Table 5: Match time per token change (in ms).

Algorithm	EP	M&C	R1-Soar	NM
Rete	1.77	2.77	3.11	2.98
Treat (dynamic)	4.94	8.06	4.82	5.72

Table 6: Fraction of total match time spent in dynamic ordering in TREAT.

EP	M&C	R1-Soar	NM
41.29%	21.10%	25.96%	22.7%

is spent in ordering, a cost completely absent in RETE.

Static ordering in TREAT can be made as efficient as RETE. However, for each production, TREAT must create a complete join order for each condition element, while RETE needs to create only one. Given that there is an average of 9 condition elements per production, loading productions in TREAT can take an average of 9 times longer than in RETE, making it quite impractical.

5 Conclusions

We have presented empirical evidence, based on four Soar programs, showing that in most cases RETE is a better match algorithm than TREAT. This seems to contradict the results that Miranker reported recently [Miranker, 1987]. Part of this difference may be because of the metric used by Miranker. Counting the number of comparisons is dependent on the implementation (for example, the use of hashing can decrease the number of comparisons 10-fold), and may not actually reflect the intrinsic differences between the match algorithms. We have also identified specific features of Soar programs that contribute to this difference.

One of the major advantages of TREAT—not having to do any joins when either there is a condition element with no matching working memory elements, or when working memory elements are removed—is completely offset by RETE not having to do any joins when the opposite memory is empty. Since the fraction of times that joins are done in RETE and TREAT is about the same, the longer chains of joins generated in TREAT lead to a larger number of tokens being generated.

The fundamental difference between RETE and TREAT—saving intermediate relations versus recomputing them—is the second main reason for TREAT's poorer performance. The matching of static structures involves recomputation of joins in TREAT, while RETE is able to match such structures just once. RETE is also better at matching monotonic structures.

Finally, we noted that combinatorial joins tend to make TREAT perform worse than RETE. In addition, TREAT tends to have more combinatorial joins than does RETE.

We also showed that both dynamic and static ordering in TREAT do not perform as well as RETE in practice. The actual process of ordering the condition elements dynamically takes a significant amount of time, making the system run very slowly. Static ordering is not good either because the time to load in productions can go up approximately 9 fold (i.e. by the average number of condition elements in a production).

While RETE seems to be better than TREAT in most cases, there are some situations under which TREAT is comparable to RETE and may even be better. This leads to the possibility of having hybrid match algorithms (TRETE?) which allow some productions to be matched using the RETE algorithm and some to be matched using the TREAT algorithm. An even closer enmeshing of the two algorithms is possible in which parts of one production are matched using the RETE algorithm while other parts are matched using the TREAT algorithm. Deciding which parts of the productions should be matched by which algorithm is not an easy task, and is a possible direction for future research.

References

[Brownston *et al.*, 1985] Lee Brownston, Robert Farrel, Elaine Kant, and Nancy Martin. *Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming.* Addison-Wesley, 1985.

[Forgy, 1982] Charles L. Forgy. Rete : A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, 19:17–37, September 1982.

[Gupta, 1987] Anoop Gupta. *Parallelism in Production Systems.* Morgan Kaufmann Publishers, Inc., 1987.

[Laird, 1986] John E. Laird. *Soar User's Manual (Version 4).* Technical Report ISL-15, Xerox Palo Alto Research Center, 1986.

[Laird *et al.*, 1987] John E. Laird, Allen Newell, and Paul S. Rosenbloom. Soar : An architecture for general intelligence. *Artificial Intelligence*, 33:1–64, September 1987.

[Miranker, 1984] Daniel P. Miranker. Performance estimates for the DADO machine : A comparison of TREAT and Rete. In *Fifth Generation Computer Systems*, ICOT, Tokyo, 1984.

[Miranker, 1987] Daniel P. Miranker. TREAT : A better match algorithm for AI production systems. In *AAAI-87 Proceedings*, American Association for Artificial Intelligence, July 1987.

[Nayak *et al.*] Pandurang Nayak, Anoop Gupta, and Paul S. Rosenbloom. Comparison of the Rete and Treat production matchers for Soar. In preparation.

[Rosenbloom *et al.*, 1985] Paul S. Rosenbloom, John E. Laird, John McDermott, Allen Newell, and Edmund Orciuch. R1-Soar : An experiment in knowledge-intensive programming in a problem-solving architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence 7*, :561–569, 1985.

