Soar as a Unified Theory of Cognition: Spring 1990

Richard L. Lewis	School of Computer Science, Carnegie Mellon University
Scott B. Huffman	Department of Electrical Engineering and Computer Science, University of Michigan
Bonnie E. John	School of Computer Science, Carnegie Mellon University
John E. Laird	Department of Electrical Engineering and Computer Science, University of Michigan
Jill Fain Lehman	School of Computer Science, Carnegie Mellon University
Allen Newell	School of Computer Science, Carnegie Mellon University
Paul S. Rosenbloom	Information Sciences Institute, University of Southern California
Tony Simon	Department of Psychology, Carnegie Mellon University
Shirley G. Tessler	School of Computer Science, Carnegie Mellon University

Soar is a theory of cognition embodied in a computer system. In 1987 it was used as the central exemplar to make the case that cognitive science should attempt unified theories of cognition (UTC) [13] [1]. Since then, much research has been done to move Soar toward being a real UTC, rather than just an exemplar. Figure 1 lists the relevant studies[2]. They have been done by a broad community of researchers in the pursuit of a multiplicity of interests. This symposium presents four of these studies to convey the current state of Soar as a UTC (their names are marked with asterisks in the figure). This short paper provides additional breadth and context.

THE SOAR ARCHITECTURE

We review here the basic structure of the Soar architecture, which has been described in detail elsewhere [8, 13, 20]. Soar formulates all tasks in *problem spaces*, in which *operators* are selectively applied to the *current state* to attain *desired states*. Problem spaces appear as triangles in Figure 2 (which describes a Soar system for comprehending natural language). Problem solving proceeds in a sequence of *decision cycles* that select problem spaces, states, and operators. Each decision cycle accumulates knowledge from a long term *recognition memory* (realized as a production system). This memory continually matches against *working memory*, elaborating the current state and retrieving preferences that encode knowledge about the next step to take. Access of recognition memory is involuntary, parallel, and rapid (assumed to take on the order of 10 milliseconds). The decision cycle accesses recognition memory repeatedly to quiescence, so each decision cycle takes on the order of 100 milliseconds.

If Soar does not know how to proceed in a problem space, an *impasse* occurs. Soar responds to an impasse by creating a subgoal in which a new problem space can be used to acquire the needed knowledge. If a lack of knowledge prevents progress in the new space, another subgoal is created and so on, creating a goal-subgoal hierarchy. Figure 2 shows how *multiple problem spaces* arise. Once an impasse is resolved by problem solving, the *chunking* mechanism adds new productions to recognition memory encoding the results of the problem solving, so the impasse is avoided in the future. All incoming perception and outgoing motor commands flow through the state in the top problem space (which occurs above the spaces in Figure 2).

FOUR EXAMPLES OF RECENT PROGRESS

NL-Soar (Huffman, Lehman, Lewis and Tessler)

The goal of the NL-Soar work is to develop a general natural language capability that satisfies the constraint of real-time comprehension. To achieve rates of 200-300 words per minute, NL-Soar must *recognitionally* bring to bear multiple sources of knowledge (e.g., syntactic, semantic, and task knowledge). In addition to

[1] Soar research encompasses artificial intelligence and human computer interaction (HCI) as well, which we will largely ignore here.

[2] We include several unpublished studies to better convey Soar's current state.

