BEHAVIORAL AND BRAIN SCIENCES (1987) 10, 467–505
Printed in the United States of America

Methodologies for studying human knowledge

John R. Anderson
Department of Psychology, Carnegie-Mellon University, Pittsburgh, Pa. 15213

Abstract: The appropriate methodology for psychological research depends on whether one is studying mental algorithms or their implementation. Mental algorithms are abstract specifications of the steps taken by procedures that run in the mind. Implementational issues concern the speed and reliability of these procedures. The algorithmic level can be explored only by studying across-task variation. This contrasts with psychology's dominant methodology of looking for within-task generalities, which is appropriate only for studying implementational issues.

The implementation–algorithm distinction is related to a number of other "levels" considered in cognitive science. Its realization in Anderson's ACT* theory of cognition is discussed. Research at the algorithmic level is more promising because it is hard to make further fundamental scientific progress at the implementational level with the methodologies available. Protocol data, which are appropriate only for algorithm-level theories, provide a richer source than data at the implementational level. Research at the algorithmic level will also yield more insight into fundamental properties of human knowledge because it is the level at which significant learning transitions are defined.

The best way to study the algorithmic level is to look for differential learning outcomes in pedagogical experiments that manipulate instructional experience. This provides control and prediction in realistically complex learning situations. The intelligent tutoring paradigm provides a particularly fruitful way to implement such experiments.

The implications of this analysis for the issue of modularity of mind, the status of language, research on human/computer interaction, and connectionist models are also examined.

Keywords: algorithm; cognitive levels; connectionism; human/computer interaction; implementation; instruction; knowledge; language; learning; methodology; modularity; tutoring

There is a set of beliefs in psychology about the nature of human thought and how to come to understand it scientifically. These beliefs are stated fairly directly by Gleitman in his introductory textbook:

> Psychology is a science and, like all other sciences, it looks for general principles – underlying uniformities that different events have in common. A single event as such means little; what counts is what any one event – or object or person – shares with others. Ultimately, of course, psychology – again, like all other sciences – hopes to find a route back to understanding the individual event. . . . Once such explanations are found, they may lead to practical applications: to help counsel and guide, and perhaps to effect desirable changes. But, at least initially, the science's main concern is with the discovery of the general principles. (Gleitman 1983, p. 11)

This passage reflects three beliefs:

1. The same general psychological principles underlie many different behaviors.
2. Psychologists should focus on discovering these principles.
3. Applying these principles may be socially desirable, but it is scientifically secondary.

These beliefs have influenced many of us to avoid certain research topics and to focus on others. It is hard to quarrel with what these beliefs lead us to focus on, but there is room to question whether we should ignore what they imply we should ignore. This target article will argue for the following points, which contradict (1)–(3) above and which lead to a wider view of research issues:

1′. To understand human cognition, one must understand both mental algorithms (procedures) and their implementation. Only the implementational level can be understood in terms of general principles of cognition that are constant across different situations. The algorithms we possess are adapted to specific task demands and are as varied as those task demands.

2′. There is important basic research to be done at the algorithmic level.

3′. A good way to understand the algorithmic level is to do pedagogical research on how these mental algorithms are taught and learned.

The motivation for writing this target article is quite straightforward: My research on human knowledge acquisition has led me to develop and study intelligent computer-based tutors. In so doing, I find myself fighting the prejudices in (1)–(3). I accordingly wish to respond in two ways. First, I would like to make a case for the scientific respectability of my research. Second, I would like to elicit some thoughtful discussion of the issue, so that if there are flaws in my thinking, I can adjust my ideas and research practices appropriately. (I should also acknowledge from the outset that much of this paper. particularly with respect to points 1′ and 2′, is merely a

 0140-525X/87 $5.00+.00

recasting of arguments already made by Newell and Simon [1972].)

1. The algorithm–implementation distinction

Assertion 1': **There is an important distinction to be made between mental algorithms and their implementation.**

The first assertion is a theoretical one about the structure of human cognition. There is a distinction to be made between (a) the mental procedures and knowledge we possess that enable us to behave adaptively, and (b) the mechanisms that implement these procedures and knowledge. The obvious analogy is to the standard computer, where programs and stored data structures correspond to the algorithm, and the actual machine and its operation correspond to the implementation. The two levels are quite independent because such abstract programs can be implemented in different CPUs (central processing units) and such data structures can be implemented in memories with different properties.

There is no reason why human cognition must mirror this computer distinction, but the distinction seems quite ubiquitous across a broad range of cognitive theories. In the standard flow chart models of information-processing psychology (e.g., Chase & Clark 1972), there is a distinction between the flow chart, which describes the algorithm, and the parameters, which determine how long it takes to implement various stages. Production system theories (e.g., Newell 1973) make a distinction between production rules and the "rules of interpretation" for production systems. Perhaps the only exception to the ubiquitousness of the distinction is the work of some (but not all) of the neural modelers (e.g., Anderson, J. R. & Hinton 1981). Although they make distinctions between the rules encoded in the neural system and the basic principles of the neural system, these authors argue that the basic principles are fundamental and the rules are only approximate and epiphenomenal.

The algorithm–implementation distinction is a theoretical one: What is algorithm and what is implementation can vary from theory to theory. For example, compare a theory like VanLehn's Sierra(1983) with a theory like ACT* (Anderson, J. R. 1983). When a procedure hits an impasse in Sierra, there is a fixed set of ways to repair the procedure built into the implementation, whereas in ACT* such repairs would be implemented by productions learned through experience. In production systems like Newell's (1973) and J. R. Anderson's (1976), goal stacks are processed at the algorithmic level, whereas they occur at the implementational level in more recent systems such as ACT* or Rosenbloom and Newell's (1986). Each theory specifies what its algorithmic language will be.

One might think that the distinction between the algorithmic and the implementational levels was a purely arbitrary choice about where to draw the line in the hierarchical decomposition of a skill. However, as we will see in the discussion of ACT*, a number of consequential theoretical assumptions can line up at this boundary. The boundary often defines the level at which learning occurs. Knowledge is acquired in terms of the units of the algorithmic level; the implementation of these units is fixed and cannot be modified by learning. In choosing a finer-grained algorithmic level (i.e., in choosing not to model certain things at a fixed implementational level), one is choosing to have learning progress in smaller increments of knowledge and to make certain things open to learning.

Pylyshyn (1980) has discussed a similar distinction between *algorithm* and *functional architecture*. He uses the term functional architecture, rather than implementation, to indicate that he is describing something abstracted from the biological level. I prefer to avoid the use of the term architecture, which I use to refer to the interface between the algorithmic and the implementational levels; that is, the architecture specifies the components out of which algorithms are built, and the architectural components have their implementation specified in terms of lower-level components (quasi-neural components in ACT*). However, it may be that the only difference between Pylyshyn's distinction and the present one is terminological. Pylyshyn likewise draws heavily on the computer analogy to articulate his distinction.

If one considers the computer analogy, it is not hard to see why there might be a bias toward regarding the implementational level as scientifically more rigorous. The implementational level is closer to something relatively "real" – the actual machine and its properties. These properties are general and will be maintained across tasks. In contrast, the program and data structure seem the whim of the programmer. At first it might seem that there could be no theory of what the programmer chooses to implement. However, the experience in computer science has been quite the opposite. The more interesting and important theory has proven to reside at the algorithmic level. The choice of program and data structure has proven to have enormous consequences for performance. These consequences are largely independent of machine implementation; moreover, they have proven to be systematic and capable of scientific study.

1.1. Marr's levels. In his very influential work, Marr (1982) specified not two, but three levels of analysis. The level he calls representational and algorithmic corresponds approximately to what I am calling the algorithmic level. This level, according to Marr, is a highly abstract specification of the representation and processing of information. Below this he places the level of hardware implementation, which is concerned with how the representations and algorithms are physically realized. This is considerably lower and more concrete than my own implementational level or Pylyshyn's architectural level. Marr, however, was thinking about computer algorithms for parts of the visual system where it would make sense to be very concrete. In the case of higher human cognition, on the other hand, we do not have the information necessary to specify the hardware level. Hence, here the implementational level is a good deal more abstract than Marr's lower level. I believe, however, that a hardware-level specification is what an implementational theory would become if we ever reached the utopian goal of adequate neurophysiological knowledge.

Marr distinguishes a third level above representations and algorithms that he calls the level of the "computational theory." Its concerns are (Marr 1982, p. 25): "What is the goal of computation, why is it appropriate, and what

is the logic of the strategy by which it can be carried out?" As Marr notes, this is very much like Chomsky's (1965) "competence" level whereas Chomsky's "performance" level is a combination of the algorithmic and implementational levels. Marr's computational level is also much like Newell's (1981) knowledge level. Newell's formulation is probably more suitable for cognition than for vision. In particular, he relates it to issues of deduction and problem-solving. The present essay does not consider the knowledge level or computational theory, but I do not mean to deny the utility of that level of analysis.

1.2. Criteria for the algorithm–implementation distinction. It is interesting to ask how we would distinguish the implementational and algorithmic levels in a particular theory. Pylyshyn (1980) offers two criteria for identifying the implementational level:

1. *Cognitive impenetrability:* The operations at the implementational level are not affected by the organism's goals and beliefs.

2. *Complexity equivalence:* The operation of the units at the implementational level should not vary as a function of the context in which they are evoked. In particular, an implementational unit should always take the same time to execute.

I am comfortable with Criterion 1, but I have reservations about Criterion 2; these have been expressed in J. R. Anderson (1979). In addition to these criteria for identifying the implementational level, I would like to add two criteria for identifying the algorithmic level:

3. *Learning:* Learning takes place in knowledge structures defined at the algorithmic level, although there must be a way of compiling these changes into the implementational level.

4. *Working memory transitions:* Cognitive steps at the algorithmic level correspond to changes in reportable states of working memory.

More will be said about these last two criteria later in the paper. All four criteria require considerable interpretation to be applicable to any existing theory. To be precise in developing further points, it will be necessary to refer to a precise interpretation of the implementation–algorithm distinction at the risk of losing generality in my arguments and conclusions. I will use the realization of this distinction in the ACT* theory.

1.3. The ACT* theory. Figure 1 illustrates the basic architecture of the ACT* system. The core concept is a "production," which is a procedural unit of knowledge. A typical production in the ACT* theory would be the following: IF the goal is to subtract the digits in a column. and the subtrahend is larger than the minuend, THEN set a subgoal to borrow. This rule makes reference to information about current goals, the state of the problem represented in working memory, and long-term declarative knowledge about the relative magnitude of digits. If the situation described in the condition side (the "if" portion of the production) is satisfied, the production's action (the "then" portion of the production) can execute, and a step of cognition will occur.

The arrows in Figure 1 illustrate the flow of information in the system. Information encoded from the environment can be deposited in working memory. Information in working memory can be stored in and retrieved from

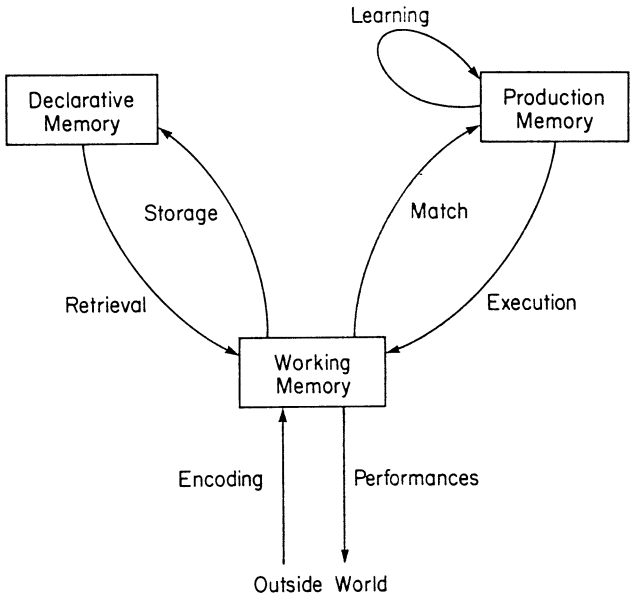

Figure 1. The overall architecture of the ACT* system.

long-term declarative memory. Matching production conditions to working memory serves to select production; when these productions execute, they deposit new information in working memory. Finally, new productions can be created by a learning process that operates on a history of what productions have been executed and what their consequences were.

There are two separate viewpoints about each of the components of the ACT* theory. On the one hand, we can analyze these components according to their abstract input–output specifications. This is like an abstract specification of a programming language. On the other hand, we can view each component according to how long it will take and how likely it is to function properly. These performance considerations are like considerations about the machine implementation of a programming language.

The performance properties of ACT* depend on a set of assumptions about activation-based processing. The critical performance factors concern the speed and success with which production conditions can be matched to working memory; speed and success are in turn a function of the level of activation of these elements. Certain nodes encoding environmental elements or goals serve as sources of activation. Activation spreads from these source nodes throughout the declarative network of knowledge. The amount of activation spread to any declarative node depends on the strength and length of paths connecting it to source nodes. Because of these activation computations at the implementational level, ACT* is a probabilistic system, capable of graded variation in its response.

We have developed separate simulations of the implementational and algorithmic levels in the ACT* theory. One simulation was concerned with the microstructure of activation spread and pattern-matching. This simulation generated the probability and time of a production's executing, and successfully predicted various patterns of data. This simulation proved much too costly, however, for any interesting problem-solving such as calculus. We therefore developed the GRAPES (Sauers & Farrell 1982)

production system, which only simulates the algorithmic level of the ACT* theory.

In GRAPES each production that should "fire" (execute) does so, and there is a unit time associated with each production's firing. Such a system completely eliminates the graded responses produced by the implementational level. It retains all the components in Figure 1, but each occurs flawlessly. The simulations have considerable psychological content in that they preserve certain assumptions in the ACT* theory about

1. the representation of knowledge in declarative and procedural memory and, more significantly, how these relate through the pattern matcher;
2. the principles of conflict resolution that determine which of the multiple matching productions actually fire (given that the conflict resolution in ACT* depends on activation, the conflict resolution principles in GRAPES are discrete approximations to these activation computations);
3. the principles for goal-directed processing, which determine how goals are set and achieved, and determine the process of conflict resolution;
4. the steps of cognition as defined by discrete production firing;
5. the principles of production learning, which determine how new productions are acquired from the trace of old production firing.

The above assumptions are very much like an abstract characterization of a programming language, specifying what the program will do without commitment to the performance specifics of its machine implementation. We can, however, make interesting predictions about behavior just from knowledge of the algorithmic level. If there are systematic errors in the production set, we can predict systematic errors or bugs in the performance. We can determine whether a particular production set will solve problems directly or with a lot of search. To continue with the computer analogy, these are like issues of correctness and efficiency, which are very important in the analysis of computer programs. Last, and most important, at the algorithmic level we can explore how learning changes the production system simulation. All these issues have been studied in GRAPES simulations. In many simulations we were able to predict the problem-solving behavior of subjects (at the input–output level) with GRAPES (Anderson, J. R. et al. 1984; Anderson, J. R. et al. in press). We have also been able to predict learning transitions in that behavior; that is, how the production set spontaneously improves with experience.

One might wonder about how realistic GRAPES simulations are, given that we have not built in any performance limitations. The one major way that GRAPES simulations can deviate from psychological reality is that they can match productions based on unlimited amounts of information in working memory. However, it is easy to restrict GRAPES productions in ways that closely approximate human performance – for example, allowing a production to refer only to long-term memory facts that are no further than a specified distance from the current goal element. This would be only an approximation to the situation created by spreading activation. However, as Simon (1969) noted, we need only approximate capacity characterizations to get close behavioral approximations. The behavior is determined more by the detail of the algorithm and the structure of the problem than by the detail of the implementation.

One thing to emphasize about our work with the GRAPES simulation is that it is intimately concerned with across-task variation. At one level of across-task variation, we are interested in how the same production set behaves given different problems. This provides convergent data on our assumptions about the production set. At another level we are concerned with how the production set changes with experience. This tests the ACT* learning theory. In no case are we engaged in the classic laboratory experiment of repeating the same test over and over again, trying to get reliable measurements of some isolated phenomenon. The enterprise is devoted to understanding variations and complexity. Cognitive principles exist, but only at the level of relations defined across very different situations, rather than at the level of properties of a single type of situation.

1.4. Role of the mental program. Work at the algorithmic level requires that one specify the actual production set that is running. Many of the predictions derive more from what specific productions are assumed than from how they are assumed to be interpreted. Productions become like constrained parameters that are estimated to fit the data. However, it is important to recognize that it is also impossible to do research at the implementational level without making assumptions about the mental program. A great deal of experimental research in psychology takes the form of assuming some fixed program for doing a task, estimating parameters that characterize its implementation, and fitting the program plus parameters to a set of data. To the degree that a model fits, the empirical package is taken as evidence for the correctness of the set of assumptions about the program and the implementation. The work of investigators such as Chase and Clark (1972) or Sternberg (1969) provides classic examples of such approaches. This is a reasonable approach in situations where one can assume a fixed program that does not change across subjects, experiences, or task variations. There are probably many experimentally tractable situations for which these assumptions are reasonable. The frequent belief, however, is that if these assumptions are not satisfied, scientific analysis is not possible. This belief is not justified. One can do a scientific analysis of the principles of variation.

2. Basic research issues at the algorithmic level

Assertion 2': **There is important basic research to be done on algorithmic issues.**

I would like to argue that there should be more concern with the algorithmic level in cognitive science. However, I am not trying to prescribe exactly how much research should be done at one level or the other; nor do I wish to commit myself to doing research only on one level. There are two basic reasons for shifting to algorithmic issues: One is to redress the imbalance that has existed in the opposite direction; the other is that issues at the algorithmic level are more interesting. Each reason will now be discussed in turn.

There are two consequences of the overemphasis on implementational issues in past research. One is that the

foundations have been laid for making rapid progress on algorithmic issues. The second is that it is getting more difficult to make further progress on implementational matters. Serious research on algorithms probably did have to wait until we had a basic understanding of how these were implemented. We had to determine which performance factors were critical in limiting the algorithmic level. Different implementational theories can imply that different factors are critical, and so one could not advance at the algorithmic level without some commitments about implementation. (For example, it seems that working memory limitations are far more fundamental in ACT* than the time it takes for a production to fire.) By analogy with computer algorithms, it is important to know whether it is a Turing machine, standard computer, or parallel device on which the algorithm is being implemented. However, there is now a rich body of research identifying the major factors influencing the execution of a mental algorithm. Whereas there is hardly a theoretical consensus, enough information is available to identify the basic type of system with which we are working. Thus, for example, there is general agreement about short-term memory capacity and retention whether or not one believes in a distinct short-term memory.

Further research on implementational issues may be reaching a point of diminishing returns relative to research on the algorithmic level. One reason is the success of the research of the past 30 years in cognitive psychology. Having achieved a consensus about the basic facts, there is no longer this consensus to be gained as the reward for further research. Rather, we are entering a stage in which researchers are stymied in their endeavors to resolve many of the theoretical issues, such as the status of short-term memory (Crowder 1982), parallel versus serial processing (Townsend 1974), semantic versus episodic memory (Tulving 1983 and *BBS* multiple book review, *BBS* 7(2) 1984), imaginal versus propositional representation (Kosslyn 1980; Pylyshyn 1981; see also Kosslyn et al. "On the Demystification of Mental Imagery" *BBS* 2(4) 1979; and Pylyshyn "Computation and Cognition" *BBS* 3(1) 1980, and so on.) If one looks at the history of research in these areas, one sees that new empirical nuances are uncovered which were unanticipated by any theory but that all theories are slightly modified to accommodate each such result. One gets the impression that existing theories are not correct; nor is the succession of theoretical modifications converging rapidly on a correct theory. Although the accumulation of such results does count as scientific progress, their incremental impact diminishes because they are failing to resolve outstanding theoretical uncertainties.

2.1. The scientific induction problem. This state of affairs can be understood when we examine the scientific induction problem faced by anyone trying to formulate a theory at the implementational level. Such theories are about events occurring at the scale of milliseconds. Moreover, it is generally thought that such events are probabilistic and to some degree state-dependent: No two people behave identically, and no person behaves identically on two successive days.

The theories proposed involve a complex sequence of internal states. For example, most theories of the Sternberg (1969) memory task propose many intervening states reflecting different degrees of comparison of the elements in the target set with the probe on various features.[1] At the implementational level, these internal states go by quickly: Theories may require as many as 100 significant state changes in a second. Yet, data are typically sampled only a few times per second at most, and often one obtains just a binary response such as yes or no. Thus, at the implementational level, the theorist is forced to reconstruct a complex phenomenon with a very impoverished data base. The ratio of data to theoretical detail is low, making for a very difficult scientific induction problem.

One might expect research at the algorithmic level to be equally limited by this induction problem; here the ratio of informative observation to theoretical detail is more favorable, however. Three things change from the implementational to the algorithmic level, one making induction harder and two making it easier. First, the theories are about more complex phenomena (e.g., how one solves a physics problem). This is the step away from a favorable ratio of theory to data. This is compensated by the second difference, however, which is that algorithm-level theories are more abstract and ignore implementational detail. For example, in simulating the Sternberg task at the algorithmic level, one can use a single production to detect a match to the target and another to detect a mismatch. At this level it does not matter how that match is implemented. If there really is an algorithmic level (which is an empirical claim, but, as argued earlier, a widely accepted one), then this is not a false abstraction, but one that captures a true, functional level of the human mind.

This third difference is that there is a richer data source at the algorithmic level. In ACT* and other theories, the steps of cognition at the algorithmic level correspond to points of discrete changes in working memory. When a production fires, it enters new information in working memory and so changes the knowledge state. In contrast, a step of cognition at the implementational level in the ACT* corresponds to a change in activation pattern. In our simulation there are 10–100 of these steps before there is a step at the algorithmic level (i.e., a production firing).

It is important that different states at the algorithmic level are correlated with major differences in the states of working memory because this creates an important advantage over the implementational level with respect to the induction problem. States of working memory are potentially reportable, and we can collect running protocols rather than just final responses. At their best, protocols offer the prospect of providing a state-by-state description of the transitions at the algorithmic level, and the scientist is simply left with the task of inducing the rules that determine these transitions. Protocols in real life are never so fine-grained as to report every state, nor are the reports sufficiently rich to discriminate between all possible pairs of states; however, they are a major advance over the situation at the implementational level. There are problems with the use of protocols, and there have been many unfounded criticisms of their use (see Ericsson and Simon, 1984, for a thorough discussion of the issues), but they are a much better source of data than what is available at the implementational level. Many of these unjustified criticisms of protocols stem from the belief that they are taken as sources of psychological

theory rather than as sources of data about states of the mind. For the latter, one need not require that the subject accurately interpret his mental states, but only that the theorist be able to specify some mapping between his reports and states of the theory. It should also be noted that there is no reason why protocols need be restricted to verbal reports. Eye movements (Just & Carpenter 1979) provide another useful protocol source. As another alternative, much of our research uses as data the keystrokes and "mouse" clicks involved in computer-terminal interactions. The essence of a protocol is that it provides a running series of responses that can be used to infer the sequence of mental states.

Protocol data are available only at the algorithmic level because only at this level do the steps of cognition correspond to reportable working-memory differences. Thus, the data are much richer for constructing theories at the algorithmic level than at the implementational level, although there is little difference in the complexity of the theories. The force of this point can be stated in information-theoretic terms. A scientist is trying to discover a theory of a certain informational complexity. Unless he claims prescience, he is going to have to perform a set of experiments whose total informational yield at least equals the informational measure of that theory. Thus, he is better off using an experimental methodology with a higher informational yield. In the worst implementational case, an experimental trial yields one bit of information – a binary choice among two responses. In a protocol experiment where one gets n responses per trial, each having m values, one gets $n \log_2(m)$ bits of data. Much of the history of scientific progress seems to depend on the availability of tools that raise the informational yield associated with an experiment.

One of the reasons I had previously failed to appreciate this argument about the relative richness of the data at the two levels is that I believed it ignored the importance of reaction-time data as a source of information. There is certainly more information in a distribution of reaction times than there is in a response probability. The problem is that the distributional information requires a proportional complication in the underlying theory. Each state transition requires a distribution of times associated with it to explain the observed distributions. Thus, whereas reaction time data increase the informativeness of a trial, the effect is cancelled by the increased complexity of the theory.

I have tested this analysis of the value of protocol data in a number of experiments on the Carnegie-Mellon University psychological community, where I have produced computer simulations of "mystery" systems that were simplified versions of known psychological theories. People had access either to the input–output of these systems or to protocols that gave partial information about intermediate states. Their task was to try to identify the rules of the system. Success was much more rapid and complete with protocols. In the case of a nondeterministic system, people had no success without the protocols, whereas they still had complete success with protocols. Although these tests were of necessity informal, their outcomes were consistent with the idea that induction is much more difficult in the absence of access to intermediate-state information.

One might think that the traditional low-information-yield experiment has been used exclusively to decide implementation-level issues where it is the only methodology available. This has not been the case, however. Many experiments in the literature (to point a finger only at myself, consider J. R. Anderson, 1976, pp. 363–75) have used the low-yield methodology to explore algorithm-level issues in situations where much better protocol data could have been obtained.

2.2. Interest. So far the argument for focusing on the algorithmic level has been that there is more opportunity for theoretical progress because it is not overresearched and it is theoretically more tractable. In addition, I would like to argue that the more interesting psychological issues are at the algorithmic level. There are two related arguments here: (1) that the algorithmic level accounts for most of the variance in human behavior and (2) that the algorithmic level is concerned with issues close to essential features of human nature. (Such arguments are necessarily appeals to personal preferences, and the reader who fails to share my aesthetic primitives will not be convinced. However, these arguments might still help explain why someone would choose to study the algorithmic level.)

An important fact about computer programs is that optimizing an algorithm can lead to much greater time savings than optimizing its implementation. For most complex programs one can predict more about the performance by counting the number of language instructions than by inquiring how these instructions are implemented. The same is true of human performance in at least some domains, such as interesting problem-solving tasks. The variance is largely accounted for by the number of productions and not by the time it takes for each production step to execute. The difference in alternative efficiency of algorithms outweighs any difference in efficiency of implementation. Thus, for example, accounts of novice–expert differences in problem-solving (e.g., Chi et al. 1981; Larkin 1981) focus on differences at the algorithmic level.

For this argument to be valid, it is not necessary that the time for a production to execute be constant (and it is not constant in the ACT* theory). It is only necessary that the variance due to the execution of the production be small compared to the variance due to the number of productions executed. When we look at complex problem-solving tasks like writing a LISP program, we easily see a 100–1 range in the number of productions required by efficient and inefficient algorithms. Even if there were a 10–1 range in the time required for a production to execute (which would be hard to get in the ACT* theory), 99% of the variance would be due to the number of productions and not to the speed per production.[2]

We have also noted that if we are to develop a theory at the algorithmic level, this theory will be focused on learning. This directs us to perhaps the most important and distinctive of human traits: our ability to acquire and utilize new knowledge. One of the great intellectual questions concerns the origins of that knowledge – how genes and environment conspire to transform the newborn into an adult adapted to its particular environment.