[Scales, 1986] Daniel J. Scales. *Efficient Matching Algorithms for the Soar/OPS5 Production System.* Technical Report KSL 86-47, Knowledge Systems Laboratory, June 1986.

[Steier, 1987] David Steier. CYPRESS-Soar : A case study in search and learning in algorithm design. In *IJCAI-87 Proceedings*, International Joint Conference on Artificial Intelligence, 1987.

[Washington and Rosenbloom] Richard M. Washington and Paul S. Rosenbloom. Neomycin-Soar : Applying search and learning to diagnosis. In preparation.

Modeling Human Syllogistic Reasoning in Soar

T. A. Polk and A. Newell, Carnegie Mellon University

Soar is an architecture for general intelligence, which has been shown to be capable of supporting a wide variety of intelligent behavior involving problem-solving, learning, designing, planning, etc. (Laird, Newell & Rosenbloom, 1987, Steier, et. al., 1987). Soar has also been put forth as a unified theory of human cognition (Newell, 1987). We provide support for this by presenting a theory of syllogistic reasoning based on Soar and some assumptions about subjects' knowledge and representation. The resulting theory (and system, Syl-Soar/S88) is plausible in its details and accounts for existing data quite well.

The Task

Syllogisms are reasoning tasks consisting of two premises and a conclusion (Figure 1, left). Each premise relates two sets of objects (x and y) in one of four ways (Figure 1, middle), and they refer to a common set (*bowlers*). A conclusion states a relation between the two sets of objects that are not common (*archers* and *canoeists*) or that no valid conclusion exists. The three terms x,y,z can occur in four different arrangements, called *figures* (Figure 1, right), producing 64 distinct syllogisms.

Premise 1: No archers are bowlers	A: All x are y	| P1 xy | P1 yx |
Premise 2: Some bowlers are canoeists	I: Some x are y	| P2 yz | P2 yz |
Conclusion: Some canoeists are not archers	E: No x are y	| P1 xy | P1 yx |
	O: Some x are not y	| P2 zy | P2 zy |

Figure 1: Syllogism task.

Syllogisms have been much studied (see Johnson-Laird 1983 for review). The essential problem has been to understand why some syllogisms are so hard while others are so easy. However, the area is also useful as a testbed for cognitive theories.

The Soar Theory of Syllogisms

The Soar architecture has the following features:

1. **Problem spaces.** All tasks, routine or difficult, are formulated as search in problem spaces. Behavior is always occurring in some problem space.
2. **Recognition memory.** All long-term knowledge is held in an associative recognition memory, realized as a production system.
3. **Decision cycle.** All available knowledge is accumulated about the acceptability and desirability of problem spaces, states and operators for the current total context, and the best alternative is chosen among those that are acceptable.
4. **Impasse driven subgoals.** Incomplete or conflicting knowledge at a decision cycle

produces an impasse. The architecture creates a subgoal to resolve the impasse. Cascaded impasses create a subgoal hierarchy.

5. **Chunking.** The experience in resolving impasses continually becomes new knowledge in recognition memory, by means of constructed productions (chunks).

6. **Annotated models.** States are represented as annotated models (to model human cognition).

Figure 2 indicates the structure of the system: the collection of problem spaces (triangles) with operators and states. Subspaces arise from impasses, usually reflecting the need to implement operators or satisfy operator preconditions. The task data structures occur in working memory and are continually viewed by the recognition memory, which contains all task-implementation and search-control knowledge. Relevant knowledge accumulates from this memory, permitting steps to be taken in the current space or, upon impasses, creating subgoals to be solved in subspaces, etc. The micromechanics are beneath the level of detail of this paper, but drive the entire system, including learning.