Name		Domain	Phenomena modelled	Notes	References
	Neuro-Soar	Neural net implementation of Soar	Implementation technology of the architecture	A	[17]
	Visual Attention	Attention within field of view	Conjunctive, disjunctive feature searches; illusory conjunction	A	[27]
	Immed. Behavior	Simple reaction tasks; typing	Performance times	Z	[13]
	ET-Soar	Intelligent tutor for electrostatics	Perception-action loop in problem-solving with external display	S	[26]
	Phone Operator	Operators assisting in collect calls	Performance time of task involving multiple modalities	Z	[4]
	* Browser-Soar	Computer browsers	Browsing an on-line reference manual	P	[5]
short	* Robo-Soar	Puma robot arm in blocks world	On-demand flexible planning, interruptibility	S	[7,9]
	* Hero-Soar	Hero mobile robot picking up cups	Reactive execution; operating in noisy, uncertain environments	S	[9]
	* NL-Soar	Natural language comprehension	Real-time comprehension; integrates multiple knowledge sources	S	[11]
	IR-Soar	Immediate reasoning tasks (< 60s)	Regularities across multiple IR tasks (syllogisms, Wason, etc.)	Q	[15,16]
	IR-Soar(Syl)	Categorical syllogisms	Individual differences	I	[15,16]
	BI-Soar	Taking instructions for IR tasks	Acquisition of new tasks from natural language instructions	S	[11]
Time Scale	* SC-Soar	Series completion	Relative difficulty of different series	Q	
	Subtraction	Multi-column subtraction	Learning new subtraction procedure; two common bugs	S, N	[20]
	Cryptarithmetic	Cryptarithmetic puzzles	Control of search in cryptarithmetic task space	P, N	[13]
	TOH-Soar	Tower of Hanoi	Strategy change, within-trial learning	P	[21]
	Verbal Learning	Paired-associate, cued recall, etc.	Learning new declarative knowledge (knowledge level learning)	S	[19]
	Inductive KLL	Concept learning	Inducing concepts from examples (knowledge level learning)	S	[18]
long	Text Editor	Learning to use a text editor	Conceptual errors in using a new text editor	Q	[28]
	Data-Soar	Learning new data structures	Use of analogy in learning "stack" data type	P	[24]
	Generic Design	Architectural, instructional design	Design of instructional material for word processor	P	[14]
	Lexical Acquis.	Early lexical acquisition	Prototype effects; underextension; one-concept-one-word bias	Q	[12]
	ABC-Soar	Analysis-based concept acquisition	Acquisition of number conservation concepts	S	[23]
	Balance Beam	Balance beam task	Developmental transitions in balance beam task	Q, N	[13]

NOTES: A: work on the architecture itself I: accounts for individual differences N: work no longer in progress P: protocol simulation
Q: qualitative coverage of phenomena S: sufficiency demonstration Z: zero parameter predictions

Figure 1: **Some efforts on Soar as a unified theory of cognition.**

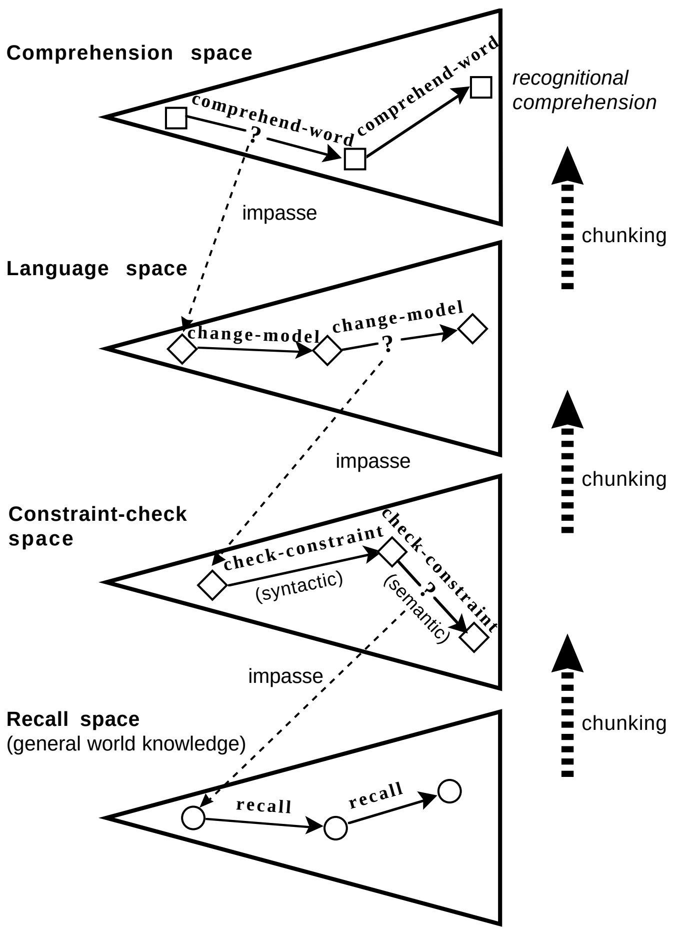

Figure 2: **Achieving recognitional comprehension from deliberate comprehension in NL-Soar.**

supporting this kind of processing, the system must integrate with the rest of Soar to work in any task that requires language understanding.