There is, of course, a great deal of research on human learning and memory, but this work has been at the

implementational level. For example, there is abundant research on the rate of retrieval of facts, the capacity of short-term memory, the effect of study time on amount learned, retention curves, inter-item confusions, and so forth. What has largely been ignored is the algorithmic question of how this knowledge is organized to allow effective performance to occur in new task environments. The interesting epistemological questions are at this algorithmic level.

The one area in which there is a longstanding tradition of concern with issues above the implementational level is language acquisition, where the dominant question has been how the child learns the rules of its language and not how these rules are implemented. Credit for the early focus on this question goes to the advocacy of the competence–performance distinction (Chomsky 1965), which directed investigation toward issues above the implementational level. (Actually, concentrating on competence, as it is traditionally construed, led to a level of abstraction even higher than the algorithmic one, with a concern for knowledge independent of how it is used. Various researchers in the field [e.g., Berwick, in press] are now coming to realize that they must also consider knowledge use, and that the problem of language acquisition is going to require work at the algorithmic level in addition to the competence level.)

Let me conclude this section by comparing research at the implementational and algorithmic levels with a choice between working two mines: It is easier to mine the algorithmic level; there is more ore left in the mine; and the ore is inherently more valuable. The third consideration should be emphasized. By studying the acquisition of new algorithms, we are addressing fundamental epistemological questions and analyzing a quintessential human trait.

3. Pedagogical experiments

Assertion 3′: **The best way to study the algorithmic level is through pedagogical experiments.**

There were two reasons given earlier for why learning is the key issue at the algorithmic level: (1) By considering learning, one can uncover the common mechanisms that explain very different behaviors, and (2) the most interesting psychological questions concern learning. We can also learn much about the structure of a particular mental procedure by studying the course of its development to the current state. When we look at a well-oiled skill, it often executes too smoothly for the performer to give any report about what is happening. In the ACT* theory, for example, a major component of skill acquisition, called knowledge compilation, eliminates the need to use working memory to hold intermediate results in the calculation of answers. Thus, one cannot use protocol data to analyze a skill that is already compiled. However, in ACT* the compiled skill preserves the logical structure of the precompiled skill. So, if one has analyzed the precompiled skill and identified its structure, one can use that analysis to infer the structure of the compiled skill.

Even accepting that the study of learning is central, there is room for debate about how we should go about studying algorithm-learning. If we were to look at research on language acquisition, we would see a strong bias toward the naturalistic study of unstructured learning situations, but there are problems with such a methodology. Key predictions from any theory will be about situations that do not occur naturally. For example, many of the theories about language acquisition contain claims about constructions that children cannot learn. Unfortunately, however, there is seldom evidence that they indeed cannot learn these constructions; only that natural languages seem to lack them. These theories also make predictions about how the types of sentences a child hears will or will not affect acquisition. Unfortunately, the naturalistic experiment excludes such manipulations of input in all but the weakest form. In addition to these difficulties, factors are confounded in naturalistic experiments. For example, it is hard to separate conceptual development from language acquisition. As a consequence, investigators of language acquisition keep returning to studies of artificial language learning in the hope that these will yield cleaner results (e.g., Braine, in press; Morgan & Newport 1981).

The problem with the attempt to introduce control into research on learning is also well illustrated in the work on artificial languages. In attempts to create a laboratory version of the learning situation, one may not be preserving the essential features of the phenomenon to be studied. This is not an idle doubt when it comes to skill acquisition. The essential feature of any interesting skill concerns the way it deals with complex and demanding problems. One might argue that our learning mechanisms are adapted to deal with complexity. In seeking laboratory simplifications, one may have thrown out the features that would reflect these adaptations to complexity.

Ideally, one would like to intervene in naturalistic learning situations, produce controlled variations on what is happening, and collect precise measurements on the outcome. Unfortunately, this is not feasible in the case of first-language acquisition, for a combination of ethical and logistical reasons. The only experimental manipulations of first-language learning in response to these constraints have been weak and narrow (e.g., Cazden 1965; Nelson et al. 1973). However, the interesting observation is that there is a large class of naturalistic learning situations where one can intervene much more effectively, although it is not totally unconstrained by logistical and ethical issues: formal education. One can look at motivated people learning representatively complex academic skills. One can intervene with major experimental manipulations (and for decades educational researchers have done so).

Thus, if one accepts the basic argument that learning is the most interesting issue at the algorithmic level (an argument based on certain aesthetic assumptions), then the best way to study the algorithmic level is through pedagogical experiments that manipulate learning history and look for differences in the algorithms acquired. This can yield the predictiveness and control associated with the best experimental rigor.

3.1. Biases against pedagogical research. The appropriateness of the pedagogical experiment seems so self-evident that I have wondered why it is not a dominant methodology in cognitive psychology. For example, why are we not exploring issues of language acquisition in

those programs that successfully teach second languages? There are questions about whether second-language acquisition involves the same mechanisms as first-language acquisition (McLaughlin 1984), but why should these questions diminish interest in second-language acquisition? Perhaps the major reason for the relative lack of interest by cognitive psychologists in pedagogical experiments is the belief it is hard to do good research in this area. Among the many reasons for this belief is the perception that much educational research is so weak. Even if this is true, it remains a separate issue whether the weaknesses of pedagogical research are inherently irremediable.

I believe that some pedagogical research has been unsuccessful because of inappropriate experimental and theoretical methodology. A great deal of work in educational psychology has been dominated by dated perceptions of the "correct" methodology in cognitive psychology. Methods and theories that are appropriate for implementation-level problems are applied to algorithm-level ones. Also, much of this research has ignored variability at the algorithmic level, summarizing structural differences in algorithms with parameters of individual differences such as intelligence measures. Given that the research has not really been designed to understand the algorithmic level, there should be little surprise that there has not been much progress. (I should add that I do think there are some stellar examples of research in educational psychology [e.g., Brown & VanLehn 1980; Chi et al. 1981; Resnick 1982], but these are the studies that did use an appropriate methodology.) Not only are pedagogical experiments the right ones for studying issues at the algorithmic level, but to make progress on these pedagogical issues, one must approach them with the methodology appropriate to that level.

3.2. Learning by instruction versus learning by induction. A second reason for the lack of interest in pedagogical research is that it is believed that learning by instruction is less interesting than learning by induction. Thus, the teaching of second languages is thought to be less interesting than the child's naturalistic learning of the first language. This makes a false dichotomy between the two learning situations, however. Naturalistic learning probably does involve at least covert instruction; and learning in a pedagogical situation certainly involves induction. VanLehn (1983) and Neves (1981) have argued that all of a child's classroom learning of skills such as algebra and subtraction is inductive. This may be too strong a claim, but there is certainly some truth in it. The instruction received in a classroom greatly underdetermines the skill to be acquired.

One could even argue that pedagogical paradigms will reveal more about the essential features of human learning than will other paradigms. Most human skills and knowledge are not reinvented anew through raw induction by each generation but are passed from one generation to the next. In studying this knowledge transfer, part instruction and part induction, we are studying the paradigm for which our learning mechanisms are most adapted.

3.3. Research agenda. Current pedagogical research addresses an interesting set of questions:

1. How is knowledge initially acquired in a new domain (the *tabula rasa* question – e.g., Laird & Newell 1983; VanLehn 1983)?
2. How does knowledge in one domain transfer to another (e.g., Moran 1983; Polson & Kieras 1985)?
3. How does declarative knowledge relate to procedural knowledge (my own research)?
4. How are experts different from novices (e.g., Chi et al. 1981; Larkin 1981)?
5. How do skills speed up with practice (e.g., Newell & Rosenbloom 1981)?
6. How do errors occur (e.g., Brown & VanLehn 1980; Norman 1981)?

3.4. Summary. Pedagogical experiments offer a nearly ideal combination of complexity, experimental rigor, and representativeness. Biases against such applied research are just that – biases. After all these years psychologists still suffer from physics envy; we believe that we should construct the human mind out of unmotivated postulates. This ignores the fact that human cognition evolved as a functional tool. We will never understand it until we understand how our cognitive mechanisms adapt to functionally important problems.

4. Intelligent tutoring

As stated previously, part of my motivation in writing this target article was to provide a defense of my own research practices. The argument to this point is that research on pedagogical issues is strongly motivated by a basic scientific interest in human cognition. My own research decisions have been more specific than just performing pedagogical experiments, however. I have chosen to develop intelligent tutors. Although I cannot argue that the development of intelligent tutors is the best way to understand human cognition, I would suggest that it is a reasonable way.

This section is aimed at two kinds of readers. The first is those who accept the three initial arguments above but who believe that research on computer-based tutors is branching off into technology and abandoning the pure intellectual objectives of cognitive psychology. The case needs to be made for the scientific integrity of this enterprise. The second is those who accept the argument for pedagogical experiments in the abstract, but who feel that the process of studying the phenomena in an academic course is overwhelming. I believe that computer-based tutors are technology's answer to this problem.

The variety of ways to use computers to educate is enormous, and the variety of ways to use artificial intelligence methods is almost equally large (Brown & Greeno 1984; Clancey, in press; Johnson & Soloway 1984). All forms of computer-based instruction allow the possibility of automatic data-collection as well as some controlled manipulation. I would like to focus, however, on a special intelligent tutoring paradigm called model-tracing, which is particularly appropriate for psychological purposes. Elsewhere (Anderson, J. R. et al. 1985) it has been argued that this methodology is also particularly effective pedagogically, but this is not the issue here.

The model-tracing methodology requires running simulations of both how subjects *should* perform tasks and how they *actually* perform tasks. In our tutoring efforts

we have models of how students write (and how they should write) programs in LISP, proofs in geometry, and algebraic solutions. In addition to these student models. an interface must be created to allow students to solve problems with the computer and to communicate their performance to it. As the student is solving a problem, one runs a real-time simulation of the student's problem-solving, noting where the actual student deviates from the ideal student. Protocols of the student's actual responses and the simulations of these responses are automatically recorded. The key factor in delivering instruction is that at all points the tutor has a rich interpretation of the student's inferred mental state. Exactly how one brings the tutor's cognitive interpretation to bear in instruction depends on one's learning theory. Based on our ACT* theory, we have developed a set of cognitive principles for instruction built heavily around immediate feedback.

There are two psychological advantages of having a tutor that simulates the student's problem-solving. First, the success of the instruction reflects on the correctness of the simulation and hence on the correctness of the cognitive theory. If the simulation is really off base, as it has been in some of our attempts, the tutoring effort fails and this feeds back on our theoretical analysis of the skill.

The second function of the simulation is to enable us to interpret the student's behavior objectively. We are no longer left to induce informally what rules the subject is following. The tutor defines an objective procedure for mapping student behavior onto inferred production rules. Thus, the tutor serves as the data collection mechanism for the ideal pedagogical experiment, which manipulates instruction and records the consequences in terms of the algorithms (productions). Apart from the gain in objectivity, one increases the rate at which protocols can be analyzed. For instance, I recently (Anderson J. R., in press) analyzed some 500 hours of protocol data with the time investment that used to go into the hand analysis of 10 hours.

Whereas any computer at all will permit one to define the instructional manipulation precisely, the model-tracing methodology allows manipulations which vary instruction in response to the student's current state of knowledge. Many predictions about learning require this kind of contingent responding.

Working in the intelligent tutoring domain naturally focuses one on searching the space of possible tutors for the one that optimizes learning. In addition to the applied motivation behind this focus, there is fundamental scientific motivation. The manipulations that optimize learning tell us a great deal about the learning mechanisms. However, in my own research I have also been interested in studying variations on the tutor that the theory predicts will be defective and will lead to slower learning. There are, of course, serious constraints on how one can deploy such "defective" tutors. One cannot use them in real classroom situations.

4.1. Possible arguments against intelligent tutoring. Although I believe that intelligent tutoring represents a major methodological advance, it is important to acknowledge problems that may be associated with it. The most serious of these may be the danger that one's time could become too consumed in understanding the domain of instruction (e.g., calculus) and that details of computer implementation could become too dominant. On the other hand, such "cost-of-development" criticisms can be made of the efforts devoted to any research tool; they are no more valid when the tool has applied significance. On the contrary, the applied value of the tool may in part justify the cost of developing it. In the long run, the cost-benefit ratio may favor intelligent tutors over most tool-building efforts in psychology or science in general.

One should not rely on the potential practical benefits of intelligent tutors to justify their development, however. The educational establishment is extremely resistant to change, and it is quite possible that improved educational technology will simply be ignored. For many reasons, educational effectiveness has little impact on educational policy. The real justification for developing intelligent tutors has to be the goal of understanding human learning. If these tutors have a beneficial effect on society, so much the better.

Another argument against tutoring is that the psychological benefits of intelligent tutoring research have been few to date. This is related to the fact that the AI techniques used to create such tutors are still evolving and to the fact that the cost of appropriate equipment is only now becoming realistic. There has been relatively rapid development in the past couple of years (Clancey, in press).

5. The importance of applied research

This paper should not be read as an assertion that there is no value in nonapplied research or in research that studies general implementation-level principles; however, it is important to stress the scientific value of applied research on the acquisition of domain-specific cognitive algorithms. The human system is not a set of principles of physics let loose in a relatively unstructured universe. We are an artifact of evolution – a system designed to achieve applications. Our most uniquely human attribute is our ability to acquire new abilities to deal with novel problems; that is, evolution has shaped us to achieve domain-specific applications. To refuse to consider what is task-specific or what is applied is to refuse to consider the most important aspects of human cognition.

6. Auxiliary theoretical issues

6.1. The modularity issue. There has been a lot of interest recently (e.g., Chomsky 1980; Fodor 1983) in whether the mind is modular, that is, whether different faculties operate according to different cognitive principles. The theory presented here implies that, at the implementational level, the mind is not modular in that the implementational principles are domain-general. The issue is more complex at the algorithmic level. The present claim is that the algorithms that arise in different domains are specific to those domains. For example, one would not be able to predict problem-solving expertise in the domain of programming as a generalization of problem-solving expertise in the domain of physics. On the other hand, these different algorithmic structures are a consequence of the same learning principles interacting with the different structures of each domain.

Cognition is hence structured differently in different domains at the algorithmic level, even if this structure is the result of domain-general learning mechanisms. The possibility of such a state of affairs complicates considerably the issue about modularity of mind. Ignoring for a moment the issue of the status of language, Fodor (1983; see also multiple book review, *BBS* 8(1) 1985) and I (Anderson, J. R. 1983) have each written books arguing that higher-level mental processes are nonmodular, in contrast to the position of Chomsky (1980), for example [see also Chomsky: "Rules and Representations" *BBS* 3(1) 1980]. Despite our agreement here, Fodor is pessimistic about a science of such central processes, whereas I am optimistic. He also views language as a modular input system, whereas I am currently agnostic about whether it is separate from other higher-level processes. In any case, a strong position of nonmodularity for the central processes would imply the following:

1. At the implementational level, the same principles describe mental operations in all domains of cognition.
2. The same learning principles apply to the acquisition of competence in all domains.
3. There is a single declarative knowledge base available for use in all domains.
4. The algorithms for processing knowledge are domain-general.

I think Fodor would accept (4). I read him as believing that all cognition takes place by means of some general inferential algorithm applying to declarative assertions. Not believing this, I had proposed (4) in 1983 only because I had not thought out carefully the implications of my theory. The ACT* theory would in fact imply that as we become skilled in a domain, the problem-solving structure of that domain becomes specialized. Note that this is a case of acquired modularity rather than of innate modularity.

6.2. The status of language. There is a good deal of confusion in cognitive science about the status of language. Some see the study of language as guiding our efforts to understand cognition, whereas others (such as Fodor 1983) see language as not being part of cognition at all but a modular input system like vision. Some see it as the one area of cognitive science where there has been real progress; others see it as a field of hopeless stagnation.

If we were to view language as a domain appropriate to the analysis offered in this paper, what would be the methodological implications for its study? There are a number of rather startling implications, and one can take their aptness as an indication of whether language is really appropriately analyzed in this framework:

1. If the implementation–algorithm distinction is applied to language, then we should identify the rules that govern language comprehension and production with the algorithmic level and the factors that control the performance of these rules with the implementational level. Adult language would accordingly be a highly overlearned skill in which conscious access to its algorithmic structure had largely been lost. Thus, experiments on the use of native language by adult speakers (perhaps the most popular psycholinguistic paradigm) are not likely to produce rapid progress on the algorithmic level because we lose the special methodological advantage of that level – namely, conscious access to intervening states.
2. If the best way to study algorithmic structure and acquisition is in paradigms that expose the sequence of internal states, then (a) we need to study language performance in beginning language users whose processing is slower and perhaps more open to report; and (b) we probably cannot use verbal protocols because of the obvious conflict within the linguistic modality if one attempts both to report and perform. Ericsson and Simon (1984) argue for the virtue of immediate retrospective reports that may avoid some of this modality conflict; however, I still suspect that the study of language processing poses a special methodological problem because the best modality for report is occupied by the behavior under study.
3. If there were a methodology for tracing the cognitive states in initial language use, apart from verbal report, then pedagogical programs for teaching second languages would be an excellent paradigm for studying language. The advantage of the instructional setting is that (a) it is concerned with directly manipulating the learning environment; (b) it works with slow processing subjects; (c) these subjects are relatively instructible; (d) it allows us to trace learning; and (e) it represents realistically complex phenomena (in contrast to work on artificial languages). Of course, there is the possibility that this approach will only teach us about second-language acquisition, not first-language learning. This still seems an eminently worthy goal.

6.3. Research on human/computer interaction. Work on intelligent tutoring can be classified as a special case of research on human/computer interaction. In the recent literature on this topic (Card et al. 1983; Card & Newell 1985; Polson & Kieras 1985; Sebrechts & Black 1982), there is a considerable overlap with the position put forward here:

1. The field of human/computer interaction is ready for cognitive applications. Although we have not resolved the theoretical debates at the implementational level, we have characterized the behavior of the system at this level to a close degree of approximation. Behavior at the algorithmic level is determined by a principle of rationality (Card et al. 1983), whereby it is adapted to the problem domain; we can accordingly predict algorithms by task analysis.
2. There will be a strong feedback from studies of human/computer interaction to cognitive psychology in that, as we try to account for complex phenomena, we will learn where the gaps are in our theoretical conceptions and where our theories are seriously off the mark. When comparing two theories, it is more important to find a phenomenon that neither theory can handle than to do a hair-splitting experiment whose only purpose is to discriminate between the two theories. The former kind of evidence is likely to be useful in assessing other theories, whereas the latter kind tends to lose its significance when both theories are proven wrong.

The difference between the research I am advocating here and more general research on human/computer

interaction concerns whether the research is focused on the computer or on learning. (Of course, specific research endeavors can focus on both.) Pedagogical research is likely to yield more scientific information than research on human/computer interaction, although it will probably have a lesser applied impact. This is because research on learning is focused on the issue that will make sense of the broad variation at the algorithmic level. In contrast, work on human/computer interaction will yield one type of algorithm for text editing, another for programming, and so forth, without achieving a higher level of analysis.

6.4. Relation to connectionist models. Connectionist models (e.g., Ackley et al. 1985; Anderson, J. A. 1983; Feldman & Ballard 1982; Hinton & Anderson 1981; McClelland & Rumelhart 1986; Rumelhart & McClelland 1986) seem to be theories at the implementational level as it is defined in this paper. So it may seem surprising to find Rumelhart and McClelland (1985), in their response to Broadbent (1985), arguing that their connectionist models correspond to Marr's representational and algorithmic level and not to his hardware implementational level. As noted earlier, however, these two levels of Marr's framework do not correspond to the algorithmic and implementational level as they are defined in this paper. Rumelhart and McClelland are clearly not working at the true hardware level, any more than most cognitive psychologists are. Theirs is nevertheless an implementation-level theory as this is defined in the present paper.

We can see that Rumelhart and McClelland's theory is not at the algorithmic level by looking at some of the criteria that have been used to define it. The authors would not want to posit an equivalence between changes in reportable states of working memory and changes in the activation levels of their connectionist models. This is a level of serial-like processing that is above their theory. Rumelhart and McClelland would be equally unwilling to accept Pylyshyn's criterion of cognitive penetrability for the algorithmic level, which would require their connectionistic computations to be affected by beliefs.

It is interesting that the ACT* implementational level embodies all the desirable features Rumelhart and McClelland (1986) derive from neural consideration: The processing units are slow; a system involves a large number of elements with a large number of connections; the elements communicate by activation and inhibition; they have continuously available output; the system is capable of graceful degradation; it has distributed control; computation takes place by relaxation; and learning involves modifying connections. Despite the fact that the ACT* implementational level has all these properties, Rumelhart and McClelland repeatedly refer to ACT* as an instance of the "other" class of models, the one that contrasts with their own PDP (parallel distributed processing) models. I think they do this because there is an exclusionary clause in their definition of a PDP model, to the effect that there is only an implementational level and any theoretical assertions at the algorithmic level are necessarily only approximations to the underlying truth. In contrast, in ACT* the algorithmic level is quite real and not an approximation. It is true that the actual implementation involves factors not specified in the algorithm, just as a computer will do things not specified by its program. However, this does not tell against the reality of ACT* algorithms or the computer's program.

McClelland and Rumelhart actually use a computer analogy to make their point about the priority of the implementational level. In their analogy, the algorithmic level corresponds to a PASCAL program and the implementational level corresponds to assembly code. My view is that the mind programs itself (i.e., learns) at the PASCAL (i.e., symbolic) level and that this learning is compiled into the assembly level (connections). Their view is that the mind programs itself at the assembly level and that the assembly-level code can be approximated only by the PASCAL code. The reason I see this otherwise is that this allows me to account for the findings on the learning of problem-solving skills. Why do Rumelhart and McClelland instead see it as they do? In Rumelhart and McClelland (1985), they write:

> Because there is presumably no compiler to enforce the identity of our higher level and lower level descriptions in science, there is no reason to suppose there is a higher level description exactly equivalent to any particular lower level description. (pp. 195–96)

Subsequently, in Rumelhart and McClelland (1986), they write:

> Since there is every reason to suppose that most of the programming that might be taking place in the brain is taking place at a "lower level" rather than a "higher level" it seems unlikely that some particular higher level description will be identical to some particular lower level description. (pp. 124–25)

So, it seems that the postulate of a symbolic level (Newell's, 1980, physical symbol hypotheses) is simply swept aside by assumption. I suggest that there is no reason to reject the algorithmic level and quite a bit of current evidence that can be accommodated only at that level. (This is not to deny that there is much evidence requiring the implementational level too.) I assume Rumelhart and McClelland's belief that there is no compiler is based on the fact that we do not yet know how the brain accomplishes the conversion from the symbolic to the neural level. But if we were to take that as a criterion for rejecting theories, we would have to go on and reject many of the connectionist proposals. So although there seems little else in the general premises of connectionist modeling to disagree about, I do find it somewhat overzealous to exclude an algorithmic level.

ACKNOWLEDGMENTS

This research is supported by grants NSF 82-08189, MDR-8470337, and IST-8318629 from the National Science Foundation. I would like to thank Lynne Reder for going over this manuscript with me.

NOTES

1. In the Sternberg task a subject is asked to hold in memory a target set of a few items; frequently digits. A probe item is presented to the subject, who must decide if the probe is in the target set.

2. If the range due to factor 1 is n_1 to 1, and the range due to factor 2 is n_2 to 1, the contribution of factor 1 to the variance is $n_1^2/(n_1^2 + n_2^2)$.

Open Peer Commentary

Commentaries submitted by the qualified professional readership of this journal will be considered for publication in a later issue as Continuing Commentary on this article. Integrative overviews and syntheses are especially encouraged.

Many levels: More than one is algorithmic

Michael A. Arbib

Program in Neural, Informational, and Behavioral Sciences, University of Southern California, Los Angeles, Calif. 90089

As Anderson states somewhat indirectly, Marr (1982) offered four levels of analysis:

1. the *functional* level, which expresses the goals of a computation and the general strategy by which it is to be carried out (it is unfortunate that Marr calls this the "computational" level, because it is far from indicating the steps – serial or parallel – whereby computation is undertaken; rather, it corresponds to what in computer program synthesis would be the specification of a program in input/output terms, plus perhaps a top-down sketch of program design);
2. the *representational* level, which provides a highly abstract specification of the representation of information;
3. the *algorithmic* level, which provides a highly abstract specification of the processing of information; and
4. the *hardware implementational* level, which is concerned with how the representation and algorithm are physically realized.

Anderson offers an analysis in terms of two levels: His *algorithmic* level corresponds to a combination of the representational and algorithmic levels; his *implementational* level is more abstract than Marr's lower level, although it aims toward a neurophysiological level, data permitting. But I would argue that the dichotomy – algorithm versus implementation – runs counter to well-established terminology. An implementation *is* an algorithm, but in a "fine-grain" language. It is not "algorithmicity" that distinguishes the levels, but rather the language in which each is to be expressed.

In computer science, we start with an informal specification of a program; generate an algorithm in a quasi-mathematical language by more or less *ad hoc* methods; translate it into a program for a fixed high-level language (e.g., PASCAL) by fairly stereotyped methods; and use a compiler or interpreter to mechanically translate it into, say, machine language. Here, subject to agreed on conventions for coding input and output, the semantics of the program is preserved through all levels from initial specification of input/output behavior to the detailed implementation in machine language.

Anderson suggests that his algorithmic level is akin to the high-level language level, whereas his implementational level is more akin to the machine-language level. However, Anderson fails to observe that in computer science we *start* with the correct specification of the program (at all levels), whereas the task of the psychologist is to go from some vague formulation of interest in some limited domain of cognition to a precise formulation of the algorithm for some related, but perhaps more narrowly constrained, domain. In particular, a specification at Anderson's algorithmic level is only a hypothesis and may receive refinement when, and only when, this hypothesis is subject to data available only at the implementational level (e.g., neural recording). More to the point: The high-level program has precisely the same semantics (for well-coded inputs) as its machine-language translation; but it is an open question whether or not a psychological "algorithm" is any more than an approximate description of the semantics of the neural net, say, that constitutes its implementation in the human brain.

In my companion target article, I offer *schemas* as providing a high level of analysis and *neural nets* as providing a low level; but I assert that a successful cognitive neuroscience must embed them in a finer net of levels to achieve an efficacious set of analytical tools. I had expected Anderson to offer his ACT* theory as his algorithmic level, because it has some similarities to what I aim for in schema theory. (I would like a clear statement about the extent to which parallelism is necessary for an ACT* analysis.) But Anderson offers ACT* as his implementational level, with GRAPES (which throws away details of activation levels) as the algorithmic level. Thus, in his theory the implementational level cannot be compiled from the algorithmic level; rather, it must be obtained by the *ad hoc* filling in of details, presumably to better match the task. But because the units are the same (only the interactions differ) in ACT* and GRAPES, I must ask where the necessary data for this refinement came from – or, in other words, what psychological-level data must the psychologist throw away in order to form an "algorithmic" model?