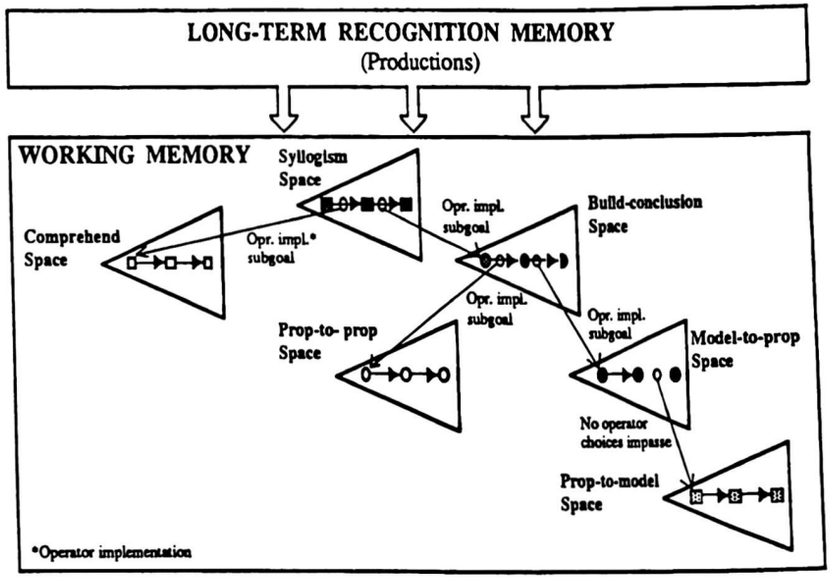

Figure 2: The structure of Soar.

A key assumption, developed strongly by Johnson-Laird (1983), is that humans represent the situations presented in syllogisms as *models*. A *pure model* is a representation that satisfies the *structure correspondence condition*: specified parts and relations of the representation data structure correspond to parts and relations of the situation, without completeness (see also Levesque, 1986). A pure model admits highly efficient *match-like* processing, but is limited in its representational power. An *annotated model* is a representation that makes principled exceptions to a pure model, which increase its representational power, while preserving essential match-like processing. An *annotation* attaches to a data-structure part, asserting a variant interpretation for the part (e.g., **not** asserts that the part is *not* to be found in the situation where the correspondence mapping would otherwise locate it). The annotations used for syllogisms are **not**, **optional**, **many**, **target** and **source**. Annotations can quantify, but are local and do not admit unbounded processing. Figure 3 (left) indicates the models that might be built from two premises. The line through the bowling pin indicates a **not** annotation.

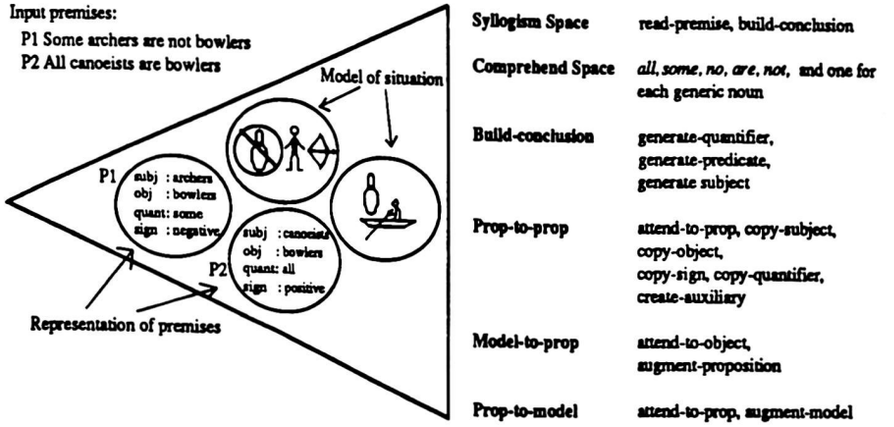

Figure 3: Annotated models, problem spaces and operators for syllogisms

Reasoning occurs by generating models to correspond to situations, inspecting the models for the properties of the situation, and forming new propositions to assert the result. Inspection is a power of the recognition memory (production match). Since models are limited, some situations can be represented only by a disjunctive set of models; reasoning then includes generating sets of models to test conjunctive properties. Reasoning with multiple models occurs in humans and has been central to model-based theories of syllogistic reasoning (Johnson-Laird, 1983, Inder, 1986), but the present theory includes only reasoning with a single model.

