The Acquisition of a Procedure Schema from Text and Experiences

Franz Schmalhofer & Bidjan Tschaitschian
German Research Center for Artificial Intelligence
University Bldg. 57, Erwin-Schroedinger-Strasse, Postfach 2080
W-6750 Kaiserslautern, Germany
e-mail: (schmalho, tschaits)@dfki.uni-kl.de

Abstract

Learning from problem solving episodes has previously been modeled in two different ways: Case-based planners (Hammond, 1989) acquire additional knowledge by storing new cases (i.e. the specific plans for different problems); search-based systems like SOAR or PRODIGY learn by chunking the result of a search process (Rosenbloom et. al., 1991; Minton et. al., 1989) and by forming macro-operators (Korf, 1985). This paper proposes comprehension-based learning as a third possibility: From specific problem solving experiences (cases) and a related problem description (text) some coarse-grained abstract representation is constructed, that may initially be inconsistent and redundant. By wholistic integration processes a coherent and consistent procedure schema is subsequently formed. Such a procedure schema can be reused for obtaining solutions to problems which are quite different at the concrete level, but have been comprehended to share abstract commonalties. The acquisition of such a procedure schema is exemplified for various solutions to different Tower of Hanoi problems (3-, 4-, and 5-disks). The utilization of these schemata is then discussed for the 4-disk problem and respective experimental data are reported.

Introduction

Procedure schemata (Rumelhart et al., 1986) or scripts play essential roles in the various performance theories of perception, planning or story understanding. How such schemata are learned from different sources of information is, however, less well understood.

The search of a problem space for resolving impasses (Rosenbloom et. al., 1991) and constructing macro-operators (Korf, 1985) or compiled knowledge (Anderson, 1987) and the formation of domain-specific search control rules from successful problem solving experiences (Minton et al. 1989) can be seen as examples for building stable structures on the basis of experience. Chunking, macro-operators, compiled knowledge and search control rules are compositions or generalizations of the result of search processes or problem solving episodes.

In this paper, we propose, however, that a procedure schema should be thought of as a more coarse-grained representation for quite different specific experiences. Such procedure schemata are represented in terms of abstract situation knowledge and result from a comprehension process (Kintsch, 1988). Procedure schemata are thus quite different from the composition or generalization of specific experiences as well as from the storage of cases (Hammond, 1989).

Michalski & Kodratoff (1990) have recently pointed out that abstraction must be distinguished from generalization: Whereas generalization transforms a description only along a set-superset dimension in a single description space (or language), abstraction involves a change in the representation space (or language) and transforms a description along the level of detail dimension resulting in a more coarse-grained description.

The current paper proposes a learning component within the framework of the construction-integration model (Kintsch, 1988) by which a procedure schema can be acquired from text (i.e. the problem description) and problem solving experiences. We do not describe how some specific problem is initially solved. Such a model has already been established by Mannes & Kintsch (1991). Instead we want to explain what is being learned while or after a problem is solved.

More specifically, we want to evaluate whether the comprehension strategies of the construction-integration model are sufficient for acquiring an abstract procedure schema from text and experiences, which will subsequently facilitate the solution of structurally different problems. Since the Tower of Hanoi task is typically used as a testbed for evaluation (Korf, 1985; Ruiz & Newell, 1989), it is also used in the current research. In this testbed the proposed learning mechanism will be compared to case-based and search-based approaches and new experimental data will be reported.

This research was supported by grant Schm 648/1-5 from the Deutsche Forschungsgemeinschaft.

The acquisition of abstract situation knowledge

The construction-integration model, which was originally developed as a text comprehension theory (Kintsch, 1988), describes how a text base and a situation model are formed when a text is studied.

Situation model. How strong a text base and a situation model become depends upon the subject's learning goal (Schmalhofer & Glavanov, 1986) and her prior domain knowledge (Kintsch et. al., 1990). While the formation of a text base requires the reading of a text, a situation model can equally well be built from other information sources, like real world experiences (or examples). Schmalhofer, Boschert & Kühn (1990) have shown that there are two different routes to building a situation model: When studying a text, a text base is built as an intermediate representation. When studying examples, templates are built as an intermediate representation. Unlike the situation model, which is an abstract representation of a domain, text base and templates are representations that are related to a specific learning material.

