ED 342 644 SE 052 498

AUTHOR	Ohlsson, Stellan
TITLE	Artificial Instruction. A Method for Relating Learning Theory to Instructional Design.
INSTITUTION	Pittsburgh Univ., Pa. Learning Research and Development Center.
SPONS AGENCY	Office of Educational Research and Improvement (ED), Washington, DC.; Office of Naval Research, Arlington, VA. Cognitive and Neural Sciences Div.
REPORT NO	UPITT-LRDC-ONR-KUL-90-05
PUB DATE	Sep 90
CONTRACT	N00014-89-J-1681
NOTE	57p.
PUB TYPE	Reports - Evaluative/Feasibility (142)
EDRS PRICE	MF01/PC03 Plus Postage.
DESCRIPTORS	Arithmetic; *Artificial Intelligence; Cognitive Processes; *Computer Simulation; Elementary Education; *Instructional Design; Learning Processes; Learning Strategies; *Learning Theories; Mathematics Education; Mathematics Instruction; *Mathematics Skills; Research Design; Research Methodology; *Subtraction; Teaching Methods
IDENTIFIERS	Learning Research and Development Center; *Regrouping (Mathematics)

ABSTRACT

Prior research on learning has been linked to instruction by the derivation of general principles of instructional design from learning theories. However, such design principles are often difficult to apply to particular instructional issues. A new method for relating research on learning to instructional design is proposed: Different ways of teaching a particular topic can be evaluated by teaching that topic to a simulation model of learning and recording the complexity of the resulting learning processes. A study to compare two mathematically correct algorithms for computing the difference between two multi-digit numbers from a conceptual or mechanical perspective was designed for both methodological and substantive purposes. The algorithms chosen to model were "regrouping" and "augmenting". Explanations of the architecture of simulation system production are provided. Learning difficulty is determined by the number of states and cycles that the simulation system carries out to learn the method carried out over all the training problems. Results of the learning runs imply that regrouping is more difficult that augmenting, and that learning subtraction conceptually is more difficult than learning it mechanically, a conclusion that would seem to contradict widely held beliefs in the mathematics education community. The presuppositions that accurate simulation models can be developed are discussed and the advantages and disadvantages of the general method of simulation use are evaluated. (MDH)

Artificial Instruction. A Method for Relating Learning Theory to Instructional Design

Stellan Ohlsson
The Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA 15260, USA

Technical Report No. KUL-90-05
September, 1990

Center for the Study of Learning

LEARNING RESEARCH AND DEVELOPMENT CENTER

University of Pittsburgh

Artificial Instruction. A Method for Relating Learning Theory to Instructional Design

Stellan Ohlsson
The Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA 15260, USA

Technical Report No. KUL-90-05
September, 1990

To appear in: P. H. Winne and M. Jones, (Eds.), *Foundations and frontiers in instructional computing systems*. New York, NY: Springer-Verlag. In press.

Prepartion of this manuscript was supported, in part, by grant no. N00014-89-J-1681 from the Cognitive Science Program of the Office of Naval Research (ONR), and in part by the institutional grant for The Center for the Study of Learning from the Office of Educational Research and Improvement (OERI), US Ministry of Education. The opinions expressed do not necessarily reflect the positions of the sponsoring agencies, and no endorsement should be inferred.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UPITT/LRDC/ONR/KUL-90-05	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Learning Research and Development Center, Univ. of Pitt 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Cognitive Science Program Office of Naval Research (Code 1142C5)
6c. ADDRESS (City, State, and ZIP Code) 3939 O'Hara Street Pittsburgh, PA 15260	7b. ADDRESS (City, State, and ZIP Code) 800 North Qunicy Street Arlington, VA 22217-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-89-2-1681
8c. ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO.
61153N	RR04206	RR04206-01	NR442a523

11. TITLE (Include Security Classification)
Artificial Instruction. A method for relating learning theory to instructional design.

12. PERSONAL AUTHOR(S)
Dr. Stellan Ohlsson

13a. TYPE OF REPORT	13b TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Technical	FROM ______ TO ______	September, 1990	48

16 SUPPLEMENTARY NOTATION
Partial funding by OERI institutional grant for the Center for the Study of Learning.

17 COSATI CODES

FIELD	GROUP	SUB-GROUP
05	02	

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Arithmetic, augmenting, computer simulation, instructional design, learning theory, regrouping, subtraction, understanding.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)
In the past, research on learning has been linked to instruction by the derivation of general principles of instructional design from learning theories. But such design principles are often difficult to apply to particular instructional issues. A new method for relating research on learning to instructional design is proposed: Different ways of teaching a particular topic can be evaluated by teaching that topic to a simulation model of learning and recording the complexity of the resulting learning processes. An application of this method to a traditional problem in mathematics education suggests that conceptual instruction in arithmetic causes more cognitive strain than mechanical instruction, contrary to a widely held belief in the mathematics education community. The advantages and disadvantages of the general method are discussed.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Susan M. Chipman	22b. TELEPHONE (Include Area Code) (202) 696-4318	22c. OFFICE SYMBOL ONR 1142C5

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE
S/N 0102-LF-014-6603 Unclassified

Knowledge and Understanding in Human Learning

Knowledge and Understanding in Human Learning is an umbrella term for a loosely connected set of activities lead by Stellan Ohlsson at the Learning Research and Development Center, University of Pittsburgh. The aim of KUL is to clarify the role of *world knowledge* in human thinking, reasoning, and problem solving. World knowledge consists of concepts and principles, and contrasts with facts (episodic knowledge) and with cognitive skills (procedural knowledge). The long term goal is to answer six questions: How can the concepts and principles of particular domains be identified? How are concepts and principles acquired? How can the acquisition of concepts and principles be assessed? How are concepts and principles encoded in the mind? How are concepts and principles utilized in performance and learning? How can instruction facilitate the acquisition and utilization of concepts and principles (as opposed to episodic or procedural knowledge)? Different methodologies are used to investigate these questions: Psychological experiments, protocol studies, computer simulations, historical studies, semantic, logical, and mathematical analyses, instructional intervention studies, and so on. A list of KUL reports appear at the back of this report.

Abstract

In the past, research on learning has been linked to instruction by the derivation of general principles of instructional design from learning theories. But such design principles are often difficult to apply to particular instructional issues. A new method for relating research on learning to instructional design is proposed: Different ways of teaching a particular topic can be evaluated by teaching that topic to a simulation model of learning and recording the complexity of the resulting learning processes. An application of this method to a traditional problem in mathematics education suggests that conceptual instruction in arithmetic causes more cognitive strain than mechanical instruction, contrary to a widely held belief in the mathematics education community. The advantages and disadvantages of the general method are discussed.

Keywords: Arithmetic, augmenting, computer simulation, instructional design, learning theory, regrouping, subtraction, understanding

On the Relation Between Learning Theory and Instruction

Instruction is an artefact, a social practice deliberately designed to achieve a particular purpose. A theory of instruction is therefore a *prescriptive* theory. The task of such a theory is to state principles that constrain search through the space of instructional designs [30]. A theory of learning, on the other hand, is a *descriptive* theory. The task of a learning theory is to state principles that accurately describe the mechanisms of cognitive change. Instructional theory and learning theory are distinct intellectual enterprises, just as agriculture and botany, medicine and physiology, engineering and physics are distinct enterprises [10, 12].

As these analogies suggest, the enterprises of instruction and learning, although distinct, are closely related. Physical therapies that ignore the chemistry and physiology of the human body are likely to do the patient more damage than good; machines that violate the laws of physics cannot work. Similarly, instructional designs that are not in accord with the mechanisms of cognitive change are unlikely to facilitate learning.

The notion that a theory of instruction should be informed by a theory of learning is hardly controversial when stated abstractly. Glaser traces this idea back to both John Dewey and Edward L. Thorndike [10], but there are many recent advocates [13, 32, 34, 35]. But how, specifically, are the two enterprises supposed to interact? How can instructional designs be informed by principles of learning? The traditional method for applying learning theory to instructional questions is to derive general principles of instruction from general principles of learning; the application of the derived principles to the design of instruction in a particular topic is left to the designer. The first systematic application of this method was launched by the behaviorists. Principles of stimulus-response relations and reinforcement gave rise to instructional principles that emphasized behavioral objectives and maximally efficient reinforcement schedules [11]. The application of piagetian research to instructional questions has taken a similar form: The principle that equilibrium requires a balance between assimilation and accomodation has given rise to training programs that deliberately induce disequilibrium in order to accelerate cognitive change [22]. David Ausubel's theory of learning as successive elaboration gave rise to Reigeluth's theory of instructional design [32]. In each approach, general principles of instruction are derived from general principles of learning, but the application of those design principles to particular instructional topics is based on intuition, common sense, and seat-of-the-pants judgments.

Modern cognitive psychology, based on information processing concepts, has surpassed past approaches with respect to the power of its theories, and with respect to the depth and the detail of its descriptions of cognitive processes. But its application to instructional questions has so far taken the same old form: General principles of instructional design are derived from general principles of learning; the application of those principles to particular instructional designs is left to the designer. For example, the principles of the ACT* theory [1] have given rise to several instructional principles, including that one should teach the goal tree for cognitive skills [2]. This principle is surely correct, but its application to a particular instructional topic is nevertheless problematic. How is this principle to be applied, for example, in the teaching of arithmetic? Should one teach the entire goal tree for subtraction with regrouping to all students, even to very young students? Are there no situations in which the complexity of the goal tree might be an obstacle to learning? Should the entire goal tree be taught at once, or should one introduce it component by component? If so, how should the components be sequenced? The general principle does not, by itself, answer instructional questions of this detailed sort.

This chapter explores a different approach to the interaction between the theory of learning and the theory of instruction. Instead of deriving *general* principles of instruction from a learning theory, this approach exploits the fact that information processing theories of learning can be embodied in runnable simulation models to answer *particular* instructional questions. A common and important type of instructional problem--perhaps the only type--is to decide between alternative ways of teaching a particular topic. Problems of this type can be solved, I suggest, by teaching the relevant topic to a simulation model of learning. To compare two ways of teaching a particular topic, we teach that topic to the learning model in both ways, and we measure the computational complexity of the learning processes induced in the two cases. If the simulation model expends less computational work to learn under one form of instruction than under another, then it predicts that the former is preferable to the latter. The main purpose of this chapter is to present an application of this method to a traditional issue in arithmetic instruction.

The method of teachable simulation models has three prerequisites. First, it requires a *runnable* model. So-called information processing models that consist of labelled boxes with arrows of varying thickness going in and out of them are of no help; neither are computer models with such shaky implementation that they can barely produce a single demonstration run without breaking; neither are programs that only embody *some* of the assumptions of the underlying theory (while the other assumptions are embodied in some other program). The method of teachable simulation models requires a robust, integrated computer model that can be run on a variety of inputs. Second, the method requires that the simulation model is capable of *learning*. A performance model is not enough. Third, the learning mechanisms of the model must be such that their inputs can be interpreted as *instruction*. A model of learning by doing is not enough; the method requires a model of learning from declarative messages that originate in an outside source. The HS model described below satisfies these three prerequisites.

The particular instructional question investigated in this chapter concerns the teaching of arithmetic. The question of how to teach an arithmetic skill like subtraction has been approached in different ways by different generations of researchers. An earlier generation focussed on the question of which subtraction algorithm is easier for children to learn. Large scale empirical research programs were launched to answer this question [5, 6]. The answer was, briefly summarized, that the method of regrouping (or "decomposition") is easier to learn than the method of augmentation (or "equal addition"), at least when subtraction is taught conceptually (as opposed to mechanically). I show in this chapter that the method of teachable simulation models implies a different answer to this question.

The current generation of researchers in mathematics education focusses on the contrast between rote and insightful learning of arithmetic algorithms. They strive to find methods that facilitate school children's acquisition of the conceptual rationale for arithmetic algorithms, in the hope that conceptual understanding will eliminate errors, improve retention, and faciliate transfer to unfamiliar problems [15]. The method of teachable simulation models leads me to a rather contrary answer to this question.

In summary, the present chapter has both a methodological and a substantive purpose. I propose a general method that exploits the fact that information processing theories of learning can be embodied in runnable simulation models to answer particular instructional questions. The method is introduced in the context of a particular application. The application is not merely a demonstration of the method. The specific conclusions reached have important implications for instruction in arithmetic.