NL-Soar realizes real-time comprehension by applying a *single operator* to comprehend each word as it is read, dropping into lower problem spaces to complete the comprehension when the required knowledge is not directly available (i.e., an impasse is reached). Figure 2 shows this behavior. As in previous work [11], we assume that comprehension builds a *situation model* of what the utterance is about, and an *utterance model* that encodes the syntactic structure of the utterance. When Soar impasses on a **comprehend-word** operator, it drops into the **Language** space, which has operators that make changes to the two models. Any kind of knowledge can propose an operator (e.g., syntactic or semantic), but operators have constraints that must be satisfied before they can apply. These constraints can be arbitrarily complex syntactic, semantic, or pragmatic tests. The constraints are checked serially in the **Constraint-check** space, which consults the situation and utterance models. Checking semantic constraints may require accessing general world knowledge, which occurs in the **Recall** space. Other spaces may be involved in comprehension; for example, referent resolution happens in the **Resolve-referent** space (not shown in the figure). Once the deliberate problem solving produces the knowledge to resolve the **comprehend-word** operator's impasse at the top, chunking adds new elements to long term memory so comprehension can proceed by recognition for similar utterances and contexts in the future. These chunks do the constraint checking in parallel, integrating into a single recognition the disparate knowledge sources tapped in the deliberate problem solving.

NL-Soar is a functioning language system: its syntactic knowledge currently comprises about 75% of James Allen's outline of English in [1], and it can handle the semantics of simple instructions for the immediate reasoning [11] and Robo-Soar blocks-world tasks [7, 9]. The novel aspect of the recent work is using chunking to routinely move knowledge up the problem space hierarchy to create an integrated recognitional comprehension system. A preliminary analysis over a very small set of test sentences indicates that about half the chunks perform lexical access (i.e., the chunks are specific to lexical items). The remaining non-lexical chunks are also fairly specific, usually tied to particular syntactic forms and semantic categories. While there is transfer to new utterances, the transfer is limited by the specificity of the chunks. This implies a model of comprehension that depends on a large number of fairly specific chunks [13]. This does not mean there is no generality in NL-Soar's linguistic knowledge. Quite the opposite, the **Language** space comprises very general sources of knowledge. It simply means that the generality is limited at the recognitional level. Additional research is necessary to fully establish the sufficiency of this model.

SC-Soar (Simon)

The goal of the SC-Soar work is to develop a model of regularity detection as it occurs in series completion tasks (e.g., A, B, A, C, A, ?). Solving these puzzles requires the capability to hypothesize and test the underlying rule of the series. The series-completion task provides a microcosm for studying concept acquisition; it involves inducing a concept from examples and encoding the environment in terms of the new concept.

SC-Soar solves a series completion problem by casting it as a comprehension task: comprehension operators are applied to each item in the series, and subspaces encode the knowledge of how to hypothesize and test the underlying rule. The structure is that of NL-Soar, although the knowledge is not about language but about the relations that characterize the series, and the focus is on the deliberate phase of this specialized comprehension. Comprehending an item involves finding relationships between the item and other items in the series (relationships can be alphabetic or identity). To establish a relationship, attention must be focused on at least one other item; the *attention operators* posited for the work on immediate reasoning [16] provide this functionality. Attending to other items may also trigger knowledge to hypothesize a period size for the series. Once a period is hypothesized, SC-Soar *re-represents* the series by structuring it into groups of items. This is not accomplished in one massive step, but occurs by re-reading the series. Thus, SC-Soar exhibits a form of progressive deepening. Comprehension chunks learned during the first pass transfer to subsequent passes, so that some of the deliberate processing can be avoided. After this re-encoding, comprehension proceeds by establishing relationships between groups rather than single items. A relationship between groups, combined with the simpler relationships within groups, establishes a hypothesis for the underlying rule of the series. This hypothesis is tested by generating expectations and matching them against the rest of the series.

The novel features of SC-Soar are structuring the system into comprehension operators, modulating problem solving by attention, and learning during the task. The first two have independent support for their plausibility from other work [11, 13, 16]. The learning is important for modeling progressive deepening behavior and improvement across trials. Currently SC-Soar solves ten out of fifteen series in the original Simon and Kotovsky set [22]. The five series it cannot solve were among the most difficult. For the ten it solves, the number of decision cycles it takes to hypothesize the correct pattern provides a measure of duration that corresponds closely to the relative difficulty of these series for the subjects. The learning has not yet been compared with human data.