I agree with Anderson (sec. 4.4, para. 3) that the ACT* "implementational level embodies all the desirable features Rumelhart and McClelland (1986) derive from neural consideration." But then I stress that connectionist models are models, perhaps just below the schema level, which are quasi-neural, but not at the neural level. In general, I believe that psychological models will involve parallel distributed models in which the units are not well represented as elements of the simple kind used in most PDP models (cf. the schemas for high-level vision in Appendix A of my target article). This same vision example leads me to doubt Anderson's claim (sec. 1.4, para. 5) that even if "different states at the algorithmic level are correlated with major differences in the states of working memory," it is also true that "states of working memory are potentially reportable," at least in a sense of "potential" that leads to easy protocol extraction. Rather, the vision example seems to show a multiplicity of processes which *must* be included in any theory that is to bear the name "algorithmic," but which are below the level of introspection.

Anderson's rich article offers many more points upon which to comment, including a discussion of tutoring *strategies* (e.g., Woolf & McDonald 1984) and of identifying "bugs" in building student models in the development of machine tutors. On another matter, I refer the reader to Arbib (1987) for my view on the modularity issue (see specifically the final section). Here, let me close with a brief summary of my reaction to Anderson's three assertions:

Assertion 1': There is an important distinction to be made between mental algorithms and their implementation.

Assertion 2': There is important basic research to be done on algorithmic issues.

Yes, but since an "algorithm" is an unambiguous recipe for carrying out some task through a structure of simple steps, I urge that the use of "algorithmic" to describe just one of the levels of algorithmic refinement be abandoned. Why not bite the bullet and call the higher level "the mental level"? There are "nonalgorithmic" levels of specification that are essential to psychological analysis, although GRAPES and ACT* by no means exhaust the useful levels of algorithmic analysis.

Assertion 3': The best way to study the algorithmic level is through pedagogical experiments.

A useful way? Yes. The best? Not proven. As a student of visuomotor coordination, I find it hard to agree with Anderson (sec. 1.5, para. 4) that "if we are to develop a theory at the algorithmic level, this theory will be focused on learning." However, I do not disagree with the importance of learning, and point to the work of Hill (1983) on language acquisition as an important example of such study at the schema level. Of course, brain damage also provides an alternative to learning for the comparison of variant mental algorithms, and Gigley (1983) offers a preliminary study in this direction with respect to

sentence understanding. The study of psycholinguistics, neurology, visual psychophysics, motor skills and language acquisition all have great value in the explication of human knowledge.

Functional principles and situated problem solving

William J. Clancey

Department of Computer Science, Knowledge Systems Laboratory, Stanford University, Palo Alto, Calif. 94304

Anderson's distinction between algorithm and implementation is useful, intuitive, and argued well. One could perhaps question his description of the algorithmic level in terms of actual "procedures that run in the mind," but a more concrete argument can be made against his claim that cognitive principles do not exist at the algorithmic level.

Anderson writes, "Only the implementational level can be understood in terms of general principles of cognition that are constant across different situations. The algorithms we possess are adapted to specific task demands and are as varied as those task demands." In contrast, recent expert systems research attempting to design "generic tools" tends to support Gleitman (1983): Recurrent knowledge organization and inference procedures for general tasks (e.g., diagnosis, planning, control) in different domains (e.g., medicine, electronics) can be abstracted from individual behavior. Indeed, these results are reflected in how expert systems researchers use the word "task" – a *kind of problem* (such as diagnosis or programming), not a specific problem to solve (patient to diagnose or program to write).

Anderson's analysis is apparently biased by research focusing on relatively formal problems, such as geometry and LISP programming. From the perspective of expert systems developed for scientific and engineering problems, cognitive science research in mathematics, typing, programming, and so forth is knowledge-impoverished. To capture what Anderson calls "a true functional level of the human mind," we must consider tasks that relate a person's behavior to some nonformal world. In general, both everyday and complex problem solving outside of formal domains like mathematics involve modeling the world in order to take action. Making selective observations, we construct and test alternative situation-specific models (e.g., alternative descriptions of disease processes in a particular patient), and relate them to action plans (e.g., therapy plans). To understand "how our cognitive mechanisms adapt to functionally important problems," as Anderson says, we must look at problems in which the problem solver is *situated;* that is, we must study problems in which the problem solver is faced with constructing a model of the outside world, within some social setting, and relating it to the needs of some task.

More specifically, engineering problems studied in expert systems research involve modeling some system in the world (a device, a manufacturing plant, a human body, a circuit, etc.) that the person is trying to design, repair, assemble, identify, diagnose, control, and so forth (called "generic tasks" [Chandrasekaran 1984; Clancey 1985]). Engineering problem solving of this type involves a modeling step to describe the world and a planning step to choose a course of action. Anderson's LISP and geometry problems involve no world to interpret; they have no functional significance in themselves. Rather, they are formal modeling tools that would be used in the context of some larger system-manipulating task, involving goals for doing something with this system in the world. This antecedent, mostly qualitative problem solving, which expert systems research focuses upon, provides the analysis that leads to a theorem to prove, a program to write, or an equation to solve.

Qualitative models can be described on different levels: the task and system, the computational method (classification vs. construction), the relational network representation (e.g., prototype hierarchy, state-transition graph, procedural hierarchy), and the implementation in a program (rules, frames, objects, etc.) (Clancey 1986). Recurrence or "principles" include both a vocabulary of relations for abstracting *processes in the world* (e.g., cause, progression over time, severity, location, flow-volume characteristics) and *cognitive processes* (often called "inference procedures") for describing complex processes in the world by explaining and predicting their behavior. For example, routine diagnostic problem solving in medicine and sand-casting can be modeled by a common set of knowledge structures and inference procedure (Thompson & Clancey 1986).

Cognitive principles of this type are not necessarily explicitly stored in the brain or even articulatable by the problem solver. Rather, they are abstractions of a grammatical form that express commonalities in the behavior of individual problem solvers. These abstractions include both *kinds of patterns* experts can articulate (familiar problem-solving situations and familiar courses of action) and recurrent *changes in attention* and *rationales for making observations* when forming a model. An example of such recurrence is the process of "triggering" a partial model on the basis of a few observations. Triggering reflects both the cognitive ability to usefully relate a new situation to past experience and the properties of a world in which processes tend to recur. Thus, the study of recurrence of processes in cognition and the world are complementary, involving the interaction of task resources and demands.

Analysis at this level contradicts Anderson's remark that "one cannot use protocol data to analyze a skill that is already compiled." In complex problem solving such as medical diagnosis, we abstract sequences of data requests (observations made by the physician) by relating them to changes in the situation-specific model (Clancey 1984). Moreover, if the problem solver has a model of how he reasons, as some good teachers do, we can ask for his description of the functional modeling goals that lie behind his questions (e.g., to detect erroneous data, or to establish temporal boundaries on the underlying cause).

The heuristic classification (HC) model of problem solving was developed to describe expert systems, but it is also a hypothesis describing human problem solving. The HC model claims that expertise (knowledge based on experience) consists of the ability to recognize situations by abstracting specific observations and relating these systems models to abstract courses of action, which are subsequently refined to meet the needs of the specific situation. Theories of problem solving based on such a model of experiential knowledge describe a computational method (HC), the modeling requirements of a task (e.g., testing hypotheses, discriminating among alternative system models), and the world (e.g., nature of the recurrence in the domain, urgency, efficiency, cost of observations, importance of ordering observations). Strikingly, problem-solving research in geometry, LISP and *pascal* programming, subtraction, algebra, and so forth ignores the inherent difficulties of *modeling nonformal systems*, in which data are uncertain and incomplete, system functionality is not axiomatic, and no written calculus exists. Consequently, this research presents an impoverished view of experiential knowledge structures and inference.

In conclusion, Anderson's call to the algorithmic level is reasonable, but application tasks for functionally important problems must be "situated," if we are to capture cognitive principles at this level. By situated, we mean, first, that the task involves explaining and predicting events in the world in order to plan courses of action (which will in turn satisfy higher goals), and second, that the problem-solving activity is itself constrained by a social context. In expert systems research focusing on "generic tasks," cognitive principles at the algorithmic level include *representation requirements for modeling processes in the world* (i.e., what is articulatable from experience must bear a

useful relation to the complexity of the world) and *inferential competence* (i.e., the constraints problem solvers for similar tasks in different domains must satisfy when gathering information and manipulating representations in the process of formulating adequate situation-specific models and action plans).

The algorithm/implementation distinction

Austen Clark

Department of Philosophy, University of Tulsa, Tulsa, Okla. 74104

I have some reservations about the way Anderson distinguishes algorithms from implementations. These reservations should not be taken to imply that there is no such distinction. Rather, if the distinction is an important one, then it is important to state it precisely.

First, it is clear that Anderson's sense of "algorithm" is much broader than that used in computer science or logic. There, an algorithm is a finite sequence of instructions, which, if followed correctly and as long as necessary, is guaranteed to terminate in finite time (see Knuth 1973, pp. 1–9; Rogers 1967, pp. 1–5). The distinction between algorithms and implementations is easy to state: The same algorithm (e.g., Euclid's algorithm) can be implemented in many different programs, perhaps written in different programming languages. Furthermore, a given program (e.g., sequence of PASCAL statements) might have very different implementations on different machines. We get the distinction between the two notions because each of those many different machine implementations implements just the same algorithm.

Perhaps this can clarify one element of the argument of Rumelhart and McClelland (1985). As long as one can point to different implementations of the same algorithm, there is good warrant for using a high-level "algorithm" vocabulary in addition to a lower-level "assembly language" vocabulary. Unfortunately, humans provide the only known implementations of cognitive-psychological "algorithms," and the warrant for using the higher-level language is correspondingly weaker (although not necessarily zero). Talk of algorithms or computations is presumably *not* talk of something over and above neural processes. Anderson sometimes writes as if there really were a distinct "algorithmic level" over and above the "implementational level"; unless we embrace dualism, these levels are dstinct only in the sense that they are distinct vocabularies we use for talking about one and the same thing.

An algorithm is essentially a *sequence* of instructions – an *ordered* set. This is not true of a set of productions, in which no rigid sequencing of instructions is established. Indeed, one of the attractions of a pattern-matching production system is precisely that one does not need to specify a rigid flow of control. To get a unique sequence of steps from a production system, one must specify principles of conflict resolution. If two productions both match patterns in working memory, competing activation levels determine which one fires.

However, the principles determining activation levels – spread of activation, inhibition, strength of productions, goal activation, and so forth – are clearly *implementational* principles. It follows that we cannot derive a unique sequence of instructions from the theory without considering details of the implementation. Now, to determine the efficiency of a production set, and sometimes even its correctness, we must be able to determine the order in which the productions will fire. If this is correct, I fail to see how efficiency or correctness can be addressed purely from the algorithmic level, as Anderson seems to suggest in Sections 1.2 and 1.5.

Finally, algorithms, strictly speaking, are guaranteed to terminate in finite time. Not all tasks have algorithmic solutions in this sense; indeed, for some problems one can prove that there *cannot* be an algorithm. Church's theorem shows that there can be no algorithm for determining which sentences of predicate logic are theorems (see Church 1936; Jeffrey 1981, pp. 125–55). If ACT* is on the right track, there is presumably some set of productions people learn for sorting sentences into theorems or nontheorems; but no such set of productions can *ever* be an algorithm for the task.

All this may seem like mere semantic quibbling: Anderson is free to use "algorithm" to mean something like "the set of productions a person learns for carrying out some cognitive task," which I'd guess is what he means. But there is an underlying substantive issue. The way in which Anderson makes the distinction seems to make far too heavy a commitment to the truth of a particular cognitive theory – the ACT* theory, or something very similar to it.

In particular. neither of the two new criteria Anderson proposes in Section 1.1 for identifying the algorithmic level seems to me to state necessary conditions for algorithms. It is startling to read that "cognitive steps at the algorithmic level correspond to changes in reportable states of working memory." This would rule out most of what Marr called "algorithms," because they have no such effect. Consider what Marr (1982) described as "algorithms" for edge detection, stereopsis, or motion perception. Why must *every* algorithm when activated have some effect on working memory? Even if we drop the claim that those effects are inevitably reportable, this still seems to give far too narrow a criterion for identifying algorithms.

Anderson says at one point that "work at the algorithmic level requires that one specify the actual production set that is running" (sect. 1.3). If he really means this, it follows that one cannot work at the algorithmic level unless there really is a production set that is "running" (i.e., unless ACT* or something similar is true), and, furthermore, that one can *specify* the actual set that is running. This is what I mean by saying his distinction entails substantive claims.

Similar sorts of substantive commitments are made in the claim that "learning takes place in knowledge structures defined at the algorithmic level" (sect. 1.1). If, as Anderson says, this is a criterion "for identifying the algorithmic level," the implication is that *all* learning is (in effect) learning of productions. Again, this might be true, but there is little ground to think the distinction will perish if it turns out to be false. Notice that Marr's edge-detection and stereopsis algorithms require no such learning. Note too that they are cognitively impenetrable.

Perhaps only philosophers will think it worthwhile to spend this much time worrying about conceptual distinctions, because that is what they spend *all* their time doing. Personally, I think Anderson's current research stands on its own merits and needs no elaborate justification or theoretical defense. To demonstrate the value of pedagogical research, it seems to me sufficient to point to some of his current work (e.g., Anderson & Skwarecki 1986) and say "See?" If others find that insufficient, I hope Anderson's target article persuades them.

The scientific induction problem: A case for case studies

K. Anders Ericsson

Department of Psychology, University of Colorado, Boulder, Colo. 80309

The dramatic increase in the complexity of theories and computer models of cognitive processes has not been matched by a methodology that yields observations sufficiently detailed for empirical evaluation of the proposed mechanisms. In his important target article, Anderson addresses this difficult problem by proposing different levels of description for cognitive processes: the level of mental algorithms and the level of their implementa-

tion. He argues that sufficiently rich observations, such as sequences of eye fixations and verbal protocols, can be used to describe the cognitive processes at the level of sequences of mental states. From such sequential descriptions of mental states, it should be possible to induce the general (algorithmic) processes specifying the sequence of steps allowing us to discover general characteristics of these processes across different task domains.

My commentary will focus on the induction problem, where algorithmic processes are induced from detailed observations of the sequential structure of a cognitive activity. I will point to some difficulties with this induction, as well as to some possible methods for dealing with these difficulties. Most observations relating to processing steps or mental states, such as eye fixations and verbal reports, reveal the heeded (attended) information at the corresponding states but provide no direct evidence about the processes responsible for bringing the information into attention (Ericsson & Simon 1984). From the protocol data, we can infer a sequence of states defined by heeded information, and it is from this sequence that processes responsible for generating those states can be hypothesized. In the ideal case, the subjects make use of the same limited knowledge so that a task analysis can specify all the alternative ways in which a given cognitive processing activity can be realized as a sequence of processing steps operating on information that is in attention. Under these conditions, protocol data are often sufficiently informative to allow us to reject most sequences of processing steps that could be hypothesized (Ericsson & Simon 1984). However, with considerable individual differences in the amount and content of the relevant available knowledge and with distinctly different learning histories, the induction problem becomes increasingly difficult.

A possible solution to this problem is to record and analyze the learning history of an individual subject, as illustrated, for example, by Pirolli and Anderson (1985). When one finds that there is adaptive use of information presented or that there is learning, one can review the protocol of the subject's previous processing of related information to induce the general processes accounting for the learning. Induction in such a complex situation will at best provide us with a plausible account rather than permitting us to reject alternative accounts. In order to significantly improve the empirical evidence for a hypothesized process, we must be able to elicit the process under controlled conditions that allow us to test our hypothesis. I will argue that it is possible to identify hypothesized processes in the context of case studies of skill acquisition and to examine the same processes under specially designed experimental conditions.

In investigations of exceptional memory performance (Chase & Ericsson 1981; 1982; Ericsson & Polson, in press; Mueller 1911; 1917; Staszewski 1986), the same subjects have been studied several times each week for months and even years. The cognitive processes of the these subjects have been monitored both through retrospective verbal reports and through conventional performance measures. From such observations it has been possible to identify reliable and stable characteristics of both encoding and retrieval processes that differ between subjects (e.g., differences relating to the types of knowledge used for encoding presented digits). A mathematics professor studied by Mueller (1911; 1917) used his extensive knowledge of numbers to form meaningful associations ($451697 \rightarrow 451 = 11 \times 41$; $697 = 17 \times 41$). A subject studied by Chase and Ericsson (1981; 1982) used his experience as a runner to encode digit groups as running times for various races ($349 \rightarrow$ 3 minutes, 49 seconds – near world-record mile time). For the latter subject, Chase and Ericsson (1981) induced the rules specifying which digit sequences would be encoded as running times and which would not. In an experiment, their subject was presented with only encodable digit sequences or only digit sequences not encodable as running times, and dramatic differences in performance were found. Many other characteristics of the cognitive processes of memory experts have been examined by specially designed experiments.

The scientific induction problem for case studies appears to involve at least three steps. The first is to form hypotheses about information-processing activity on individual trials, something Anderson has already discussed. The second step is to identify stable information-processing activity across trials for a given individual and, ideally, to validate its induced structure with experimental tests under systematically controlled conditions. The last and most critical step is to seek generalizable characteristics of memory skill across different individuals. There is remarkable consistency in the general characteristics of memory skill; Ericsson (1985) showed that the skill of a large number of memory experts could be readily characterized using the principles of skilled memory proposed by Chase and Ericsson (1982).

Once the knowledge and cognitive processes used by a given subject in a task are well understood, issues of generalizability across subject matter and task domains can be addressed in transfer experiments. By systematically varying the content and structure of tasks. not only can detailed hypotheses about generalizability of processes and knowledge be evaluated against performance data, but the access and mediation of particular knowledge can be monitored through verbal reports, as shown, for example, in Ericsson and Polson (in press).

In sum, the case studies described above can be viewed as illustrations of the general methodological approach proposed by Anderson. These studies not only show that such an approach is feasible but also support Anderson's conjecture that generalizable mental procedures can be identified across individuals and across knowledge domains.

The evolutionary aspect of cognitive functions

J.-P. Ewert

Department of Neuroethology, University of Kassel, D-3500 Kassel, Federal Republic of Germany

Anderson emphasizes that one may never understand human cognition until one realizes how it adapts to functionally important problems. He provides convincing methodological material to support the claim that the study of cognition at the algorithmic (rather than the implementational) level will yield important insights into fundamental properties of human knowledge, and that this may have an impact on progress in psychological research per se. However, because these methods do not allow one to compare comparable cognitive mechanisms of vertebrate animals, Anderson's notions about the uniqueness of human cognition are not well supported. An algorithm resulting from the analysis of human mental functions is probably different from the algorithm that the underlying neuronal substrate is using (see Creutzfeldt 1983, p. 426; 1986, p. 16). One can only compare algorithmic levels of cognitive processes between man and animal in terms of neuronal structures and related information processing. It appears to me that the way fundamental processes of cognition (perception, memory, retention, recall, retrieval; see Lynch et al. 1984) are implemented in the human brain can come to be understood in terms of the evolution of related structures and functions of the vertebrate brain (for discussions see Bullock 1983; Griffin 1982; Northcutt 1981; 1986; Ploog & Gottwald 1974). In the following, we will mention some general aspects of cognition and then ask whether an algorithmic/implementational terminology can also be applied to neural systems that mediate primordial homologues in lower vertebrates.

According to Anderson, human cognition – a system designed to achieve domain-specific applications – is an "artifact of evolution." Putting aside the question of whether this term is

appropriate, we agree with Anderson's notion that the "ability to acquire new abilities to deal with novel problems" is a uniquely human attribute as far as the linguistic system and its verbal capacity is concerned (e.g., see Arbib et al. 1982). Anything that depends significantly on language, its grammar, its symbolic structure, and its logic, may not be available to nonverbal species (Schopenhauer, 1883, even suggests that the superior structure of certain languages, such as classical Greek, permits better construction of thoughts and their connections). All this implies that there are mental perceptual processes that are built out of linguistic competence and experience. Walker (1983) points out that the question here is whether language-dependent mental processes are added to, develop from, or take advantage of neural organizations and representations that arise in the brains of vertebrate animals in general, or whether nothing would be left that resembled human cognitive functions if linguistic devices were excluded.

To prevent misunderstanding, some definitions are in order here. *Cognition* is the process by which knowledge is gained about the world; this is not necessarily bound up with linguistic descriptions. *Mental function* refers to internal activities influenced by memories of past experiences and expectations of future ones (Paillard 1987). More specifically, these may be organized events in neural tissue occurring in response to antecedent (internal or external) signals that they classify, transform, and coordinate prior to initiating actions (at least potentially) whose effects can be anticipated to the extent that available information permits. *Acquired knowledge* is thus a kind of representation of previously experienced situations that can be recognized and eventually foreseen. *Perception* is the use of the senses to acquire abstract knowledge of the outside world by combining certain pieces of sensory information into coherent, perceptual wholes (e.g., see Ewert 1987a). This means that a couple of separate events are not sufficient to define a significant stimulus pattern or spatial structure, respectively; an algorithm describes the rules about the relationships between events, to which the appropriate perceptual systems are adapted.

Piaget (1971) proposes the term *schema* for these "perceptual wholes" that correspond to units of knowledge – mental entities – that are prerequisites for cognitive functions. A schema encodes local and contextual knowledge about the criteria for recognizing an object. Uexküll (1909), Lorenz (1943), and Tinbergen (1951) have shown that animals possess schemas as internal "small-scale models of external reality" (Craik 1943); the fundamentals of animal cognition can accordingly be evaluated in terms of the qualities and interactions of their perceptual and motor schemas (cf. target articles by Arbib and Ewert, this issue; see also commentary by Ewert on Arbib's target article). Schemas are the means whereby man and animal perceive a symbolic picture of their physical environments (Creutzfeldt 1986, p. 22). These schemas do not copy the environment but provide significant information about behaviorally relevant sensory and functional aspects of it. With the aid of its innate releasing mechanisms, a toad uses inherited abstract knowledge about its prey by comparing the spatiotemporal aspects of configural visual cues with an inborn prey schema. A bird uses inherited knowledge about nest building by placing pieces of straw in the form of a nest according to an inborn nest schema. An important property of schemas is (1) that they refer to the physical original only *approximately,* in an abstract fashion, and (2) that they can be modified by accumulating individual experience through the introduction of new features (e.g., by classical conditioning), or they can be elaborated by utilizing new feature relationships (e.g., operant conditioning). The latter processes, together with the context-dependent formation and combination of schemas into symbols and abstract concepts (ideas), characterize an aspect of human cognition. So far, we follow Hume (1902) when he argued that a theory of human cognition will acquire additional authority if we find that the same theory is requisite to explain the same phenomena in other animals, thus implying common, basic neural mechanisms underlying human and animal knowledge and suggesting that human superiority is a matter of degree. (The discontinuity results from language, which obviously involves special tuning of neural systems, permitting stages of intelligence that only humans can reach.)

Primate and nonprimate mammals possess comparable kinds of "neuronal machines" (Creutzfeldt 1983; Eccles et al. 1967; Hebb 1949; Hubel & Wiesel 1977; Mountcastle 1957) suitable for constructing schemas as internal representations. Being percept- or act-related cell assemblies, these functional units amount to perceptual and motor schemas and can be traced back to equivalent units in lower vertebrates such as toads (see target articles by Ewert and Arbib, this issue). According to our current concepts, object perception and cognition proceed in parallel/hierarchically organized distributed connectionist networks (Rummelhart & McClelland 1986) consisting of interacting functional modular units (e.g., Creutzfeldt 1983) that represent "presently excited, distinctive neural gnostic organizations" (Bindra 1976). Hence, cognition can be analyzed in terms of aspects of brain function and its evolution with respect to differentiation, take-over, addition, and conservation of function (e.g., Ebbesson 1984; Walker 1983).

It is known from comparative neurology that many functional properties of the human brain can be traced back to related properties of anatomically homologous structures of lower vertebrates (see Ebbesson 1980; Northcutt 1986; Vanegas 1984). In general, we can say, for example, that all terrestrial vertebrates, including man, have certain species-specific sensorimotor coordinations for head and body turning wired into the midbrain and hindbrain, and that object recognition and adaptation to novel signals require forebrain (telencephalic/diencephalic) intervention. The optic tectum, whose intrinsic circuitry is responsive to retinal input alone, functions for "noticing," whereas its connectivity with prosencephalic structures enables the system to "examine" and "recognize" signals; the latter is cognitive and controlled by context (Ewert 1987b). According to Paillard (1987), the turning of sensorimotor instruments can be regarded as a primordial form of adaptive response to environmental constraints that have prepared the way for the emergence of cognitive instruments for the control of perception and action. In mammals there are forebrain structures – for example, the hippocampus, a component of a limbic/diencephalic system significantly involved in memory, and homologous structures of the medial pallium (or "primordium hippocampi," Herrick 1933) with partly comparable functions – that can be traced back to amphibians (Ewert & Finkenstädt 1987). These homologous structures receive appropriate inputs from sensory systems and serve to distinguish between previous familiar and novel events (Gaffan 1976; Vinogradova 1975) and to accumulate knowledge of local geography by providing "cognitive maps" (O'Keefe & Nadel 1979; Tolman 1948).

We therefore conclude that it is not fully justified to view the "ability to acquire new abilities to deal with novel problems" as a uniquely human attribute, because essential equivalents can be traced back to homologous neural substrates and functional properties in vertebrate animals down to amphibians. Consequently, one can inquire into the phyletic roots of cognitive processes at both algorithmic levels (defined by Anderson as constructed by the mental procedures that enable us to behave adaptively) and implementational levels (defined as the mechanisms that implement those procedures). For example, in toads the algorithm for prey-catching is given by Figure 1 (see Ewert's target article, this issue). This algorithm may be inherent in "sensorimotor codes" of command releasing systems, CRSs (as described in sect. 6.3). Does the implementational level refer to sensorimotor functions of CRSs? Can sensorimotor codes – whose prosencephalic/mesencephalic circuitry determines innate rules according to which the relations among visual cues describe prey in space – be interpreted in terms of species-

specific cognition? Do the functional units of interconnected cells that determine tectal T5(2) prey-selective neuronal properties in the manner of a prey schema implement their function according to a "wired-in algorithm"? Does information provided by "modulatory limbic-diencephalic circuits" that enable toads' CRSs to behave adaptively in relation to past and present events pertain to an algorithmic level?

In summary, we think that the comparative study of brain function helps us understand why animals may use different algorithms for similar tasks or similar algorithms for different tasks, respectively (see also commentary by Ewert on Arbib's target article, this issue). The algorithm the animal uses is the expression of its implementation. The implementation cannot be fully understood without an evolutionary history. The evolutionary story cannot be told at the algorithmic level.

The study of cognition and instructional design: Mutual nurturance

Robert Glaser

Learning Research and Development Center and Department of Psychology, University of Pittsburgh, Pittsburgh, Pa. 15260

From a historical perspective, Anderson's target article continues a line of thinking that Dewey, Thorndike, Skinner, and others pursued on the relationships among learning theory, pedagogical experimentation, and instructional science. The underlying assumption with each has been that behavioral science research will uncover fundamental laws of learning and parameters of individual differences that can be applied to educational practice. In Thorndike's (1922) analysis of S-R (stimulus-response) bonds in learning school subjects, in Skinner's attempts to extrapolate operant analysis to programmed instruction [see Skinner: "An Operant Analysis of Problem-Solving" *BBS* 7(4) 1984], and in attempts by statistical learning theorists such as Atkinson (Atkinson & Paulson 1972) to develop optimal practice programs for verbal performance, this assumption was clear. Gestalt psychologists such as Wertheimer (1959) also considered the potential of their theories for informing the teaching of thinking, problem solving, and understanding. Not often explicitly stated was the underlying proposition that attempts at pedagogical application can provide a significant test of scientific knowledge, raising issues that help amend current theory, generate new theory, and force more complete explanations of human performance. Certainly in other sciences this two-way communication between theoretical research and attempts at application has generated active fields of basic research and raised fundamental research questions.