Regrouping versus Augmenting

There are several mathematically correct algorithms for computing the difference between two multi-digit integers. Educational researchers at the beginning of this century asked whether one of these algorithms is easier to learn than the others, a very reasonable question. In the *regrouping* algorithm non-canonical columns, i. e., columns in which the minuend digit is smaller than the subtrahend digit, are dealt with by incrementing the relevant minuend digit with one place-value unit. To keep the value of the minuend constant, this change in the minuend is compensated by decrementing the first non-zero minuend digit with a higher place value than the incremented digit. In the *augmenting* algorithm non-canonical columns are also dealt with by incrementing the minuend digit, but in this case the change in the minuend is compensated by *incrementing* the *subtrahend* digit with the next higher place value. (Strictly speaking, the entities which are incremented and decremented are the *numbers* which the digits refer to. Since no ambiguity results, I use the somewhat inaccurate locution "decrementing a digit" instead of the accurate but tedious "decrementing the number a particular digit refers to" .)

Which algorithm is easier?

The regrouping and augmenting algorithms build on different mathematical ideas. The regrouping algorithm is based on *the associative law*

$$(a + b) + c = a + (b + c).$$

The associative law implies that the value of the minuend remains constant through the regrouping operation. (A complete derivation of the regrouping algorithm from first principles is available in [25].) The augmenting algorithm, on the other hand, is based on *the constant difference law*

$$a - b = (a + k) - (b + k).$$

This law implies that the difference between the minuend and the subtrahend remains constant through the augmenting operation. (A more detailed discussion of the rationale for the augmenting algorithm is available in [8].) Since the two algorithms build on different mathematical ideas, it is entirely plausible that one of them is easier to learn and/or to execute than the other.

Large-scale classroom studies were performed in the early decades of this century in an effort to settle this issue empirically. William Brownell concluded: "Even a cursory survey of the ... experimental results ... reveals the impossibility of deciding simply and finally between D [the regrouping method] and EA [the equal addition method] as the better procedure for teaching 'borrowing'" [5, p. 169]. Augmenting was found to be easier than regrouping more often than the other way around, but the observed difference was small in magnitude. Brownell argued that the results were only in favor of augmenting when subtraction was taught as a mechanical performance. If subtraction was taught conceptually, he claimed, the results favored regrouping [5, 6]). Brownell's argument was widely accepted and politically instrumental in settling the issue in favor of teaching the regrouping method in American schools. Educators in other nations were not equally convinced, and the augmenting method is still taught in some European schools.

The empirical studies did not clearly distinguish between performance and learning. They confused the question *which algorithm is easier to use*? with the question *which algorithm is easier to learn*? One reason for the lack of separation of these two questions is that pure measures of learning are hard to come by. We can only observe by recording performances, so most empirical measures will confound the two questions. In the context of a simulation model, the two questions can be cleanly separated. This section investigates which algorithm is easier to use, while the next section investigates which algorithm is easier to learn.

In information processing terminology, the question of which algorithm is easier to use can be reformulated as follows: What is the relation between the cognitive complexity of the mental procedure corresponding to the regrouping algorithm and the cognitive complexity of the procedure corresponding to the augmenting algorithm? This question can be answered by implementing the two algorithms as psychologically plausible simulation models, run those models, and measure their relative complexity.

Simulating regrouping and augmenting

The hypothesis that cognitive skills (mental procedures) are encoded as production systems was first proposed by Allen Newell and Herbert A. Simon [23], and has been adopted by a number of researchers [1, 18, 19]. According to the production system hypothesis, cognitive skills are encoded in sets of *production rules*, where each production rule has the general form

Goal + Situation --> Action.

The symbol "Goal" stands for a specification of a desired situation, "Situation" stands for a description of the relevant features of the current situation, and "Action" refers to something the person knows how to do. The intended interpretation of such a rule is that when the person has the specified goal, and he or she is in a situation that fits the situation description, then he or she will consider the specified action. A collection of interrelated production rules is a *production system*. Each cognitive skill is hypothesized to correspond to a production system.

A production system *architecture* is a program that can interpret a production system. In this context, to interpret means to (a) decide which production rules (in a particular production system) are satisfied in the current situation, (b) select one or more rules to be evoked, and (c) execute the actions of the evoked rules. Each pass through the three steps (a)-(c) is one production system *cycle*, or operating cycle. The number of cycles required to execute a production system is one of the measures of cognitive complexity used in this chapter.

The satisfied rules are identified by matching the Situation against the so-called working memory, a data base which contains the system's information about the current state of affairs, and by matching the Goal against the system's current goal. If both components match, the rule is satisfied and is therefore a candidate for being evoked. Selecting which rules to evoke is sometimes called *conflict resolution* [21]. A typical conflict resolution scheme is to select those rules that match against the most recent information in working memory. Execution of the primitive actions must involve calls on motor programs that control the muscles of the relevant limbs, e. g., the finger muscles for the action of writing a digit, but production system theories do not have much to say about this aspect of human cognition.

The HS architecture is a relatively standard production system architecture. It has a single working memory which contains information about both the current state of affairs, and the systems' current goal(s). All available rules are matched against working memory in each operating cycle. There is no conflict resolution. Every satisfied rule is evoked. There is no complexity limitation on the left-hand side of the rules, but the right-hand side (the action part) is limited to a single action. The system continues to match and evoke rules until either there are no satisfied rules, or the current problem is solved. Detailed descriptions of the HS architecture are available in [28, 29].

Table 1. The distribution of production rules in two canonicalization algorithms.

Rule type	Regrouping	Augmenting
Visual	4	3
Motor	11	11
Write & cross out	6	6
Say answer	5	5
Cognitive	20	17
Create expressions	11	12
Revise expressions	9	5
Memory	3	4
All rules	38	35

In order to simulate subtraction with regrouping, the HS system was extended with a (simulated) task display and a (simulated) visual-motor interface consisting of an eye and a hand. The *task display* is a data structure in the computer which contains the same information as a piece of paper with a subtraction problem written on it. Technically speaking, the task display is a two-dimensional array of digits. (I am assuming that the subtraction problem is written in vertical format.) Information about the task display enters into the working memory of the HS system through a simulated *eye*, a program module which can only access one digit at a time. When the simulated eye 'looks' at a digit, information about that digit is entered into working memory. In order to gather information about some other digit, the eye has to be moved. The eye can move left, right, up, and down. Eye movements are distinct computational steps, so control of visual attention is encoded in production rules. The model can alter the external task display only through the use of a simulated *hand*. The hand can cross out an existing digit and write a digit in a blank space. These two primitive actions count as distinct computational steps, so the hand is also controlled by production rules. In short, the model simulates subtraction at the level of individual eye movements and individual writing actions, a very fine-grained level of analysis compared to most simulation models.

HS was also equipped with a long-term memory for number facts, e. g., 8 - 7 = 1. Retrieval of number facts was simulated with a function which returns the (correct) answer to any query about relations between two numbers. HS does not simulate the probabilistic nature of memory retrieval, nor the existence of incorrect number facts. Like attention allocation and writing, memory retrieval is a distinct computational step which is controlled by production rules.

The HS models of regrouping and augmenting consist of 38 and 35 production rules, respectively. The number of different rules in different categories are shown in Table 1. The distribution of rules over visual steps (i. e., move the eye), motor steps (i. e., write, cross out, and say the answer), cognitive steps (i. e., the creation and revision of working memory expressions), and memory steps (i. e., retrievals from long-term memory) is approximately the same for both models. The details of the rules themselves are not important for present purposes. Examples of complete production rules are available in [8].

In order to estimate the cognitive complexity of the two subtraction algorithms, the two simulation models were run on a subtraction test consisting of 66 subtraction problems which varied with respect to number of columns, number of non-canonical columns, and number of *blocking zeroes*, i. e., zeroes immediately to the left of a non-canonical column (or another blocking zero). The number of production system cycles required by each model to complete each problem was recorded. In addition, each cycle was classified with respect to the type of rule that was evoked in that cycle.

The results are shown in Figure 1. The figure shows the cognitive complexity of the regrouping and augmenting algorithms on eleven different problem types. Problem types 1-4 have two, three, four, or five canonical columns, respectively, but no non-canonical columns. The number of cycles required to complete such problems is the same for both models. Problem types 5-8 have one, two, three, or five non-canonical columns, respectively. The regrouping model requires more steps to handle each such column than the augmenting model. The difference is small in magnitude. The difference is located entirely in the visual-motor interface, i. e., the regrouping algorithm requires more cycles because it involves more complicated attention allocation.

Problem types 9, 10, and 11 have one, two, or three blocking zeroes, respectively. (A blocking zero is immediately to the left of a non-canonical column or another blocking zero.) The regrouping model has a slight advantage on these problem types. The reason is that once a set of columns have been traversed by the regrouping procedure, no further regrouping of those columns is needed. The augmenting algorithm, on the other hand, has to augment every column with zero as the subtrahend digit and a non-zero minuend digit. Consequently, if there are several blocking zeroes in a problem, the regrouping algorithm completes that problem in slightly fewer operating cycles than the augmenting algorithm. Once again, the difference is small in magnitude. A more extensive discussion of these results is available in [8].

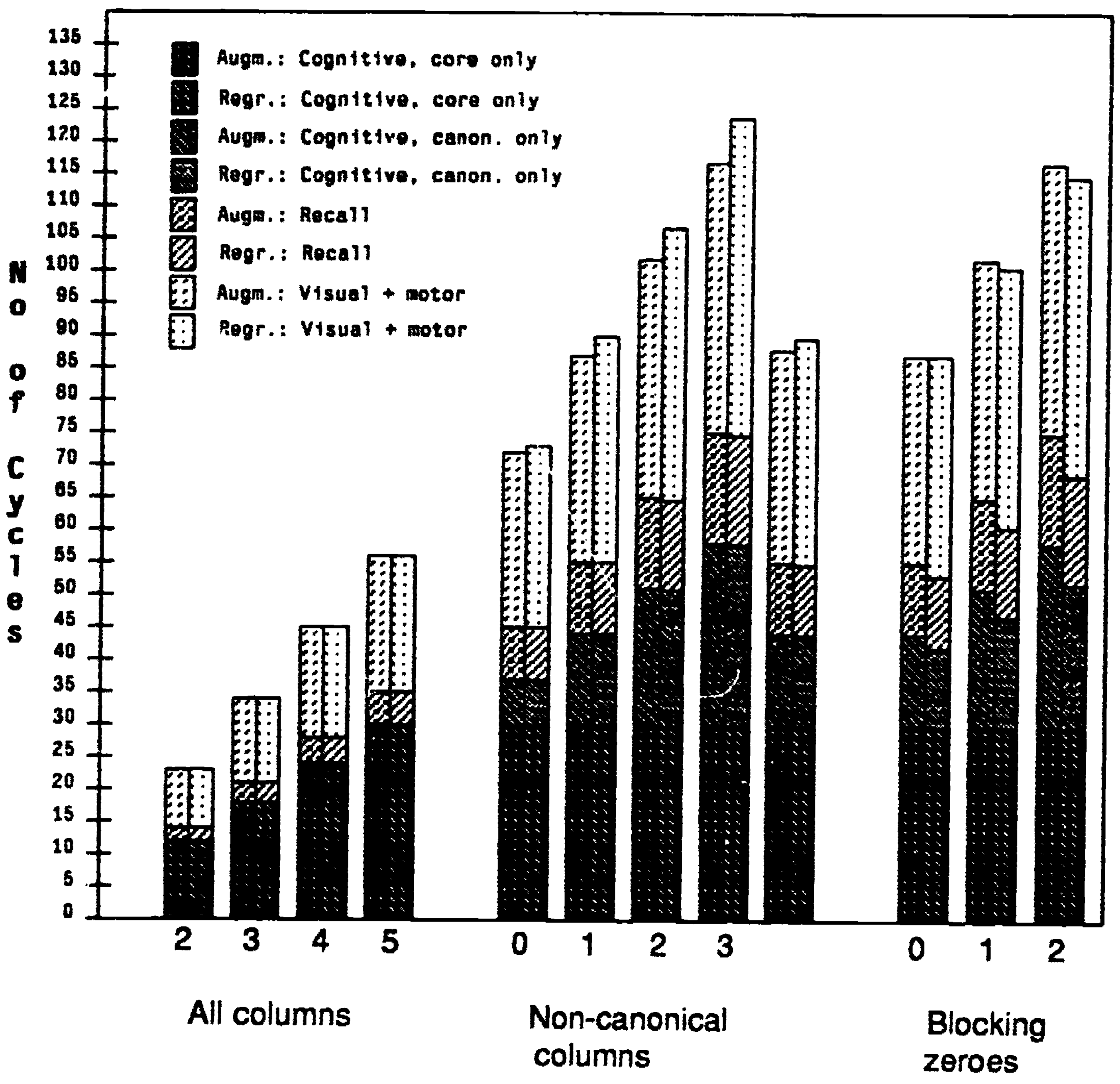

Figure 1. The number of production system cycles required to execute the regrouping and augmenting algorithms in eleven different problem types. The regrouping bar is to the right and the augmenting bar to the left for each problem type. Each bar is segmented to show the number of cognitive steps for canonical columns (bottom segment), cognitive steps for non-canonical columns (second segment from bottom), memory steps (third segment from bottom), and the number of eye and hand movements (top segment).