Robo-Soar and Hero-Soar (Laird)

The goal of the Robo-Soar and Hero-Soar work is to get Soar to interact with an external environment. The functional demands of working in a dynamic environment include *reactive execution*, *interruptibility*, *flexible on-demand planning*, and the *conversion of deliberate planning to reactive execution*. Robo-Soar and Hero-Soar are robotic systems developed to explore how Soar can support these capabilities [7, 9]. Robo-Soar controls a Puma arm and receives perceptual input through a camera; its task is to align blocks in a work area, unless interrupted by a flashing "trouble" light, in which case it should immediately push a button. Hero-Soar controls a mobile robot with grippers and sonar sensors; its task is to navigate around a room looking for cups to retrieve.

Robo-Soar and Hero-Soar cope with the demands of external interaction by taking advantage of the three different ways that Soar can intend actions: by recognition, by deliberate operator selection, and by unrestricted problem solving. At the lowest level, actions can be evoked by a single chunk in recognition memory. This supports high-speed response for automatic reflexes, such as stopping the wheel motors when Hero-Soar detects an object in its path. No deliberation is involved, hence the action cannot be modulated by additional knowledge. At the next level of control, Soar chooses an action by selecting an operator in a problem space. For example, in Robo-Soar operators exist for **close-gripper**, **open-gripper**, and **move-gripper**. At each decision cycle, recognition memory retrieves all the relevant preferences about what action to do next, and the decision procedure interprets these preferences to make a unique selection. Thus, the decision is based upon an integration of all immediately available knowledge, rather than an isolated stimulus. This makes Soar *interruptible,* as at each decision cycle any operator can be proposed and considered (e.g., the operator to push the button). The cost of bringing more knowledge to bear is time. However, the decision cycle is fast enough (about ten decisions per second) that the operator level effects reactive execution. When Soar lacks the knowledge to select the next action uniquely, an impasse arises. Soar responds by formulating this selection as a problem to be solved in a problem space. Planning is but one of a number of methods available. For example, Soar could ask an outside agent for help. Once the impasse is resolved, chunks are added to recognition memory that will allow the decision to be made in the future without problem solving. Thus, as in NL-Soar (Figure 2), chunking supports the conversion of deliberate processing to reactive execution.

Robo-Soar and Hero-Soar are not psychological models of perception and action. However, whether the goal is to build a mobile robot or model human behavior, the functional demands of working in a dynamic, uncertain environment are the same. From the UTC perspective, the Robo/Hero-Soar work establishes that Soar is sufficient to handle some of these demands and permits the exploration of how perception and motor behavior integrate with central cognition and learning. It paves the way for adopting psychologically realistic perceptual and motor systems.

Browser-Soar (John)

The goal of the Browser-Soar work is to model how humans interact with a reactive environment (a computer browser) in a rapid perception-action loop [5]. Browser-Soar is intended to be a detailed psychological model of the moment-by-moment interaction.[3] Browsers are application programs that allow multiple access paths to data bases via recognition of pointers (as opposed to retrieval by a query command language). The use of browsers is characterized by both deliberate search and opportunistic switches in search strategy. The specific browser studied is the on-line help system for cT, an interactive graphics programming language [4]. This browser consists of three windows: one provides access to topics via a hierarchical menu, another

[3]Browser-Soar does not yet interface with actual robotic effectors and perceptual devices.

[4]cT was developed at the Center for the Design of Educational Computing (CDEC), Carnegie Mellon University.

provides access via an alphabetic list, and the third displays the help text for a selected topic.