Many behavioral scientists now believe that the potential of their science for influencing pedagogical practice has grown markedly. Greater understanding of cognition and the structures of human knowledge have led to increasing information about complex human activity. Findings are accumulating on how humans acquire competence and skill in language, mathematics, and science, and in various forms of expertise learned over the course of schooling, training, and experience. This newly uncovered richness in scientific descriptions of human performance contrasts with the psychological analysis of information-lean processes that were the focus of older theories of human intelligence, learning, and perception. The modern discovery that human performance entails complex structures of knowledge has advanced our understanding; our awareness of the influence of these memory structures on cognition demands a new level of lawful explanations of human behavior. Anderson's target article is a significant contribution to much-needed discussion of methodological questions that arise in the study of knowledge acquisition.

In support of Anderson's analysis, the argument for research at the algorithmic level is currently overwhelming. Many areas of investigation show changes in theory and data that are reflected in Anderson's suggested approach. Early studies of information processing in problem solving, of development in children, and of the design of artificial intelligence expert systems concentrated on knowledge-lean, generally intelligent processes, such as general heuristics of search, increases in the power of memory with maturation and the computational power of computer systems. With research experience and accumulating evidence, a further shift in paradigm has occurred. The study of high levels of human competence and expertise has revealed the involvement of organized, integrated structures that are integral to effectiveness in thinking and problem solving. In developmental psychology, studies on competent children's knowledge in a subject-matter domain indicate that structure and cohesiveness facilitate access to, and use of, the knowledge in a way that makes them act more developmentally mature than novice children their own age. In the design of expert systems, it has been more productive to emulate the specialized knowledge structure and methods of an expert than to simulate the processes of general intelligence and the heuristics of general problem solving.

These findings uncover the significant qualities of competent performance that emerge when an individual engages a highly organized knowledge structure or appropriate schema that includes concepts, models, planning mechanisms, and metacognitive strategies. These structures do engage fundamental processes of the human system, but learning ability and intelligence also seem to be significant functions of knowledge and knowledge-organizing capability. Thus, the present challenge is to study competence and learning in terms of the interplay between knowledge structure and processing ability – at what I interpret to be Anderson's algorithmic level. Performance analyzed at this level illuminates a critical difference that is of major significance for education, a difference between individuals who have acquired considerable competence in particular domains of knowledge and skill and those who are less competent. The acquisition and efficient utilization of an organized body of conceptual and procedural knowledge should now become a major arena of study for cognitive science.

Although I agree with Anderson that substantial promise for the study of learning and more interesting and important theory reside at what he calls the algorithmic level, I believe we should also consider how the study of implementational processes will be influenced. To use his computer analogy, the program and data structure are of enormous consequences to performance, but they are significantly influenced by machine implementation or architectural features of the system (Anderson instead suggests they are largely independent). In certain research endeavors, the connection between research at the algorithmic and the implementational levels will be important to maintain; research at the implementational level on phenomena of memory and perception will be forced to revise its concepts by research findings at higher levels of performance. The memory capacity of a child who is knowledgeable in a specific domain and the rapid pattern recognition of the expert in a particular field will both need to be explained eventually at these two levels of description. With this in mind for the long term, investigation and theory development at the algorithmic level on how humans acquire domain-specific knowledge and skill should proceed now as directly as possible. As Anderson proposes, a reasonable methodology is to attempt to produce, through pedagogical experimentation, the performances that humans can attain. Theories of learning will be generated and tested once we attempt to understand the factors that produce or retard the acquisition of knowledge and the properties of the tutorial situations involved.

Pedagogical "engineering" experiments are highly appropriate at our current state of knowledge about individual differences and the nature of competent performance. A lesson from experience, however, is in order. In the past, attempts in educational psychology to study individual abilities in learning were weakened by an unfruitful combination of two streams of psychological science – the study of individual differences in the psychometric tradition, on the one hand, and the study of conditions of learning in relatively simple situations, on the other. No common theory connected these two endeavors. Nevertheless, for some years studies were carried out on what was called aptitude–treatment interaction (ATI). The results of these studies were carefully documented by Cronbach and Snow (1977) and others, and indicated a singular lack of empirical regularities. In hindsight, it was an impossible task. Variables studied in the learning laboratory, such as massed and spaced practice and inductive and deductive learning, were juxtaposed with traditional assessments of mental abilities and personality variables; apparently, at the analytical levels investigated, the underlying processes and mechanisms involved had little in common. At present, cognitive psychology shows progress that may produce more fruitful contact. Individual differences are being analyzed in information-processing terms, as are the learning strategies of individuals; and protocol-analysis techniques are bringing these together in the study of single cases. From this point of view, attempts to make advances in learning theory that encompass individual abilities and dispositions may become more theoretically and practically effective.

Anderson's discussion of pedagogical variables and conditions for learning is somewhat one-sided. Based upon the kinds of tutors he has built, the subject matter he has studied, and the tutorial tactics he has used, he claims that learning by induction – that is, working with examples, making mistakes, correcting them, and inducing the correct procedures – is much more "interesting" than learning by instruction – that is, learning from being told. The latter, I assume, includes learning from listening, from modeling, from apprenticeship situations, and from being assisted in using metacognitive strategies for learning. These are certainly forms of learning that cannot be ignored and that will need to be investigated regardless of one's tutorial predilections. So when Anderson says that his claim that one form of learning is more interesting than another may be too strong a claim, I concur.

To conclude, investigations in different fields with various approaches – the study of highly competent performance and expertise, the study of the nature of human development and its structure-dependent transitions, and the study of expert systems in AI – all converge on the significance of "knowledge structure-process interactions" and the importance of knowledge organization and representation in facilitating human ability. Anderson's proposal for work at the algorithmic level of analysis and his methodological statement are supported by this evidence. The form of investigation he proposes should assist in the development of learning theory that meets the complexity of human performance, and it should also inform the study of fundamental human architecture at the implementational level. A major challenge at the algorithmic level is to derive connections between learning theory and instructional theory more systematically so that, as learning theory is developed in various domains, normative principles for achieving knowledge and skill are identified. Instructional intervention methodology can test the adequacy of theory, not only as a scientific description of experimental data, but also as a source of prescriptive heuristics that can guide the design of instructional systems. In keeping with the spirit of Anderson's paper, Miller (1986) has written, "When the next experiment in any program of research comes to seem too trivial to justify the effort of doing it, a useful application can do wonders to revitalize and redirect the work" (p. 295).

Ambiguities in "the algorithmic level"

Alvin I. Goldman

Department of Philosophy, University of Arizona, Tucson, Ariz. 85721

I am not unsympathetic to the idea of research "at the algorithmic level," but there seem to be some ambiguities in what Anderson means by this phrase. In some places, he apparently means the study of algorithms themselves – that is, mental procedures, or what he calls (in the ACT* framework) "productions." In other places, "research at the algorithmic level" appears to refer to the study of how procedures are *acquired* or *learned*. These interpretations are quite different. As Anderson points out, procedures themselves are typically task-specific. On the other hand, principles for the learning or acquisition of procedures could be the same across all tasks and domains. Indeed, it seems clear that Anderson believes that acquisition (and tuning) principles *are* the same for all domains, because he formulates such a unique set of principles in Anderson (1983). It is therefore puzzling to find him writing that "only the implementational level can be understood in terms of general principles of cognition that are constant across different situations" (para. 3). Given the theory of production learning in Anderson (1983), he seems committed to the notion that there are principles of algorithmic learning that can be understood in terms of general principles of cognition. So if research at the algorithmic level is, or at least includes, the study of production acquisition, then the algorithmic level must also feature principles of cognition that are constant across situations. Of course, the researcher who wishes to investigate such learning principles is well advised, as Anderson stresses, to sample what goes on in many task domains. But this only means that empirical evidence should be drawn from varied instances; it does not conflict with the claim that there is a common set of learning principles.

This apparent inconsistency might suggest that the better interpretation of what Anderson intends by algorithmic level is the study of algorithms themselves, not the study of principles of their acquisition. There are two problems with this interpretation. First, it doesn't sit well with the fact that he often formulates his endorsement of algorithm-level research in terms of the *learning* of algorithms (see his statement of point 3′, para. 3). Second, it is doubtful that the study of algorithms themselves is a suitable topic for psychological theory. The class of cognitive domains is indefinitely large and varied: science, religion, telephone dialing, bridge playing, and so on. Moreover, the particular sets of procedures that different cognizers possess in any one of these domains will be peculiar to their specific experience, historical situation, tutors, and the like. It seems implausible to suppose that psychology is the appropriate discipline to study this enormous range of differences, or all the variables that influence it. Psychology seems well equipped to study the general mental mechanisms of procedure acquisition, not the variety of procedures themselves. To use the phraseology I employ in Goldman (1986), psychology can shed light on basic mental "processes," but not on the innumerable intellectual "methods" that different cognizers encounter and internalize.

Anderson points out that differences in production sets probably account for most of the variance in human behavior. Does it follow from this, in addition to the assumption that psychology is interested in the study of human behavior, that psychology should study these differences? No. Of equal importance in accounting for the variance in human behavior are differences in information stored in declarative memory, the cognizer's "beliefs" or units of (unproceduralized) "knowledge." But it does not follow from this that psychology should try to study all factual subject matters. Only the common principles of learning – both belief acquisition (and retention) and procedure acquisition (and retention) – are natural targets of research in cognitive psychology.

There is another aspect of the algorithmic level that is prominent in Anderson's target article. This is the conception of a level that *abstracts from implementational details*. This marks a distinction familiar from the work of many previous writers (as Anderson notes): the program/hardware distinction, the algorithmic/functional architecture distinction, and so on. However, this way of drawing the distinction does not seem to be equivalent to either of the two others ways of drawing the distinction discussed above. For example, because production acquisition can be studied at either a higher or a lower level of abstraction, the algorithmic level as defined by the study of production acquisition cannot be equated with the algorithmic level as defined by abstraction from implementational details. Thus, Anderson's exposition suggests two or more *different* distinctions, not a single distinction.

The possibility of multiple distinctions is further suggested by another feature of the discussion. Anderson picks out speed and reliability as the critical factors in activation-based processing, and chooses these as the ones from which the algorithmic level should abstract. But why choose precisely these factors as the ones from which the higher level of analysis should abstract? In principle, there seem to be indefinitely many processing details from which a researcher might wish to abstract (for certain purposes). The general idea of abstracting from processing details does not fix a unique distinction between implementational and algorithmic levels. I would invite Anderson to elaborate on just which processing details should be omitted and the rationale for this choice.

These comments are not meant to express doubts about either the legitimacy or utility of algorithm-level research in general or Anderson's current research program in particular. But before we assess the cogency of his case for this general type of research, it would help to have a more precise delimitation of the way in which such research is to be understood.

A flawed analogy?

James Hendler

Department of Computer Science, University of Maryland, College Park, Md. 20742

Anderson, with appropriate reference to Marr (1982), bases much of his target article on a discussion prompted by a division of cognition into two relatively decomposable layers: the *implementational* and the *algorithmic*. Anderson says that "the obvious analogy is to the standard computer, where programs and stored data structures correspond to the algorithm, and the actual machine and its operation correspond to the implementation." He goes on to liken his two levels to Pylyshyn's (1980) distinction between *algorithm* and *functional architecture* and points out that "Pylyshyn likewise draws heavily on the computer analogy to articulate his distinction." In this commentary, I wish to question whether the arguments based on the near decomposability of these levels are valid. In particular, I hope to show that the very analogy drawn on, that of the "standard computer," helps demonstrate problems with the algorithmic–implementational dichotomy and the assertions based thereon.

Let us consider what happens in a standard computer during a standard programming task. We start with a high-level goal – for example, "compute the mean for a set of grades." This goal is broken down into an algorithm, a carefully specified procedure giving step-by-step operations on a set of data structures, which must also be defined. In our averaging example, we start with the algorithm (sum the grades into a total and divide this by the number of grades) and some decisions about how the grades are to be entered and stored (e.g., are they to be read from the keyboard and stored in an array, or are they read from a file and operated on as read, etc.). Most computer scientists would refer to the latter decisions as *implementational* details, but Anderson would have us categorize these as algorithmic structures.

As the code is prepared to run on the computer, it goes through many stages. At the top level are the many programs for computing an average. One such is

```
(defun AVG (x)
        (quotient (apply 'plus x) (length x)))
```

But, of course, we could have entered many variants, some with loops, some with recursion, or we could even have hand-coded the program for some particular number of arguments. The code is now run through a compiler to provide intermediate code such as a LISP machine's

```
20 MOVE D-PDL FEF|6
21 MOVE D-PDL ARG|0
22 (MISC) APPLY D-PDL
23 MOVE D-PDL ARG|0
24 (MISC) LENGTH D|PDL
25 (MISC) %DIV D-RETURN
```

This intermediate code would look very different on any of the many other non-LISP machines that run LISP. However, we could simply have typed this as the program for computing an average and thus by the definition above, we are still looking at an algorithm.

As the standard computer prepares to run this program, yet more decomposition occurs until we reach the memory instructions, a set of 1's and 0's that represent the program at the machine level. At this point, we'd certainly like to think we are at the implementational level, but we still have a program, with associated data structures (which we could have typed in directly on many machines), and thus we might still have to consider ourselves as discussing algorithmic issues.

The problem here is that we have been focusing on the code itself as it passes through various transformations between levels of description. We start at what is clearly an algorithmic level and proceed by small steps to the implementational. Where does the transition take place? At the editor level, when we compose the LISP code? At the compiler level, when the intermediate code is produced? At the assembler level, when that code is turned into machine instructions?

How can we claim that all these levels are nearly decompositional and claim to have an understanding of this program, or of programming in general, if we cannot describe the algorithms and mechanisms that produce the transformations of the code? How can we gain an understanding of cognition without theories that bridge levels and describe the mental algorithms and cognitive mechanisms that allow information flow through these layers? If we focus on the distinction between algorithmic and implementational, we may miss important issues arising from these transformations. If we base our research on this distinction, we may ignore important cognitive mechanisms.

This problem arises if we try to pursue Anderson's lines of argument about the validity of different methodologies and models by examining some spreading-activation models. If we look at programs like ACT* or NETL (Fahlman 1979), we see architectures that run the activation over a network and supply information that higher-level processes can use. Thus, the implementational and algorithmic levels can be distinguished fairly easily, assuming one is willing to make the cutoff at the point where those programs do. If, however, we look at two other spreading-activation systems, local connectionist models (cf. Feldman & Ballard 1982) and symbolic marker-passing algorithms (the modern descendants of Quillian 1966), the distinction is less clear.

Anderson proposes that connectionist models are implementational. He justifies this, in his discussion of the Rumelhart and McClelland work, with the claim that "the authors would not want to posit an equivalance between changes in reportable states of working memory and changes in the activation levels of their connectionist models." Recent local connectionist models, however, do have states that may correspond to these. For example, Shastri (1985) proposes a model for doing hierarchical reasoning on such an architecture, and Cottrell (1985) proposes a model for performing word-sense disambiguation. Thus, what seem to be algorithmic functions are occurring at what Anderson calls the implementational level.

Symbolic marker-passing algorithms seem to go the other way. Charniak's (1983) marker-passer and my own (Hendler, in press) use algorithms that propagate symbolic information over a network to find inference chains used by a natural language processor and planner, respectively. This model of marker-passing involves the propagation of a large amount of information and uses various types of heuristics for analyzing the returns. I would be hesitant to call our models implementational (I certainly wouldn't claim they were neurophysiologically correct), but they seem to operate below what Anderson regards as the algorithmic level.

To get a handle on the psychological reality of local connectionism, marker-passing, and other such models, it is not enough to run only the experiments Anderson proposes at the algorithmic level or the traditional experiments at the implementational. Clearly, the field of cognitive psychology must expand its methods to include new innovations such as Anderson's learning experiments, but we shouldn't (as his target article might cause us to believe) limit experimentation to looking solely at the two levels. New experimentation and cognitive modeling must examine the boundaries between different phenomena, the transformation of information from high-level symbols to the neurophysiological level. Without this we will never be able to describe the overall architecture and mechanisms of cognition.

Generality and applications

Jill H. Larkin

Department of Psychology, Carnegie-Mellon University, Pittsburgh, Pa. 15213

Decisions about what to work on are among the most important ones a researcher makes. Anderson's target article, while arguing the promise of one kind of research, provides a thoughtful discussion of general criteria for good research. The purpose of this commentary is to consider further three issues Anderson raises – the implementation–algorithm dichotomy, the need for general results, and the role of applications – and to elaborate their merit and basis as general principles for productive science.

The implementation–algorithm dichotomy. Anderson argues that psychological algorithms are different from their implementational mechanisms and that there is a clear and nonarbitrary boundary between them. But throughout science there are always difficult problems about how to cut a domain into manageable pieces, either according to phenomena (e.g., high-energy physics) or by level of detail (e.g., atomic physics and chemistry). When the cut is appropriate. progress is independently possible on both sides. Chemists can understand the interaction of atoms by largely ignoring all but the outer shell of electrons, whereas atomic physicists study these internals largely ignoring interactions between atoms. Yet this partition of fields *is* somewhat arbitrary, and there are always regions of overlap. The trick is to cut at a point that yields a theoretical structure that (a) covers an interesting range of phenomena and (b) can operate relatively independently of phenomena outside its domain.

Unfortunately, in new and complex fields, the place for the theoretical cut may not be obvious. Anderson argues that the algorithmic category should include processes that are reportable and correspond to working-memory transitions, and processes that change through learning. Although these criteria are sensible, there may be other equally sensible cuts. For example, there are processes (e.g., language, vision. and imagery) that seem to be particularly easy and efficient for humans. Furthermore, because the mechanisms of these rapid processes seem to interact little with slower, more conscious logical reasoning, it might be useful to treat these processes as theoretical black boxes, akin to Anderson's implementational level.

In short, cutting out a useful algorithmic domain depends on some subtle judgments about what are basic units of human thought, units whose internal details do not strongly affect the algorithmic processing. If this cut can be made well, then productive research can occur on both sides of the cut, each informing the other, but neither dependent on progress in the other.

The need for generality. My only serious disagreement with Anderson's target article concerns its downplaying of the need for general scientific results. Anderson states in several ways that generality is to be found in implementation and not in algorithms; he never clearly describes the kind of scientifically useful results that might come from studies of algorithms. The reader is left with the unfortunate inference that these results may be a grab-bag of domain-specific models, plus a few practical applications. This is not satisfactory; science does not progress unless last year's results are something that can be built upon this year. Fortunately, I think Anderson is wrong in saying that algorithmic results must be domain-specific. First, as he suggests briefly, it is possible that a relatively small number of learning mechanisms, operating in a wide variety of environments, could account parsimoniously for many domain-specific task algorithms. Anderson (1982) has proposed such mechanisms himself, and this is the central point made by Simon (1981). Second, as we understand domain-specific algorithmic models better, we may develop a set of constraints that can be placed on all such models. Finally, we may develop tools (like the production system) that are consistently useful in building algorithmic models. Not only do such tools make work easier. but understanding why they are useful can be a path to understanding general properties of human information-processing algorithms.

Contributions of an applied science. Anderson discusses the special value of using pedagogical experiments to test algorithmic theories – that is, using an applied domain as an experimental setting. However, merely using an applied science as a testbed overlooks its value as a source of theoretical questions. Basic research scientists rightly emphasize *answering* questions through disciplined laboratory work in which much is suppressed so that selected results can be seen clearly. But *finding* good questions requires a broad view, without premature narrowing that may exclude the most interesting issues. Connection with applications can be a valuable source of new questions, because successful applications cannot neglect any issue of practical importance. For example, I recently studied "traditional" computer-based instructional programs that are (a) used for instruction and (b) tutor the student in problem solving. In all cases, far more than half the computer code is devoted to the screen presentation. This observation suggests that using computer tutors as a test of psychological theory will require good models of knowledge presentation. This aspect of pedagogy has been largely neglected by cognitive scientists working with intelligent tutors and illustrates the

importance of using an applied domain as a resource for identifying central questions.

Connectionism and motivation are compatible

Daniel S. Levine

Department of Mathematics, University of Texas, Arlington, Texas 76019

The distinction between algorithmic and implementational levels drawn in the target article is a useful one. I believe, though, that this distinction has implications for connectionist (or neural network) models beyond those that Anderson has explored.

The question of algorithmic understanding of human problem solving is inseparable from the study of goal-directed behavior. Studying goals, in turn, implies studying motivational influences. The pedagogical issues Anderson raises clearly include motivational effects. For example, learning is influenced by the learner's attitudes toward his teacher, toward previous encounters with the body of knowledge to be learned, or even toward computers if they are used in the instructional process. Tikhomirov (1985) has shown, moreover, that actual problem-solving methods used in a given task differ with the motivational level of the solver. In particular, those who are given higher motivation (for example, by being told that a task is a test of their abilities) formulate more detailed subgoals and alternative strategies than those given lower motivation.

Connectionist models have often ignored motivational variables, but not always. The three major exceptions have been Grossberg (1971; 1975; 1982); Barto and his colleagues (Barto et al. 1983; Sutton & Barto 1981; and Klopf (1982). Grossberg (1975) and Grossberg and Levine (1975) have included drive representations, so that an organism can choose which incoming events are motivationally significant and can selectively enhance those events. A similar role, though arising from a different mechanism, is played by the "adaptive critic" of Barto et al. (1983). Grossberg (1982) develops this theory further with distinctions between the methods for processing familiar and novel events.

Higher-level cognitive processes and problem solving in humans are generally conceded to involve the association areas of the neocortex, particularly the frontal cortex. Levine (1986) has suggested ways in which the frontal lobes are likely to influence some previously developed neural network models due to Grossberg and his colleagues. These suggestions are based on many experimental and clinical studies (Fuster 1980; Nauta 1971) showing that the frontal lobes are important precisely because they integrate inputs from subcortical emotional and visceral processing areas as well as inputs from the sensory cortices. Hence frontal damage weakens the influence of current reward or punishment on behavior. The weakening of reinforcement signals can lead to effects such as inability to change category boundaries once established (which has been simulated by Leven and Levine, submitted) and excessive attraction to novel stimuli.

For the level Anderson advocates studying I am uncomfortable with the word "algorithmic" because of its connotations of a premeditated set of strategies to handle a definable task (which is not, I believe, what Anderson had in mind). Perhaps "heuristic" would be a better word for this level. The heuristics for real-time problem solving are not only different, as the target article emphasized, between novice and expert, but they often evolve in the course of solving the problem. To understand the laws (not rigid rules, but rules of thumb) by which this takes place, it is helpful to have a good connectionist understanding of the mutual interactions within the "triune brain" (MacLean 1970). Leven and Levine (submitted) instantiate triunity in neural networks by studying the interdependent effects of emotion, habit, and novelty on cognition.

The neural network studies to which I have referred are not necessarily definitive, but they are illustrative of one direction in which connectionist models are now proceeding. As these models evolve further, they will be less restricted to what Anderson calls the implementational level and will include more of the algorithmic or heuristic level. With the current explosion of both theory and data, this should be a matter of a few years rather than decades. The "physics envy" in psychology that Anderson rightly criticizes will soon be laid to rest, not, one hopes, by constructing the "human mind out of unmotivated postulates" but by describing important mental processes with motivated principles.

Nonverbal knowledge as algorithms

Chris Mortensen

Department of Philosophy, University of Adelaide, Adelaide, S. A. 5001, Australia

Anderson's case for algorithms is nicely argued. However, it is worth stressing a limitation of the proposed methodology, which he himself notes (sect. 1.1, para. 1) but sets aside. Because, as he points out, we are evolved creatures, much of our information acquisition is bound to be prelinguistic (or as I prefer to put it, nonverbal). Anderson remarks that in the case of the visual system, algorithms will be much closer to quite concrete implementational structure. I will argue that the ubiquity of sensory information in working memory requires us to admit algorithms that are not so easy to treat on his verbalist model and that, though open to computer pedagogy, point to an expanded conception thereof.

The advantage of the algorithmic approach is that it operates nicely on a level with working memory differences and with reports (protocols) of processes inside the CNS. I think it is reasonable to dismiss with Anderson (sect. 4.4, para. 7) the objection that such a level should not be too realistically conceived (as a Dennettian might argue): The realism is to be justified by the fecundity of the approach. So a philosopher might ask whether *verbal* protocols exhaust the information we consciously and unconsciously store and use in production memory. The point is that the sensory information that is in us and is used in matching and action far outruns our capacity to report its content (compare police identikit pictures with verbal descriptions). One might feel inclined to dub this "nonverbal or not wholly verbal knowledge or belief," except that its role in action (productions) forces us to expand the notion to include ideas like intention and desire as well; thus the more general "nonverbal propositional attitudes."

To describe sensory information as nonverbal should not be misconstrued as divorcing it from *algorithmic* investigation, however, if only because of the strong interaction with memory and procedures whose contents are more fully verbal. Any account of the workings of the mind has to deal with the (partial) translations of information between the parallel processors, especially the important special case of verbal protocols of sensory knowledge. To this can be added the observation that discredited empiricism was not so grotesquely wrong: Our information-storage and processing systems are simply saturated with the sensory. It is hardly unexpected that learning theory felt that it had to deal with this. On the other hand, the evident parallelism of the processors should incline one to accept more easily the significant autonomy or irreducibility of different modes of representation or content that the brain uses, parallelism also being the natural cashing of Anderson's "domain specificity." Nor should the present suggestion be misconstrued

as necessarily maintaining some sort of *literal* mental-image story, because, as I have argued elsewhere (Mortensen 1985), this is a further issue, the determination of which is at least partly neurophysiological.

But now we might ask how Anderson's model of a production adapts to procedures that go beyond the verbal. It is not so unthinkable (e.g., IF the scene looks like *that*, THEN set subgoal), but it is different. The interesting thing is that the propositional contents here look to be controlled much more by the very concrete implementational systems, in accordance with his (sect. 1.1, para. 1). It is hard to escape the conclusion that the investigation of the algorithms for such systems will have to be driven much more by the admittedly very difficult investigation of evolved hardware.

One might speculate on uncovering algorithms using a computer pedagogy that expands on such things as learning LISP and calculus, in the direction of geometry, spatial reasoning in games, colour and sound structures, musical composition, and much more. Anderson mentions the case of proofs in geometry, but here one can find a clash of paradigms between philosophers and mathematicians who maintain the primacy of the verbal/symbolic in geometry (suggested by Anderson's reference to "proofs" here), and those who maintain the primacy of purely spatial knowledge, or "spatial intuition." My basic point is that the protocols we have informally available to us suggest the fruitfulness of such computer pedagogy in investigating the latter thesis using the algorithmic approach.