Discussion

The simulations of the regrouping and augmenting algorithms teach us several lessons. First, the difference between the two algorithms with respect to cognitive complexity is small in magnitude. Since the two algorithms are derived from different mathematical ideas, it is not obvious why this is so. Closer reflection reveals the reason. Both the law of associativity and the constant difference law are instances of a more general law which says that a quantity remains constant if every change in it is compensated by a corresponding counterchange. The structure of this law implies that the goal structure of the corresponding algorithm will contain two main subgoals: a change goal and a compensate goal. This is indeed the case for both algorithms. Furthermore, the internal structure of each change or compensation is always the same: Cross out a digit, compute the replacement digit, and write the replacement digit. Since the structure of the goal tree is similar in both algorithms, the number of cycles of operation is nearly equal. This equality is, in a sense, accidental. In general, there is no reason to expect different mathematical laws to generate algorithms with similar goal structures.

Second, the simulations show that the differences between the two algorithms have different directions on different types of problems. There is no difference on canonical problems. The difference is in favor of augmenting on problems which have non-canonical columns but no blocking zeroes. The difference is in favor of regrouping on problems which have two or more blocking zeroes. The implication of this result is that empirical measures of the cognitive complexity of the two algorithms will depend on the composition of the test. A test without blocking zeroes will favor the agumenting algorithm, but a test with many blocking zeroes will favor regrouping. In a mixed test the differences will tend to cancel each other. Unfortunately, some of the pre-World War II studies did not specify which subtraction problems were used to measure the students' performance.

The outcome of the simulation runs are consistent with the pattern of empirical results in the literature. If there are only small differences, and if those differences go in different directions for different classes of problems, then we would expect empirical measurements to give inconsistent results. Sometimes one algorithm should appear to be easier, sometimes the other, due either to the composition of the test problems or to sampling error. This is exactly what the literature shows [5, 6].

These simulations imply that it does not matter which algorithm is taught. Regrouping and augmenting are equally complicated; the differences in cognitive complexity are too small to be of pedagogical significance. This conclusion is consistent with the fact that both algorithms are, in fact, taught in different school systems, without noticable higher degree of success in one system than in the other. However, the study summarized in this section (and reported in more detail in [8]) only concerned the *execution* of the two algorithms. The two algorithms are equally complex to use, once learned. But Brownell's argument was that regrouping is easier to learn than augmenting, at least if subtraction is taught conceptually. We therefore need to investigate the cognitive complexity of the *construction* (as opposed to execution) of the two algorithms. In addition, we need to compare the cognitive complexity of the construction under both conceptual and mechanical instruction.

Conceptual versus Mechanical Instruction

As mathematics educators deepen their analysis of mathematical cognition, they become more and more concerned with the question of conceptual understanding [15]. This concern is partly fuelled by research into childrens' mathematical errors. Catalogues of error patterns have been compiled for a number of mathematical tasks, including subtraction [4, 38, 39, 41] and fractions [9, 14, 17, 27, 31, 36, 37]. Most of the error patterns described in these catalogues are senseless; they have no discernable relation to the correct mathematical operations. To observe children making senseless mistakes is a frustrating experience, and it is impossible not to believe that if children only understood what they are doing, they would not make those mistakes. Following this line of reasoning, mathematics educators have tried to design conceptually based instruction in arithmetic.

Does conceptual understanding help?

The purpose of many instructional interventions in arithmetic is to show that if children are taught the conceptual rationale for the arithmetic algorithms, they will have less difficulty in learning those algorithms, and their performance will be less error prone and more flexible in response to changing task demands [15]. Unfortunately, this enterprise has not been spectacularly successful.

A training study by Resnick and Omansson can serve as an example [33]. Children with faulty subtraction performance were taught the conceptual rationale of the regrouping algorithm with the help of Diene's blocks. The instruction was designed to force children to map back and forth between blocks and numbers. The children first performed a step with the blocks, and then performed the same step with the symbols. At the end of the instruction, several of the children could explain the correct subtraction procedure. When they were given subtraction problems to perform, they nevertheless made errors. As a second example, Ohlsson, Bee, and Zeller taught children how to add fractions with an interactive computer tool that enabled children to switch back and forth between graphical and numerical representations of fractional quantities [27]. A change in one representation was automatically mirrored by the corresponding change in the other representation. A detailed analysis of the children's performance on the pre- and posttests revealed that they could map back and forth between the fraction symbol x/y and concrete representations of fractional quantities. All of them nevertheless committed the standard error of adding fractions by adding both numerators and denominators on the posttest. In both of these studies, instruction that was carefully designed to make the meaning of the mathematical operations evident failed to prevent or cure senseless errors.

These empirical failures focus attention on the lack of theoretical analysis of conceptual understanding in the context of arithmetic. What is meant by conceptual understanding, and what is its (supposed) function in procedural learning? How does conceptual instruction interact with the construction of a mental procedure? Why should we believe that knowledge of the rationale of an arithmetic procedure facilitates the learning of that procedure? In spite of the recent emphasis on conceptual understanding in arithmetic instruction, little effort has been spent in answering these questions.

My approach to these questions is to extend the HS architecture with a learning mechanism that enables the model to learn procedures on the basis of instruction. The instruction is modeled as a set of declarative knowledge [illegible] that the user gives to the system. Such a learning mechanism enables us to *teach* the model how to do subtraction. We supply the system with a set of declarative knowledge units which correspond to the instructions a teacher would give a student, and the system learns by converting those knowledge units into a cognitive skill, i. e., into production rules. By giving the system *different* sets of declarative knowledge units, we can simulate the effects of *different* ways of teaching subtraction. In particular, we can compare conceptual instruction with mechanical instruction.

Making HS teachable

In a production system architecture, a *learning mechanism* is any process that can revise existing production rules or generate new ones. When a new rule is added to a production system, the behavior of the system changes. The new rule will control behavior in those situations in which it matches working memory. Since the new rule is different from previous rules, the system's behavior will be different. The fact that the behavior changes is the main reason to regard the generation of new rules as a simulation of (procedural) learning.

Artificial Instruction

A number of simulation systems model procedural learning as the construction of new production rules (see, e. g., [1, 3, 16, 18, 19, 24, 39]). These models simulate learning by doing, i. e., they model the effects of *practice*. In spite of their differences, they all instantiate the same abstract theory. The first principle of this abstract theory is that humans have access to one or more weak problem solving mechanisms (analogy, hill climbing, planning, search, etc.) which can generate task oriented behavior on unfamiliar problems. The second principle is that information about each problem solving step--the reasons for taking it, the desirability of the outcome, the temporal order of the steps, and so on--is stored in long-term memory. The third principle of the abstract theory is that the learning mechanisms construct new rules through some form of induction over the individual steps. For example, the SAGE system described by Langley carries out forward search and stores steps in which a particular action had good outcomes, as well as steps in which that action had bad outcomes [20]. The system learns by identifying one or more situation features that discriminate between the two classes of situations, and it constructs a new rule by incorporating those features into the rule that controls that action. Different models of learning by doing differ with respect to which weak methods they postulate, which information they assume is stored in memory, and which induction procedure they use, but they all instantiate the three abstract principles stated above.

Simulation models that instantiate the abstract theory of learning by doing are quite successful in modeling the effects of practice. But models of practice are not sufficient for present purposes. There is nothing in such systems that correspond to instruction, i. e., to a set of messages that originate outside the system and which are used to construct new procedural knowledge. A learning mechanism which is to simulate learning from instruction must take declarative knowledge units among its inputs.

In the HS system, general world knowledge, including knowledge imparted by instruction, is assumed to consist of *constraints* on cognitive processes. For example, the laws of the number system impose constraints on arithmetic operations. Unless an addition procedure yields the same result for (a + b) + c as for a + (b + c), i. e., unless it satisfies the constraint imposed by the associative law, it is not a correct addition procedure. The notion of general knowledge as constraints is not limited to arithmetic, or, indeed, to mathematics. For example, the laws of conservation of energy, mass, and momentum are examples of natural science principles which are naturally cast as constraints. Traffic laws are good examples of constraints in everyday life. I do not claim that all general knowledge can be formulated as constraints, only that constraints is one important form of knowledge, a form, moreover, which is particularly relevant to arithmetic. In the HS system, constraints are encoded in knowledge elements which are distinct from both working memory elements and from production rules.

An incorrect or incomplete arithmetic procedure typically leads to results that *violate* one or more of the relevant constraints. For example, an incorrect or incomplete regrouping procedure might violate the constraint that the value of the subtrahend is to remain constant over regrouping. The basic idea behind the HS system is that a constraint violation contains information about how to revise the faulty procedure so that similar constraint violations are avoided in the future. In each operating cycle, the system matches all available constraints against the current state of affairs. If a constraint is satisfied, no action is taken. If one or more constraints are violated, the learning mechanism is triggered. This corresponds to having a tutor who watches a problem solution and provides instruction when needed. (The HS system is given all the constraints at the beginning of the simulation run, rather than single constraints--instructions--at select points during a problem solving. Since the system effectively does not 'see' a constraint until it is violated, this difference to real tutoring is less significant than it first appears.) The learning mechanism analyzes the constraint violation, and revises the faulty rule accordingly. The technical details of the learning mechanism are not important for present purposes. A detailed description of the HS learning mechanism is available in [28, 29].

Since learning happens when the behavior of the system causes a constraint violation, there must be some initial rules which can generate behavior. HS must be supplied with at least one initial rule for each problem solving operator. In the simulation runs reported in this chapter, the initial rules are minimal, i. e., their condition sides contain only the applicability conditions for the relevant action. These incomplete rules generate almost random behavior. Each action is considered in every situation in which its applicability conditions are satisfied. The probability of causing a constraint violation is high. The system detects the violation, revises the faulty rule, and then starts over on the problem. The cycle of trying to solve the problem, detecting a violation, revising the faulty rule, and starting over continues until the problem can be solved without any constraint violations. This is a reasonable first approximation model of learning to solve problems under tutelage.

In summary, the HS system encodes declarative knowledge, including instructions, as constraints on behavior. In arithmetic, the effect of faulty or incomplete procedural knowledge is typically to generate results that violate the constraints imposed by the laws of numbers. HS learns by analyzing a constraint violation and revising the rule that caused the violation in such a way that similar constraint violations are avoided in the future. This capability makes HS *teachable*: To teach HS a particular procedure, the user supplies the system with an initial set of (incomplete) rules and the constraints that define the correct procedure. Each constraint corresponds to an instruction. The system tries to solve problems, makes mistakes, and learns from the instructions it has been given. If the instructions are complete enough, the system will eventually arrive at the correct procedure.

Teaching HS subtraction

The HS system was taught both the regrouping and the augmenting algorithms for subtraction, and both algorithms were taught in two different ways, corresponding to conceptual and mechanical instruction. This subsection describes the inputs to the four simulation experiments, and the next subsection describes the results.

What does it mean to do subtraction procedurally, as a mechanical skill? A person who does subtraction mechanically is not thinking about the mathematical objects--the numbers--symbolized by the digits in the problem display, nor about the mathematical relations between those numbers. For example, he/she does not think about the fact that the "3" in the numeral "32" denotes the number 30. Instead, he/ she thinks about the digits themselves. He or she performs crossing out and writing actions on the physical display (i. e., the paper) without considering the mathematical meaning of those actions.

Consistent with this interpretation of what it means to do subtraction mechanically, HS was supplied with a representation of a subtraction problem that was isomorphic to the information available in a standard problem display (vertical format). The representation contained information about which digits occurred in which spatial arrangement, but little else. In particular, there was no representation of the place values of the different digits, nor of the current value of either the subtrahend or the minueno. In this representation, a subtraction problem appears as two strings of digits. The representations for the regrouping and the augmenting algorithms were very similar.

If the learner thinks of a subtraction problem in terms of physical operations on the digits in the problem display, he or she cannot benefit from conceptual instruction. For example, instructions that mention the place value of a particular digit can have no impact on a learner who has not internally represented that place value. There is nothing for such an instruction to relate to. The constraints we supplied to HS in the mechanical case were shallow and superficial. They were not derived from the laws of the number system, and they did not mention the conceptual or mathematical meaning of the operations involved.