Other domain knowledge. A situation model is, however, not the only conceptual model that represents domain knowledge. For example, when solving a word arithmetic problem involving the exchange of marbles between children, an abstract model consisting of mathematics equations (Nathan, Kintsch, & Young, 1992) may become related to the more concrete model about marble exchanges. The two different levels of domain knowledge are called system and situation models. Reference knowledge is needed for relating the models to one another. Domain knowledge thus consists of system and situation models and reference knowledge.

The formation of task knowledge. A procedure schema, on the other hand, represents abstract task knowledge, which is expressed in terms of situation knowledge. Task knowledge describes how a certain goal can be achieved, by performing certain actions. Figure 1 gives an overview of how a procedure schema can be acquired from text and experiences and which knowledge is being used for its formation. During the reading of the problem description (i.e. a text), partial system and situation models will be formed as domain knowledge in addition to a text base.

When a problem is successfully solved, problem solving experience is first generated as a sequence of concrete actions. From these problem solving experiences and related domain knowledge a procedure schema can be acquired. In the PABS-model (Schmalhofer et. al., in press), the formation of a procedure schema is described by five phases (I - V), that are grouped into construction and integration episodes.

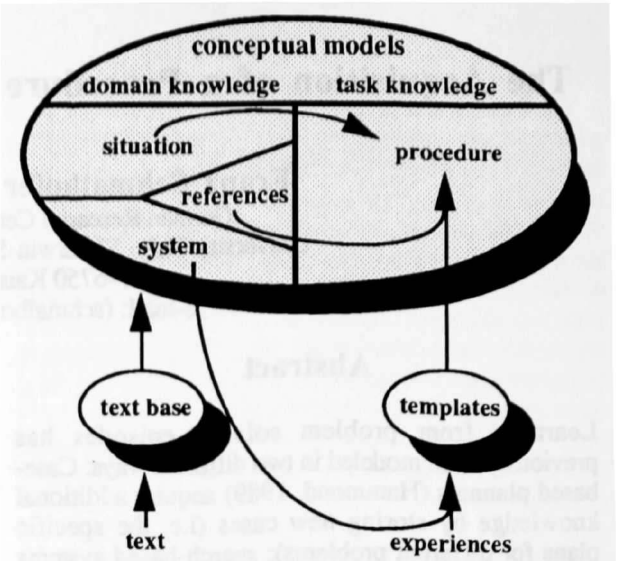

Figure 1. The formation of domain and task knowledge

The Construction Episode. The construction episode consists of three phases. For the given sequence of individual actions robust production rules construct: I) state descriptions (initial, intermediate and final) induced by the actions, II) abstract descriptions of the concrete states, and III) abstract operators which describe various possible transitions among the abstract states (i.e. descriptions in terms of situational knowledge). What is being constructed depends upon the learner's prior domain knowledge. The prior domain knowledge, which is represented in the long-term memory, comprises: a) a system model about a specific task environment, b) a situation model about an application domain, and c) the references between the system model and the situation model (see Figure 1).

The Integration Episode. The representation resulting from this construction process may be redundant, incoherent and even contradictory. For example different abstract operators may have been constructed to describe the same transition between two states. The integration process transforms this initial representation into a coherent form and an abstract procedure schema is obtained. Previously, a spreading activation mechanism has been used for performing this integration process (Mannes & Kintsch, 1991). In order to obtain a syntactically sound procedure schema, which indeed describes a valid transition from the initial to the final state a symbolic component and a marker-passing algorithm (Hendler, 1989) are required.

The integration process consists of two phases: In phase IV a consistent path of abstract operations is established by a dependency analysis. In phase V an additional integration is performed by generalizing the path of abstract operations into the final procedure schema. This schema is added to the task knowledge and may later on be used for solving similar and different but related problems.

The five phases have been implemented in the PABS-simulation, which stands for Plan or Program Abstraction. Its sufficiency has already been demonstrated for the formation of an algorithm schema from a concrete program text. In addition, a formal characterization in terms of concrete and abstract STRIPS-worlds and state- and sequence abstraction mappings has been presented (Schmalhofer et. al., in press).

The Tower of Hanoi Task

Problem description. The Tower of Hanoi task (Simon, 1975) requires that a *tower* of n disks which are graded in size is transferred from a start peg (say peg A) to some goal peg (say peg C). At the outset, all the disks are arranged *pyramidically* on peg A. An additional peg (say B) may be used as an auxiliary location. The following rules must be observed: At any time only a single disk may be moved and a *larger disk* must never be put on top of a smaller disk.