To sum up, Anderson's methodology of algorithms, protocols, and computer pedagogy looks highly fruitful at the present stage of investigation; but, in the long run, a broadened conception of nonverbal algorithms and their ubiquity will be an essential step toward "the utopian goal of adequate neurophysiological knowledge."

Ways and means

Adam V. Reed

AT&T Bell Laboratories, Middletown, N.J. 07748

Like Anderson, I would like to see cognitive psychology make faster progress in identifying the structure of mental operations. Unfortunately, I have serious misgivings about the dichotomy on which Anderson bases his recommendations, about the terminology with which he labels one side of this dichotomy, and about his prescription for progress.

The dichotomy. The dichotomy between studying algorithms and studying their implementation is both imprecise and superficial. There are at least six kinds of questions one may ask about an information-processing system:

1. What functions is the system capable of performing?
2. What functions does it actually perform?
3. What procedures does the system carry out in order to perform those functions?
4. How much time and what other resources are required to carry out each of those procedures? What are the trade-offs, for each procedure, between time and resource availability, and the quality (accuracy) of results?
5. What are the elementary operations from which procedures are assembled?
6. What are the time and resource requirements, and the reliability, of each elementary operation?

Each subdiscipline of computer science is concerned with the answers to a specific category of questions from the above list. The theory of computability is concerned with (1). The theory of specification is concerned with (2). The various software disciplines, such as heuristic programming and algorithm design, are concerned with (3). The (somewhat misnamed) theory of complexity is concerned with (4). The theory of instruction sets is concerned with (5), whereas (6) is the subject of processor and, where applicable, microcode design.

The historical categorizations of questions about human cognition tend to correspond cleanly to ordered sets of categories from the above list. For example, Chomsky (1965) cut between category (1), which he called competence, and categories (2)–(6), which he called performance. Pylyshyn (1980) cut between (1)–(4), "algorithm," and (5)–(6), "functional architecture." Marr (1982) called (1)–(2) "representational," (3)–(5) "algorithmic," and (6) "hardware." Anderson also attempts a dichotomy, but he doesn't cut clean. The intended cut may be between (3) and (4), but what is (3) concerned with, if not with how functional capabilities are actually *implemented?* And what is the point of (4), if not the characterization of procedures – in Anderson's terminology, *"algorithms"?* Would it not be cleaner to follow the example of computer science and explicitly acknowledge all six of the above categories?

By setting up more precise categories, it becomes possible to discern among them a network of mutual constraints. For example, with respect to any given task, the answers to (3), (5), and (6) mutually determine the answer to (4). The answers to (4), (5), and (6) would necessarily constrain the answer to (3), just as the answers to (3), (4), and (5) would constrain the answer to (6). Answers to (2) and (1) mutually constrain each other; and the answer to (3) can only be formulated with at least a presumptive set of answers to (2) and (5). Often, the most fruitful way to attack a question in one category is to answer some questions in the constraining ones. Thus, the claim that attention to the "wrong" questions in the above set is responsible for lack of progress on others ought to elicit skepticism.

The terminology. In the customary terminology of computer science, an *algorithm* is not just any procedure: It is, specifically, a procedure that has been proven to yield a correct result in finite time. In contrast, a *heuristic* is a procedure that has a nonzero expectation of yielding, within a specified time, a result that satisfies some criterion of usefulness. Computer simulation of any but the simplest of cognitive tasks is far more likely to be heuristic than algorithmic. Indeed, artificial intelligence techniques are often taught under the label of "heuristic programming." I see no justification for using the term "algorithmic" in reference to human cognition. Communication between psychologists and computer scientists doing cognitive science is confused enough already.

The prescription. Anderson claims that the identification of procedures used to perform psychological tasks may be achieved only through the study of across-task variation in protocol data. He dismisses as irrelevant for that purpose the measurement of performance variables, such as time and accuracy, as a function of parametric variation of the same generic task. Yet protocol data can provide, at best, selective glimpses at intermediate points in the performance of a task. In general, these intermediate points will be shared by any number of structurally different algorithms or heuristics.

Anderson notes, correctly, that in computer applications the choice of program and data structure has proven to have enormous consequences for performance . . . largely independent of machine implementation." This is one of the key results in the branch of computer science known as *complexity theory*. Complexity theory is concerned with the relationship between the structure of algorithms and heuristics on the one hand and their resource requirements and relative performance on the other. These relationships usually take the form of an expression relating some resource requirement or performance variable, such as time, to a parametrically variable task characteristic, such as the size of the input set, for each category of procedural structure. For example (Aho et al. 1974), sorting algorithms of the *quicksort* structure has an expected time to completion of $O(n \log n)$, where n is the number of items to be sorted.

Now suppose a psychologist hypothesized that humans used a procedure analogous to quicksort when performing a sorting

task. Given the result derived by Aho et al., the obvious prediction of this hypothesis is that sorting time for humans should follow the above pattern, except possibly for a fixed component A and an input/output processing component Bn; that is, expected sort completion time would be predicted to vary as $[A + Bn + O(n \log n)]$ with input set size n. If measured performance did not conform to this prediction, the hypothesis that humans used a variant of the quicksort algorithm to perform sorting tasks would be disconfirmed. Conversely, confirmation would constitute presumptive evidence for the hypothesis, because other sorting algorithms have other patterns of expected time. A more definitive confirmation could be obtained by looking at completion-time distributions, and especially at their worst-case extreme, because the worst-case performance of the quicksort algorithm is known to be quadratic with n.

Because heuristics, unlike algorithms, are not guaranteed to produce the correct result, they require the use of time-accuracy curves (Reed 1973; 1976) rather than reaction-time distributions. In any case, parametric data remain necessary to tell which specific procedures, of the many that are compatible with any given set of protocol data, are actually carried out in the human mind.

Is there more than one type of mental algorithm?

Ronan G. Reilly

Educational Research Centre, St. Patrick's College, Dublin 9, Ireland

There are a number of ways in which Anderson's definition of mental algorithm is unsatisfactorily ambiguous. He initially asserts that "mental algorithms are abstract specifications of the steps taken by procedures that run in the mind." He then moves from this definition to a more concrete one, in which he states that the steps at the algorithmic level "correspond to points of discrete changes in working memory," or indeed to the firing of a production rule. Thus, he appears to move from something of a "competence" definition of the algorithmic level to a "performance" one. Yet, when specifically discussing language, Anderson equates competence with algorithm and performance with implementation. This indicates a somewhat loose definition of the algorithmic concept. In the rest of this commentary, I will propose a more precise definition. In particular, I will argue that there are two forms of algorithmic representation, that it is essential to distinguish between them, and that the distinction has implications for the viability of Anderson's research agenda for cognitive science.

There is a need, I feel, to distinguish between an abstract algorithm (algorithm^1) and an executable algorithm (algorithm^2). To use a computer analogy, algorithm^1 is equivalent to the high-level specification of a program, algorithm^2 is the specification rendered in some machine-independent algorithmic language, and, finally, the implementational level is equivalent to the compiled program within a specific operating system on a specific machine. In using the term "executable" here, I do not imply that the algorithm is implemented or compiled in Anderson's sense ("performable" might be a more accurate term). Anderson (1983) claims that we learn a skill by the proceduralization of its initially declarative representation. He would equate algorithm^1 with a declarative representation, and algorithm^2 with its subsequently proceduralized form. In Anderson's ACT* model, both algorithmic forms would be considered executable, whereas I argue that they are not.

Some examples will help illustrate what I mean. Take the following instructions:

1. Draw a symmetric 6-pointed star.
2. Draw an isosceles triangle. Superimpose another isosceles triangle of the same size, but rotated by 45° about the two triangles' common center.
3. Draw an isosceles triangle. Superimpose an inverted isosceles triangle of the same size, where both triangles have the same center.

Each of the above is a means of achieving the same end: drawing a "Star of David." Instructions (2) and (3) are equivalent in their degree of specificity, whereas instruction (1) is somewhat more abstract. Yet, they are all underspecified with respect to the following instructions:

4. Draw this: [an actual picture of a 6-pointed star].
5. Draw like this: [somebody drawing a 6-pointed star].

Instruction (4) has greater specificity with regard to size and location on the page. Instruction (5) is the most specific of all, in that it provides the learner with location information, as well as information about how to draw the components of the figure. In terms of speed of task completion, instruction (5) will almost certainly be the most effective, and instruction (1), the least effective. Instructions (1) through (5) represent gradations of specificity from the abstract to the concrete. They also represent gradations of speed of task performance.

Although the speed at which the algorithm can be constructed and executed is determined, in part, by instruction specificity, this is not the full story. One could imagine ever more detailed and ever more precise instructions being provided to the learner, with this level of detail interfering with the execution of the task. Yet the wealth of detail available to the learner from seeing the task performed (as in instruction 5) does not interfere, but rather aids, in speeding the performance of the task. Why is this? One possibility is that the abstract, verbally based encoding is somewhat removed from its executable form. The verbal instructions need to be transformed into an executable form prior to execution, thus giving rise to a processing bottleneck, whereas the visual percept maps directly, or less indirectly, onto an executable form. The verbal form corresponds to what I have called algorithm^1, and the executable form to algorithm^2. It is also possible to map from algorithm^2 back to algorithm^1. This occurs when we ask learners to provide a verbal report of their mental algorithms. Note that algorithm^2 is not the proceduralized or compiled version of the algorithm being learned. It can be in a declarative form and can be operated upon by general problem-solving productions, as envisaged by Anderson (1983). It is, however, a separate and, I maintain, a psychologically real representation. Skill acquisition can take place via either route, but the processes involved can be reported only if rendered as an algorithm^1.

I have dwelt on tasks that, when learned, manifest themselves in overt action. When one looks at "purer" cognitive tasks, the distinction between algorithm^1 and algorithm^2 becomes less easy to discern. For example, take learning to do mental arithmetic and, in particular, the addition of 2-digit numbers. The instructions for performing such tasks (encoded as algorithm^1) bear about as much relation to the algorithm that performs the task (algorithm^2) as, say, key-presses on a pocket calculator do to the electronic processes that perform the calculations. There is a necessary relation between the two algorithms, but one should not be mistaken for the other. This would be the equivalent of mistaking the map for the terrain. In tasks in which the medium of instruction is primarily verbal and, as a result, in which the primary route to algorithm^2 is via algorithm^1, the distinction between the two types of algorithm becomes less significant. However, even in this latter case, there may be independent routes to algorithm^2 (such as through observing the use of an abacus) that may make the distinction behaviorally relevant.

The main implication the distinction has for Anderson's research agenda is that verbal reports of mental algorithms do not directly tap the algorithms used in the performance of a task. Such verbal reports are therefore less reliable than Anderson seems to assume. Furthermore, protocol analysis will throw direct light only on the structure of algorithm^1, which repre-

sents merely a part – albeit an important part – of the whole learning process. To put the preceding argument another way, there is a valid distinction to be made between learning by observation, learning by instruction, and learning by doing – a distinction not captured by Anderson's definition of algorithm.

Weak versus strong claims about the algorithmic level

Paul S. Rosenbloom

Departments of Computer Science and Psychology, Stanford University, Stanford, Calif. 94305

While reading Anderson's target article, I constantly found myself in the seemingly paradoxical position of agreeing with his arguments, but disagreeing with his conclusions. His arguments do support a weak form of his conclusions: (1) The algorithmic level is important, understudied, and relatively tractable; (2) learning is one of the important issues at the algorithmic level; and (3) intelligent tutoring does provide a powerful methodology for studying learning at the algorithmic level. However, the same arguments become severely overextended in the attempt to defend the stronger conclusions actually presented in his article: (1) "Issues at the algorithmic level are more interesting"; (2) "learning is the key issue at the algorithmic level"; and (3) "the best way to study the algorithmic level is through pedagogical experiments." In proposing these stronger conclusions, Anderson is attempting to go beyond his stated goal of justifying his own research enterprise to the implicit claim that this is where the field as a whole ought to focus. Such claims are notoriously difficult to justify by reasoned arguments, because they are often more a question of personal research strategy than of established fact. For similar reasons, they are also difficult to argue against. Nonetheless, in the remainder of this commentary I will attempt to cast some doubt on the first two of Anderson's strong conclusions.

The first strong claim is that the algorithmic level is the "most interesting and fundamental" one. There are a variety of ways to counter this claim. One is to deny that any one level is more fundamental or interesting than the others. This could be argued by making an analogy to the levels of physics, chemistry, and biology. Each level has its own set of important questions to answer and its own set of techniques to be used in answering them. A second way to counter the claim is to argue that some other level is the most interesting and fundamental one. For example, from the point of view of the knowledge level, much of the algorithmic level is simply implementational detail. At the knowledge level, we can ask the fundamental questions about what knowledge the person has and how that knowledge changes over time. A third way is to consider focusing on interfaces between levels rather than on the levels themselves. The interface between the implementational and algorithmic levels – what Anderson refers to as the architecture – has an existence that is independent of the contents of the individual levels. As Anderson mentions, the GRAPES (Sauers & Farrell 1982) and ACT* (Anderson, J. R. 1983) architectures have different implementations but support essentially the same algorithmic level. The architecture is like the implementational level in being understandable in terms of general principles. However, in studying the architecture, we have the advantage of being able to use data from both the implementational and the algorithmic levels. Knowledge of the architecture provides leverage by constraining the contents of the implementational and algorithmic levels. Given a characterization of the architecture and a description of a task to be performed, it should be possible to predict the set of algorithms that subject could be using. Despite the importance of the architecture, there is not yet a consensus about its structure.

The second strong claim is that "learning is the key issue at the algorithmic level." There are at least two reasons to doubt this. The first is that learning is only one of many influences on the behavior of the algorithmic level. In general, the algorithm exhibited by a subject on an experimental task is determined by a combination of the demands of the task, the existing contents of the subject's algorithmic level, and the subject's architecture. Even if we ignore all of the nonarchitectural influences, performance is influenced not only by the learning component, but also by the representational, problem-solving, perceptional, and motor components. For example, the problem-solving capabilities provided by the architecture determine what algorithms the subject can and cannot perform (at least in any straightforward fashion). The problem-solving component may also play a key role in the process of determining what will be learned. The second reason for doubting that learning is the key issue at the algorithmic level is that the study of the algorithmic level is tractable, as Anderson mentions, because of the existence of reportable cognitive steps in novice performance, not because of learning. Learning clearly influences both how the novice performs and how the novice is transformed into an expert, but it need not occur at the same level as reportable cognitive steps, as Anderson claims. For example, in the SOAR system (Laird et al., in press), learning occurs at the level of productions – as in ACT* and GRAPES – but reportable cognitive steps occur at a level above the productions, at the problem-solving level.

Levels of research

Colleen Seifert and Donald A. Norman

Institute for Cognitive Science, University of California at San Diego, La Jolla, Calif. 92093

We believe that Anderson has made a valuable contribution in his target article, but not necessarily the major one he thought he was making. He creates a controversy by claiming that there is a level of psychological investigation that is unfruitful to pursue because the data are so limited. In particular, he decries the very level at which cognitive psychologists have focused most of their energy. He then argues for the usefulness of protocol studies, claiming that not only do they provide valuable information, but that the information is of superior quality, thereby making the usual measures obsolete. Along the way, he separates psychological processes into implementation and algorithm, despite years of demonstrations and arguments that these two levels are fundamentally intertwined, that today's algorithms are tomorrow's mechanisms. Indeed, one of the major points of the connectionist revolution [see Ballard: "Cortical Connections and Parallel Processing" *BBS* 9(1) 1986.] is that mechanism has taken over the work of what was heretofore thought to require high-level, symbol-processing algorithms. Even if the connectionist approach is wrong (or, more likely, somewhat overambitious), its very existence shows the volatile nature of any distinction between these two so-called levels.

Anderson argues that psychological phenomena at the implementational level take place in the millisecond region, yet the experiments we perform can give only sparse data at this level. In particular, he speaks of reaction-time data and percent-correct data. But this is not all there is, and surely Anderson does not mean to neglect the rich range of data that psychologists have relied upon for years; for example:

1. the nature and form of the response
2. the kind of error
3. the relative ordering of difficulties
4. the relative ordering of times
5. scaling results, whether direct or indirect (as in multidimensional scaling techniques)
6. time–error tradeoff functions

The perplexing aspect of Anderson's argument is that protocol data, which Anderson has suddenly discovered, are thought by many to be too sparse to yield powerful constraints for theories. At best, protocol data can only illuminate the states through which people pass on their way to a solution. Yet Anderson argues now that these methods are superior!

We agree with Anderson that research at the algorithmic level is essential, that protocol methods are valuable and necessary, and that matching the power of the data with the nature of the hypotheses is crucial. But there are very rich sets of tools available for all levels of investigation (witness the partial list we gave above, which yields very subtle analyses and powerful constraints on interpretations), and there are strong limitations on the power of tools at all levels, especially the well-known weaknesses of protocol methods).

Anderson does provide good arguments about the value of research questions at the "algorithmic level," however. The intention behind the target article, we believe, was to call for increased attention to other types of research issues. Our view is that Anderson did not go far enough in explaining the problems that face researchers interested in algorithm-level questions. Consider the following in which Anderson remarks on the failure of experimenters to utilize protocol information: "Many experiments in the literature (to point a finger only at myself, consider J. R. Anderson, 1976, pp. 363–75) have used the low-yield methodology to explore algorithm-level issues in situations where much better protocol data could have been obtained" (sect. 1.4, para. 9).

The question Anderson raises, but does not really address, is *why* experimenters have stayed with "low-information-yield" experiments when they could have used protocols. Three reasonable causal factors come to mind:

1. Protocol methodology has never been developed beyond the notion of transcripts.
2. The traditional view of appropriate methodology requires every observation to be reported with a significance level.
3. The mainstream view of "algorithm-level" work is that any study utilizing a task domain is about the domain and not about cognition.

Anderson's paper should be a demonstration of the science of protocol study and a call for further developing techniques that allow constrained and reliable observation of performance on cognitive tasks. Of course, as Anderson notes, more information about the operation of cognitive processes is better than less, but the question is, What *kind* of information can be collected and productively observed? Anderson presents only a limited solution to the problem of the richness of protocol data and the difficulties of analysis and representation. He limits the content of the protocols to the output of the reasoning task – that is, written computer programs and geometric proofs. Observing the output of the cognitive process (the series of output programs produced in trying to find a solution) provides interesting results. As Anderson demonstrates, it has the added benefit of allowing the automatization of the protocol analysis, where the output programs are compared via computer analysis. However, this technique throws away much of the richness in protocol data that Anderson argues is necessary for good observations: There is no record of the subjects' reports of the planning process, and no possibility of observing, for example, expressions of difficulty or confusion. In order to automate analysis, many interesting aspects of the subjects' cognitive behavior will necessarily be thrown out.

Whereas comparing written output of attempts at solving a problem will provide much useful information, there is a good deal more to be gained from protocol studies. Unfortunately, protocol methodology has not been developed much beyond taking transcriptions. Too often, protocol analysis appears to be an art rather than a science. Protocol analysis needs new methods that will aid in gathering systematic observations and that will provide a way to consider a variety of explanations and points of view about the behavior observed. These methods must go beyond transcript analysis to provide a window on cognition of the process-tracing flavor (see, for example, Card et al. 1983). Of course, protocols are most useful when a model exists that is actually capable of generating the trace, because then the subjects' traces can provide very detailed feedback about where the model is right or wrong (for example, Rumelhart and Norman's, (1982), connectionist model of typing). If verbal protocols are taken, then some model is required of how the natural language trace is generated. A method to verify hypotheses about cognitive behavior, objective and falsifiable, is the goal; the claim is that, along with an important role in generating ideas, protocol studies of cognition can be used to evaluate hypotheses. (By the way, we do not mean to restrict the definition of protocols: What about simple observation of cognitive behaviors? An analysis that approaches the content of a sample of behavior from a variety of levels and perspectives about what is occurring may provide suggestions about model development and some new ideas about hypothesis testing.)

Part of the agenda for this new methodology includes efficient methods for presenting results in ways that demonstrate to someone who hasn't watched hours of tapes that the conclusions do indeed hold. Newell and Simon's (1972) development of the Problem Behavior Graph was one such attempt, but it floundered, in part because of the tremendous amount of effort required for its generation. Perhaps one of the reasons this work is not accepted as readily as traditional statistical studies is that it is difficult to evaluate how good a job an experimenter has done in analyzing the protocols. Traditional studies use a standard measuring stick, the *F* test, to decide whether a researcher is right in drawing particular conclusions. Quantitative methods provide a standard of objectivity; is it the only one possible? Unfortunately, with few exceptions, psychologists have adapted their questions to the methods available rather than to inventing new methods that are better suited to their questions. On the other hand, current protocol methods go too far in the other direction, sacrificing objectivity in order to get the data that some questions require. No method exists for reporting the content of protocols so as to verify observed trends in the data; instead, a few examples are simply drawn from the transcripts. A method for communicating protocol content that is understandable to outsiders and allow the quality of the study to be evaluated would increase the usefulness and publishability of this work.

We agree with Anderson's important sociological observation that mainstream interests in the field consign any work on "applied"-domain problems to a field other than psychology. Any study using computers in the task is deemed more appropriate in a human-factors journal, and any study of the learning of "real-world" information belongs in the educational journals. This is not true for the implementational-level examples Anderson cites; for example, a study of short-term memory using number recall as a task is not considered to be an experiment about numbers, but about memory. For problems at the algorithmic level, the domain of the task is considered more important than any generalities about learning or cognition that can be taken from the work; this must stop. Studies at all levels of behavior can yield general principles of psychology (or, more generally, general principles of cognitive science). Even if the behavior studied does depend on a domain context, the results can provide principles about the role of artifacts and structured tasks in the cognitive processes that occur in natural settings (Hutchins 1986; Norman, in press). As long as mainstream psychology continues to view work at the algorithmic level as domain-related rather than as basic science, researchers interested in these problems will be forced to seek refuge elsewhere, hiding out in education or computer science in order to publish and receive grants under peer review (much to the benefit of computer science and education!).

Even the computational modeling Anderson describes as a

successful test of his ideas is not widely accepted in psychological journals as a compelling methodology. Anderson might well have written his target article as an attempt to justify the use of cognitive modeling in psychology. Although interest in computational models, especially parallel-distributed processing or connectionist models, has increased in the field, the mainstream journals still expect experimental work. It seems clear that some questions, at least, will require modeling to refine and develop ideas about cognition; even the sample study that Anderson presents includes computer simulation. The repertoire of "empirical" methods need not be limited to experimental studies; it can be extended to include both computational models and protocol methods, and can be further developed through new approaches in each area. Broader questions require a wider range of appropriate methodologies, a greater variety of tasks in which to observe cognitive functioning, and the use of domains involving real-world behavior. Questions proposed by Anderson as algorithmic level should be applauded for their naturalistic basis, rather than ruled out as too applied and uninteresting for those truly concerned with cognitive processes.

It is not necessary to agree with Anderson's distinction between implementational and algorithmic levels to agree with this paper's message; a precise distinction is not required so much as a precise definition of interesting research questions for the field. Anderson's point, slightly interpreted, is that we should broaden the scope of questions that are considered to be "core psychology," and therefore encourage the development of a variety of tools and methods to examine those questions. Anderson has come up with a good solution for his own questions; his article should be read as a challenge to consider the possibilities of a more diverse view of what constitutes basic research in psychology.

In conclusion, we agree with Anderson's argument that we must do work on both implementation and algorithms and that we must use all the types of data-collection devices at our disposal. We must certainly also look at both pure research and applications. Perhaps the fuss that Anderson has generated will encourage others to be more broad-minded in their questions and methods. Now all that is left is just to do it.

Connectionism and implementation

Paul Smolensky

Department of Computer Science and Institute of Cognitive Science, University of Colorado, Boulder, Colo. 80309

Anderson is careful not to advocate the abandonment of other research programs while advocating his own. The question addressed in this commentary is, How can Anderson's research program be reconciled with one that he argues against – a connectionist approach? The argument I summarize here is presented at length in Smolensky (1987b).

The basic argument is this: The character of cognition varies dramatically with the scale of the description. On a coarse scale, the programs of Anderson's algorithmic level describe the sequential transitions between mental states. Symbols denoting the conscious-domain concepts, and rules explicitly mediating serial transitions, are appropriate descriptive tools. On a fine scale – so the connectionist hypothesis goes – conscious-domain concepts are composed of multiple constituents, not consciously accessible, and mental processing consists of massively parallel satisfaction of numerical constraints among these fine-grained constituents. These numerical constraints comprise the knowledge on the fine scale.

The questions addressed by Anderson's research program and those addressed by the connectionist program that emerges from the above hypothesis are quite complementary. Anderson's algorithms specify the sequential computations of the coarse scale, but as he points out in his discussion of language acquisition, not all cognitive tasks can be described at this scale. The methodology he advocates is severely limited in domains such as first-language acquisition, where the processes of interest do not involve extended sequential transitions among multiple, consciously accessible states. Let us call behavior to which Anderson's methods apply Type A and the rest Type B. It is with Type B behavior that the connectionist approach offers its greatest potential contribution.

The connectionist hypothesis on the nature of the fine-scale description of cognition puts the coarse-scale description of Anderson's algorithms in a different light from the one that comes from the traditional view of cognitive architecture. Anderson defines the cognitive architecture to be the set of computational components out of which mental algorithms or programs are constructed. He points out that his theories at the algorithmic level are predicated on the consensus view, according to which the cognitive architecture provides such symbolic computational components as variables, recursive symbolic structures, pattern matchers, and random-access memory – all the ingredients that make possible an algorithm or program stated as a production system. How are the symbolic components of this architecture implemented? Implementations of the sort provided by conventional computers are not consistent with the connectionist hypothesis concerning the fine-scale description of cognition. This means that a major problem for the connectionist program is the implementation of symbolic computation. More precisely, the question is, How can a system with connectionist microstructure support a coarse-level description, for Type A behavior, like that of symbolic algorithms of the Anderson type? For Type B behavior, there is no reason to demand such a description. The goal is *not* to provide a connectionist implementation of a symbolic architecture that can be used to perform *all* higher cognition. It *is* a connectionist goal to explain why the connectionist fine structure admits Anderson-style symbolic algorithmic accounts of the coarse structure in Type A behaviors – and not in Type B behaviors.

This view places two important items on the connectionist research agenda: explicitly relating fine- and coarse-scale descriptions of connectionist accounts of Type A behavior on the one hand and of Type B on the other. Symbolic algorithms such as Anderson's provide higher-level accounts of Type A behavior, and it is important to see explicitly how such accounts can be implemented in a connectionist microstructure.[1] At the same time, it is also important to develop accounts of Type B behavior at levels higher than the "implementation-level" descriptions in terms of individual units and connections. The concepts for coarse descriptions of connectionist accounts of Type B behavior are quite approximate and provide incomplete descriptions that cannot by themselves serve as the basis for formalization (Rumelhart et al. 1986; Smolensky 1986; 1987a).