What does it mean to do subtraction conceptually? The learner who does subtraction conceptually thinks about the numbers symbolized by the digits in the problem display, and he/she is aware of the mathematical interpretation of the actions performed on that display. Consistent with this view, the HS representation for conceptual learning was very different from the HS representation for mechanical learning. In the conceptual representation, a subtraction problem is encoded at the top level as a difference between two numbers. The subtrahend and the minuend are both associated with particular additive decompositions, i. e., sets of numbers that add to those numbers. The elements of the additive decompositions are associated with a face value and a place value. In the conceptual representation, the distinction between numbers and digits is explicit, and the face values of the additive components are associated with the digits in the problem display. The operations of crossing out and writing digits correspond to internal, mental operations on the numbers symbolized by those digits. The representations for the regrouping and augmenting algorithms were once again very similar.

In addition to the representation of the problem and the constraints, HS must also be given some initial procedural knowledge. Without initial rules HS cannot generate behavior, and so cannot discover constraint violations. In the simulation runs presented in this subsection, HS was *given* the correct procedure for *canonical* subtraction problems, i. e., problems in which the minuend digit is larger than the subtrahend digit in every column. The system *learned* to solve *non*-canonical problems, i. e., problems for which the minuend digit is larger than the subtrahend digit in at least one column. In common parlance, the system learned to 'borrow'. I shall refer to this process as *canonicalization*, since the purpose of 'borrowing' is to bring a non-canonical problem onto canonical form. In summary, the system learned two different canonicalization methods, regrouping and augmenting, with two different representations of each method.

In each training run the system tries to solve its current problem. Since the rules for canonical problems cannot handle non-canonical problems, the system commits mistakes. The mistakes are identified by the constraints, and the system applies its learning mechanism to revise the rules. It then starts over. Eventually it learns to solve the problem correctly. If the system is given a second training problem, it may or may not solve that problem correctly. It depends on the relation between the training problems. If it fails to solve the second training problem correctly, it revises its procedure further. In the simulation runs reported below, the system was fed successive training problems until it arrived at the correct subtraction procedure. The number of training problems required varied between two and four, depending on condition. The correctness of the learned procedure was verified by running it on the 66-item subtraction test described earlier in this chapter.

Computational results

Table 2 shows the amount of computational work required to learn to canonicalize in each of the four conditions, summed over all training problems in each condition. It contains several interesting effects. First, the regrouping models require more learning to handle columns with blocking zeroes than columns without blocking zeroes. The augmenting models, on the other hand, are not affected by blocking zeroes. Second, regrouping is computationally more expensive than augmenting. The only exception is that if we disregard blocking zeroes, then regrouping is easier to learn than augmenting *with a mechanical representation.* Third, conceptually based learning is more complex than mechanical learning for both regrouping and augmenting. Also, the difference between the conceptual and the mechanical representations is larger in the case of regrouping than in the case of augmenting. The conceptual regrouping model required 2.3 as many cycles as the mechanical one, while the conceptual augmenting model required 1.3 as many cycles as its mechanical counterpart. Finally, it makes no difference whether we measure the computational complexity by the number of cycles or by the number of search states visited during learning. All effects mentioned here occur in both variables.

Table 2. The amount of computation required by the HS model to learn to canonicalize under four different conditions, measured both in terms of the number of search states visited and the number of production system cycles required.

	Type of representation			
	Conceptual		Mechanical	
Algorithm learned	States	Cycles	States	Cycles
Regrouping				
No blocking zoroes	968	940	464	449
Blocking zeroes	1843	1815	828	794
Augmenting				
No blocking zeroes	889	862	689	687
Blocking zeroes	889	862	689	687

It is, of course, possible to question the psychological relevance of both the number of production system cycles and the number of search states visited. Both measures are heavily dependent on the theoretical assumptions behind the simulation model. If the human learner is not doing search, or if human cognition is not a production system architecture, there might be no relation between these measures and measures of cognitive work in humans. In addition, both measures depend on the particular implementation of the four simulation models.

But the complexity of the four learning processes can also be measured in terms of the number of learning events and the number of rules learned. A learning event is an event in which the system discovers a constraint violation, and revises its current rule set. A learning event might lead to the construction of one or more new rules. The number of learning events required is not primarily a function of the theoretical assumptions behind the models or of the implementation details. It is a measure of how many 'things' there are to learn before the correct procedure has been acquired; it is primarily a function of the logic of the learning task.

Table 3. The amount of lear. ing required by the HS model to learn to canonicalize under four different conditions, measured both in terms of the number of learning events required and the number of new rules created.

	Type of representation			
	Conceptual		Mechanical	
Algorithm learned	Events	Rules	Events	Rules
Regrouping				
No blocking zeroes	23	35	16	23
Blocking zeroes	32	50	24	32
Augmenting				
No blocking zeroes	20	29	18	24
Blocking zeroes	20	29	18	24

Table 3 shows the amount of learning required to master regrouping and augmenting, measured in terms of the number of learning events as well as the number of new rules learned. All the effects observed in Table 2 are reproduced in Table 3: Regrouping is more complex to learn than augmenting (except for problems without blocking zeroes, in the mechanical representation), the conceptual versions require more learning than their mechanical counterparts, and the difference between the conceptual and the mechanical versions is larger in the case of regrouping than in the case of augmenting. All effects appear with both measures. The main difference between Tables 2 and 3 is that both the absolute values and the relative size of the various effects are smaller.

Table 4. The amount of instruction required by the HS model to learn to canonicalize under four different conditions, measured both in terms of the number of constraints (instructions) required and the number of training problems needed.

	Type of representation			
	Conceptual		Mechanical	
Algorithm learned	Constraints	Problems	Constraints	Problems
Regrouping	31	4	21	4
Augmenting	25	2	20	5

Table 4 shows yet another way to measure the outcome of the simulation experiments. Instead of measuring the amount of *learning*, Table 4 measures the amount of *instruction* needed to teach the HS model the two subtraction algorithms. The amount of instruction is measured in terms of how many constraints--instructions--we had to provide HS with in order to bring it up to correct performance. All the relevant effects from the other tables are reproduced in this variable. Regrouping requires more constraints than augmenting, and the difference is larger in the conceptual than in the mechanical case.

The amount of instruction can also be measured in terms of the number of training problems needed to bring the model up to correct performance. This measure shows a different pattern: With respect to regrouping, the number of training problems is the same for both conceptual and mechanical representations. Augmenting requires one more training problem than regrouping in the mechanical representation. Finally, to learn augmenting with the conceptual representation requires only two training problems, the lowest of the four measures. This is the only case where the conceptual representation has an advantage. The number of training problems is a coarse measure of the complexity of the learning processes involved, and this result carries little weight against the consistent pattern across the five other measures.

Discussion of substantive conclusions

The results from the learning runs imply, briefly put, that regrouping is more difficult than augmenting, and that learning subtraction conceptually is more difficult than learning it mechanically. Since these results go against current wisdom in the mathematics education community, it is natural to ask what confidence we can place in them. The simulation model that produced these results might not be an accurate model of human learning. There is the possibility that the production system hypothesis is wrong. Also, the particular learning mechanism implemented in HS might not correspond to any type of learning that humans do. In either case, we would have to admit that HS does not simulate human performance or learning. The relevance of the computational results to instruction is then doubtful.

Another possibility which would lessen the relevance of the computational results is that the production system hypothesis is correct, but HS is the wrong implementation of it. Simulation models are always underdetermined by the theories they embody [26]. There is always the possibility that the computational results depend upon this or that technical detail of the implementation. It would clearly be capricious to base instruction on results which depend on programming style.

Although both of these objections to computer simulations are valid in principle, I believe that the particular computational results reported here are principled. The effects in Tables 2 through 4 are not caused by this or that exotic feature of the implementation of HS, but by the fact that the gap between principles and procedures in arithmetic is wide, much wider than the intuitions of mathematically literate people suggest. To support this claim, I will discuss three aspects of that gap: the role of spatio-temporal relations, the function of expediency in algorithm design, and the importance of attention allocation.

The role of spatio-temporal relations. Equality relations between quantities are timeless and without spatial interpretation. For example, the associative law

$$(a + b) + c = a + (b + c)$$

states that the sum of any two numbers x and c, where x is the sum of any two numbers a and b, is equal to the sum of the two numbers a and y, where y is the sum of b and c. The law does not say anything about spatial locations or directions. The fact that the law has a left-to-right linear structure is a property of the paper medium. If the law was encoded as a list-structure in a computer, the individual symbols might be distributed in a very different spatial pattern, but the law would have the same meaning. Neither does the law speak about temporal order. The addition operations mentioned in the law are not related through relations such as *before* and *after*; and concepts like *first, next,* and *last* have no role in the understanding of the law. The laws of the number system express equality relations abstracted from time and space.

The control of action, on the other hand, is all about spatio-temporal relations. The main function of an algorithm or a problem solving procedure is to order primitive actions in time, to regulate which action is to be done before or after which other action. Furthermore, the actions, to the extent that they are motor actions, have to be performed at some particular location in space, on some particular object. If a digit is to be crossed out, the spatial coordinates for that object must be known. If the right action is performed in the wrong spatial location, an error is likely to result. To learn a cognitive skill is to acquire a structure for the spatio-temporal control of action.

If the mathematical structure--the set of laws that constitute the rationale for a particular algorithm--ignores time and space, and if the cognitive skill involved in executing that algorithm is a structure for spatio-temporal organization, it follows that the mathematical structure does not fully determine the skill. One cannot derive that *this* action has to be performed before *that* action from mathematical laws which do not speak about temporal relations; one cannot direct an action to *this* spatial location rather than *that* with the help of laws which do not speak about space. Information about time and space has to be added to the mathematical principles in order to control action. Knowledge about the mathematical rationale for an algorithm is not sufficient for the construction of the algorithm.

The role of expediency in algorithm design. The belief that mathematical principles determine mathematical action ignores the role of expediency in the design of the place value algorithms. Why, for example, do we solve place value problems by processing the columns in order from lower to higher place values? There is no *mathematical* reason for this rule. It is equally correct to begin subtracting to the left, i. e., with the highest place value column, and work towards the right, i. e., towards columns with lower place values. Unlike the standard procedure, this alternate procedure, although mathematically correct, requires that the already processed columns have to be processed again every time the minuend is regrouped. Beginning with the lowest place value column saves work; it is a choice dictated by expediency, not by correctness. Indeed, there is no mathematical reason to regroup in the first place. It is possible to perform subtraction by processing each column independently of the others, recording negative results when appropriate, and then combining the column results into the final answer. The decision to regroup is dictated by economy considerations, not by mathematical principles.

The place value algorithms evolved over a long period of time as efficient means of performing calculations. The main reason to adhere to those algorithms is that they save work, as compared to other, equally correct procedures. But there is no relation between the mathematical theory of place value and the expediency of the algorithms that build on it. One cannot derive that *this* way of doing subtraction is more efficient than *that* way from the laws of the number system. The shape of these algorithms is not determined by the underlying mathematical principles, so understanding those principles contributes little to the learning of the algorithms. Any aspect of a procedure which is grounded in expediency rather than in mathematical concepts and relations will appear arbitrary and incomprehensible regardless how well the conceptual rationale for that procedure is understood.

School children cannot be aware of the expediency of the place value algorithms. In order to realize how economical they are, one must have something to compare them to. Since children are taught the efficient algorithms, they have no experience of *less* efficient ways of doing calculations. Also, since children are not doing calculations for a living, they have no interest in expediency.

The importance of attention allocation. One of the most robust findings of cognitive psychology is that there are severe limits on how much information can be kept in working memory at any one point in time. This limitation is simulated in the HS system by letting working memory elements decay as time passes. The main consequence of this limitation is that the control of attention is a central issue in all action, including mathematical action. If you cannot keep all information in the problem display in your head simultaneously, then you have to access it sequentially, by moving your eye over it in a carefully controlled manner. *To learn subtraction is to learn where to look.* Obviously, mathematical principles have nothing to say about this aspect of mathematical action. No matter how well one understands the concept of place value, one still has to figure out where to look at each moment during subtraction.