Browser-Soar consists of a set of problem spaces that provide the capability to search deliberately through the help windows, while allowing recognition of new items to trigger knowledge at any time that may change the search strategy. The top problem space in Browser-Soar (**Browse**) is entered when an impasse arises in the task space for programming in the cT language. A browsing episode involves bringing up the help window, finding the appropriate help, and applying the newly found information to the problem at hand. Each of these activities corresponds to an operator in the **Browse** space. Currently, Browser-Soar implements the **find-appropriate-help** operator. Applying this operator results in an impasse because the operator cannot be implemented by recognition. Soar responds by setting up another problem space, with operators that define the search criteria (e.g., what labels to look for in the help windows), define the evaluation criteria (how to decide that some piece of information will actually help resolve the impasse in the task space), carry out the search, and evaluate the search results. Each of these operators is also implemented in a problem space; for example, carrying out the search for the defined criteria is accomplished in a space with operators that select among search methods and execute them. At the bottom of the problem space hierarchy are motor operators that control mouse and keyboard actions, and cognitive operators that can be applied with directly available knowledge. The operators in Browser-Soar can be viewed as deliberate goals, and this organization is useful for modeling the goal-oriented component of browsing as well as the mechanics of manipulating the windows used for browsing. Data-driven, opportunistic behavior emerges because an additional operator, **evaluate-new-items**, is proposed whenever new information is brought within the scope of attention (**evaluate-new-items** is available in every Browser-Soar problem space). The current problem solving is thus *interrupted* (as in Robo/Hero-Soar) so the new items may be considered, possibly suggesting a more relevant path to pursue.

Browser-Soar simulates in detail a 67 second episode of a cT programmer using the on-line help to try to find a particular graphics command. The simulation shows the user's behavior to be largely GOMS-like [2], i.e. a *routine* cognitive skill, even though, because of the continuous rapid feedback, the behavior might seem more open and intelligently spontaneous. On the other hand, the episode contains three failed accesses of help text and two changes of search strategy. To handle these requires forms of control that lie outside of the GOMS-style procedural hierarchy (e.g., the interrupt). Thus, Browser-Soar not only shows that this browsing behavior is GOMS-like, but characterizes the ways in which it departs from that simple model.

MAKING PROGRESS TOWARD A UNIFIED THEORY OF COGNITION

We now step back from an examination of specific models to look at how Soar as a whole is developing. Breadth should be the hallmark of a UTC. In fact, a UTC effort that engages a large number of independent investigators will by nature proceed breadth-first. Soar certainly fits this model, and continues to cover a wide range of cognitive phenomena (Figure 1)[5]. The task variety exhibited in Figure 1 ranges from traditional problem solving tasks like Tower of Hanoi, to highly skilled tasks such as natural language, to planning and interacting in a dynamic external environment. The span of time-scales is also evident from Figure 1, ranging from immediate response tasks to developmental tasks. We are beginning to take advantage of *cross-domain integration.* By integrating different capabilities to model a whole task, we can bring more constraints to bear and reduce theoretical degrees of freedom. This kind of integration is evident in the close relation between NL-Soar and the series completion model (SC-Soar), and in work on taking

[5]We talk about individual attempts to use Soar as a cognitive theory as if they were separate systems, and give them unique names, such as XX-Soar. In fact, each such system is the same Soar, but with different knowledge. Such knowledge is often ad-hoc, which poses a problem of identifying the contributions of the theory, but this is a problem for all of psychology. See [11] for an approach to this problem with Soar.

instructions (NL-Soar, BI-Soar, and IR-Soar [11]).

In addition to achieving wide coverage, Soar must ultimately provide theories that press the scientific frontiers of specific domains. Soar is beginning to achieve such depth in diverse ways. The immediate behavior models provide *zero-parameter predictions* of performance times for simple reaction, choice reaction, transcription typing, and stimulus-response compatibility tasks [4, 13]. IR-Soar(Syl), the syllogistic reasoning model [15, 16], has recently been extended to give a detailed account of *individual differences*. The Browser-Soar, Data-Soar [24], Generic Design [14], and TOH-Soar [21] models provide *simulations of thinking aloud protocols* for their respective tasks. The Tower of Hanoi simulation is particularly interesting as it uses chunking to model strategy change. Figure 1 notes some of the different ways that Soar theories model behavior: accounting for individual differences (I), simulating protocols (P), providing qualitative coverage of phenomena (Q), demonstrating the sufficiency of Soar to accomplish the task (S), and making zero parameter predictions (Z).