Anderson points out that theories about complex behaviors require considerable data to constrain them, and that the best data supply rich information about mental states many times during the execution of a complex task. He points out that verbal protocols are not the only possible sources of such data. It is a point well taken that adequate constraint of connectionist models requires the development of methodologies that provide rich real-time data during processing. One promising source of such data are detailed time courses of probabilities for many possible responses (e.g., Kounios et al. 1987). It is clear that the lack of conscious access to the fundamental elements of connectionist models presents serious methodological challenges to the research advocated by Anderson.

Anderson tries to motivate his research program by explaining that what he finds most interesting are questions such as: What algorithms determine the sequences of conscious states that people go through (in Type A behaviors)? How do old concepts and production rules give rise to new ones as people learn? The connectionist approach is instead motivated by such questions as: How can the consciously accessible aspects of

mental states be characterized in terms of lower-level constituents? Why do the consciously accessible aspects of mental states follow sequential symbolic algorithms sometimes but not at other times? How do consciously accessible concepts or symbols emerge from nonsymbolic constituents (rather than from other symbols)? How do symbolic production rules emerge from experiences (rather than from other rules)? The "implementational" questions that arise from the connectionist hypothesis call for *explanations* of higher-level processes – explanations that are considerably more ambitious than those previously attempted in cognitive science.

NOTE

1. Although it is true that Anderson's implementation of ACT* incorporates a number of the elements of connectionist computation, it lacks the crucial feature of distributed representation (Hinton et al. 1986a) and critically involves a number of elements of symbolic computation that are not part of an orthodox connectionist implementation; the hardest parts of the job of distributed connectionist implementation of symbolic computation remain, for example, pointers (Hinton 1987; Touretzky 1986) and variable binding (Smolensky 1987c; Touretzky & Hinton 1985).

Interactive instructional systems and models of human problem solving

Edward P. Stabler, Jr.

Canadian Institute for Advanced Research and Department of Computer Science, University of Western Ontario, London, Ontario, Canada N6A 5B7

One notable feature of the recent research on human problem-solving protocols and on interactive instructional systems is its emphasis on what Larkin (1981) calls "formal domains" – that is, on domains such as "mathematics, applied mathematics, most of what are called 'hard' sciences, as well as sophisticated games (e.g., chess, go)." Larkin distinguishes these formal domains from such domains as "biology, psychology, and English literature [in which] it is extremely hard to find unambiguous sets of principles sufficient to solve problems" (p. 311). In his target article, Anderson does not mention the emphasis on formal domains in this research (or in the sample of this research that he cites), but a consideration of how this emphasis might be explained provides some insights into the methodologies for the sort of "pedagogical research" he advocates.

Larkin describes the formal domains that have been of particular interest as "involving considerable amounts of rich semantic knowledge but characterized by a set of principles logically sufficient to solve problems in that domain" (p. 311). This provides one obvious explanation of the emphasis on formal domains in protocol studies and in the development of automated tutors: In these domains we have formal representations of the principles the students must learn and apply in problems. If we also suppose that the students' knowledge is actually encoded in much the way it is formalized and that the principles for using that knowledge are much like the strategies used in searching for formal proofs, then we also have a basis for modeling the mental state of the student. In the informal domains, no formalized representation is available to serve either as a specification of what the student should do nor as a basis for modeling the student.

This explanation of the emphasis on formal domains also suggests a way to reconcile Anderson's methodological recommendations with Chomsky's and Marr's. Anderson defends the following assertions:

Assertion 1': There is an important distinction to be made between mental algorithms and their implementation.

Assertion 2': There is important basic research to be done on algorithmic issues.

Assertion 3': The best way to study the algorithmic level is through pedagogical experiments.

The algorithmic level is compared to Chomsky's performance level and to Marr's algorithmic level. Anderson's proposals are similar to Chomsky's and Marr's in recommending that attention be given to psychological accounts above the "implementational" level, but both Chomsky and Marr regard the "highest" levels of theorizing as methodologically prior to the lower levels. Chomsky suggests that we cannot expect to learn much by studying the mechanisms of linguistic performance until we have some grip on what linguistic knowledge the subject has – the "competence theory" (cf. for example, Chomsky 1975, pp. 16–17). Marr makes a similar point about his "first level" theory of what function is being computed: "The theory of a computation must precede the design of algorithms for carrying it out, because one cannot seriously contemplate designing an algorithm or a program until one knows precisely what it is meant to be doing" (Marr 1979, p. 19). Anderson's recommendations square with these remarks if we take him as suggesting that, in the formal domains, we have an account of precisely what computation is being carried out. In that case, we can proceed to consider what algorithms are involved; that is, the implicit assumption of researchers in this field may be:

Assertion 4: In formal domains, we have an account of what overall computations are being carried out, and so in these areas we can most profitably study the algorithmic level.

This would explain the emphasis on formal domains, and it would allow Anderson's methodological recommendations to fit neatly with Chomsky's and Marr's. Let's consider this idea more carefully.

Notice that in the case of human problem-solving in formal domains, the account of what computation is being carried out can depend in an obvious way on whether we "include" the external medium of computation, such as the subject's marks on a piece of paper; that is, in considering precisely what computation is being carried out, we should distinguish the computation being carried out overtly from the psychological processing that underlies such a performance. In adding two large integers, for example, the common rule involves adding the least significant digits first, recording the sum, and then proceeding to the next pair of digits, possibly with a "carry," and so on. But the psychological processes underlying this rule are certainly rather different – for one thing, when the integers being added have many digits, there may not be any psychological representation of the two integers at all. In this case, at any one time, the subject needs a mental representation of only the relevant pair of digits and the carry. The psychological process must only involve adding a pair of digits with a carry, keeping track of what is recorded, and so on.

In other, more complex tasks in formal domains, this distinction becomes less transparent, because the psychologically relevant properties of what is recorded and the psychologically relevant but unrecorded aspects of a problem are often far from obvious. Consider, for example, the Anderson et al. (1981) study of a student's attempt to construct proofs in geometry. At several points, Anderson et al. notice that the student apparently failed to understand certain things he had read or written. A case like this illustrates the obvious point that the mental representation does not generally correspond exactly to the external expressions being read and produced, and hence we cannot suppose that the properties of the mental representations of geometric results or methods correspond to similar properties in the formulae.

In formal domains, then, although we have external representations of the principles needed to solve problems (e.g., formulae, logical statements), we do not *ipso facto* have an account of what psychological computation is being carried out – that is, it is not plausible to assume that the psychological representation of knowledge of formal domains is similar to the statements or formulae that comprise the external representation of that

knowledge. (It is interesting to note that the "schema" representation of geometric propositions attributed to the subject in Anderson et al. [1981] *is* very much like the external representation, and this is why the problem with the inability to deal with one of the concepts in the schema was remarked upon and actually marked in the representation.) In the second place, it is not plausible that the strategies humans use for solving problems in formal domains are much like the strategies used by automatic systems searching for formal proofs. This point is clear from the failure, in spite of great efforts, to find a formal system for constructing proofs that behaves at all like a mathematician, or even as well as an undergraduate logic student (cf. Bledsoe 1977; Bundy 1983; Loveland 1984). The problem in finding proofs in formal systems is similar to the problem in learning: The ways in which students and mathematicians select from the enormous space of possible paths to a proof are unknown, as are the ways in which learners select from the enormous space of possible hypotheses that fit the data. If the protocol approach had discovered how humans (experts *or* students) do these things, it would not be in need of defense, and lacking such an account undermines our ability to model a student.

Another aspect of the Anderson et al. (1981) study illustrates the difficulties of drawing inferences from a protocol without a prior theory of the mental representations and the computations over them. After suggesting that one of the fundamental mechanisms of learning is analogical, the authors note that their computational model of analogical reasoning finds only superficial, syntactic properties that are shared by pairs of problems. They carefully do not commit themselves to the view that the student relies exclusively on such superficial similarities, and they provide no general account of what analogical reasoning involves. In fact, the range of relevant properties the student might consider in this sort of reasoning – based on his psychological representations of previous problems, together with what he can gather from the record of his work and the text – seems quite unlimited. It really is implausible that the properties that suggest analogies are generally superficial, syntactic properties, or that the relevant properties would generally be reportable by the subject – so little of cognitive psychology is that simple! Even if Fodor's (1983; see also multiple book review in *BBS* 8(1) 1985) skepticism about the prospects of any account of such reasoning is rejected, a consideration of any of the well-known studies casts doubt on the prospects for a simple syntactic account of analogical reasoning – even simple recall tasks are more complex. The point here is again just that the psychologically relevant features of the task may not be definable over the external representation at all, and the lack of a good theory regarding the mental representations being used by the subject will seriously complicate the inference from the protocol to the algorithm. In other words, we cannot accept Assertion 4, and we find the protocol approach to the study of the algorithmic level to be subject to the difficulties that the views of Chomsky and Marr would lead one to expect. We are forced to try to develop the knowledge and computation-level theory, together with the account of the algorithm used – a strategy that one would expect to be difficult.

The real test of Anderson's methodology will be its empirical success in leading us to substantial new theories of human problem solving and learning. Unfortunately, Anderson does not attempt to argue for his proposals with examples of psychological principles that his methodology has uncovered. A survey of the protocol research literature, however, shows that it is full of interesting informal insights. It *is* surprising that careful protocol studies have not been pursued more vigorously in the past. Whether the new interest in this research will ultimately lead us to algorithmic accounts of how people go about solving problems, however, has not yet been convincingly established. It is still an open question whether the individual variability in problem-solving strategies will be constrained enough to allow for effective error diagnosis even in the majority of cases, whether there are general-purpose skill acquisition strategies, and so on.

In any case, the *methodology* of protocol analysis and pedagogical research should be distinguished from empirical claims that require support. Anderson has sprinkled a number of unsupported empirical claims through his discussion that can be sorted out from the methodological ideas. One of the empirical claims in need of support is that in reality, as in ACT*, "if one has analyzed the precompiled skill and identified its structure, one can use that analysis to infer the structure of the compiled skill." No study is cited to support the idea that our "precompiled skills" are really similar in structure to the well-practiced "compiled skills." Another empirical claim that is perfectly open to testing is Anderson's idea that at the algorithmic level, all the psychologically relevant states are reportable, and hence that they do not pass at a very rapid rate. Obviously, the test of these claims is empirical, and their falsity would not necessarily undermine the usefulness of protocol analysis, but only the directness of its bearing on the theory.

It is worth noting that the prospects for instructional technologies are not so bleak. Even if we cannot get an algorithmic account of how naive students solve geometry problems, for example, it will be useful to have a catalog of common misunderstandings. Instructional systems can then be designed to watch for symptoms of these misunderstandings and to administer corrective instruction when they are detected. Obviously, this does not require a very "rich account of the student's mental state," and the value of the technology does not depend on the success of attempts to provide algorithmic accounts of all the problem-solving strategies a student might use. Of course, a correct algorithmic account of the user's skill-acquisition strategies might well be very useful in designing an instructional system, but no such account is needed for building a system with great practical value. In other words, for practical purposes, we can do quite well if we have only the following:

Assertion 4': In formal domains, we have an account of what overall, overt computations should be carried out, and so in these areas we can most profitably develop interactive instructional systems.

I think this is the explanation of the emphasis on formal domains in the development of automated tutors. It is also clear that protocol research in formal domains has been inspired in large part by these practical efforts to design tutors that can be more responsive to the needs of individual students, rather than by the promise of new psychological breakthroughs in computational theories of learning and problem solving. In my opinion, the practical potential of this research suffices to make it worth doing; the possibility of important new psychological insights is just an added bonus.

Applying Marr to memory

Keith Stenning

Centre for Cognitive Science, Edinburgh University, Edinburgh EH8 9LW, Scotland

Can we apply Marr's distinction between computational, algorithmic/representational, and implementational questions to the analysis of "central" processes such as memory, learning, and reasoning as usefully as Marr applied it to the analysis of perceptual domains?

Let us take memory as a generic area and ask how Marr's distinction can be applied. It is at the computational level that the differences between perception and central processes emerge most clearly. One approach that suggests itself immediately trivializes the computational level: The function any memory system computes during errorless performance is the identity function. Laying aside the difficulties of specifying the cueing conditions under which retrieval takes place, this is still a

curious picture. One of the main functions of any memory system is to forget "irrelevant" information, and specifying just what is relevant and what is irrelevant would be the main task in pursuing a computation-level description of memory. But memory "failure" is normally thought of as the paradigmatic performance limitation. It is an open question whether we can adequately distinguish the abstractive properties that are the computational purpose of such system from their performance failure due to overload.

Anderson's research interests have moved in the direction of the study of learners' cognitive skill acquisition and away from the study of "pure" memory tasks. He now mounts a series of arguments which purport to show that the old mine is worked out and that the new mine is more promising. His arguments suppose that the relation between these two areas is that between the implementational and the algorithmic/representational in Marr's hierarchy of distinctions. He supposes that the two areas are in this relation because, according to his theory, the accessibility of facts in immediate declarative memory is one of the main constraints on the operation of the production system with which he models procedural knowledge. Thus, in this particular theory, one narrow class of characteristics of memory constitutes part of a theory of the implementation of procedural-knowledge deployment.

This particular theory also leads Anderson to suppose that memory is studied using measures of error probability and reaction time. whereas complex learning tasks are studied using protocol techniques. Yet, from his own definition of protocol techniques ("the essence of a protocol is that it provides a running series of responses that can be used to infer the sequence of mental states" sect. 1.4, para. 5), it is clear that free recall, self-paced reading time, eye movements, and most other measurement techniques can be used as protocol techniques.

I would not wish to argue against anyone adopting this new field of study, but I believe Anderson has not given us an acceptably worked-out application of Marr's distinctions. If he were to give us such an application worked out for the subsystem that accomplishes the learning of complex cognitive skills, there would still be every reason to suppose that the distinctions would have to be reworked to apply to the subsystem(s) that accomplish(es) factual memory. This would be true even if there were overlap between the operation of the two systems. Anderson himself acknowledges that "what is algorithm and what is implementation can vary from theory to theory" (sect. 2, para. 4), and however good the arguments are for a particular interface between algorithm and implementation in the study of skill learning, they are not arguments that it has to be the same boundary between algorithm and implementation in studying factual memory. The main reason for skepticism about the applicability of Marr's distinctions to central abilities is the requirement that it places on the isolation of subsystems responsible for these abilities. This is difficult enough in the visual system, in which components really are dedicated processors. In memory systems in which there are several ways of accomplishing tasks, the problem is much more acute. Much of the debate in memory has always been about how to divide up the systems.

The substance of the specification of memory systems consists of determinations of their abstractive properties and their cueing characteristics over time. The ACT* system, developed as it has been chiefly to study the interaction between the application of procedures and the availability of premises for them, specifies only a small part of what would have to be specified to capture a computation level specification of memory. That seems wholly appropriate. But it should not be allowed to obscure the omissions that are other researchers' main topics.

To take just one example, Rumelhart and McClelland's (1985) PDP (parallel distributed processing) simulations of memory are mainly concerned with abstraction and cueing. They quite rightly believe that, inasmuch as Marr's distinctions are applicable to central abilities, their work primarily concerns the algorithmic/representational level and, to some extent, a somewhat abstract level of implementation. I think they are quite right in doubting the possibility of specifying the computational level first, as Marr advocated in perception. The algorithms they are chiefly concerned with are ones giving rise to phenomena with which ACT* does not concern itself.

Some of the confusion in Anderson's uses of these distinctions is brought out well by juxtaposing two quotes:

> The level [Marr] calls representational and algorithmic corresponds approximately to what I am calling the algorithmic level. (sect. 1.1, para. 1)

and

> So it may seem surprising to find Rumelhart and McClelland (1985), in their response to Broadbent (1985), arguing that their connectionist models correspond to Marr's representational and algorithmic level and not to his hardware implementational level. As noted earlier, however, *these two levels of Marr's framework do not correspond to the algorithmic and implementational level as they are defined in this paper.* (sect. 4.4, para. 1, my italics)

Although PDP systems can look as if they are concerned with hardware, their implementational concerns are better construed as abstract implementations, just as Anderson's are. At their algorithmic/representational level, they focus heavily on the degree of distribution of representations, an issue that has been neglected by ACT* and by most of the rather ill-defined argument that has surrounded issues of representation. When focus is on the issue of distribution and redundancy of representations, rather immediate evidence is forthcoming that this dimension is both important and empirically resolvable (e.g., see Stenning et al. 1987).

What is the algorithmic level?

M. M. Taylor and R. A. Pigeau

Defence and Civil Institute of Environmental Medicine, Downsview, Ontario, Canada M3M 3B9

One must assume that Anderson is making a coherent statement about a program of research into cognition. Such coherence can be attributed to the paper *only* if everything is interpreted according to the computer metaphor of cognition. A consistent interpretation is very difficult if one proceeds from consideration of the human rather than from consideration of the computer. We do not, for example, see any contradiction at all between the three beliefs that Anderson introduces as having misled psychologists and the three beliefs that he proposes to substitute, except in the context of the computer metaphor.

Anderson uses the terms *algorithmic level* and *algorithm* in more than one way. Is the algorithmic level in the scientist, or is it in the cognitive machine being observed? Sometimes it seems as if algorithms are procedures carried on at a defined algorithmic level in the cognitive machine, and sometimes as if algorithms are performed in the cognitive machine but the algorithmic level is the appropriate way to think about the cognitive machine. In the latter sense, the algorithmic level could well refer to the algorithms that determine how a synapse changes its chemistry in response to neural events. But we are sure Anderson would consider this to be at the implementational level (or lower). On the other hand, if the algorithmic level is taken to be where algorithms are performed in the cognitive machine, we do not believe it to represent a valid concept for biological systems. We exemplify this assertion by reference to a model of cognitive performance (reading) that has a wide range of application.

The Bilateral Cooperative Model (BLC; Taylor 1984; Taylor & Taylor 1983) has been elaborated for the case of reading, but it is intended as a general model for cognitive processing of symbolic data. In it there are algorithmic modules at all levels of

data abstraction, but under most circumstances they are little used in mapping input to behaviour. Only when exact interpretations are required, or when the input is ambiguous or novel, is there much recourse to the algorithmic modules as a primary means of controlling perception or behaviour. Normally, algorithmic modules serve only as an occasional check on the progress of operations that are performed in a more connectionist (analogic) way.

In the BLC model, analogic functions form one of two *tracks* of processes that produce successively higher levels of abstraction. Algorithmic processes form the other track. The development of levels is one aspect of learning; the development of categories within a level is another. Algorithmic processes are important in both aspects of learning, which is taken to occur in three phases: Phase 1, gross pattern recognition, culminating in Phase 2, the identification of elements that recur in patterns of interest, and rules that link them; in Phase 3 these elements are used as input for a more refined pattern-recognition stage. Only operations related to Phase 2 are presumed to be available to overt instruction. Phase 3 corresponds to Anderson's compilation but is not identical because it does not use most of the rules developed in Phase 2.

If we invoke the four criteria proposed by Anderson for distinguishing the implementational level from the algorithmic level, we find that in the BLC model (1) the operations of the lowest level are not unaffected by the organism's goals and beliefs; (2) the operations of the low-level units are affected by the context in which they are invoked, and the time they take to execute is strongly affected both by the context and by the data; (3) learning takes place throughout the system; and (4) cognitive steps in algorithmic modules are reportable, although they may not correspond to changes in working memory, because that is only one of the levels at which algorithmic modules operate. Hence, according to the four criteria, the BLC model cannot be segmented into algorithm and implementation.

It is, however, a viable model of reading, one that accounts for experimentally observable effects in the early stages of learning to read, the effects of brain damage, lexical decisions, semantic priming, visual masking, and so forth; some of these effects are a priori surprising, such as the sparing of speed-reading in a brain-damaged man who cannot read slowly. Furthermore, something like the BLC model seems to be required to solve the problem that there is a mathematically inevitable tension between error reduction and rapid response in any symbolic communication system, whether biologic or artificial (Taylor, in press). The two tracks allow context-plausible interpretations to be processed rapidly, with later correction if necessary, and permit accurate but slow interpretation of other items.

The BLC model leads us to agree with Anderson's final assertion that "the best way to study [algorithms] is through pedagogical experiments," because it is only during the learning phase that algorithms are important for behaviour. Moreover, overt teaching can have the greatest effect on learning in Phase (2), the development of pattern elements and the rules that link them. This assertion appears to be true for reading, and we have no doubt that it is equally valid in other areas of cognitive endeavour.

Algorithm cannot be separated from implementation. Biological organisms exist in a world of critical timing requirements. A Von Neumann computer can execute any algorithm, and (as Anderson says) changes in algorithm can make significant changes in speed. But a massively parallel computer can perform some algorithms several orders of magnitude faster than any Von Neumann computer that is limited by the speed of light and the atomic size of its computing elements. Implementation is very much tied up with what algorithms are permissible to the biological machine that must survive in a world that will not wait.

ACKNOWLEDGEMENT

This is DCIEM Technical note 87-N-14.

Connectionist models are also algorithmic

David S. Touretzky

Computer Science Department, Carnegie Mellon University, Pittsburgh, Pa. 15213

Most proponents of the connectionist view undoubtedly appreciate that sequential problem-solving behavior may be profitably described at an algorithmic level such as the one Anderson has developed in ACT*. A theory of LISP programming or geometry theorem proving could conceivably be expressed in several million weighted connections between perceptron-like units, but it would be silly and uninformative to do so. For his part, Anderson rightly concedes the inappropriateness of modeling ultra-low-level neural phenomena, such as the computations the retina performs, using production systems (personal communication). Thus, at the high and low ends of the complexity scale, the target article gives no cause for disagreement. Intermediate-level cognitive phenomena are the battleground on which controversial connectionist claims have been made and must be defended.

I will define an intermediate-level phenomenon, such as understanding an ambiguous sentence or recognizing a figure from a collection of separate cues, as one that results in a conscious change of mental state but that is not decomposable into consciously accessible transition states. Anderson's second criterion for identifying what he calls the algorithmic level eliminates nondecomposable phenomena by definition. But because he presents ACT* as having implications for connectionist models, which operate almost exclusively at this level, I will ignore the restriction. The principal connectionist claim is that intermediate-level phenomena are best described as massively parallel subsymbolic computations, rather than sequential rule-based ones. ACT*'s success as a model of sequential high-level problem-solving behavior has little bearing on claims about the intermediate level of cognition.

As Anderson points out, ACT* shares important features with PDP theories: spreading activation, large numbers of units, graded activity levels, computation by parallel relaxation, and learning that involves weight modification. It is the sort of high-level theory a connectionist can feel comfortable with, because in modeling high-level phenomena, one needn't be concerned that spreading activation across localist "grandmother cells" is semantically vague and anatomically implausible, or that the tricode theory of representation appears suspiciously LISP-like and unmotivated by neurological considerations. A certain degree of abstraction is necessary when discussing high-level reasoning.

It does not follow that PDP models, which in part address these concerns, may be dismissed as "implementational." PDP models involve much more than the speed and reliability of computations (Anderson's definition of the implementational level). They include a commitment to an alternative, distributed representation for knowledge (Derthick & Plaut 1986; Hinton *et al.* 1986) that is decidedly not rule based. Distributed representations give rise to useful automatic generalization phenomena (McClelland & Rumelhart 1986a; Sejnowski & Rosenberg 1986). This allows some behaviors that appear to be rule-based to be reproduced in PDP networks without resorting to rules. Anderson views this as an attempt to exclude the algorithmic level from psychological theories. But this presupposes that algorithmic descriptions require rules. The significance of rules for learning in cognitive tasks at the intermediate level (as defined here) is not as well established as for high-level tasks.

Another interesting property of distributed representations is their ability to represent different shades of meaning of a concept, as determined by context, using slight variations on the concept's canonical activity pattern (McClelland & Kawamoto 1986). One reason ACT* is not viewed as a PDP model, despite its "connectionist" flavor, is that it represents declarative memory structures and the components of patterns appearing on the

left-hand sides of rules in the classical LISP way, as atomic objects rather than as distributed patterns of activation.

The target article dredges up an early attempt at dismissing PDP models as implementational, the old "PASCAL is better than assembly language" argument, and misapplies it to the domain of problem-solving skills. Rumelhart and McClelland (1985; 1986a) prefer a different analogy for contrasting the classical and connectionist models: Newtonian versus quantum mechanics. They contend that macro-level Newtonian theories expressed in terms of rules and symbols can only approximate what really takes place in the mind, and that there will be psychological phenomena observable at the macro level that have no explanation except that they are the result of subsymbolic "quantum" effects. Although it may be possible to describe the very highest levels of cognition in purely Newtonian terms, these aren't the levels today's PDP models are primarily concerned with. (They are, however, the ones to which ACT* has been principally applied.)

The representational theories and architectural ideas being explored in the PDP school promise a fundamentally different notion of "symbol" than is in use today (Touretzky & Derthick 1987). Our notions of inference are also likely to change. For example, it is presently unclear how ACT*'s spreading activation metaphor could be realized in a system that represents concepts in a distributed fashion. Suppose the spreading activation metaphor is not such a good approximation of what really goes on in the brain? If this turns out to be the case, then some other mechanism, based perhaps on shared microfeatures or chaotic attractors, will eventually replace it. [See Skarda & Freeman: "Brains Make Chaos in Order to Make Sense of the World" *BBS* 10(2) 1987.]

Because we don't yet know how the brain works, we can't say what the best high-level approximations of cognition will look like. ACT*'s primitives are as good as any that have been proposed for modeling sequential problem solving. PDP models generally aren't concerned with this level. For the level of processing they do address, PDP models are evidence for the appropriateness of a radically different language for algorithmic descriptions.

Learning is critical, not implementation versus algorithm

James T. Townsend

Psychological Sciences Department, Purdue University, West Lafayette, Ind. 47907

Anderson puts forward the distinction between "implementational" and "algorithmic" levels and argues that, for a number of specified reasons, there should be a shift of research priority toward the algorithmic level.

The status of metatheory about psychological modeling is not well enough developed currently to provide any definitive answer to this question; perhaps it never will be. Thus, the matter becomes an issue of persuasion. Anderson writes quite cogently, but, in my opinion, fails to win the case for the general utility of the distinction, at least in a cross-theoretic fashion. On the other hand, several of his supporting points effectively argue for increased attention to experiments on learning and to related theory.

The distinction between algorithm and implementation is most cogent in the field of computer science, probably because of the inseparable historical link with digital computers. The latter evolved, of course, as a class of automata in which the state changes depend primarily on a program, or software. This clear-cut distinction is not always so compelling, as Anderson admits. In fact, when one approaches models based on the concept of neural nets or soup-like holographic storage systems, what is "program" and what is "hardware" depend greatly on the predilections of the theorist.

I do not even believe that this distinction needs to be very closely attached to the *importance* of learning. Thus, learning may modify a program or the hardware; we certainly haven't gotten the final answer from the neurophysiologists. Furthermore, the most interesting thing about the early Perceptron work (Rosenblatt 1959) was the purported ability to adapt or learn. The efforts (skirting the well-known pitfalls) of that group, flowing as they did out of the pioneering work of Rashevsky (1931), McCulloch and Pitts (1943), Ashby (1952), and others, seem independent of the present distinction. Similar points can be made, I suspect, about more recent neuralistic and distributed models, such as Grossberg's (e.g., 1980), J. A. Anderson's (e.g., 1973), Willshaw's (e.g., 1981), Murdock's (e.g., 1982), and Hopfield's (e.g., 1982), among many others.