In summary, there are at least three principled reasons to believe in a wide derivational gap between mathematical principles and mathematical action. First, mathematical principles ignore questions of space and time, while a cognitive procedure is a structure for the spatio-temporal control of action. Second, mathematical principles ignore the cost of computing a result, while the standard place value algorithms are designed for maximum expediency. Children cannot understand those features of place value algorithms which are designed with expediency in mind, because they have no experience of the less expedient alternatives; and, unlike the professional calculators who developed the algorithms, children have no particular interest in economy. Third, the limited capacity of human working memory implies that all task information cannot be kept active at all times. Consequently, any cognitive skill must specify how attention is to be allocated over the task information. But mathematical principles have nothing to say about the allocation of attention.

If the gap between mathematical principles and mathematical action is as wide as the above discussion suggests, then how is an understanding of the mathematical concepts and principles underlying a particular algorithm supposed to facilitate the construction of the cognitive skill? This question has not been clearly answered by any current theory of mathematical cognition, and I suggest that no answer exists. The gap between mathematical knowledge and mathematical action *is* difficult to bridge; that is why it took two millennia to develop the place value algorithms, and that is why school children make mistakes even after they have grasped the rationale of an algorithm.

The fact that the derivational distance between mathematical principles and mathematical action is large does not in and of itself explain why the HS model needs to compute more in the case of a conceptual representation than in the case of a mechanical representation. Granted that the derivational distance is large, we still need an explanation for why it is *larger* in one case than in the other. The explanation is simple: There is more work involved in updating and processing a rich representation than an impoverished one. There are more relations to keep track of, and therefore more operations to perform. Each of those operations has to be controlled by some procedural rule; hence, there are more rules to learn, or more complicated conditions for the rules. The same must be true of humans; updating and maintaining a richer mental representation must require more cognitive work.

Because the gap between the mathematical principles and the mathematical procedures is so wide, I believe that any reasonable simulation model of knowledge-based acquisition of an arithmetic procedure will reproduce the results reported here. The reader who disbelieves this is urged to prove me wrong by developing a simulation model that can learn subtraction both conceptually and mechanically and which expands less computation in the former case than in the latter.

According to the results reported here, William Brownell could not have been more wrong. Regrouping is more difficult to learn than augmenting. In particular, regrouping in a conceptually rich representation is more difficult to learn than regrouping done mechanically, and the disadvantage of the conceptually rich representation as compared to the mechanical case is much larger for regrouping than for augmenting. These results directly contradict Brownell's conclusion that regrouping is easier than augmenting, particularly when taught conceptually [5, 6].

At first glance, this contradiction seems devastating for the model. After all, Brownell's conclusion was based on empirical observations, and in the case of a contradiction between theory and data, it is the theory that must go. However, unlike simulation studies, empirical studies cannot differentiate between learning and performance, between the amount of cognitive work needed to learn an algorithm and the amount of cognitive work needed to execute it, once learned. The only way to measure the cost of learning is to observe performance, so any empirical measure will necessarily confound the two. As the reader might recall, the simulation of performance in the first study reported in this chapter did produce results which fit the empirical data rather well. It is reasonable to interpret those data as measures of the cognitive cost of executing the algorithms rather than of the cognitive cost of learning them. We then have a good fit between the theory and *the data themselves*, but no support for Brownell's *interpretation* of the data.

The result that conceptual instruction requires more computational work than mechanical instruction is comforting to the researcher who desperately wants to know why well-intended, carefully planned and skillfully executed instructional interventions that aim to impart conceptual understanding do not succeed in producing correct performance [27, 33]. But it is less comforting to the educator or teacher who is responsible for designing efficient instruction. The simulation results imply that it is a mistake to expect conceptual understanding to *facilitate* procedural learning. Instead, the results indicate that conceptually based instruction will be more costly in terms of time and effort than mechanical instruction. The relation between the conceptual rationale of an arithmetic procedure and the procedure is an instructional topic in its own right, a topic, moreover, which is complicated and therefore requires time and effort on the part of both instructor and student. Instead of being a tool for teaching (the same old) arithmetic, *conceptually based instruction in arithmetic constitutes a higher pedagogical ambition*, as compared to mechanical instruction.

It is easy to feel sympathy with this higher ambition. We obviously want students to grasp the rationale behind the arithmetic algorithms. The present discussion is not meant to imply that conceptual instruction in arithmetic is wrong or undesirable. What is wrong is the expectation that such instruction can be digested easier and with less effort than mechanical instruction.

Conceptually based instruction in arithmetic might need to revisit the idea of a spiral curriculum [7, pp. 52-54]: Teach the algorithms with a small amount of conceptual interpretation at an early age; teach them again with a deeper presentation of the conceptual rationale when the students have aquired more mathematical knowledge; and so on. The topic could be visited as many as four our five times between third grade and college, each visit probing deeper into the conceptual rationale, until the students are able to carry out a relatively tight derivation of the algorithms (e. g., as in [25]). To the best of my knowledge, no large scale empirical evaluation of such a spiral curriculum for arithmetic has yet been done.

Evaluation of the General Method

The specific conclusions about arithmetic instruction presented in this chapter are controversial and unlikely to be accepted without a debate. Such a debate would be welcome. But the controversial nature of the *domain-specific conclusions* should not be allowed to obscure the fact that the present study also contributes a *general method* with a potentially greater impact.

The main method of traditional educational research is well exemplified by the studies conducted in order to choose between the regrouping and augmenting algorithms: To determine the relative advantage of an instructional design A as compared to an alternative design B, teach one set of students with design A and a second set of students with design B, and compare the outcomes. This empirical method is laborious and time consuming. In addition, it is rarely successful in settling the instructional issue at hand. Measures of instructional outcomes are so imprecise and coarse that a negative outcome is unconvincing. The opponents of the hypothesis favored by the author of such a study can always feel justified in questioning whether the measures used were sensitive enough to register even quite significant effects. On the other hand, a positive effect is equally unconvincing. An observed effect cannot be ascribed to the instructional intervention with any certainty, because it is almost impossible to achieve control over all the determinants of an instructional outcome. Empirical comparisons between alternative instructional designs carry little intellectual authority, regardless of outome.

Teachable simulation models enable an alternative method for investigating instructional questions. Instead of teaching the relevant instructional topic in different ways to different groups of students, we can teach it in different ways to a model of learning, if that model takes the form of a robust, runnable simulation. The simulation runs provide us with measures of the amount of computational work required to learn the target topic under different modes of instruction. A significantly lower value for mode A than for its rival B constitutes a prediction that A is the preferred way of teaching the target topic.

Using this method, an instructional designer can invent a new approach to a particular topic, use it to teach that topic to the model, and have a preliminary outcome, all in a matter of days. Preparing the inputs (the initial procedural knowledge and the instructions) to a teachable simulation model is not a trivial task, but it is measured in hours or days, rather than in months or years. Such rapid turnaround between an instructional idea and its evaluation has the potential to facilitate search through the space of instructional designs [30]. Many different designs can be tried and compared at a relatively low cost and in a relatively short time.

A teachable simulation model can also help identify fruitless questions and inappropriate techniques. Consider once again the large scale classroom studies of the pre-World War II era that attempted to settle the controversy between regrouping and augmenting empirically. My simulation results show that there is no reason to expect any differences between regrouping and augmenting on measures of *performance*. The two algorithms are nearly equal in cognitive complexity, once learned. Hence, trying to measure the difficulty of the two algorithms by measuring performance is not a useful endeavor. The differences between the algorithms only affect the amount of cognitive work required to *learn* the algorithms. But pure empirical measures of learning are hard to come by. One possibility is to count the number of learning events per unit time as revealed by think-aloud protocols, a measure hardly ever used in learning research (but see [40] for an exception). No such measure was employed in the pre-World War II studies that compared regrouping and augmenting. Those studies could not, in principle, resolve the issue they were addressing, because they were approaching it with the wrong tools. Theoretical clarification is a necessary prerequisite for meaningful data collection in instructional science as in other sciences. Implementing and using a teachable simulation model is one way to achieve such clarification.

A second traditional approach to instructional design, over and above empirical comparisons between alternative teaching methods, is to base particular decisions on general design principles, which, in turn, are derived in some more or less intuitive way from a learning theory. The debate about how to teach subtraction could conceivably be decided by the application of such a principle. For example, we could apply the principle of successive elaborations: A topic should be taught by first presenting a kernel idea, an epitome, which is then successively elaborated [32]. But this principle does not discriminate between the different ways of teaching subtraction. Both regrouping and augmenting can be taught by first presenting the basic idea of the algorithm, and then elaborating it. As a second example, consider the principle, proposed by Anderson, Boyle, Farrell, and Reiser, that one should teach the goal hierarchy of the target skill [2]. Once again, this principle does not discriminate between alternative subtraction algorithms. As a last example, a colleague of mine suggested that one should prefer regrouping over augmenting on the principle that teaching should facilitate *future* learning, and the regrouping operation is more generally useful than the augmenting operation. But it is unclear in what sense the law of associativity is more generally useful than the constant difference law; both seem equally necessary for continued study in mathematics. In short, the disadvantage of using general design principles as mediators between theories of learning and instructional designs is that the application of those design principles is seldom straightforward.

The method of teachable simulation models links learning theory to instructional design in a different way. The method brings learning theory to bear on particular issues, without mediation by general design principles. For example, the simulation runs presented in this chapter tell us that augmenting is easier to learn than regrouping and that the advantage of augmenting is increased with conceptually based instruction. The simulation runs resolve the particular issue of regrouping versus augmenting, but they do not suggest any principle of arithmetic instruction, let alone any general design principle. The principles of learning embedded in the model are applied directly to the instructional issue at hand. Whether this is, in general, a better way to proceed than via general design principles cannot be determined here. The two different ways of linking learning theory to instructional design are not incompatible. A mixture of both approaches will probably prove most advantageous.

Testing instructional designs by trying them out on a simulation model seems to presuppose that we have accurate simulation models. There are three answers to this objection. First, the lack of accuracy of today's models and theories is a temporary disadvantage. As research into human learning progresses, we will be able to construct more accurate theories. It is desirable to have a method which allows us to channel increased theoretical understanding into improved instructional designs. The dependence on the accuracy of our learning theory is not (only) a bug, it is (also) a feature. Second, the extent to which particular computational results depend upon the accuracy of the model is a matter for debate. In the preceeding section I argued that the results reported in this chapter are consequences of deep features of arithmetic, and hence relatively independent of the particulars of the HS model. (It is clear how to provide evidence for or against claims of this kind: A claim about independence of results from a particular model is supported if the results can be reproduced with a different model.) Third, a theory need not be entirely accurate to be useful. Even approximate theories can often supply information that improve upon common sense and rules of thumb.

Answering questions through theoretical calculations goes against the grain in a discipline that was shaped in the heydays of the peculiar brand of empiricism advocated by the logical positivists. It is therefore useful to look up from our local concerns and observe that the ratio of theoretical calculation to empirical observation tends to grow as scientific disciplines mature. Once upon a time, geometers measured angles in order to decide whether a triangle was a right triangle or not. By the time Euclid wrote his great treatise, geometry was already a purely theoretical discipline in which answers to questions are derived from first principles. Mechanics went through a similar development. Brahe and Galileo needed observations, but since the "rational mechanics" of the 19th century, questions like how much force it takes to lift a particular payload into orbit are answered by calculation, not by observation. If it were necessary to send up hundreds of rockets with different payloads and different thrusts in order to decide the issue empirically, space travel could never have gotten off the ground. In short, to observe is to confess ignorance; it is what scientists do when they have little or no theoretical understanding. As a science matures, calculations replace (some) empirical measurements. There is every reason to expect instructional science to develop similarly. The present chapter is but a small step in that direction.

Acknowledgements

The research reported in this chapter is supported by grant No. N00014-89-J-1681 from the Cognitive Science Program of the Office of Naval Research, and the institutional grant for The Center for the Study of Learning from the Office of Educational Research and Improvement, US Ministry of Education.