Representation of task environment (system model). A problem solver will represent the individual disks in the concrete description language as part of a system model. A problem state is represented by a list. The initial states of the 3-, 4- , and 5- disk problems are thus represented by *[[1 2 3] [] []], [[1 2 3 4] [] []]* and *[[1 2 3 4 5] [] []]* respectively: *1* refers to the smallest disk, *2* to the second smallest disk and so on.

Situation representation. The reader of the problem description will not only represent the individual disks (as part of a system model), but coarse-grained knowledge construction processes could also bring abstract background knowledge into play. Such knowledge may denote perceptual chunks, like a pyramidically arranged set of disks and associate them to terms that are semantically related to respective parts of the problem description (i.e. *tower*, *upper_tower*, *lower_tower*, *large_disk*, and *empty*)

In the abstract description language a pyramid consisting of the smallest to the i-th smallest disk (1<i<n) can thus be denoted as an *upper_tower*. The pyramid consisting of the largest disk to the j-th largest disk (with 1<j<n) is denoted as a *lower_tower*. A pyramid consisting of all the disks of a problem is denoted as *tower*. Because of their significance, certain disks which are distinctively large may also be represented as part of the situation model. All disks k with 2<k≤n are simply denoted as *large_disk*. The concept *empty* is in addition used for describing a peg that is empty.

For all different Tower of Hanoi problems (e.g. the 3-, 4- and 5-disk problem), a situation model would thus consist of the terms: *tower*, *upper_tower*, *lower_tower*, *large_disk*, *empty*. At the situational level, states and moves are expressed in these terms. The initial states of different problems like the 3-, 4-, and 5-disk problems would consequently all be represented by *[tower empty empty]*.

Note that not all concrete states can be represented in the abstract description language. Because the abstract language is more coarse-grained and less expressive, problem states like *[[1 2 3 5] [4] []]* cannot be represented at the abstract level. Because disk *4* is missing *[1 2 3 5]* is neither a *tower* nor an *upper_tower* nor a *lower_tower*. For the same reason different concrete problem states may have an identical abstract description. For example, *[[3 4 5] [1 2] []]* and *[[4 5] [1 2 3] []]* are both described as *[lower_tower upper_tower empty]*.

Procedure Schemata from Different Tower of Hanoi Problems

Acquisition. Consider for example some specific (optimal or non-optimal) solution path to the 5-disk problem, where concrete state descriptions are obtained for the visited states in the first construction phase (phase I). The result of this phase is shown in section I of Figure 2.

In phase II the abstract description language is applied to characterize the different concrete states, whenever possible. The arrows in section II of Figure 2 show the resulting associations between concrete and abstract states. They are part of the reference knowledge by which system and situation models can be related. The empty rectangles indicate states, which cannot be described in abstract terms.

In the third phase (section III) operators are then constructed for describing the transitions between abstract states. There are two different ways for constructing an abstract operator. An operator may be (a) constructed purely at the abstract level or (b) the concrete level may be taken into consideration to obtain a more operational description:

a) The STRIPS descriptions of two consecutive abstract states are used to induce a respective abstract STRIPS operator. The transitions from *[[3 4 5] [1 2] []]* to *[[4 5] [1 2] [3]]* and from *[[4 5] [1 2 3] []]* to *[[4 5] [1 2] [3]]* are thus both described by *[lower_tower upper_tower empty]* to *[lower_tower upper_tower large_disk]* and the same STRIPS-operator would consequently be induced.

b) The abstract description language is applied to characterize the dynamics of the transition at the concrete level between two such states. For the two transitions (see above) two different operators can now be constructed: Since a *large_disk* is moved from A to C, the first transition is described as *abstract_move(A,C)*. In the second transition the largest disk of the *upper_tower* on B is pulled out from underneath and moved to the *empty* location C. This transition is therefore described as *complicated_split(B,C)*. This construction process maintains more properties of the concrete level, but in terms of an abstract description language. From an optimal solution of the 5-disk problem, the following sequence of abstract operators is constructed: *split_tower(A,B), abstract_move(A,C), join_upper_tower(B,C), abstract_move(A,B), join_upper_tower(C,B), abstract_move(A,C), split_upper_tower(B,A), abstract_move(B,C), split_upper_tower(A,B), abstract_move(A,C), join_tower(B,C).*