Soar theories can be novel in their predictions and explanations, and/or synthetic, by incorporating existing theories and well understood mechanisms. For example, besides providing a new theory of individual differences, IR-Soar(Syl) is synthetic in its incorporation of the theory of mental models [6]. Browser-Soar is essentially a GOMS-like model, but introduces new control mechanisms. NL-Soar is synthetic in its assimilation of existing linguistic knowledge, but is novel in its method for automatically becoming recognitional. In contrast, the immediate behavior theory builds mostly on previous work in HCI [2, 3], while the model of the balance beam task [13] is completely unique to Soar.

FUTURE DIRECTIONS

We believe it is now fair to assess Soar as a genuine candidate for becoming a generally useful UTC. Soar's breadth is impressive and there are beginning to be places where it makes specific contributions at the frontier. However, the coverage remains quite patchy, which substantially diminishes the assessment of Soar as a coherent theory. Also, Soar efforts often evolve along idiosyncratic paths, viewed against the standard ways theory and data develop in psychology. For instance, NL-Soar's mechanism for the transition from deliberate to recognitional comprehension emerges well before NL-Soar confronts any standard linguistic or psycholinguistic phenomena. There are reasons for this priority, namely, the functional necessity of creating and growing the mass of **comprehend-word** operators. But it leaves NL-Soar deficient when evaluated by current criteria in either linguistics or psycholinguistics. More examples can be easily found. Thus, our net assessment is that Soar must still be viewed as nascent, and a substantial wait-and-see attitude is still justified.

Yet if one looks for encouraging results, they are also easily discerned. Integration of perception, cognition, and motor behavior is proceeding, although much still needs to be done. The little cluster of integration of language comprehension (NL-Soar), immediate reasoning (IR-Soar), instruction-taking (BI-Soar), and concept formation (SC-Soar) is an important harbinger.

Important changes are occurring that involve evolution of the Soar architecture itself. We have shifted to an underlying processing of states by continuous modification rather than discrete copying [10]. We continue to move toward using an *annotated models* representation [16] in all of our systems. The effort is already underway to extend Soar perception to model visual attention [27]. We have become convinced that the existing recognition memory is computationally unrealistic, so work is in progress to examine alternative formulations [25]. We are also exploring the possibility of implementing the Soar architecture using a neural network [17].

Acknowledgements

Many thanks to Thad Polk and Kathryn Swedlow for useful comments. This work was supported by the Information Sciences Division, Office of Naval Research, under contract N00014-86-K-0678, and by the National Science Foundation fellowship program in which Richard Lewis participates. The views expressed in this paper are those of the authors and do not necessarily reflect those of the supporting agencies. Reproduction in whole or in part is permitted for any purpose of the United States government. Approved for public release; distribution unlimited.

References

[1] Allen, J. **Natural Language Understanding**. Benjamin/Cummings, Menlo Park, CA, 1987.

[2] Card, S., Moran, T. P., and Newell, A. **The Psychology of Human-Computer Interaction**. Erlbaum, Hillsdale, NJ, 1983.

[3] John, B. E. *Contributions to Engineering Models of Human-Computer Engineering*. Department of Psychology, Carnegie Mellon University, May 1988. *Ph.D. Thesis*.

[4] John, B. E. *Extensions of GOMS analyses to expert performance requiring perception of dynamic visual and auditory information*. To appear in the *Proceedings of the Conference on Human Factors and Computing Systems*, April, 1990.

[5] John, B. E., Newell, A., and Card, S. *Browser-Soar: A GOMS-like model of a highly interactive task*. *Talk presented as the Human Computer Interaction Consortium Winter Workshop, San Diego, CA, February 12, 1990.*

[6] Johnson-Laird, P. **Mental Models**. Harvard, Cambridge, MA, 1983.

[7] Laird, J. E., Hucka, M., Yager, E. S., and Tuck, C. M. *Correcting and extending domain knowledge using outside guidance*. in: **Proceedings of the Seventh International Conference on Machine Learning**. 1990.

[8] Laird, J. E., Newell, A., and Rosenbloom, P. S. *Soar: An architecture for general intelligence*. **Artificial Intelligence**, vol. 33 (1987), pp. 1–64.

[9] Laird, J. E. and Rosenbloom, P. S. *Integrating planning, execution, and learning in Soar for external environments*. February 1990. *Artificial Intelligence Laboratory, The University of Michigan. An earlier version of this paper will appear in the 1990 AAAI Spring Symposium Workshop on Planning in Uncertain, Unpredictable, and Changing Environments*.