References

1. Anderson, J. R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.
2. Anderson, J. R., Boyle, C. F., Farrell, R., & Reiser, B. J. (1987). Cognitive principles in the design of computer tutors. In P. Morris, (Ed.), *Modelling Cognition.* New York, NY: Wiley.
3. Anzai, Y., & Simon, H. A. (1979) The theory of learning by doing. *Psychological Review, 86,* 124-140.
4. Brown, J. S., & Burton, R. R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science, 2,* 155-192.
5. Brownell, W. A. (1947). An experiment on "borrowing" in third-grade arithmetic. *Journal of Educational Research,* 41, 161-263.
6. Brownell, W. A. ,& Moser, H. E. (1949). *Meaningful vs. mechanical learning: A study in Grade III subtraction.* Durham, NC: Duke University Press.
7. Bruner, J. S. (1960). *The process of education.* New York, NY: Vintage Books.
8. Ernst, A. M., & Ohlsson, S. (1989). *The cognitive complexity of the regrouping and augmenting algorithms for subtraction: A theoretical analysis* (Technical Report No. KUL-89-06). Pittsburgh, PA: University of Pittsburgh.
9. Evertz, R. (1982). *A production system account of children's errors in fraction subtraction* (Technical Report No. CAL 28). Milton Keynes, UK: The Open University.
10. Glaser, R. (1976). Components of a theory of instruction: Toward a science of design. *Review of Educational Research, 46,* 1-24.
11. Glaser, R. (1978). The contributions of B. F. Skinner to education and some counterinfluences. In P. Suppes, (Ed.), *Impact of research on education: Some case studies.* Washington, D. C.: National Academy of Education.
12. Glaser, R. (1982). Instructional psychology: Past, present, and future. *American Psychologist, 37,* 292-305.
13. Greeno, J. G. (1980). Some examples of cognitive task analyses with instructional implications. In R. E. Snow, P.-A. Federico, & W. E. Montague, (Eds.), *Aptitude, learning, and instruction.*

Volume 2: Cognitive process analyses of learning and problem solving. Hillsdale, NJ: Erlbaum.

14. Haseman, K. (1985). Die Beschreibung von Schulerfehlern mit kognitionstheoretischen Modellen. *Der Mathematikunterricht, 31*, 6-15.
15. Hiebert, J., (Ed.), (1986). *Conceptual and procedural knowledge: The case of mathematics.* Hillsdale, NJ: Erlbaum.
16. Hollan, J. H., Holyoak, K. J., Nisbett, R. E., & Thagard, P. R. (1986). *Induction. Processes of inference, learning, and discovery.* Cambridge, MA: MIT Press.
17. Hunting, R. P. (1983). Alan: A case study of knowledge of units and performance with fractions. *Journal for Research in Mathematics Education, 14*, 182-197.
18. Klahr, D., Langley, P., & Neches, R., (Eds.), (1987). *Production system models of learning and development.* Cambridge, MA: MIT Press.
19. Laird, J., Rosenbloom, P., & Newell, A. (1986). *Universal subgoaling and chunking. The automatic generation and learning of goal hierarchies.* Boston, MA: Kluwer.
20. Langley, P. (1987). A general theory of discrimination learning. In Klahr, D., Langley, P., & Neches, R., (Eds.), (1987). *Production system models of learning and development.* Cambridge, MA: MIT Press.
21. McDermott, J., & Forgy, C. (1978). Production system conflict resolution strategies. In D. A. Waterman & F. Hayes-Roth, (Eds.), *Pattern-directed inference systems.* New York, NY: Academic Press.
22. Murray, F. B., Ames, G. J, & Botvin, G. J. (1977). Acquisition of conservation through cognitive dissonance. *Journal of Educational Psychology, 69,* 519-527.
23. Newell, A., & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, NJ: Prentice-Hall.
24. Ohlsson, S. (1987). Transfer of training in procedural learning: A matter of conjectures and refutations? In L. Bolc, (Ed.), *Computational models of learning.* Berlin, West Germany: Springer-Verlag.

25. Ohlsson, S. (1988). *The conceptual basis for of subtraction with regrouping: A mathematical analysis* (Technical Report No. KUL-02-88). Pittsburgh, PA: University of Pittsburgh.
26. Ohlsson, S. (1988). Computer simulation and its impact on educational research and practice. *International Journal of Educational Research, 12*, 5-34.
27. Ohlsson, S., Bee, N. V., & Zeller, P. A. (1989). *Empirical evaluation of a computer-based environment for fractions* (Technical Report No. KUL-89-07). Pittsburgh, PA: University of Pittsburgh.
28. Ohlsson, S. & Rees, E. (in press). Adaptive search through constraint violations. *Journal of Experimental and Theoretical Artificial Intelligence.*
29. Ohlsson, S. & Rees, E. (in press). The function of conceptual understanding in the learning of arithmetic procedures. *Cognition & Instruction.*
30. Pirolli, P. L., & Greeno, J. G. (1988). The problem space of instructional design. In J. Psotka, L. D. Massey, & S. A. Mutter, (Eds.), *Intelligent tutoring systems. Lessons learned.* Hillsdale, NJ: Erlbaum.
31. Post, T. R., Wachsmuth, I., Lesh, R., & Behr, M. J. (1985). Order and equivalence of rational numbers: A cognitive analysis. *Journal for Research in Mathematics Education, 16*, 18-36.
32. Reigeluth, C, M., & Stein, F. S. (1983). The elaboration theory of instruction. In C. M. Reigeluth, (Ed.), *Instructional-design theories and models: An overview of their current status.* Hillsdale, NJ: Erlbaum.
33. Resnick, L. B., & Omanson, S. F. (1987). Learning to understand arithmetic. In R. Glaser, (Ed.), *Advances in instructional psychology* (Vol 3, pp. 41-95). Hillsdale, NJ: Erlbaum.
34. Scandura, J. M. (1977). *Problem solving. A structural/process approach with instructional implications.* New York, NY: Academic Press.
35. Shuell, T. J. (1986). Cognitive conceptions of learning. *Review of Educational Research, 56*, 411-436.
36. Smith, J. (1990). *Learning rational numbers.* Unpublished doctorial dissertation, School of Education, University of California at Berkeley.

37. Tatsuoka, K. K. (1984). *Analysis of errors in fraction addition and subtraction problems* (Technical Report January 1984). Urbana-Champaign, IL: University of Illinois.

38. VanLehn, K. (1982). Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills. *Journal of Mathematical Behavior, 3*, 3-71.

39. VanLehn, K. (1990). *Mind bugs: The origins of procedural misconceptions.* Cambridge, MA: MIT Press.

40. VanLehn, K. (in press). Rule acquisition events in the discovery of problem solving strategies. *Cognitive Science.*

41. Young, R. M., & O'Shea, T. (1981). Errors in children's subtraction. *Cognitive Science, 5*, 153-177.

Kul Reports

1985

Ohlsson, S., & Langley, P. (April, 1985). *Psychological evaluation of path hypotheses in cognitive diagnosis* (Technical Report No. 1985/2). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

1986

Ohlsson, S. (January, 1986). *Some principles of intelligent tutoring* (Technical Report No. 1986/2). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (June, 1986). *Computer simulation and its impact on educational research and practice* (Technical Report No. 1986/14). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson S. (October, 1986). *Sense and reference in the design of interactive illustrations for rational numbers* (Technical Report No. 1986/18). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

1987

Ohlsson, S. (April, 1987). *A semantics for fraction concepts* (Technical Report No. KUL-87-01). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (September, 1987). *Trace analysis and spatial reasoning: An example of intensive cognitive diagnosis and its implications for testing* (Technical Report No. KUL-87-02). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., Nickolas, S., & Bee, N.V. (December, 1987). *Interactive illustrations for fractions: A progress report* (Technical Report No. KUL-87-03). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., & Rees, E. (December, 1987.) *Rational learning: Deriving arithmetic procedures from state constraints* (Technical Report No. KUL-87-04). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

<u>1988</u>

Ohlsson, S. (February, 1988). *Mathematical meaning and applicational meaning in the semantics for fractions and related concepts* (Technical Report No. KUL-88-01). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (March, 1988). *Principled understanding of subtraction with regrouping: A mathematical analysis* (Technical Report No. KUL-88-02). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., & Rees, E. (August, 1988). *An information processing analysis of conceptual understanding in the learning of arithmetic procedures* (Technical Report No. KUL-88-03). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., (December, 1988). *Towards intelligent tutoring systems that teach knowledge rather than skills: Five research questions* (Technical Report No. KUL-88-04). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

<u>1989</u>

Ohlsson, S. (January, 1989). *Knowledge requirements for teaching: The case of fractions* (Technical Report No. KUL-89-01). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (April, 1989). *Cognitive science and instruction: Why the revolution is not here (yet)* (Technical Report No. KUL-89-02). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Robin, N., & Ohlsson, S. (August, 1989). *Impetus then and now: A detailed comparison between Jean Buridan and a single* *contemporary subject* (Technical Report No. KUL-89-03). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., (Ed.), (October, 1989). *Aspects of cognitive conflict and cognitive change* (Technical Report No. KUL-89-04). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Leinhardt, G., & Ohlsson, S. (November, 1989). *Tutorials on the structure of tutoring from teachers* (Technical Report No. KUL-89-05). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ernst, A., & Ohlsson, S. (December 1989). *The cognitive complexity of the regrouping and augmenting algorithms for subtraction: A theoretical analysis* (Technical Report No. KUL-89-06). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., Bee, N.V., & Zeller, P.A. (December, 1989). *Empirical evaluation of a computer-based learning environment for fractions* (Technical Report No. KUL-89-07). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

1990

Ohlsson, S., & Rees, E. (January, 1990). *Adaptive search through constraint violations.* (Technical Report No. KUL-90-01). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., & Hall, N. (February, 1990). *The cognitive function of embodiments in mathematics instruction* (Technical Report No. KUL-90-02). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., & Rees, E. (March, 1990). *Comparative evaluation of knowledge-based simulation models of procedural learning* (Technical Report No. KUL-90-03). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (May, 1990). *The mechanism of restructuring in geometry* (Technical Report No. KUL-90-04). Pittsburgh, PA:

Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (September, 1990). *Artificial instruction. A method for relating learning theory to instructional design* (Tachnical Report No. KUL-90-05). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S. (December, 1990). *The cognitive skill of theory articulation: A neglected aspect of science education?* (Technical Report No. KUL-90-06). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

<u>1991</u>

Ohlsson, S. (January, 1991). *Instructional theory versus system hacking. An essay on the standards of good research in AI and Education* (Technical Report No. KUL-91-01). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Ohlsson, S., & Bee, N. (February, 1991). *Strategy variability: A challenge to models of procedural learning* (Technical Report No. KUL-91-02). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Hall, N, & Ohlsson, S. (February, 1991). *A procedural-analogy theory of concrete illustrations in arithmetic learning* (Technical Report No. KUL-91-03). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Distribution List

Ms. Lisa B. Achille
Code 5530
Naval Research Lab
Overlook Drive
Washington, DC 20375-5000

Dr. Edith Ackermann
Media Laboratory
E15-311
20 Ames Street
Cambridge, MA 02139

Dr. Beth Adelson
Department of Computer Science
Tufts University
Medford, MA 02155

Technical Document Center
AFHRL/LRS-TDC
Wright-Patterson AFB
OH 45433-6503

Dr. Robert Ahlers
Code N711
Human Factors Laboratory
Naval Training Systems Center
Orlando, FL 32813

Dr. Robert M. Aiken
Computer Science Department
038-24
Temple University
Philadelphia, PA 19122

Mr. Tejwansh S. Anand
Philips Laboratories
345 Scarborough Road
Briarcliff Manor
New York, NY 10520

Dr. James Anderson
Brown University
Department of Psychology
Providence, RI 02912

Dr. John R. Anderson
Department of Psychology
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Dr. Thomas H. Anderson
Center for the Study of Reading
174 Children's Research Center
51 Gerty Drive
Champaign, IL 61820

Dr. Stephen J. Andriole, Chairman
Department of Information Systems
and Systems Engineering
George Mason University
4400 University Drive
Fairfax, VA 22030

Prof. John Annett
University of Warwick
Department of Psychology
Coventry CV4 7AL
ENGLAND

Edward Atkins
Code 61Z1210
Naval Sea Systems Command
Washington, DC 20362-5101

Dr. Patricia Baggett
School of Education
610 E. University, Rm 1302D
University of Michigan
Ann Arbor, MI 48109-1259

Dr. James D. Baker
Director of Automation and Research
Allen Corporation of America
209 Madison Street
Alexandria, VA 22314

Dr. Meryl S. Baker
Navy Personnel R&D Center
San Diego, CA 92152-6800

prof. dott. Bruno G. Bara
Unita di ricerca di
intelligenza artificiale
Universita di Milano
20122 Milano - via F. Sforza 23
ITALY

Dr. Jonathan Baron
80 Glenn Avenue
Berwyn, PA 19312

Dr. Gautam Biswas
Department of Computer Science
Box 1688, Station B
Vanderbilt University
Nashville, TN 37235

Dr. John Black
Teachers College, Box 8
Columbia University
525 West 120th Street
New York, NY 10027

Dr. Michael Blackburn
Code 943
Naval Ocean Systems Center
San Diego, CA 92152-5000

Dr. Arthur S. Blaiwes
Code N712
Naval Training Systems Center
Orlando, FL 32813-7100

