense longterm implications for the development of AI. It permits each of the separate aspects to be adequately developed technically.
- Knowledge is not representable by a structure at the symbol level. It requires both structures and processes. Knowledge remains forever abstract and can never be actually *in hand*.
- Knowledge is a radical approximation, failing on many occasions to be an adequate model of an agent. It must be coupled with some symbol level representation to make a viable view.
- Logic is fundamentally a tool for analysis at the knowledge level. Logical formalisms with theorem-proving can certainly be used as a representation in an intelligent agent, but it is an entirely separate issue (though one we already know much about, thanks to the investigations in AI and mechanical mathematics over the last fifteen years).
- The separate knowledge level may lead to constructing a generative class of rational systems, although this is still mostly hope.

As stated at the beginning, I have no illusions that yet one more view on the nature of knowledge and representation will serve to quiet the cacophony revealed by the noble surveying efforts of Brachman and Smith. Indeed, amid the din, it may not even be possible to hear another note being played. However, I know of no other way to proceed. Of greater concern is how to determine whether this theory of knowledge is correct in its essentials, how to find the bugs in it, how to shake them out, and how to turn it to technical use.

ACKNOWLEDGMENT

I am grateful for extensive comments on an earlier draft provided by Jon Bentley, Danny Bobrow, H.T. Kung, John McCarthy, John McDermott, Greg Harris, Zenon Pylyshyn, Mike Rychener and Herbert Simon. They all tried in their several (not necessarily compatible) ways to keep me from error.

REFERENCES

1. Bell, C.G. and Newell, A., *Computer Structures: Readings and Examples* (McGraw-Hill, New York, 1971).
2. Bell, C.G., Grason, J. and Newell, A., *Designing Computers and Digital Systems Using PDP16 Register Transfer Modules* (Digital Press, Maynard, MA, 1972).
3. Berliner, H.J. Backgammon computer program beats world champion, *Artificial Intelligence* **14** (1980) 205–220.
4. Bobrow, D., A panel on knowledge representation, *Proc. Fifth Internat. Joint Conference on Artificial Intelligence* (Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, 1977).
5. Bobrow, D.G., Ed., Special issue on non-monotonic logic, *Artificial Intelligence* **13** (1980) 1–174.
6. Bobrow, D. and Raphael, B., New programming languages for AI research, *Comput. Surveys* **6** (1974) 153–174.
7. Brachman, R.J. and Smith, B.C., Special issue on knowledge representation, *SIGART Newsletter* **70** (1980) 1–138.
8. Brentano, F., *Psychology from an Empirical Standpoint* (Duncker and Humbolt, Leipzig, 1874). Also: (Humanities Press, New York, 1973).
9. Chomsky, N., Knowledge of language, in: Gunderson, K., Ed., *Language, Mind and Knowledge* (University of Minnesota Press, Minneapolis, 1975).
10. Dennent, D.C., *Brainstorms: Philosophical Essays on Mind and Psychology* (Bradford, Montgomery, VT, 1978).
11. Erman, L.D., Hayes-Roth, F., Lesser V.R. and Reddy, D.R., The Hearsay-II speach-understanding system: Integrating knowledge to resolve uncertainty, *Comput. Surveys* **12** (2) (1980) 213–253.
12. Hayes, P., In defence of logic, *Proc. Fifth Internat. Joint Conference on Artificial Intelligence* (Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, 1977).
13. Hintikka, J., *Knowledge and Belief* (Cornell University Press, Ithaca, 1962).
14. Konolige, K., A first-order formalization of knowledge and action for a multiagent planning system, Tech. Note 232, SRI International, Dec. 1980.
15. Loveland, D.W., *Automated Theorem Proving: A Logical Basis* (North-Holland, Amsterdam, 1978).
16. Luce, R.D. and Raiffa, H., *Games and Decisions* (Wiley, New York, 1957).
17. McCarthy, J., Predicate calculus, *Fifth Internat. Joint Conference on Artificial Intelligence* (Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA 1977).
18. McCarthy, J., Ascribing mental qualities to machines, in: Ringle, M, Ed., *Philosophical Perspectives in Artificial Intelligence* (Harvester, New York, 1979).
19. McCarthy, J., First order theories of individual concepts and propositions, in: Michie, D., Ed., *Machine Intelligence* **9** (Edinburgh University Press, Edinburgh, 1979).