Six problem spaces are used in syllogistic reasoning (Figure 3 lists them, with operators, Figure 2 shows how they link together). **Comprehend**, **Syllogism** and **Build-conclusion** form the top-level path between the presented premises and the response. The knowledge to form them comes from the definition of the task, plus general skills in reading and writing. **Comprehend** is an expectation-based scheme that associates both syntactic and semantic knowledge with individual words. It constructs an initial (possibly incomplete) model; it also leaves as a byproduct a model of each premise as a proposition, with parts *subject*, *object* and *sign* (the predicate), and *quantifier*. **Prop-to-prop**, **Model-to-prop**, and **Prop-to-model** have operators required to manipulate models of situations and models of propositions, as well as attention operators to instantiate the manipulations.

The Behavior of the System

Figure 4 illustrates the system's behavior. (1) It starts in **Syllogism** and applies read-premise, implemented in **Comprehend**, to the first and then the second premise. (2) This results in an initial model, plus the two internal propositions. This encoding only extracts information about the subject of the premise. (3) Since the overall task is to produce a conclusion, build-conclusion is applied. Its space (**Build-conclusion**) puts together legal propositions. The task decomposes into discovering the subject, predicate and quantifier of the conclusion. Task knowledge permits determining some parts without other parts being specified. Incomplete or incorrect knowledge leads to composing invalid conclusions.

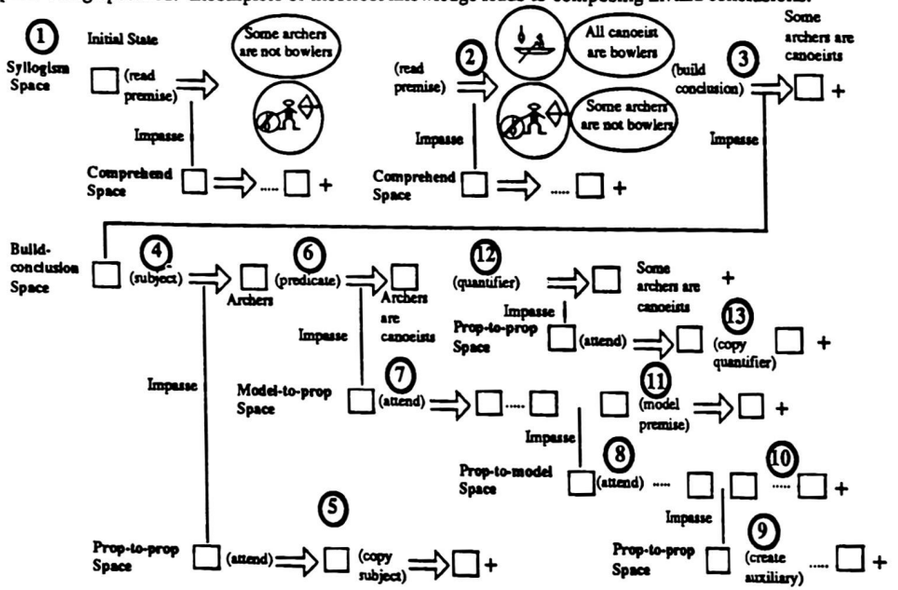

Figure 4: Behavior on *Some archers are not bowlers, All canoeists are bowlers.*

(4) Generating the subject is tried first, which uses **Prop-to-prop** because the propositions, not the model, distinguish between subjects and objects. (5) Attend-to-prop selects the first proposition and copy-subject creates the subject of the conclusion (*archers*). (6) Next, generate-predicate is selected, which uses **Model-to-prop**, because the propositions contain no useful information about the predicate. (7) The attend-to-object operator applies, but no others, because the model is incomplete. This leads to augmenting the model, using **Prop-to-model**. (8) Attend-to-prop selects premises to extract more information, but neither premise yields anything. (9) Create-auxiliary produces a new proposition in **Prop-to-prop**. It attends to the second premise and applies operators which convert it, creating the new premise *All bowlers are canoeists*. (10) This allows solving in **Prop-to-model** to resume, by focusing attention on this new proposition and using it to augment the model. (11) The model now suggests a

predicate, so solving is able to continue in **Model-to-prop** to obtain the predicate for the conclusion (*are canoeists*). (12) All that remains in **Build-conclusion** is to generate the quantifier. The model does not represent quantifiers, so **Prop-to-prop** is used again. (13) It attends to the first premise and copies its quantifier (*some*), finally obtaining, *Some archers are canoeists*. This is incorrect, but many humans fail this syllogism as well. Correctness depends on knowledge being available at many local choices.