Theorists have and will continue to emphasize, usually implicitly, more static or more adaptive aspects of their systems' architecture and behavior, depending on their unique perspectives. And, these aspects will be more or less "divied up" into implementational versus algorithmic, again depending on perspective.

Even some of the relatively simple instances of implementation the author mentions may fall into the algorithm as opposed to the implementation hopper, depending on one's outlook and the degree of detail provided by the theorist. For instance, in our own work (e.g., Townsend & Ashby 1983), four major issues – parallel versus serial processing, self-terminating versus exhaustive processing, limited versus unlimited capacity, and different types of stochastic independence versus dependence – may be interpreted under algorithmic or implementational formats, depending on the way the theorist considers them. One or more might be taken at a descriptive level with little or no "micro" interpretation, or they might be generated through finer-grained mechanisms.

All this being said, there is little doubt that learning has been sorely neglected in a cognitive science, especially in theory. I would maintain that *motivation* has been given even less attention. Providing our cognitive machines with a propelling motivation would partly ameliorate some of the complaints of Dreyfus (e.g., 1979) and others.

Underestimating the importance of the implementational level

Michael Van Kleeck

Department of Psychology, Harvard University, Cambridge, Mass. 02138

Although research at the algorithmic level defined by Anderson is undoubtedly worthwhile, in arguing for an increase in such research, he tends to underestimate the importance, theoretical tractability, and research prospects of the implementational level relative to the algorithmic level. In addition, it seems to me that he unfairly charges psychologists with using unmotivated postulates and neglecting the functional role of human cognition.

Relative importance of the two levels. Comparing the algorithmic and implementational levels, Anderson claims that the algorithmic level is more interesting, important, and fundamental. One of his arguments is that the algorithmic level accounts for most of the variation in human behavior. However, this is true only in certain areas of cognition, such as the acquisition and use of skills in symbol manipulation. In other equally important areas such as visual cognition and musical ability, the algorithmic level plays a much smaller role, whereas implementation-level factors such as attentional resources and memory capacity take on greater importance.

As a further argument for the importance of the algorithmic level, Anderson states that in computer science, algorithms and data structures have powerful effects on performance that are largely independent of hardware implementation. Implementation, however, plays a critical role in determining which algorithms are appropriate for real-world applications; silicon chips, for example, have made possible the use of algorithms that would have been prohibitively slow in the days of vacuum-tube computers. Similarly powerful implementational constraints on algorithmic appropriateness apply in human cognition. For example, the slowness of neural response rates implies that strictly serial algorithms could not account for the rapidity of visual processing (Feldman 1981).

Anderson also believes that the algorithmic level is more important and interesting than the implementational level because the former bears upon epistemological questions and essential human qualities. The implementational level, however, is equally relevant to these issues and has special advantages in dealing with some of them. For example, fundamental issues concerning knowledge representation and functional specialization have been clarified by PET (Positron-Emission Tomography) data. These data reveal striking differences in the metabolic activity of various brain areas according to stimulus modality (e.g., visual versus auditory), stimulus class in a given modality (e.g., musical versus linguistic stimuli in the auditory modality), and task strategy in a given stimulus class (e.g., visual imagery versus nonimagery strategies in a musical task) (Mazziotta et al. 1982; Phelps & Mazziotta 1985). The acquisition of new skills, cited by Anderson as a quintessentially human capacity particularly suited for study at the algorithmic level, can also be examined at the implementational level. Indeed, if we turn from symbolic skills to motor skills, the implementational level is probably the more appropriate one, as suggested by the success of explanations of invertebrate motor learning in terms of changes in the effectiveness of chemical synaptic connections (Kandel 1985). Furthermore, one can argue that even though cognitive capacities such as memory and visual cognition – capacities perhaps best studied at the implementational level – are more equally shared with other species than is skill acquisition, they are no less essential to being human.

Ratio of data to theoretical detail. Anderson claims that the ratio of data to theoretical detail is greater at the algorithmic level than at the implementational level. He argues that although phenomena at the algorithmic level are more complex, this greater complexity is offset by the greater abstraction of algorithm-level theory and by the availability of richer sources of data, especially protocols. Although these judgments of phenomenal complexity and theoretical abstraction are necessarily somewhat subjective, Anderson has probably underestimated the tremendous increase in complexity of the phenomena studied at the algorithmic level. In pedagogical experiments, for example, not only are the tasks more difficult and complex, but the subject's previous knowledge and experience also have a much greater potential impact on performance than in a typical implementation-level task such as a test of short-term memory for digits. Anderson also slights the potential richness of data at the implementational level. Behavioral data at that level, however, need not be limited to binary responses and reaction times, but instead can include, for example, multiple response alternatives. confidence ratings, and measures of the strength and speed with which response keys are depressed. Nor is one confined to behavioral data; Anderson states that protocol data are available only at the algorithmic level, but if we accept his definition of a protocol as providing "a running series of responses that can be used to infer the sequence of mental states," then informationally rich protocols are certainly available at the implementational level in the form of electrophysiological measures such as GSRs (galvanic skin responses; e.g., Dawson & Schell 1982), ERPs (event-related potentials; e.g., Desmedt 1977), and PET scans (e.g., Phelps & Mazziotta 1985).

Progress in research. According to Anderson, research on the implementational level is reaching a point of diminishing returns: The basic facts are already known, so there is not as much "ore left in the mine" at the implementational level as at the algorithmic level, and there is little progress in resolving such theoretical dichotomies as serial versus parallel processing. The establishment of basic facts, however, has by no means signaled the slowing of productive research. On the contrary, it has provided the foundation for a proliferation of fruitful investigations drawing on new methodologies, such as those using the physiological measures of brain activity mentioned above, and new theoretical orientations, such as connectionism (McClelland & Rumelhart 1986; Rumelhart & McClelland 1986). For example, ERP data have clarified the relation between attention and the parallel processing of visual inputs (Hillyard et al. 1985).

Cognition as a functional tool. In recommending increased emphasis on pedagogical experiments as a way of exploring the important issues found at the algorithmic level, Anderson claims that psychologists have ignored the evolution of human cognition as a functional tool and have instead wished to "construct the human mind out of unmotivated postulates." There may be psychologists who are guilty on these counts, but Anderson's charges are too sweeping. Visual cognition, for example, is a functional tool *par excellence* and has been insightfully analyzed in terms of the problems for whose solution it evolved (e.g., Kosslyn 1987; Marr 1982). Connectionist modelers draw on physiological plausibility as well as computational adequacy to motivate their postulates (McClelland & Rumelhart 1986; Rumelhart & McClelland 1986).

More ore. Research at the algorithmic level is undoubtedly valuable, but there is an abundance of interesting, important ore left in the implementation-level mine, and we have well-motivated theories and highly efficient techniques for the extraction of that ore. Many mother lodes still await prospectors who know where and how to look.

ACKNOWLEDGMENTS

I thank Stephen Kosslyn for his helpful discussion regarding this commentary, preparation of which was supported by a National Science Foundation Graduate Fellowship held by the author and by Office of Naval Research Contract N0014-85-K-0291 to S. Kosslyn.

Author's Response

Implementations, algorithms, and more

John R. Anderson

Department of Psychology, Carnegie-Mellon University, Pittsburgh, Pa. 15213

With some notable exceptions, I found myself largely nodding as I read through these commentaries, wishing I had made some of the points the commentators made and regretting that I had left enough ambiguities in my target article to allow the commentators to make the misinterpretations they did. (Often when the commentators provided corrections, they were asserting interpretations I had originally intended.) So, from the point of view of a reader trying to extract some useful knowledge from both the article and the commentaries, this should be a nice,

complementary package. There are, however, a few remarks I wish to make here just to set the record straight and to add a few points of embellishment. This will, one hopes, add to the informativeness of the package.

Contrary to **Rosenbloom**'s reading, it seems that most commentators found the conclusions of my article acceptable but were bothered by apparent holes in the arguments. Let us review the three conclusions in reverse order:

I was genuinely surprised that no one took issue with the third conclusion about the value of applied research in the science of cognition or about the value of pedagogical research in particular. In fact, **Glaser** and **Larkin** have even added to my assertion that pedagogical experiments are a good test bed for ideas with their observation that such experiments are a good source of theoretical ideas. I wholeheartedly agree – an omission on my part. It would be nice if we could consider this issue settled, and I would not have to deal with reviewers' assertions to the contrary the next time I submit an article or research proposal. **Seifert & Norman**'s obvious frustration has been my constant experience.

As to the second conclusion – that there is important research to be done at the algorithmic level – there seems to be a general consensus, although some (**Rosenbloom, Seifert & Norman, Townsend, Van Kleeck**) have questioned my judgment that it is more important than that at the implementational level. As conceded in the target article, this is ultimately a matter of taste, and I can do no more than try to convey the basis for my judgment. I would be quite happy to see research on the two levels proceed on equal footing. I have every intention of continuing to do research on processes at the implementational level.

With respect to the first conclusion – that there is an algorithm–implementation distinction – most commentators seem to accept some form of this distinction, but it is clear that there are a lot of questions about whether I have really gotten it right. It is to these questions that the major portion of this response will be addressed.

The algorithm–implementation distinction. I would like to start out by setting some points straight. First, the reference to the computer-science distinction was given as an analogy, and no analogy is perfect (**Arbib, Clark, Hendler, Reed**). As those who study analogies know, there is always the danger of pursuing them too literally. Once one has extracted from the analogy what it has to offer, one would do best to focus on the target domain and ignore the source domain. However, as I hope will become apparent in this response, I do not think the analogy has outlived its usefulness yet, and I will continue to use it in attempting to communicate. The many expositions of this distinction in the commentaries are enlightening. The various interpretations offered just illustrate the richness of the concept in computer science. One hopes that my communication goals will not rest on resolving the definition of the distinction in computer science. However, to help specify the mapping I have in mind in using this analogy, let us equate "algorithm" with LISP code and "implementation" with the final realization of the code that runs on the machine.

Second, I did not intend the algorithm–implementation distinction, as I was developing it, to extend to vision (**Arbib, Clark, Larkin, Mortensen, Seifert & Norman, Van Kleeck**). I even have questions about whether it extends to language (**Larkin**). It is clearly applicable in those domains that might be called reasoning and problem solving. For this reason, Marr's development of his levels is not totally appropriate for my purposes. I actually included a discussion of Marr only because a *BBS* referee had criticized me for lack of scholarship in not relating my distinction to such a thoughtful and influential analysis. I was surprised at how many commentators took it as the reference point for my development of levels. Apparently my paper does not read the same way to others as it does to me. Pylyshyn's (1984) development is much closer to mine and more appropriate to the area of reasoning and problem solving. Pylyshyn should also be consulted for a discussion of the binary distinction between algorithm and implementation (or functional architecture) in the human system in contrast to the many steps of a decomposition in a computer system (**Arbib, Hendler**).

The only way to be fully precise (**Arbib, Clark, Goldman, Reilly**) about the algorithm–implementation distinction is to point to its manifestation in a well-defined theory, as I did with respect to ACT*. For those (such as **Clark**) who are uneasy with this, three points were noted that ought to line up with the algorithmic level in any theory that makes the distinction – Pylyshyn's cognitive penetrability, my reportable working-memory changes, and my learning transitions. The last is most critical to the argument because it ties the algorithmic level to fundamental epistemological issues. It is undoubtedly the case that these criteria do not pick out a unique level in some psychological theories (e.g., those of **Taylor & Pigeau** and of **Hendler,** apparently). It is not logically necessary that the criteria should apply to all theories – some theories may not make the distinction.

Independence of levels. It is also true that even in the ACT* theory, the levels are not independent, although they are distinct. The clear point of nonindependence is the penetration of implementation-level properties into conflict resolution, which is part of the definition of the algorithmic level (**Clark, Larkin**). Without the conflict resolution specified, we have what amounts to a nondeterministic algorithm (not a nonalgorithm as **Clark** suggests) in that we cannot specify which of the applicable rules will be executed. If we consider the activation computation that underlies conflict resolution in ACT* (an implementation-level matter), we still have a nondeterministic algorithm, but we can now specify the probabilities of various paths being followed. However, the penetration of activation computation into the algorithmic level does not deny the reality of that level any more than the penetration of memory-size limitations denies the reality of a LISP program that is running on the computer.

When **Seifert & Norman** assert that the two levels are "fundamentally intertwined," I am not sure whether they are asserting that there are interactions or that the two levels are inextricably intertwined. If they mean the latter, I must simply reject their assertion. The success that scientists have had in pursuing one level and ignoring the other supports my view. Moreover, if the levels were

inextricably intertwined, we would see little progress in cognitive science. We would simply be unable to analyze a maze of interactions between two levels. In other words, "near decomposibility" (Simon 1969) is a precondition for scientific analysis and, as Simon further argues, it may also be a precondition for a functioning system. I particularly think that the near independence of the algorithmic level from the implementational level is a prerequisite for successful learning. If low-level interactions were going to blunt the attempt to acquire knowledge, there would be no hope of organizing a system that could solve complex problems. It would be a disaster if it were calculated at the algorithmic level that some piece of knowledge was needed but the implementational level could turn the manifestation of that knowledge into something entirely unintended. Just such a disaster has been the experience in machine learning when people have tried to intertwine functions such as activation computation into their rule-learning system (Rosenbloom 1983).

The psychological reality of the levels. In response to questions raised by **Arbib** and by **Taylor & Pigeau,** I really am committed to the reality of the algorithmic level. It is a reality in the subject's head, not just a convenient fiction used by the theorist. The level is real because it is the one at which the significant learning takes place. This is not to deny that ultimately everything, including the algorithmic learning, must be implemented in neural computations. In terms of the LISP analogy again, we can have a LISP program that writes LISP code (i.e., a self-programming system at the symbolic level). This LISP program may be compiled into machine code, and the code it writes may be compiled into machine code. However, the fact that everything is taking place in compiled machine code does not deny the reality that the learning transitions are defined at the symbolic level (i.e., LISP).

The machine code does not simply "implement" the algorithm without adding anything to behavior (**Arbib, Goldman**). In computers, the machine realization imposes certain temporal properties and memory limitations that were not present in the algorithm when it was first specified in the higher-level language. So, too, the implementational level in ACT* adds properties not present at the algorithmic level, namely, timing information and probability of failure.

I am less confident that the implementational level as it exists in ACT* or in the typical connectionist theory has the same psychological reality (**Hendler**) as the algorithmic level. The first reality of that sort below the algorithmic level might be something like Marr's level of hardware implementation. Because we know so little about such neural details, the implementational aspects of the typical cognitive theories may be just approximate descriptions to help us predict the details of our data. Thus, if there is an approximation (**Touretzky**), it may well exist at the implementational level and not at the algorithmic level.

Of course, all this is scientific hypothesis, and there may be no algorithmic level, either real or approximate. The hypothesis has to be judged by its empirical fruitfulness. I am no doubt biased, but it seems to me that the judgment will prove to be extremely positive. It would be a glorious discovery for science if the mind operates in terms of symbols – Newell and Simon's (1972) physical symbol hypothesis – that turned out to be true. As Newell (1980) has argued, this would rank with the greatest of scientific discoveries. It is far from obvious that such a symbol level should exist.

I am not sure how unique symbolic processing is to humans; hence I am unsure how to react to the suggestions of **Ewert.** Theorizing about the evolution of human cognition can be as perilous as theorizing about the origins of language, and for the same reason: There are no records of the cognitive activities of our ancestors. If symbol processing is to some degree unique to humans, Ewert's question about whether it depends on language is a good one. Mathematical problem solving is not exclusively linguistic (although it may well be based on the same symbol-manipulation capabilities that underlie language). Many of the production rules in our geometry tutor involve components that can be construed as verbal tests only in the most distorted sense, as **Mortensen** suggests. For example, our geometry tutor uses tests of whether points are in the same half plane, which is surely a visual pattern.

Stabler raises the issue of whether there may be some level of analysis like Marr's computational level or Chomsky's competence level and whether our inquiry into cognition ought to begin there. He may be right, and he is certainly right that the formal mathematical theories of the domains I study do not provide that level. If there is a higher level, I suspect it will prove to be something like Newell's (1981) principle of rationality: The behavior is in some sense a rational solution to the information-processing demands imposed on it. Such a principle implies that behavior is largely a function of the computational problems posed to the information-processing system, as **Stenning** emphasizes. Note that this is really a metalevel constraining our theory of the other levels and not a description of something in the head. Marr's computational level is really a metalevel in this sense.

Where is the science? It is apparent even to me that I was unclear about where the science is at the algorithmic level. **Goldman** is right in pointing out that identifying an algorithm for a particular domain cannot be the ultimate goal of basic research, although it may be so for applied research (see my remarks on human–computer interaction). The most we can hope for at this level is what **Clancey** describes, namely, a taxonomy of the characteristics of algorithms that have been uncovered. **Goldman** and **Larkin** are right too in asserting that the science is in the learning principles determining how the mind encodes the structure of the domain. In these learning principles, we all hope there is across-domain generality. Of course, one has to study algorithms in some domain to study their acquisition. Thus, the study of a domain algorithm per se is an important subgoal of the higher goal, and it is proper for scientists to publish their conclusions about domain-specific algorithms so that other scientists can have the benefit of their analyses. This is just to say that research does not have to achieve its ultimate goals to be important.

My point about generalities was really two points. First, if we look at the structure of particular algorithms, we are not going to see much in the way of generalities, because these algorithms reflect the idiosyncratic structure of the

domains to which they apply. Second, although generalities are present at the algorithmic level in the learning principles that relate the structure of the domain to the algorithms, the methodology for uncovering these learning generalities is different from the methodology for uncovering generalities at the implementational level. In studying generalities at the implementational level, we can repeat over and over again the same, or nearly the same, situation in a subject, and we can measure the phenomena precisely. When we are studying the learning of an algorithm, we focus on what is of necessity a moving target, and we cannot repeat an observation. The first episode will cause learning and, hence, we will have a different person for the second episode. It is also unlikely that two individuals will follow the same trajectory, so we cannot simply aggregate over individuals. Thus, in studying learning at the algorithmic level, the generalizations we are looking for are relationships among our observations, not the same pattern appearing over and over again in all the observations.

Protocols. As **Seifert & Norman** note, I am a recent convert to protocols, at least as measured in terms of the length of my career. Seven years ago I would have had no difficulty in writing a commentary very similar to theirs. I am hardly an expert in the use of protocols, but if serious methodological discussion is needed, I again point the reader to Ericsson and Simon (1984). The interpretation of protocols does not have to be any more subjective than our interpretation of what word a subject scribbled in a free-recall test. Ultimately, any data source has to be filtered through a theory, and protocols are no exception. To repeat a point made in the target article, tutoring is one means of objectivizing protocol analysis. **Seifert & Norman** are right in stressing that the implementational level is not always restricted to a binary response and that a multi-valued response is certainly more informative. However, these are still single measures at the end of the process of interest. Think how much more informative it is to get a running trace consisting of hundreds of such responses through the use of protocols! The other items **Seifert & Norman** list in their "rich range of data" are just analytic techniques for squeezing the most out of what is a rather impoverished set of data. **Van Kleeck** points out that we may discover some methodology permitting us to increase the information gain in implementation-level experiments and even to have running traces of the processes. Unfortunately the candidates he lists have largely proven unsatisfactory, but this does not imply that we should not explore new methodologies or try to perfect the ones **Van Kleeck** lists.

I share some of **Seifert & Norman**'s concern about statistical tests, not so much with respect to the falsification issue but with respect to the overall difficulty of testing the generality of conclusions drawn from a protocol. With a few notable exceptions (e.g., Larkin 1981) there have not been real efforts to assess the generality of one's conclusions from protocols. The reason for this generalizability problem is not something inherent in the methodology of protocols per se. It is just that it takes a great effort to collect and analyze one protocol, let alone to do this for many protocols and then analyze the relationship among them. This is a point **Seifert & Norman** make, and they really hit the nail on the head here. It represents a fourth factor I failed to mention, one that reduces the advantage of protocol analysis over "implementation methodology." Our tutors have eliminated that roadblock, however, by automating the process. We collect automatically analyzed protocols in the way we have been used to collecting response and timing data in our implementation-level experiments.

I have little to say in response to **Ericsson**'s comments except to reinforce them. My one point would be that, in contrast to his research on memory expertise, it is not possible to repeat an observation in studying the acquisition of a rapidly changing skill such as computer programming. **Rosenbloom**'s point – that in studying learning one is two steps of induction from the protocols – is well taken. The first step of induction is to infer from the protocols the rules that define the transitions. The second step is to infer the learning principles that determine the changes in the rules.

While we are on the subject of protocols, I should stress two points from the target article. First, the function of protocols is to get a running record of correlates of the states of working memory. One does not require that the student report his algorithm (**Reilly**) – that is left as an induction task for the scientist. Second, protocols are not restricted to verbal reports; methodologies such as eye movements allow one to study nonverbal problem solving (**Mortensen**). I also appreciate **Stenning**'s pointing out that free recall is a type of protocol. It allowed me to understand what I was doing in large part with my early FRAN theory (Anderson 1972) – building a theory of one algorithm that students use for performing free recall. This work had few implications for implementational issues concerned with human memory.

Connectionism. The level of commentary generated on the subject of connectionism versus symbol processing (**Smolensky, Touretzky**) is very encouraging and provides a hopeful sign that psychology has finally matured and that we are not going to play out another fruitless dichotomy such as serial versus parallel processing, propositional versus imaginal representations and so forth. The only matter on which I have to take issue with **Smolensky** and **Touretzky** concerns their suggestion that ACT* does not use distributed representations. These commentators should know better. It is perfectly possible to implement the nodes in ACT* as distributed patterns of activation over sets of elements and the associations in ACT* as associations among these patterns.

Individual differences. I am sympathetic with **Glaser**'s concerns about making contact with individual differences. Indeed, it is my major misgiving about algorithm-level theories such as ACT* that they do not give a good account of individual differences in learning. It is easy to explain performance differences in terms of various implementation-level parameters that specify the capacity of the system. Differences in learning algorithms may not be so simply explained.

Narrowness of topics. Finally, there are the comments about the narrowness of my theoretical vision. **Levine** and **Townsend** complain about the short shrift given to emotions and motivation. **Clancey, Glaser,** and **Stabler** all comment that the topics I have studied (mathematical

problem solving and computer programming) are rather narrow and that other domains would raise issues that I have ignored. In response to such comments, I can only say, "Yes, and I am glad you are studying these topics."

ACKNOWLEDGMENTS

Preparation of this response is supported by National Science Foundation grants BNS 82–08189, MDR–8470337, and IST–8318629. I would like to thank Lynne Reder for going over this manuscript with me.

References

Ackley, D. H., Hinton, G. E. & Sejnowski, T. J. (1985) A learning algorithm for Boltzmann machines. *Cognitive Science* 9:147–69. [aJRA]

Aho, A. V., Hopcroft, J. E. & Ullman, J. D. (1974) *The design and analysis of computer algorithms*. Addison-Wesley. [AVR]

Anderson, J. A. (1973) A theory for the recognition of items from short memorized lists. *Psychological Review* 80:417–38. [JTT]

(1983) Cognitive and psychological computation with neural models. In: *IEEE Transactions on Systems, Man, and Cybernectics*. IEEE. [aJRA]

Anderson, J. A. & Hinton, G. E. (1981) Models of information processing in the brain. In: *Parallel models of associative memory*, ed. G. E. Hinton & J. A. Anderson. Erlbaum. [aJRA]

Anderson, J. R. (1972) FRAN: A simulation model of free recall. In: *The psychology of learning and motivation*, vol. 5, ed. G. H. Bower. Academic Press. [rJRA]

(1976) *Language, memory, and thought*. Erlbaum. [aJRA]

(1979) Further arguments concerning representations for mental imagery: A response to Hayes-Roth and Pylyshyn. *Psychological Review* 86:395–406. [aJRA]

(1982) Acquisition of cognitive skill. *Psychological Review* 89:369–406. [JHL]

(1983) *The architecture of cognition*. Harvard University Press. [aJRA, AIG, RGR, PSR]

(in press) Analysis of student performance with the LISP tutor. In: *Diagnostic monitoring of skill and knowledge acquisition*, ed. N. Frederiksen, R. Glaser, A. Lesgold & M. Shaffo. Erlbaum. [aJRA]

Anderson, J. R., Boyle, C. F. & Reiser, B.J. (1985) Intelligent tutoring systems. *Science* 228:456–62. [aJRA]

Anderson, J. R., Farrell, R. & Sauers, R. (1984) Learning to program in LISP. *Cognitive Science* 8:87–129. [aJRA]

Anderson, J. R., Greeno, J. G., Kline, P. J. & Neves, D. M. (1981) Acquisition of problem-solving skill. In: *Cognitive skills and their acquisition*, ed. J. R. Anderson. Erlbaum. [EPS]

Anderson, J. R., Pirolli, P. & Farrell, R. (in press) Learning to program recursive functions. In: *The nature of expertise*, ed. M. Chi, R. Glaser & M. Farr. Erlbaum. [aJRA]

Anderson, J. R. & Skwarecki, E. (1986) The automated tutoring of introductory computer programming. *Communications of the ACM* 29:842–49. [AC]

Arbib, M. A. (1987) Modularity and interaction of brain regions underlying visuo-motor coordination. In: *Modularity in knowledge representation and natural language understanding*, ed. J. L. Garfield. MIT Press/Bradford Books. [MAA]

Arbib, M. A., Caplan, D. & Marshall. J. C., eds. (1982) *Neural models of language processes*. Academic Press. [J-PE]

Ashby, W. R. (1952) *Design for a brain*. Wiley. [JTT]

Atkinson, R. C. & Paulson, J. A. (1972) An approach to the psychology of instruction. *Psychological Bulletin* 78:49–61. [RG]

Barto, A. G., Sutton, R. S. & Anderson, C. W. (1983) Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics* SMC 13:834–46. [DSL]

Berwick, R. (in press) Parsability and learnability. In: *Mechanisms of language acquisition*, ed. B. MacWhinney. [aJRA]

Bindra, D. (1976). *A theory of intelligent behavior*. Wiley. [J-PE]

Bledsoe, W. W. (1977) Non-resolution theorem proving. *Artificial Intelligence* 9:1–35. [EPS]

Braine, M. D. S. (in press) What is learned in acquiring word classes: A step toward an acquisition theory. In: *Mechanisms of language acquisition*, ed. B. MacWhinney. [aJRA]

Broadbent, D. (1985) A question of levels: Comment on McClelland and Rumelhart. *Journal of Experimental Psychology: General* 114:189–92. [aJRA, KS]

Brown, J. S. & Greeno, J. (1984) *Research briefings 1984*. National Academy Press. [aJRA]

Brown, J. S. & VanLehn, K. (1980) Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science* 4:379–426. [aJRA]

Bullock, T. H. (1983) Implications for neuroethology from comparative neurology. In: *Advances in vertebrate neuroethology*, ed. J. -P. Ewert, R. R. Capranica & D. J. Ingle. Plenum. [J-PE]