[10] Laird, J. E., Swedlow, K., Altmann, E., and Congdon, C. B. *Soar 5 User's Manual*. 1990. *In preparation*.

[11] Lewis, R. L., Newell, A., and Polk, T. A. *Toward a Soar theory of taking instructions for immediate reasoning tasks*. in: **Proceedings of the Eleventh Annual Conference of the Cognitive Science Society**. 1989, pp. 514–521.

[12] Miller, C. S. and Laird, J. E. *A Simple, Symbolic Model for Associative Learning and Retrieval*. March 1990. *Artificial Intelligence Laboratory, The University of Michigan. Unpublished*.

[13] Newell, A. **Unified Theories of Cognition**. Harvard University Press, Cambridge, Massachusetts, 1990. *In press*.

[14] Pirolli, P. *Generic design*. *Talk presented at the Seventh Soar Workshop, Carnegie Mellon University, Pittsburgh, PA, February 23-25, 1990.*

[15] Polk, T. A. and Newell, A. *Modeling human syllogistic reasoning in Soar*. in: **Proceedings of the Tenth Annual Conference of the Cognitive Science Society**. 1988, pp. 181–187.

[16] Polk, T. A., Newell, A., and Lewis, R. L. *Toward a unified theory of immediate reasoning in Soar*. in: **Proceedings of the Eleventh Annual Conference of the Cognitive Science Society**. 1989, pp. 506–513.

[17] Rosenbloom, P. S. *A symbolic goal-oriented perspective on connectionism and Soar*. in: **Connectionism in Perspective**, edited by R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, and L. Steels. Elsevier (North-Holland), Amsterdam, The Netherlands, 1989, pp. 245–263.

[18] Rosenbloom, P. S. and Aasman, J. *Knowledge Level and Inductive Uses of Chunking (EBL)*. February 1990. *Information Sciences Institute, University of Southern California. Unpublished*.

[19] Rosenbloom, P. S., Laird, J. E., and Newell, A. *The chunking of skill and knowledge*. in: **Working Models of Human Perception**, edited by H. Bouma and A. Elsendoorn. Academic Press, London, England, 1988, pp. 391–410.

[20] Rosenbloom, P. S., Laird, J. E., Newell, A., and McCarl, R. *A preliminary analysis of the Soar architecture as a basis for general intelligence*. in: **Proceedings of the Workshop on Foundations of Artificial Intelligence**, edited by D. Kirsh and C. Hewitt. MIT Press, Cambridge, Massachusetts, 1989.

[21] Ruiz, D. and Newell, A. *Tower-noticing triggers strategy-change in the Tower of Hanoi: A Soar model*. in: **Proceedings of the Eleventh Annual Conference of the Cognitive Science Society**. 1989, pp. 522–529.

[22] Simon, H. A. and Kotovsky, K. *Human acquisition of concepts for sequential patterns*. **Psychological Review**, vol. 70 (1963), pp. 534–546.

[23] Simon, T. *Modeling conceptual development: A computational account of conservation learning*. in: **Computational Approaches to Concept Formation**, edited by D. Fisher and M. Pazanni. Morgan Kaufmann Publishers, Inc., Los Altos, California, 1990. *In press*.

[24] Steier, D. and Adelson, B. *Data-Soar: Simulating a student learning about data structures*. *Talk presented at the Seventh Soar Workshop, Carnegie Mellon University, Pittsburgh, PA, February 23-25, 1990.*

[25] Tambe, M. and Rosenbloom, P. S. *Eliminating expensive chunks by restricting expressiveness*. in: **Proceedings of the Eleventh International Joint Conference on Artificial Intelligence**. 1989, pp. 731–737.

[26] Ward, B. *A Soar-Based Intelligent Tutor for Electrostatics*. May 1988. *Thesis proposal, Computer Science Department, Carnegie Mellon University*.

[27] Wiesmeyer, M. and Laird, J. E. *A Computer Model of 2D Visual Attention*. March 1990. *Artificial Intelligence Laboratory, The University of Michigan. Unpublished*.

[28] Young, R. M. and Whittington, J. *Using a knowledge analysis to predict conceptual errors in text-editor usage*. To appear in the *Proceedings of the Conference on Human Factors and Computing Systems*, April, 1990.