In the following integration phase (section IV) a type of symbolic marker passer is additionally used for analyzing consistency and completeness. Thereby a wholistic integration between concrete and abstract descriptions is obtained. For some problem solutions, it turned out that the induced abstract STRIPS operators were inconsistent with the concrete level (see example in phase III) and are thus

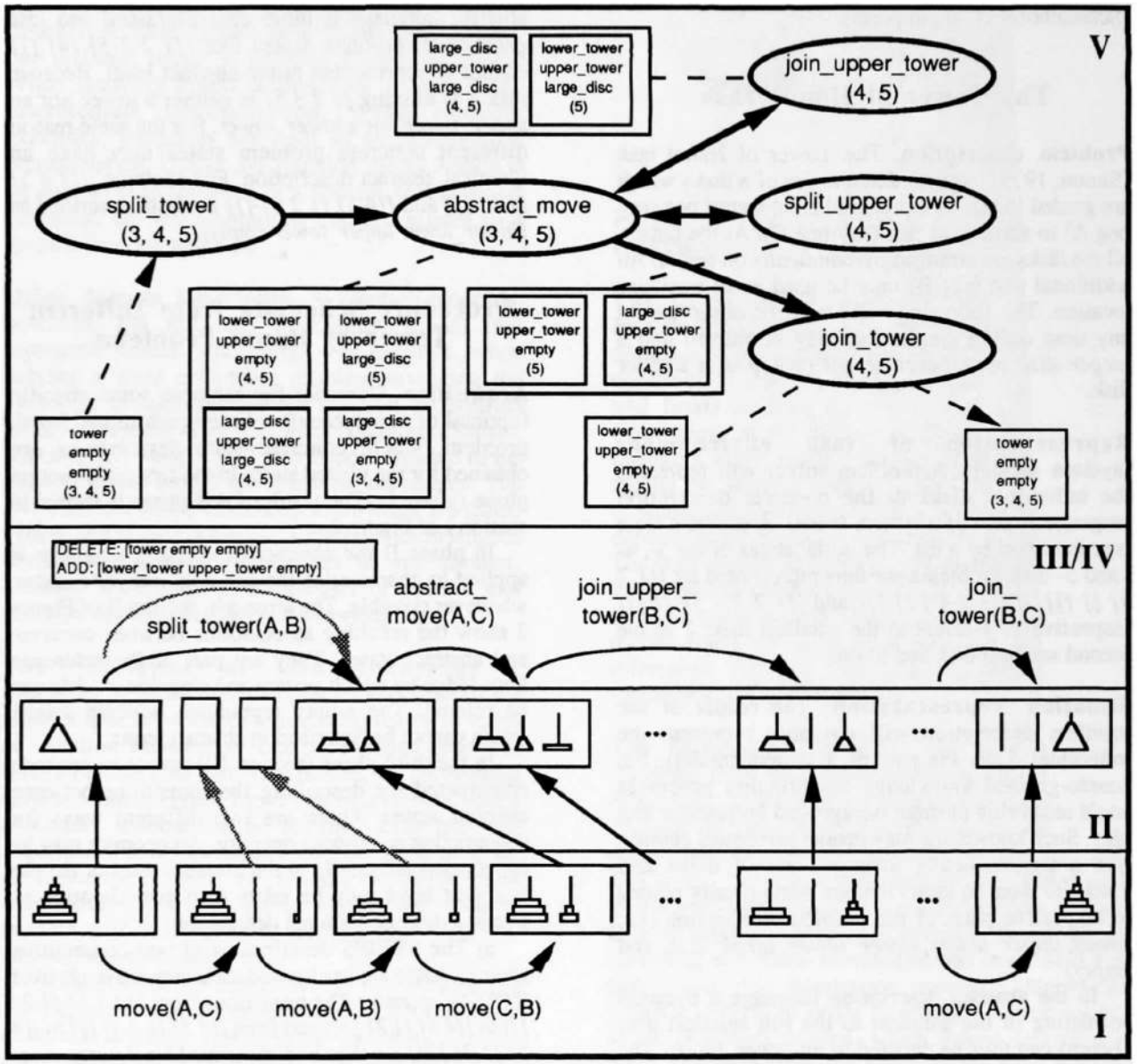

Figure 2. The construction (I, II, III) and integration phases (IV, V) for acquiring a procedure schema from 3-, 4-, and 5-disk problems

eliminated in this phase. We will therefore restrict our further discussion to the abstract operators, which were formed by the construction process b) in phase III.