Human Data and Soar Performance

Figure 5 presents data from (Johnson-Laird & Bara, 1984) by 20 University of Milan students on all 64 syllogisms (with unlimited time) and also the responses by Soar. The four sections of the chart correspond to the four figures (Figure 1,right). Each row corresponds to one of the 9 legal responses. The top number in each cell indicates the number of subjects giving that response to a particular syllogism. *Some archers are not bowlers* and *All canoeists are bowlers* (Figure 4) is abbreviated Oxy,Azy, and occurs in the lower left quadrant, where we see that 8 subjects responded Ixz (*Some archers are canoeists*), 7 responded Oxz (*Some archers are not canoeists*), 3 responded NVC (*no valid conclusion*) and 2 subjects gave illegal responses. Valid responses are shaded (Oxz for 7/20 correct). Only 38% percent of all responses were correct and 7 syllogisms were solved by no one.

Individual humans behave differently from each other and from themselves over time, due to learning and other factors. The data of Figure 5 are a composite, as shown by multiple responses. A family of Soar systems is required to correspond to this human variation. We varied the theory along 3 dimensions: (1) whether auxiliary propositions are created, as in our example (2 choices); (2) how premises augment objects with **not** annotations (3 choices); and (3) whether premises about *some x* augment objects about *x* (2 choices). The first dimension is one of reasoning power, the other two involve the semantics of interpreting premises. These dimensions form a family of 12 variants.

This small family accounts for 980 out of 1154 (85%) observed legal responses (126/1280 responses were illegal and not recorded) by covering 131 out of the 193 cells (68%) that contain 1 or more responses (all cells with more than 6 subjects are predicted with one exception [Oyx, Ayz = Izx]). Only one response is predicted that is not given by any subject [Oyx, Ayz = Ozx]. Frequencies were assigned to the different members of the family to produce the fit shown in parentheses in Figure 5 (15/20 subjects were assumed in the family since 23% of responses, many illegal, were unpredicted). No simple measure of fit is available, but the correlation between subjects and systems is .87.

The theory produces the classical effects, such as the *atmosphere effect* (Woodworth & Sells, 1935), the *conversion hypothesis* (Chapman & Chapman, 1959) and the *figural effect* (Johnson-Laird, 1983). Space does not permit showing the analysis, but they need only be traced out in Figure 5. The atmosphere and figural effects arise because the syntactic form of the premises serves as search control in the construction of the conclusion. The conversion effect arises when this search control is insufficient and a new proposition is created.

According to the theory, there are three main sources of difficulty: (1) making unwarranted assumptions about the premises; (2) failing to consider all the implicit ramifications of the premises; and

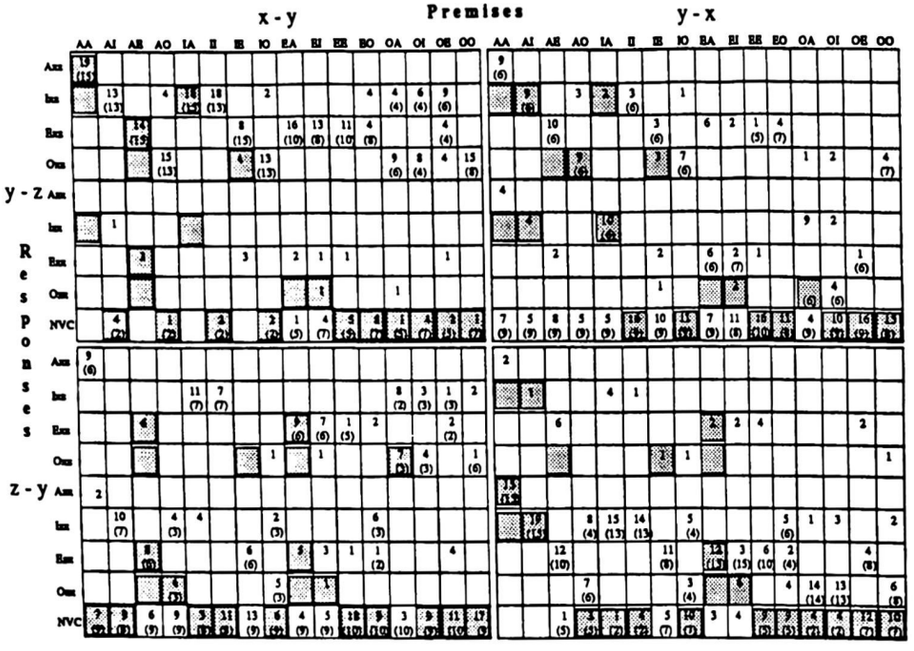

Figure 5: Data (from Johnson-Laird & Bara, 1984) and Soar predictions in ( ).