Dr. Deborah A. Boehm-Davis
Department of Psychology
George Mason University
4400 University Drive
Fairfax, VA 22030

Dr. Sue Bogner
Army Research Institute
ATTN: PERI-SF
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Dr. Jeff Bonar
Guidance Technology, Inc.
800 Vinial Street
Pittsburgh, PA 1'212

Dr. J. C. Boudreaux
Center for Manufacturing
Engineering
National Bureau of Standards
Gaithersburg, MD 20899

Dr. Lyle E. Bourne, Jr.
Department of Psychology
Box 345
University of Colorado
Boulder, CO 80309

Dr. Hugh Burns
Department of English
University of Texas
Austin, TX 78703

Dr. Robert Calfee
School of Education
Stanford University
Stanford, CA 94305

Dr. Joseph C. Campione
Center for the Study of Reading
University of I''inois
51 Gerty Drive
Champaign, IL 61820

Dr. Joanne Capper, Director
Center for Research into Practice
3545 Albemarle Street, NW
Washington, DC 20008

Dr. Jaime G. Carbonell
Computer Science Department
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Dr. Gail Carpenter
Center for Adaptive Systems
111 Cummington St., Room 244
Boston University
Boston, MA 02215

Dr. John M. Carroll
IBM Watson Research Center
User Interface Institute
P.O. Box 704
Yorktown Heights, NY 10598

Dr. Ruth W. Chabay
CDEC, Hamburg Hall
Carnegie Mellon University
Pittsburgh, PA 15213

Dr. Fred Chang
Pacific Bell
2600 Camino Ramon
Room 3S-450
San Ramon, 'A 94583

Dr. Davida Charney
English Department
Penn State University
University Park, PA 16802

Mrs. Ola Clarke
818 South George Mason Drive
Arlington, VA 22204

Dr. Norman Cliff
Department of Psychology
Univ. of So. California
Los Angeles, CA 90089-1061

Dr. Stanley Collyer
Office of Naval Technology
Code 222
800 N. Quincy Street
Arlington, VA 22217-5000

Dr. Jere Confrey
Cornell University
Dept. of Education
Room 490 Roberts
Ithaca, NY 14853

Dr. Lynn A. Cooper
Department of Psychology
Columbia University
New York, NY 10027

Dr. Meredith P. Crawford
3563 Hamlet Place
Chevy Chase, MD 20815

Dr. Hans F. Crombag
Faculty of Law
University of Limburg
P.O. Box 616
Maastricht
The NETHERLANDS 6200 MD

Dr. Kenneth B. Cross
Anacapa Sciences, Inc.
P.O. Drawer Q
Santa Barbara, CA 93102

Dr. Cary Czichon
Intelligent Instructional Systems
Texas Instruments AI Lab
P.O. Box 660246
Dallas, TX 75266

Brian Dallman
Training Technology Branch
3400 TCHTW/TTGXC
Lowry AFB, CO 80230-5000

Mr. John F. Dalphin
Chair, Computer Science Dept.
Towson State University
Baltimore, MD 21204

Margaret Day, Librarian
Applied Science Associates
P.O. Box 1072
Butler, PA 16003

Geory Delacote
Directeur de L'Informatique
Scientifique et Technique
CNRS
15, Quai Anatole France
75700 Paris, FRANCE

Dr. Denise Dellarosa
Psychology Department
Box 11A, Yale Station
Yale University
New Haven, CT 06520-7447

Dr. Sharon Derry
Florida State University
Department of Psychology
Tallahassee, FL 32306

Dr. Thomas E. DeZern
Project Engineer, AI
General Dynamics
PO Box 748/Mail Zone 2646
Fort Worth, TX 76101

Dr. Ronna Dillon
Department of Guidance and
Educational Psychology
Southern Illinois University
Carbondale, IL 62901

Dr. J. Stuart Donn
Faculty of Education
University of British Columbia
2125 Main Mall
Vancouver, BC CANADA V6T 1Z5

Defense Technical
Information Center
Cameron Station, Bldg 5
Alexandria, VA 22314
(2 Copies)

Dr. Pierre Duguet
Organization for Economic
Cooperation and Development
2, rue Andre-Pascal
75016 PARIS
FRANCE

Dr. Ralph Dusek
V-P Human Factors
JIL Systems
1225 Jefferson Davis Hwy.
Suite 1209
Arlington, VA 22201

Dr. John Ellis
Navy Personnel R&D Center
Code 51
San Diego, CA 92252

Dr. Susan Epstein
144 S. Mountain Avenue
Montclair, NJ 07042

ERIC Facility-Acquisitions
2440 Research Blvd, Suite 550
Rockville, MD 20850-3238

Dr. K. Anders Ericsson
University of Colorado
Department of Psychology
Campus Box 345
Boulder, CO 80309-0345

Dr. Debra Evans
Applied Science Associates, Inc.
P. O. Box 1072
Butler, PA 16003

Dr. Lorraine D. Eyde
Office of Personnel Management
Office of Examination Development
1900 E St., NW
Washington, DC 20415

Dr. Jean-Claude Falmagne
Irvine Research Unit in
Mathematical & Behavioral Sciences
University of California
Irvine, CA 92717

Dr. Beatrice J. Farr
Army Research Institute
PERI-IC
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Marshall J. Farr, Consultant
Cognitive & Instructional Sciences
2520 North Vernon Street
Arlington, VA 22207

Dr. P-A. Federico
Code 51
NPRDC
San Diego, CA 92152-6800

Dr. Jerome A. Feldman
University of Rochester
Computer Science Department
Rochester, NY 14627

Dr. Paul Feltovich
Southern Illinois University
School of Medicine
Medical Education Department
P.O. Box 3926
Springfield, IL 62708

Dr. Elizabeth Fennema
Curriculum and Instruction
University of Wisconsin
225 North Mills Street
Madison, WI 53706

CAPT J. Finelli
Commandant (G-PTE)
U.S. Coast Guard
2100 Second St., S.W.
Washington, DC 20593

Prof. Donald Fitzgerald
University of New England
Department of Psychology
Armidale, New South Wales 2351
AUSTRALIA

Dr. Michael Flaningam
Code 52
NPRDC
San Diego, CA 92152-6800

Dr. J. D. Fletcher
Institute for Defense Analyses
1801 N. Beauregard St.
Alexandria, VA 22311

Dr. Kenneth D. Forbus
University of Illinois
Department of Computer Science
1304 West Springfield Avenue
Urbana, IL 61801

Dr. Barbara A. Fox
University of Colorado
Department of Linguistics
Boulder, CO 80309

Dr. Carl H. Frederiksen
Dept. of Educational Psychology
McGill University
3700 McTavish Street
Montreal, Quebec
CANADA H3A 1Y2

Dr. John R. Frederiksen
BBN Laboratories
10 Moulton Street
Cambridge, MA 02238

Dr. Norman Frederiksen
Educational Testing Service
(05-R)
Princeton, NJ 08541

Department of Humanities and
Social Sciences
Harvey Mudd College
Claremont, CA 91711

Dr. Alfred R. Fregly
AFOSR/NL, Bldg. 410
Bolling AFB, DC 20332-6448

Dr. Alinda Friedman
Department of Psychology
University of Alberta
Edmonton, Alberta
CANADA T6G 2E9

Dr. Michael Friendly
Psychology Department
York University
Toronto ONT
CANADA M3J 1P3

Col. Dr. Ernst Frise
Heerespsychologischer Dienst
Maria Theresien-Kaserne
1130 Wien
AUSTRIA

Dr. Robert M. Gagne
1456 Mitchell Avenue
Tallahassee, FL 32303

Dr. C. Lee Giles
AFOSR/NE, Bldg. 410
Bolling AFB
Washington, DC 20332

Dr. Philip Gillis
ARI-Fort Gordon
ATTN: PERI-ICD
Fort Gordon, GA 30905

Mr. Lee Gladwin
305 Davis Avenue
Leesburg, VA 22075

Dr. Robert Glaser
Learning Research
& Development Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15260

Dr. Marvin D. Glock
101 Homestead Terrace
Ithaca, NY 14856

Dr. Dwight J. Goehring
ARI Field Unit
P.O. Box 5787
Presidio of Monterey, CA 93944-5011

Dr. Joseph Goguen
Computer Science Laboratory
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

Mr. Richard Golden
Psychology Department
Stanford University
Stanford, CA 94305

Mr. Harold Goldstein
University of DC
Department Civil Engineering
Bldg. 42, Room 112
4200 Connecticut Avenue, N.W.
Washington, DC 20008

Dr. Sherrie Gott
AFHRL/MOMJ
Brooks AFB, TX 78235-5601

Dr. T. Govindaraj
Georgia Institute of
Technology
School of Industrial
and Systems Engineering
Atlanta, GA 30332-0205

Dr. Wayne Gray
Artificial Intelligence Laboratory
NYNEX
500 Westchester Avenue
White Plains, NY 10604

H. William Greenup
Dep Asst C/S, Instructional
Management (E03A)
Education Center, MCCDC
Quantico, VA 22134-5050

Dr. Dik Gregory
Admiralty Research
Establishment/AXB
Queens Road
Teddington
Middlesex, ENGLAND TW110LN

Dr. Stephen Grossberg
Center for Adaptive Systems
Room 244
111 Cummington Street
Boston University
Boston, MA 02215

Michael Habon
DORNIER GMBH
P.O. Box 1420
D-7990 Friedrichshafen 1
WEST GERMANY

Dr. Henry M. Halff
Halff Resources, Inc.
4918 33rd Road, North
Arlington, VA 22207

Mr. H. Hamburger
Department of Computer Science
George Mason University
Fairfax, VA 22030

Dr. Bruce W. Hamill
Research Center
The Johns Hopkins University
Applied Physics Laboratory
Johns Hopkins Road
Laurel, MD 20707

Dr. Patrick R. Harrison
Computer Science Department
U.S. Naval Academy
Annapolis, MD 21402-5002

Janice Hart
Office of the Chief
of Naval Operations
OP-111J2
Department of the Navy
Washington, D.C. 20350-2000

Dr. Wayne Harvey
Center for Learning Technology
Education Development Center
55 Chapel Street
Newton, MA 02160

Dr. Barbara Hayes-Roth
Knowledge Systems Laboratory
Stanford University
701 Welch Road
Palo Alto, CA 94304

Dr. Frederick Hayes-Roth
Teknowledge
P.O. Box 10119
1850 Embarcadero Rd.
Palo Alto, CA 94303

Dr. James Hendler
Dept. of Computer Science
University of Maryland
College Park, MD 20742

Dr. James Hiebert
Department of Educational
Development
University of Delaware
Newark, DE 19716

Dr. Geoffrey Hinton
Computer Science Department
University of Toronto
Sandford Fleming Building
10 King's College Road
Toronto, Ontario M5S 1A4 CANADA

Dr. James E. Hoffman
Department of Psychology
University of Delaware
Newark, DE 19711

Dr. Keith Holyoak
Department of Psychology
University of California
Los Angeles, CA 90024

Ms. Julia S. Hough
Cambridge University Press
40 West 20th Street
New York, NY 10011

Dr. William Howell
Chief Scientist
AFHRL/CA
Brooks AFB, TX 78235-5601

Dr. Steven Hunka
3-104 Educ. N.
University of Alberta
Edmonton, Alberta
CANADA T6G 2G5

Dr. Jack Hunter
2122 Coolidge Street
Lansing, MI 48906

Dr. Bonnie E. John
Wean Hall 8124
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Dr. Daniel B. Jones
U.S. Nuclear Regulatory
Commission
NRR/ILRB
Washington, DC 20555

Mr. Paul L. Jones
Research Division
Chief of Naval Technical Training
Building East-1
Naval Air Station Memphis
Millington, TN 38054-5056

Mr. Roland Jones
Mitre Corp., K-203
Burlington Road
Bedford, MA 01730

Dr. Marcel Just
Carnegie-Mellon University
Department of Psychology
Schenley Park
Pittsburgh, PA 15213

Dr. Ruth Kanfer
University of Minnesota
Department of Psychology
Elliott Hall
75 E. River Road
Minneapolis, MN 55455

Dr. Michael Kaplan
Office of Basic Research
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Dr. A. Karmiloff-Smith
MRC-CDU
17 Gordon Street
London
ENGLAND WC1H 0AH

Dr. Milton S. Katz
European Science Coordination
Office
U.S. Army Research Institute
Box 65
FPO New York 09510-1500

Dr. Frank Keil
Department of Psychology
228 Uris Hall
Cornell University
Ithaca, NY 14850

Dr. Wendy Kellogg
IBM T. J. Watson Research Ctr.
P.O. Box 704
Yorktown Heights, NY 10598