Bundy, A. (1983) *The computer modelling of mathematical reasoning*. Academic Press. [EPS]

Card, S. K., Moran, T. P. & Newell, A. (1983) *The psychology of human-computer interaction*. Erlbaum [aJRA, CS]

Card, S. K. & Newell, A. (1985) The prospects for psychological science in human–computer interaction. *Human–Computer Interaction* 1:209–42. [aJRA]

Cazden, C. G. (1965) *Environmental assistance to the child's acquisition of grammar*. Doctoral dissertation, Harvard University. [aJRA]

Chandrasekaran, B. (1984) Expert systems: Matching techniques to tasks. In: *AI applications for business*, ed. W. Reitman. Ablex Publishing. [WJC]

Charniak, E. (1983) Passing markers: A theory of contextual influence in language comprehension. *Cognitive Science* 7:171–90. [JH]

Chase, W. G. & Clark, H. H. (1972) Mental operations in the comparison of sentences and pictures. In: *Cognition in learning and memory*, ed. L. Gregg. Wiley. [aJRA]

Chase, W. G. & Ericsson, K. A. (1981) Skilled memory. In: *Cognitive skills and their acquisition*, ed. J. R. Anderson. Erlbaum. [KAE]

(1982) Skill and working memory. In: *The psychology of learning and motivation*, vol. 16, ed. G. H. Bower. Academic Press. [KAE]

Chi, M. T. H., Feltovich, P. J. & Glaser, R. (1981) Categorization and representation of physics problems by experts and novices. *Cognitive Science* 5:121–52. [aJRA]

Chomsky, N. (1965) *Aspects of the theory of syntax*. MIT Press. [aJRA, AVR]

(1975) *Reflections on language*. Pantheon Books. [EPS]

(1980) Rules and representations. *Behavioral and Brain Sciences* 3:1–61. [aJRA]

Church, A. (1936) A note on the Entscheidungs problem. *Journal of Symbolic Logic* 1:40–41. [AC]

Clancey, W. J. (1984) *Acquiring, representing, and evaluating a competence model of diagnosis*. Heuristic Programing Project Memo 84–2, Stanford University. (To appear in M. Chi, R. Glaser & M. Farr, eds., *The nature of expertise*, in preparation.) [WJC]

(1985) Heuristic classification. *Artificial Intelligence* 27:289–350. [WJC]

(1986) *Viewing knowledge bases as qualitative models* (Knowledge Systems Laboratory Report 86-27). Stanford University. [WJC]

(in press) Qualitative students models. *Annual Review of Computer Science*. [aJRA]

Cottrell, G. W. (1985) *A connectionist approach to word sense disambiguation*. Doctoral dissertation, University of Rochester. [JH]

Craik, K. J. W. (1943) *The nature of explanation*. Cambridge University Press. [J-PE]

Creutzfeldt, O. (1983) *Cortex cerebri*. Springer-Verlag. [J-PE]

(1986) Gehirn und Geist. *Bursfelder Universitätsreden* 5:3–39. [J-PE]

Cronbach, L. J. & Snow, R. E. (1977) *Aptitudes and instructional methods*. Halstead Press. [RG]

Crowder, R. G. (1982) *The psychology of reading: An introduction*. Oxford University Press. [aJRA]

Dawson, M. E. & Schell, A. M. (1982) Electrodermal responses to attended and nonattended significant stimuli during dichotic listening. *Journal of Experimental Psychology: Human Perception and Performance* 8:315–24. [MVK]

Derthick, M. A. & Plaut, D. C. (1986) Is distributed connectionism compatible with the physical symbol system hypothesis? *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Amherst, Mass., pp. 639–44. [DST]

Desmedt, J. E., ed. (1977) *Language and hemispheric specialization in man: Cerebral event-related potentials*. S. Karger. [MVK]

Dreyfus, H. L. (1979) *What computers can't do*. Harper & Row. [JTT]

Ebbesson, S. O. E., ed. (1980) *Comparative neurology of the telencephalon*. Plenum. [J-PE]

(1984) Evolution and ontogeny of neural circuits. *Behavioral and Brain Sciences* 7:321–66. [J-PE]

Eccles, J. C., Ito, M. & Szentagothai, J. (1967) *The cerebellum as a neuronal machine*. Springer-Verlag. [J-PE]

Ericsson, K. A. (1985) Memory skill. *Canadian Journal of Psychology* 39:188–231. [KAE]

Ericsson, K. A. & Polson, P. G. (in press) Memory for restaurant orders. In: *The nature of expertise*, ed. M. T. H. Chi, R. Glaser & M. J. Farr. Erlbaum. [KAE]

Ericsson, K. A. & Simon, H. A. (1984) *Protocol analysis: Verbal reports as data*. MIT Press. [arJRA, KAE]

Ewert, J. -P. (1987a) Measuring visual discrimination: Principles in configurational perception. In: *Methods and aims in neuroethology*. ed. D. M. Guthrie. Manchester University Press. [J-PE]

(1987b) Neurethology: Toward a functional analysis of stimulus-response mediating and modulating neural circuitries. In: *Cognitive processes and spatial orientation in animal and man*, vol. 1, ed. P. Ellen & C. Thinus-Blanc. Dordrecht: Martinus Nijhoff. [J-PE]

Ewert, J. -P. & Finkenstädt, T. (1987) Modulation of tectal functions by prosencephalic loops in amphibians. *Behavioral and Brain Sciences* 10:122–23. [J-PE]

Fahlman, S. E. (1979) *NETL: A system for representing and using real world knowledge*. MIT Press. [JH]

Feldman, J. A. (1981) A connectionist model of visual memory. In: *Parallel models of associative memory*, ed. G. E. Hinton & J. A. Anderson. Erlbaum. [MVK]

Feldman, J. A. & Ballard, D. H. (1982) Connectionist models and their properties. *Cognitive Science* 6:205–54. [aJRA, JH]

Fodor, J. A. (1983) *The modularity of mind*. MIT Press/Bradford Books. [aJRA, EPS]

Fuster, J. (1980) *The prefrontal cortex*. Raven. [DSL]

Gaffan, D. (1976) Recognition memory in animals. In: *Recognition and recall*, ed. J. Brown. Wiley. [J-PE]

Gigley, H. M. (1983) HOPE – AI and the dynamic process of language behavior. *Cognition and Brain Theory* 6:39–88. [MAA]

Gleitman, H. (1983) *Basic psychology*. Norton. [aJRA, WJC]

Goldman, A. I. (1986) *Epistemology and cognition*. Harvard University Press. [AIG]

Griffin, D. R. (1982) *Animal mind – human mind*. Springer-Verlag. [J-PE]

Grossberg, S. (1971) On the dynamics of operant conditioning. *Journal of Theoretical Biology* 33:225–55. [DSL]

(1975) A neural model of attention, reinforcement, and discrimination learning. *International Review of Neurobiology* 18:263–327. [DSL]

(1980) How does a brain build a cognitive code? *Psychological Review* 87:1–51. [JTT]

(1982) Processing of expected and unexpected events during conditioning and attention: A psychophysiological theory. *Psychological Review* 89:529–72. [DSL]

Grossberg, S. & Levine, D. S. (1975) Some developmental and attentional biases in the contrast enhancement and short-term memory of recurrent neural networks. *Journal of Theoretical Biology* 53:341–80. [DSL]

Hebb, D. O. (1949) *The organization of behavior*. Wiley. [J-PE]

Hendler, J. A. (in press) *Integrating marker-passing and problem solving: A spreading activation approach to improved choice in planning*. Erlbaum. [JH]

Herrick, C. J. (1933) The amphibian forebrain, 8, Cerebral hemispheres and pallial primordia. *Journal of Comparative Neurology* 58:737–59. [J-PE]

Hill, J. C. (1983) A computational model of language acquisition in the two-year-old. *Cognition and Brain Theory* 6:287–317. [MAA]

Hillyard, S. A., Munte, T. F. & Neville, H. (1985) Visual-spatial attention, orienting, and brain physiology. In: *Attention and performance*, vol. 11, ed. M. I. Posner & O. S. M. Marin. Erlbaum. [MVK]

Hinton, G. E. (1987) Representing part–whole hierarchies in connectionist networks. Manuscript. [PS]

Hinton, G. E. & Anderson, J. A. (1981) *Parallel models of associative memory*. Erlbaum. [aJRA]

Hinton, G. E., McClelland, J. L. & Rumelhart, D. E. (1986) Distributed representations. In: *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 1, ed. D. E. Rumelhart, J. L. McClelland & the PDP Research Group. MIT Press/Bradford Books. [DST]

(1986a) Distributed representations. In: *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 2, *Psychological and biological models*, ed. J. L. McClelland, D. E. Rumelhart & the PDP Research Group. MIT Press/Bradford Books. [PS]

Hopfield, J. J. (1982) Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences* (U.S.A.) 79:2554–58. [JTT]

Hubel, D. H. & Wiesel, T. N. (1977) Functional architecture of macaque monkey visual cortex. *Proceedings of the Royal Society (London)* B198:1–59. [J-PE]

Hume, D. (1902) *Enquiries concerning the human understanding and concerning the principles of morals*, ed. L. A. Selby-Bigge. Clarendon Press. [J-PE]

Hutchins, E. (1986) Mediation and automatization. *Quarterly Newsletter of the Laboratory of Comparative Human Cognition* 8(2). University of California, San Diego. [CS]

Jeffrey, R. (1981) *Formal logic: Its scope and limits*, 2nd ed. McGraw-Hill. [AC]

Johnson, L. & Soloway, E. (1984) Intention-based diagnosis of programming errors. *Proceedings of the National Conference on Artificial Intelligence*, Austin, Tex. [aJRA]

Just, M. A. & Carpenter, P. A. (1979) The computer and eye processing pictures. *Behavioral Research Methods and Instrumentation* 11:172–76. [aJRA]

Kandel, E. (1985) Cellular mechanisms of learning and the biological basis of individuality. In: *Principles of neural science* (2nd ed.), ed. E. R. Kandel & J. M. Schwartz. Elsevier. [MVK]

Klopf, A. H. (1982) *The hedonistic neuron*. Hemisphere. [DSL]

Knuth, D. E. (1973) *The art of computer programming*, vol. 1, *Fundamental algorithms*. Addison-Wesley. [AC]

Kosslyn, S. M. (1980) *Image and mind*. Harvard University Press. [aJRA]

(1987) Seeing and imagining in the cerebral hemispheres: A computational approach. *Psychological Review* 94:148–75. [MVK]

Kounios, J., Osman, A. M. & Meyer, D. E. (1987) Structure and process in semantic memory: New evidence based on speed-accuracy decomposition. *Journal of Experimental Psychology: General* 116:3–25. [PS]

Laird, J. E. & Newell, A. (1983) Universal weak method: Summary of results. In: *Proceedings of the Eight IJCAI*, vol. 1. International Joint Conference on Artificial Intelligence. [aJRA]

Laird, J. E., Newell, A. & Rosenbloom, P. S. (in press) SOAR: An architecture for general intelligence. *Artificial Intelligence* 33. [PSR]

Larkin, J. H. (1981) Enriching formal knowledge: A model for learning to solve textbook physics problems. In: *Cognitive skills and their acquisition*, ed. J. R. Anderson. Erlbaum. [arJRA, EPS]

Leven, S. J. & Levine, D. S. (submitted) A theory of decision-making in psychology and economics: Emotion, reason, and optimality revisited. [DSL]

Levine, D. S. (1986) A neural network theory of frontal lobe function. In: *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Erlbaum. [DSL]

Lorenz, K. (1943) Die angeborenen Formen möglicher Erfahrung. *Zeitschrift für Tierpsychologie* 5:235–409. [J-PE]

Loveland, D. W. (1984). Automated theorem proving: A quarter century review. In: *Automated theorem proving: After 25 years*, ed. W. W. Bledsoe & D. W. Loveland. American Mathematical Society. [EPS]

Lynch, G., McGaugh, J. L. & Weinberger, N. M., eds. (1984) *Neurobiology of learning and memory*. Guilford Press. [J-PE]

MacLean, P. (1970) The triune brain, emotion, and scientific bias. In: *The neurosciences second study program*, ed. F. O. Schmitt. Rockefeller University Press. [DSL]

Marr, D. (1979) Representing and computing visual information. In: *Artificial intelligence: An MIT perspective*, vol. 2, ed. P. H. Winston & R. H. Brown. MIT Press [EPS]

(1982) *Vision: A computational investigation into the human representation and processing of visual information*. W. H Freeman. [aJRA, MAA, AC, JH, MVK, AVR]

Mazziotta, J. C., Phelps, M. E., Carson, R. E. & Kuhl, D. E. (1982) Tomographic mapping of human cerebral metabolism: Auditory stimulation. *Neurology* 32:921–37. [MVK]

McLaughlin, B. (1984) *Second language acquisition in childhood*. Erlbaum. [aJRA]

McClelland, J. L. & Kawamoto, A. H. (1986) Mechanisms of sentence processing: Assigning roles to constituents. In: *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 2, ed. J. L. McClelland, D. E. Rumelhart & the PDP Research Group. MIT Press/Bradford Books. [DST]

McClelland, J. L. & Rumelhart, D. E., eds. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 2, *Psychological and biological models*. MIT Press/Bradford Books. [aJRA, MVK]

McClelland, J. L. & Rumelhart, D. E. (1986a) On learning the past tenses of English verbs. In: *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 2, ed. J. L. McClelland, D. E. Rumelhart & the PDP Research Group. MIT Press/Bradford Books. [DST]

McCulloch, W. S. & Pitts, W. (1943) A logicalculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* 5:115–33. [JTT]

Miller, G. (1986) Dismembering cognition. In: *One hundred years of psychological research in America*, ed. G. S. Hulse & B. F. Green, Jr. Johns Hopkins University Press. [RG]

Moran, T. (1983) Getting into a system: External–internal task-mapping analysis. In: *Proceedings of CHI 1983: Human factors in computing systems.* Boston: CHI (Computer Human Interaction). [aJRA]

Morgan, J. L. & Newport, E. L. (1981) The role of constituent structure in the induction of an artificial language. *Journal of Verbal Learning and Verbal Behavior* 20:67–85. [aJRA]

Mortensen, C. (1985) Mental images: Should cognitive science learn from neurophysiology? Presented at AAP Conference on Cognitive Science, University of N.S.W. (To appear in the published proceedings, ed. P. Slezack. Reidel. [CM]

Mountcastle, V. B. (1957) Modality and topographic properties of single neurons of cat's somatic sensory cortex. *Journal of Neurophysiology* 20:403–34. [J-PE]

Mueller, G. E. (1911) Zur Analyse der Gedaechtnistaetigkeit und des Vorstellungsverlaufes: Teil I. *Zeitschrift für Psychologie Ergaenzungsband 5.* [KAE]

(1917) Zur Analyse der Gedaechtnistaetigkeit und des Vorstellungsverlaufes: Teil II. *Zeitschrift für Psychologie Ergaenzungsband* 9. [KAE]

Murdock, B. B. (1982) A theory for the storage and retrieval of items and associative information. *Psychological Review* 89:609–26. [JTT]

Nauta, W. (1971) The problem of the frontal lobe: A reinterpretation. *Journal of Psychiatric Research* 8:167–87. [DSL]

Nelson, K. E., Carskadden, G. & Bonvillian, J. D. (1973) Syntax acquisition: Impact of experimental variation in adult verbal interaction with the child. *Child Development* 44:497–504. [aJRA]

Neves, D. M. (1981) Learning procedures from examples. Unpublished doctoral dissertation. Department of Psychology, Carnegie-Mellon University. [aJRA]

Newell, A. (1973) Production systems: Models of control structures. In: *Visual information processing,* ed. W. G. Chase. Academic Press. [aJRA]

(1980) Physical symbol systems. *Cognitive Science* 4:135–83. [arJRA]

(1981) The knowledge level. *AI Magazine* 2:1–20. [arJRA]

Newell, A. & Rosenbloom, P. (1981) Mechanisms of skill acquisition and the law of practice. In: *Cognitive skills and their acquisition,* ed. J. R. Anderson. Erlbaum. [aJRA]

Newell, A. & Simon, H. A. (1972) *Human problem solving.* Prentice-Hall. [arJRA, CS]

Norman, D. A. (1981) Categorization of action slips. *Psychological Review* 88:1–15. [aJRA]

(in press) *The psychology of everyday things.* Basic Books. [CS]

Northcutt, R. G. (1981) Evolution of the telencephalon in nonmammals. *Annual Review of Neuroscience* 4:301–50. [J-PE]

(1986) Strategies of comparison: How do we study brain evolution? *Verhandlungen der Deutschen Zoologischen Gesellschaft* 79:91–103. [J-PE]

O'Keefe, J. & Nadel, L. (1979) Précis of O'Keefe & Nadel's *The hippocampus as a cognitive map. Behavioral and Brain Sciences* 2:487–533. [J-PE]

Paillard, J. (1987) Cognitive versus sensorimotor encoding of spatial information. In: *Cognitive processes and spatial orientation in animal and man,* vol. 2, ed. P. Ellen & C. Thinus-Blanc. Dordrecht: Martinus Nijhoff. [J-PE]

Phelps, M. E. & Mazziotta, J. E. (1985) Positron emission tomography: Brain function and biochemistry. *Science* 228:799–809. [MVK]

Piaget, J. (1971) *Biology of knowledge.* Edinburgh University Press. [J-PE]

Pirolli, P. L. & Anderson, J. R. (1985) The role of learning from examples in the acquisition of recursive programming skills. *Canadian Journal of Psychology* 39:240–72. [KAE]

Ploog, D. & Gottwald, P. (1974) *Verhaltensforschung: Instinkt, Lernen, Hirnfunktion.* Urgan & Schwarzenberg. [J-PE]

Polson, P. G. & Kieras, D. E. (1985) A quantitative model of the learning and performance of text editing. In: *Proceedings of the conference on human factors in computer systems,* ed. CHI (Computer Human Interaction). Association for Computing Machinery. [aJRA]

Pylyshyn, Z. W. (1980) Computation and cognition: Issues in the foundations of cognitive science. *Behavioral and Brain Sciences* 3:111–69. [aJRA, JH, AVR]

(1981) The imagery debate: Analogue media versus tacit knowledge. *Psychological Review* 88:1–24. [aJRA]

(1984) *Computation and cognition.* MIT Press/Bradford Books. [rJRA]

Quillian, M. R. (1966) *Semantic memory.* Doctoral dissertation, Carnegie Institute of Technology. (Available as [1966] Report 2, Project 8668, Bolt, Baranek, and Newman.) [JH]

Rashevsky, N. (1931) Learning as a property of physical systems. *Journal of General Psychology* 5:207–29. [JTT]

Reed, A. V. (1973) Speed-accuracy tradeoff in recognition memory. *Science* 181:574–76. [AVR]

(1976) List length and the time-course of recognition in immediate memory. *Memory and Cognition* 4:16–30. [AVR]

Resnick, L. (1982) Syntax and semantics in learning to subtract. In: *Addition and subtraction: A cognitive perspective,* ed. T. Carpenter, J. Moser & T. Romberg. Erlbaum. [aJRA]

Rogers, H., Jr. (1967) *Theory of recursive functions and effective computability.* McGraw-Hill. [AC]

Rosenblatt, F. (1959) Two theorems of statistical separability in the perception. *Proceedings of a symposium on the mechanization of thought processes,* Her Majesty's Stationary Office, London. [JTT]

Rosenbloom, P. S. (1983) The chunking of goal hierarchies: A model of practice and stimulus-response compatibility. Unpublished Ph.D. thesis, Carnegie-Mellon University. [rJRA]

Rosenbloom, P. S. & Newell, A. (1986) The chunking of goal hierarchies: A generalized model of practice. In: *Machine learning II,* ed. R. S. Michalski, J. G. Carbonell & T. M. Mitchell. Morgan Kaufman. [aJRA]

Rumelhart, D. E. & McClelland, J. L. (1985) Levels indeed! A response to Broadbent. *Journal of Experimental Psychology: General* 114:193–97. [aJRA, AC, KS, DST]

Rumelhart, D. E. & McClelland, J. L., eds. (1986) *Parallel distributed processing: Explorations in the microstructure of cognition,* vol. 1, *Foundations.* MIT Press/Bradford Books. [aJRA, MAA, J-PE, MVK]

Rumelhart, D. E. & McClelland, J. L. (1986a) PDP models and general issues in cognitive science. In: *Parallel distributed processing: Explorations in the microstructure of cognition,* vol. 1, ed. D. E. Rumelhart, J. L. McClelland & the PDP Research Group. MIT Press/Bradford Books. [DST]

Rumelhart, D. E. & Norman, D. A. (1982) Simulating a skilled typist: A study of skilled cognitive-motor performance. *Cognitive Science* 6:1–36. [CS]

Rumelhart, D. E., Smolensky, P., McClelland, J. L. & Hinton, G. E., (1986) Schemata and sequential thought processes in parallel distributed processing models. In: *Parallel distributed processing: Explorations in the microstructure of cognition,* vol. 2. *Psychological and biological models,* ed. J. L. McClelland, D. E. Rumelhart & the PDP Research Group. MIT Press/Bradford Books. [PS]

Sauers, R. & Farrell, R. (1982) *GRAPES user's manual* (Technical Report ONR-82-3). Carnegie-Mellon University. [aJRA, PSR]

Schopenhauer, A. (1883) *The world as will and idea.* (Translation by R. B. Haldane & I. Kemp) London: Kegan Paul, Trench, Trubner. [J-PE]

Sebrechts, M. M. & Black, J. B. (1982 Software technology: A rich new domain for applied psychology. *Applied Psycholinguistics* 3:123–43. [aJRA]

Sejnowski, T. J. & Rosenberg, C. R. (1986) *NETtalk: A parallel network that learns to read aloud* (Technical Report JHU/EECS-86/01). The Johns Hopkins University Electrical Engineering and Computer Science Department. [DST]

Shastri, L. (1985) *Evidential reasoning in semantic networks: A formal theory and its parallel implementation.* Doctoral dissertation, University of Rochester. [JH]

Simon, H. A. (1969) *The sciences of the artificial.* MIT Press. [arJRA]

(1981) *The sciences of the artificial* (rev. ed.). MIT Press. [JHL]

Smolensky, P. (1986) Neural and conceptual interpretations of parallel distributed processing models. In: *Parallel distributed processing: Explorations in the microstructure of cognition,* vol. 2. *Psychological and biological models,* ed. J. L. McClelland, D. E. Rumelhart & the PDP Research Group. MIT Press/Bradford Books. [PS]

(1987a) Connectionist AI, symbolic AI, and the brain. *AI Review* 1:95–109. [PS]

(1987b) *On the proper treatment of connectionism* (Technical Report CU-CS-359-87). Department of Computer Science, University of Colorado at Boulder. In preparation for *Behavioral and Brain Sciences,* vol. 11 (1988). [PS]

(1987c) *On variable binding and the representation of symbolic structures in connectionist systems* (Technical Report CU-CS-355-87). Department of Computer Science, University of Colorado at Boulder. [PS]

Staszewski, J. (1986) The psychological reality of retrieval structures: An investigation of expert knowledge. Unpublished Ph.D. dissertation, Cornell University. [KAE]

Stenning, K., Shepherd, M. & Levy, J. P. (1987) *On the construction of representations for individuals during text comprehension* (Research Paper No. 9). Centre for Cognitive Science, University of Edinburgh. [KS]

Sternberg, S. (1969) Memory scanning: Mental processes revealed by reaction time experiments. *American Scientist* 57:421–57. [aJRA]

Sutton, R. S. & Barto, A. G. (1981) Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review* 88:135–70. [DSL]

Taylor, M. M. (1984) The Bilateral Cooperative Model of reading: A human paradigm for artificial intelligence. In: *Artificial and human intelligence,*

ed. A. Elithorn & R. Banerji. Elsevier. (First published in 1981 by Defence and Civil Institute of Environmental Medicine, Ontario, Canada, as Research Paper 81-P-4). [MMT]

(in press) Natural dialogue is not natural language: Response timing in a layered protocol. In: *Structure of multimodal dialogue*, ed. M. M. Taylor, F. Néel & D. G. Bouwhuis. North-Holland. [MMT]

Taylor, I. K. & Taylor, M. M. (1983) *The psychology of reading*. Academic Press. [MMT]

Thompson, T. & Clancey, W. J. (1986) A qualitative modeling shell for process diagnosis. *IEEE Transactions on Software* 3(2):6–15. [WJC]

Thorndike, E. L. (1922) *The psychology of arithmetic*. Macmillan. [RG]

Tikhomirov, O. (1985) Informal heuristic principles of motivation and emotion in human problem solving. In: *Methods of heuristics*, ed. R. Groner, M. Groner & W. Bischof. Erlbaum. [DSL]

Tinbergen, N. (1951) *The study of instinct*. Clarendon Press. [J-PE]

Tolman, E. C. (1948) Cognitive maps in rats and men. *Psychological Review* 55:189–208.[J-PE]

Touretzky, D. S. (1986) BoltzCONS: Reconciling connectionism with the recursive nature of stacks and trees. *Proceedings of the 8th Conference of the Cognitive Science Society*, Amherst, Mass. [PS]

Touretzky, D. S. & Derthick, M. A. (1987) Symbol structures in connectionist networks: Five properties and two architectures. COMPCON Spring 87, 32nd IEEE Computer Society International Conference, San Francisco. [DST]

Touretzky, D. S. & Hinton, G. E. (1985) Symbols among the neurons: Details of a connectionist inference architecture. *Proceedings of the International Joint Conference on Artificial Intelligence*. [PS]

Townsend, J. T. (1974) Issues and models concerning the processing of a finite number of inputs. In: *Human information processing: Tutorials in performance and cognition*, ed. B. H. Kantowitz. Erlbaum. [aJRA]

Townsend, J. T. & Ashby, F. G. (1983) *Stochastic modeling of elementary psychological processes*. Cambridge University Press. [JTT]

Tulving, E. (1983) *Elements of episodic memory*. Oxford University Press. [aJRA]

Uexküll, J. V. (1909) *Umwelt und Innenwelt der Tiere*. Springer-Verlag. [J-PE]

Vanegas, H., ed.(1984) *Comparative neurology of the optic tectum*. Plenum. [J-PE]

VanLehn, K. (1983) *Felicity conditions for human skill acquisition: Validating an AI-based theory* (Technical Report CIS-21). Palo Alto: Xerox Parc. [aJRA]

Vinogradova, O. (1975) Hippocampus and the orienting reflex. In: *Neuronal mechanisms of the orienting reflex*, ed. E. N. Sokolov & O. Vinogradova. Erlbaum. [J-PE]

Walker, S. (1983) *Animal thought*. Routledge & Kegan Paul. [J-PE]

Wertheimer, M. (1959) *Productive thinking*. Harper & Row. [RG]

Willshaw, D. (1981) Holography, associative memory, and inductive generalization. In: *Parallel models of associative memory*, ed. G. E. Hinton & J. A. Anderson. Erlbaum. [JTT]

Woolf, B. & McDonald, D. D. (1984) Building a computer tutor: Design issues. *IEEE Transactions on Computers* 17:61–73. [MAA]