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 subconscious "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 (Tourtelot & Dereth 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 "divided up" into implementational versus algorithmic, again depending on perspective.

Even some of the relatively simple instances of implementa-
tion 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 exhaus-tive 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 the domain of problem-solving 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.

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 al-
gorithmic and implementational levels, Anderson claims that the algorithmic level is more interesting, important, and funda-
mental. 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 mechanics of simple reaction time, and use of simple skills. 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.