In the V-th phase (also an integration phase) the sequence of abstract operators resulting from phase IV, is transformed into the final procedure schema, which is a more compact and structurally richer representation. The operators are generalized by variabilization, e.g. from *abstract_move(A,C)* to *abstract_move(X,Y)*, yielding five different generalizations (see section V of Figure 2). Two generalizations are connected whenever one operation was directly followed by the other in its phase IV representation. The abstract preconditions are also maintained in the V-th phase, resulting in the procedure schema shown in section V of Figure 2. The numbers (3, 4, 5) show which components of the procedure schema are formed from the optimal solution of the 3-, 4-, and 5-disk problems.

Reuse for similar problems. Identical problems may be solved by using the processing product of phase IV, which decomposes the problem space into more manageable segments and often yields a performance improvement in relation to the first problem solution. In the schema formation several detours of the original solutions were abstracted away.

Similar problems are problems which have the same number of disks, but the location of the tower in the initial state was changed (e.g. from A to B). The procedure schema and its stepwise refinements (starting from phase V) will be used for solving these problems and yield similar improvements.

Reuse for different problems. A different problem occurs for example when the number of disks has been changed from 5 to 4 disks. These problems are quite different in their solutions at the concrete level: Problems with an odd number of disks require the first move to be opposite to the first move in problems with an even number of disks (Simon, 1975). Also, the 4-disk problem requires fewer moves than the 5-disk problem. Because of its abstract representation the procedure schema obtained from the 5-disk problem is, however, equally well suited for solving a 4-disk problem as the schema acquired from the 4-disk problem. The schema acquired from the 3-disk problem contains only certain parts (see section V of Figure 2) and is therefore less suited for solving the 4-disk problem.

Experiment

In three different conditions (30 subjects each), subjects had to solve two 3-disk, two 4-disk, or two 5 disk-problems in a row and their number of moves was recorded. The two consecutive problems were similar. The first time the tower of disks was on peg A in the initial state. The second time it was located on peg B. Both times, it had to be transferred to peg C.

Thereafter, the knowledge which they had acquired from these problem solving episodes was tested. In order to obtain a more complete assessment of their knowledge with respect to the 4-disk problem, which served as the criterion task, all subjects were presented with the 81 different states of the four disk problem, one at a time. Rather than completely solving the Tower of Hanoi problem, the subject had to select the best move for each of the 81 different problem states, which were randomly divided in three sets of 27 states. For selecting a move, subjects were allowed 15, 30 or 45 seconds. The allowed processing time and the three sets of 27 states were counterbalanced by a Latin-Square design.

Results. Table 1 shows the average number of moves for two similar problems with 3, 4 or 5 disks. Clearly, fewer moves were required when solving the problem for the second time as compared to the first time. Table 2 shows the performance in the 4-disk criterion task. The percentage of correct moves is shown as a function of the allowed processing time and the three different training tasks.

Table 1. Average number of moves in three different training tasks

Mean number of moves	3-disk problem	4-disk problem	5-disk problem
first problem	11.1	30.9	66.0
similar problem	9.1	21.4	55.0

Table 2. Percentage of optimal moves in the 4-disk criterion task as a function of pretraining and processing time

Pretraining / Time	3-disks	4-disks	5-disks
15-sec.	63	68	70
30-sec.	69	75	77
45-sec.	69	76	81

Throughout, the subjects with the 5-disk problem solving experience performed better than the subjects with the 4- or 3-disk problem solving experiences. Thus, the subjects did not solely store the individual moves from the specific Tower of Hanoi problem nor solely compiled solution knowledge from it. Instead, subjects must have also utilized background knowledge which allowed them to form a more abstract representation from the problem solving

episodes, that could be efficiently transferred from the 5-disk training task to the 4-disk test task. Since more elaborate abstractions can be acquired from the 5-disk problem, the subjects with this training performed better in the criterion task than the subjects with the 4- or 3-disk training.

Conclusions

For the Tower of Hanoi tasks we can now compare the acquisition and utilization of knowledge in case-based, search-based, and comprehension-based systems.