(3) failing to consider all the possible conclusions based on a (possibly correct) model. Syllogisms are difficult to the extent they present opportunities for these processing difficulties (e.g., have implicit ramifications relevant to the conclusions). This predicts that better subjects will extract more information from the premises without making unwarranted assumptions or that they will search for conclusions more extensively.

We designed a family of systems based on 10 parameters, which includes the current 3-parameter family, with the values (mostly binary) of each parameter being independently ordered by validity (so that better values correspond to more powerful and correct ways of building models). When all parameters take on their optimal values, perfect performance should occur. Better solvers should occur within this space with interpretable parameter settings. To test this, we analyzed another set of 20 subjects 58% of whose responses were correct (Johnson-Laird & Steedman, 1978). We implemented a small sub-family (24 variants including the 12) that covered 87% of the responses and 67% of the cells; it did however predict 11 responses not given by any subjects. The parameter settings of the modal system for the new distribution are better (higher in validity ordering) than those of the old distribution's modal system on 3 parameters and the same on the other 7.

The explanatory power of this theory appears better than existing theories. Their predictions are less accurate in that they predict a large number of responses that were not observed in any subjects and they do not make strong frequency predictions. Most theories only explain highly aggregate data. However, the data used here (Figure 5) is still aggregated over subjects, and nothing has yet been done with timing and protocol data. So ample opportunity remains to challenge and improve the present theory.

This theory has much to recommend it generally. It predicts flexible activity, e.g., going back to the premises to try to extract more information. Its spaces (especially executive ones) are substantially less arbitrary than prior simulations (e.g., **Comprehend** embodies a theory of elementary language comprehension). Although not reported on here, the present theory involves a theory of learning, which is an essential part of any general account of human cognitive behavior. These attributes and others arise primarily from this theory of syllogism being embedded in Soar as a unified theory of cognition.

Acknowledgements

We thank the members of the Soar project for support and criticism, especially Rick Lewis who is working on **Comprehend**; also Norma Pribadi for making the beautiful figures and Phil Johnson-Laird for comments on this theory. This research was supported by the Information Sciences Division of the Office of Naval Research under Contract N00014-86-K-0678 and also by the NSF under the Engineering Research Center Program, Contract CDR-8522616. 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

Chapman, I. J., & Chapman, J. P. (1959). Atmosphere effects re-examined. *Journal of Experimental Psychology, 58*, 220-226.

Inder, R. (1986). Modeling syllogistic reasoning using simple mental models. In Cohn, A. G., & Thomas, J. R. (Eds.), *Artificial Intelligence and its Applications*. New York: Wiley.

Johnson-Laird, P. (1983). *Mental Models*. Cambridge, MA: Harvard.

Johnson-Laird, P. N., & Bara, B. G. (1984). Syllogistic inference. *Cognition, 16*, 1-61.

Johnson-Laird, P. N., & Steedman, M. (1978). The psychology of syllogisms. *Cognitive Psychology, 10*, 64-99.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence, 33*, 1-64.

Levesque, H. J. (1986). Making believers out of computers. *Artificial Intelligence, 30*, 81-108.

Newell, A. (1987). Unified Theories of Cognition. The William James Lectures. Harvard University, Spring 1987. (Available in vidocassette from Harvard Psychology Department).

Steier, D. E., Laird, J. E., Newell, A., Rosenbloom, P. S., Flynn, R. A., Golding, A., Polk, T. A., Shivers, O. G., Unruh, A. & Yost, G. R. (1987). Varieties of Learning in Soar: 1987. In *Proceedings of the Fourth International Workshop on Machine Learning*. Los Altos, CA: Morgan Kaufman.

Woodworth, R. S., & Sells, S. B. (1935). An atmosphere effect in formal syllogistic reasoning. *Journal of Experimental Psychology, 18*, 451-460.