20. McCarthy, J. and Hayes, P.J., Some philosophical problems from the standpoint of artificial intelligence, in: Meltzer, B. and Michie, D., Eds., *Machine Intelligence* 4 (Edinburgh University Press, Edinburgh, 1969).
21. Mead, G.H., *Mind, Self and Society from the Standpoint of a Social Behaviorist* (University of Chicago Press, Chicago, 1934).
22. Minsky, M., Plain talk about neurodevelopmental epistemology, in: *Proc. Fifth Internat. Joint Conference on Artificial Intelligence* (Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, 1977).
23. Moore, R.C., Reasoning about knowledge and action, Tech. Note 191, SRI International, Oct. 1980.
24. Newell, A. Limitations of the current stock of ideas for problem solving, in: Kent, A. and Taulbee, O., Eds., *Conference on Electronic Information Handling* (Spartan, Washington, DC, 1965).
25. Newell, A., AAAI President's message, *AI Magazine* 1 (1980) 1–4.
26. Newell, A., Physical symbol systems, *Cognitive Sci.* 4 (1980) 135–183.
27. Newell, A., Review of Nils Nilsson, Principles of Artificial Intelligence, *Comtemporary Psychology* 26 (1981) 50–51.
28. Newell, A. and Simon, H.A., *Human Problem Solving* (Prentice-Hall, Englewood Cliffs, 1972).
29. Newell, A. and Simon, H.A., Computer science as empirical inquiry: Symbols and search, *Comm. ACM* 19(3) (1976) 113–126.
30. Nilsson, N., *Principles of Artificial Intelligence* (Tioga, Palo Alto, CA, 1980).
32. Robinson, J.A., A machine-oriented logic based on the resolution principle, *J. ACM* 12 (1965) 23–41.
33. Schank, R.C., *Conceptual Information Processing* (North-Holland, Amsterdam, 1975).
34. Schank, R. and Ableson, R., *Scripts, Plans, Goals and Understanding* (Erlbaum, Hillsdale, NJ, 1977).
35. Shortliffe, E.H., *Computer-based Medical Consultations: MYCIN* (American Elsevier, New York, 1976).
36. Simon, H.A., *Administrative Behavior* (MacMillan, New York, 1947).
37. Simon, H.A., Cognitive science: The newest of the artificial sciences. *Cognitive Sci.* 4 (1980) 33–46.
38. Stockton, F.R., The lady or the tiger?, in: *A Chosen Few: Short stories* (Charles Scribner's Sons, New York 1895).
39. Von Neumann, J. and Morgenstern, O., *The Theory of Games and Economic Behavior* (Princeton University Press, Princeton, NJ, 1947).

CHAPTER 5

Learning by Chunking: A Production-System Model of Practice

P. S. Rosenbloom and A. Newell, Carnegie Mellon University

Performance improves with practice. More precisely, the time to perform a task decreases as a power-law function of the number of times the task has been performed. This basic law—known as the *power law of practice* or the *log-log linear learning law*—has been known since Snoddy (1926).[1] While this law was originally recognized in the domain of motor skills, it has recently become clear that it holds over the full range of human tasks (Newell and Rosenbloom 1981). This includes both purely perceptual tasks such as target detection (Neisser, Novick, and Lazar 1963) and purely cognitive tasks such as supplying justifications for geometric proofs (Neves and Anderson 1981) or playing a game of solitaire (Newell and Rosenbloom 1981).

The ubiquity of the power law of practice argues for the presence of a single common underlying mechanism. The *chunking theory of learning* (Newell and Rosenbloom 1981) proposes that *chunking* (Miller 1956) is this common mechanism—a concept already implicated in many aspects of human behavior (Bower and Winzenz 1969; Johnson 1972; DeGroot 1975; Chase and Simon 1973). Newell and Rosenbloom (1981) established the plausibility of the theory by showing that a model based on chunking is capable of producing log-log linear practice curves.[2] In its present form, the chunking theory of learning is a macro theory: it postulates the outline of a learning mechanism and predicts the global improvements in task performance.

This chapter reports on recent work on the chunking theory and its interaction with production-system architectures (Newell 1973).[3] Our goals are fourfold: (1) fill out the details of the chunking theory; (2) show that it can form the basis of a production-system learning mechanism; (3) show that the full model produces power-law practice curves; and (4) understand the implications of the theory for production-system architectures. The approach we take is to implement and analyze a production-system model of the chunking theory in the context of a specific task—a 1,023-choice reaction-time task (Seibel 1963). The choice of task should not be critical because the chunking theory claims that the *same* mechanism underlies improvements on *all* tasks. Thus, the model, as implemented for this task, carries with it an implicit claim to generality, although the issue of generality will not be addressed.