A case-based planner stores the specific experiences and utilizes these experiences by adapting them to new problems. The specific solutions of 5-disk (odd number of disks) and 4-disk problems (even number of disks) are quite different (Simon, 1975). A case-based system with a 5-disk training should therefore solve a 4-disk task relatively poorly, which clearly contradicts the experimental findings of Table 2.

Search-based systems acquire new knowledge by searching a problem space and forming macro-operators or chunks. Since the Tower of Hanoi task results in an ill-suited problem decomposition, macro-operators are formed which yield quite inefficient problem solutions (Korf, 1985). Furthermore, these macro-operators cannot be transferred between problems with a different number of disks. When multiple levels of descriptions are available like in SOAR, more useful chunks of problem solving experiences can be formed (Ruiz & Newell, 1989).

With a comprehension-based approach an abstract procedure schema is formed in terms of situation knowledge. The same abstract schema is thus acquired for all Tower of Hanoi tasks with more than 4 disks. The schema acquired from the 3- and 4-disk problems is still similar but less elaborated. Comprehension-based learning from a 5-disk training therefore produces better solutions for the 4-disk criterion task than a 4-disk training. This result was obtained in the reported experiment.

References

Anderson, J. R. 1987. Skill Acquisition: Compilation of weak-method problem solutions. *Psychological Review* 94(2):192-210.

Hammond, K. 1989. *Case-based planning*. London: Academic Press.

Hendler, J. A. 1989. Marker-passing over micro-features: Towards a hybrid symbolic/connectionist model. *Cognitive Science* 13:79-106.

Kintsch, W. 1988. The use of knowledge in discourse processing: A construction-integration model. *Psychological Review* 95:163-182.

Kintsch, W., Welsch, D., Schmalhofer, F., and Zimny, S. 1990. Sentence memory: A theoretical analysis. *Journal of Memory and Language* 29:133-159.

Korf, R. E. 1985. Macro-operators: A weak method for learning. *Artificial Intelligence*, 26:35-77.

Mannes, S. M., and Kintsch, W. 1991. Routine computing tasks: Planning as understanding. *Cognitive Science* 15:305-342.

Michalsky, R. S., and Kodratoff, Y. 1990. Research in machine learning: Recent progress, classification of methods, and future directions. In Michalsky, R. S., and Kodratoff, Y. eds. *Machine learning: An artificial intelligence approach* (Vol. 3, pp. 3-30). San Mateo, CA: Morgan Kaufmann.

Minton, T. M., Carbonell, J. G., Knoblock, C. A., Kuokka, D. R., Etzioni, O., and Gil, Y. 1989. Explanation-based learning: A problem solving perspective. *Artificial Intelligence* 40:63-118.

Nathan, M. J., Kintsch, W., and Young, E. 1992. A theory of algebra word problem comprehension and its implications for the design of computer learning environments. *Cognition and Instruction* 9(4):329-389.

Rosenbloom, P. S., Laird, J. E., Newell, A., and McCarl, R. 1991. A preliminary analysis of the SOAR architecture as a basis for general intelligence. *Artificial Intelligence* 47: 289-325.

Rumelhart, D. E., Smolensky, P., McClelland, J. L., and Hinton, G. E. 1986. Schemata and sequential thought processes in PDP models. In McClelland, J. L. and Rumelhart, D. E. eds. *Parallel distributed processing* (Vol. 2, pp. 7-57). Cambridge, MA: MIT Press.

Ruiz, D. & Newell, A. 1989. Tower-noticing triggers strategy-change in the Tower of Hanoi: A Soar Model. *Cognitive Science Proceedings*.

Schmalhofer, F. & Glavanov, D. 1986. Three components of understanding a programmer's manual: Verbatim, propositional and situational representations. *Journal of Memory and Language* 25:279-294.

Schmalhofer, F. Boschert, St., & Kühn, O. 1990. Der Aufbau allgemeinen Situationswissens aus Text und Beispielen (The construction of general situation knowledge from text and examples) *Zeitschrift für Pädagogische Psychologie* 4(3):177-186.

Schmalhofer, F., Bergmann, R., Boschert, St., and Thoben, J. (in press). Learning program abstractions: Model and empirical validation . In Strube, G., and Wender, K. F. eds. *The cognitive psychology of knowledge. The German "Wissenspsychologie" project.* Amsterdam: Elsevier.

Simon, H. A. 1975. The functional equivalence of problem-solving skills. *Cognitive Psychology* 7: 268-288.