Dr. Douglas Kelly
University of North Carolina
Department of Statistics
Chapel Hill, NC 27514

Dr. David Kieras
Technical Communication Program
TIDAL Bldg., 2360 Bonisteel Blvd.
University of Michigan
Ann Arbor, MI 48109-2108

Dr. Thomas Killion
AFHRL/OT
Williams AFB, AZ 85240-6457

Dr. Jeremy Kilpatrick
Department of
Mathematics Education
105 Aderhold Hall
University of Georgia
Athens, GA 30602

Dr. J. Peter Kincaid
Army Research Institute
Orlando Field Unit
c/o PM TRADE-E
Orlando, FL 32813

Dr. Walter Kintsch
Department of Psychology
University of Colorado
Boulder, CO 80309-0345

Dr. Alex Kirlik
Georgia Institute of
Technology
Center for Human-Machine
Systems Research
Atlanta, GA 30332-0205

Dr. Janet L. Kolodner
Georgia Institute of Technology
School of Information
& Computer Science
Atlanta, GA 30332

Dr. Stephen Kosslyn
Harvard University
1236 William James Hall
33 Kirkland St.
Cambridge, MA 02138

Dr. Kenneth Kotovsky
Community College of
Allegheny County
808 Ridge Avenue
Pittsburgh, PA 15212

Dr. Keith Kramer
HCI Lab, Code 5530
Naval Research Laboratory
4445 Overlook Avenue
Washington, DC 20375-5000

Dr. Gary Kress
628 Spazier Avenue
Pacific Grove, CA 93950

Dr. Lois-Ann Kuntz
3010 S.W. 23rd Terrace
Apt. No. 105
Gainesville, FL 32608

Dr. David R. Lambert
Naval Ocean Systems Center
Code 772
271 Catalina Boulevard
San Diego, CA 92152-5000

Dr. Pat Langley
NASA Ames Research Ctr.
Moffett Field, CA 94035

Dr. Robert W. Lawler
Matthews 118
Purdue University
West Lafayette, IN 47907

Dr. Eugene Lee
Naval Postgraduate School
Monterey, CA 93943-5026

Dr. Yuh-Jeng Lee
Department of Computer Science
Code 52Le
Naval Postgraduate School
Monterey, CA 93943

Dr. Jill F. Lehman
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Dr. Jim Levin
Department of
Educational Psychology
210 Education Building
1310 South Sixth Street
Champaign, IL 61820-6990

Dr. John Levine
Learning R&D Center
University of Pittsburgh
Pittsburgh, PA 15260

Matt Lewis
Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

Dr. Doris K. Lidtke
Software Productivity Consortium
1880 Campus Commons Drive, North
Reston, VA 22091

Dr. Marcia C. Linn
Graduate School
of Education, EMST
Tolman Hall
University of California
Berkeley, CA 94720

Dr. Robert Lloyd
Dept. of Geography
University of South Carolina
Columbia, SC 29208

Dr. Jack Lochhead
University of
Massachusetts
Physics Department
Amherst, MA 01003

Vern M. Malec
NPRDC, Code 52
San Diego, CA 92152-6800

Dr. William L. Maloy
Code 04
NETPMSA
Pensacola, FL 32509-5000

Dr. Mary Marlino
Director, Educational Technology
HQ USAFA/DFTB
USAF Academy, CO 80840-5000

Dr. Sandra P. Marshall
Dept. of Psychology
San Diego State University
San Diego, CA 92182

Dr. John H. Mason
Centre for Maths Education
Mathematics Faculty
Open University
Milton Keynes MK7 6AA
UNITED KINGDOM

Dr. Manton M. Matthews
Department of Computer Science
University of South Carolina
Columbia, SC 29208

Dr. Richard E. Mayer
Department of Psychology
University of California
Santa Barbara, CA 93106

Dr. David J. McGuinness
Gallaudet University
800 Florida Avenue, N.E.
Washington, DC 20002

Dr. Joseph C. McLachlan
Code 52
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr. Douglas L. Medin
Department of Psychology
University of Michigan
Ann Arbor, MI 48109

Mr. Stig Meincke
Forsvarets Center for Lederskab
Christianshavns Voldgade 8
1424 Kobenhavn K
DENMARK

Dr. Arthur Melmed
Computer Arts and
Education Laboratory
New York University
719 Broadway, 12th floor
New York, NY 10003

Dr. Jose Mestre
Department of Physics
Hasbrouck Laboratory
University of Massachusetts
Amherst, MA 01003

Dr. D. Michie
The Turing Institute
George House
36 North Hanover Street
Glasgow G1 2AD
UNITED KINGDOM

Dr. Vittorio Midoro
CNR-Istituto Tecnologie Didattiche
Via All'Opera Pia 11
GENOVA-ITALIA 16145

Dr. James R. Miller
MCC
3500 W. Balcones Center Dr.
Austin, TX 78759

Dr. Jason Millman
Department of Education
Roberts Hall
Cornell University
Ithaca, NY 14853

Dr. Christine M. Mitchell
School of Indus. and Sys. Eng.
Center for Man-Machine
Systems Research
Georgia Institute of Technology
Atlanta, GA 30332-0205

Dr. Andrew R. Molnar
Applic. of Advanced Technology
Science and Engr. Education
National Science Foundation
Washington, DC 20550

Dr. William Montague
NPRDC Code 13
San Diego, CA 92152-6800

Dr. Melvin D. Montemerlo
NASA Headquarters
Code RC
Washington, DC 20546

Prof. John Morton
MRC Cognitive
Development Unit
17 Gordon Street
London WC1H OAH
UNITED KINGDOM

Dr. Allen Munro
Behavioral Technology
Laboratories - USC
250 N. Harbor Dr., Suite 309
Redondo Beach, CA 90277

Dr. William R. Murray
FMC Corporation
Central Engineering Labs
1205 Coleman Avenue
Box 580
Santa Clara, CA 95052

Chair, Department of Weapons and
Systems Engineering
U.S. Naval Academy
Annapolis, MD 21402

Dr. T. Niblett
The Turing Institute
George House
36 North Hanover Street
Glasgow G1 2AD
UNITED KINGDOM

Library, NPRDC
Code P201L
San Diego, CA 92152-6800

Librarian
Naval Center for Applied Research
in Artificial Intelligence
Naval Research Laboratory
Code 5510
Washington, DC 20375-5000

Dr. Harold F. O'Neil, Jr.
School of Education - WPH 801
Department of Educational
Psychology & Technology
University of Southern California
Los Angeles, CA 90089-0031

Dr. Paul O'Rorke
Information & Computer Science
University of California, Irvine
Irvine, CA 92717

Dr. Stellan Ohlsson
Learning R & D Center
University of Pittsburgh
Pittsburgh, PA 15260

Dr. James B. Olsen
WICAT Systems
1875 South State Street
Orem, UT 84058

Dr. Gary M. Olson
Cognitive Science and
Machine Intelligence Lab.
University of Michigan
701 Tappan Street
Ann Arbor, MI 48109-1234

Dr. Judith Reitman Olson
Graduate School of Business
University of Michigan
Ann Arbor, MI 48109-1234

Office of Naval Research,
Code 1142CS
800 N. Quincy Street
Arlington, VA 22217-5000
(6 Copies)

Dr. Judith Orasanu
Basic Research Office
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Jesse Orlansky
Institute for Defense Analyses
1801 N. Beauregard St.
Alexandria, VA 22311

Dr. Everett Palmer
Mail Stop 239-3
NASA-Ames Research Center
Moffett Field, CA 94035

Dr. Okchoon Park
Army Research Institute
PERI-2
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Roy Pea
Institute for Research
on Learning
2550 Hanover Street
Palo Alto, CA 94304

Dr. David N. Perkins
Project Zero
Harvard Graduate School
of Education
7 Appian Way
Cambridge, MA 02138

Dr. C. Perrino, Chair
Dept. of Psychology
Morgan State University
Cold Spring La.-Hillen Rd.
Baltimore, MD 21239

Dr. Nancy N. Perry
Naval Education and Training
Program Support Activity
Code-047
Building 2435
Pensacola, FL 32509-5000

Dept. of Administrative Sciences
Code 54
Naval Postgraduate School
Monterey, CA 93943-5026

Dr. Peter Pirolli
School of Education
University of California
Berkeley, CA 94720

Prof. Tomaso Poggio
Massachusetts Institute
of Technology E25-201
Center for Biological
Information Processing
Cambridge, MA 02139

Dr. Peter Polson
University of Colorado
Department of Psychology
Boulder, CO 80309-0345

Dr. Steven E. Poltrock
Boeing Advanced Technology Center
PO Box 24346 m/s 7L-64
Seattle, WA 98124

Dr. Joseph Psotka
ATTN: PERI-IC
Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333-5600

Mr. Paul S. Rau
Code U-33
Naval Surface Weapons Center
White Oak Laboratory
Silver Spring, MD 20903

Dr. James A. Reggia
University of Maryland
School of Medicine
Department of Neurology
22 South Greene Street
Baltimore, MD 21201

Dr. J. Wesley Regian
AFHRL/IDI
Brooks AFB, TX 78235

Dr. Fred Reif
Physics Department
University of California
Berkeley, CA 94720

Dr. Charles M. Reigeluth
330 Huntington Hall
Syracuse University
Syracuse, NY 13244

Dr. Daniel Reisberg
Reed College
Department of Psychology
Portland, OR 97202

Dr. Lauren Resnick
Learning R & D Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

Dr. J. Jeffrey Richardson
Center for Applied AI
College of Business
University of Colorado
Boulder, CO 80309-0419

Dr. Edwina L. Rissland
Dept. of Computer and
Information Science
University of Massachusetts
Amherst, MA 01003

Mr. William A. Rizzo
Code 71
Naval Training Systems Center
Orlando, FL 32813

Dr. Linda G. Roberts
Science, Education, and
Transportation Program
Office of Technology Assessment
Congress of the United States
Washington, DC 20510

Dr. Ernst Z. Rothkopf
AT&T Bell Laboratories
Room 2D-456
600 Mountain Avenue
Murray Hill, NJ 07974

Dr. Alan H. Schoenfeld
University of California
Department of Education
Berkeley, CA 94720

Lowell Schoer
Psychological & Quantitative
Foundations
College of Education
University of Iowa
Iowa City, IA 52242

Dr. Janet W. Schofield
816 LRDC Building
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15260

Dr. Kay Schulze
Computer Science Dept
U.S. Naval Academy
Annapolis, MD 21402-5018

Dr. Miriam Schustack
Code 52
Navy Personnel R & D Center
San Diego, CA 92152-6800

Dr. Judith W. Segal
OERI
555 New Jersey Ave., NW
Washington, DC 20208

Dr. Robert J. Seidel
US Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

Dr. Colleen M. Seifert
Institute for Cognitive Science
Mail Code C-015
University of California, San Diego
La Jolla, CA 92093

Dr. Michael G. S[illegible]fto
NASA Ames R[illegible]search Ctr.
Mail Stop 239-1
Moffett Field, CA 94035

Mr. Colin Sheppard
AXC2 Block 3
Admiralty Research Establishment
Ministry of Defence Portsdown
Portsmouth Hants P064AA
UNITED KINGDOM

Dr. Lee S. Shulman
School of Education
507 Ceras
Stanford University
Stanford, CA 94305-3084

Dr. Randall Shumaker
Naval Research Laboratory
Code 5510
4555 Overlook Avenue, S.W.
Washington, DC 20375-5000

Dr. Edward Silver
LRDC
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15260

Dr. Herbert A. Simon
Department of Psychology
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Robert L. Simpson, Jr.
DARPA/ISTO
1400 Wilson Blvd.
Arlington, VA 22209-2308

Dr. Zita M. Simutis
Chief, Technologies for Skill
  Acquisition and Retention
ARI
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Derek Sleeman
Computing Science Department
The University
Aberdeen AB9 2FX
Scotland
UNITED KINGDOM

Ms. Gail K. Slemon
LOGICON, Inc.
P.O. Box 85158
San Diego, CA 92138-5158

Dr. Edward E. Smith
Department of Psychology
University of Michigan
330 Packard Road
Ann Arbor, MI 48103

Dr. Alfred F. Smode
Code 7A
Research and Development Dept.
Naval Training Systems Center
Orlando, FL 32813-7100

Dr. Elliot Soloway
Yale University
Computer Science Department
P.O. Box 2158
New Haven, CT 06520

Linda B. Sorisio
IBM-Los Angeles Scientific Center
1600 Wilshire Blvd., 4th Floor
Los Angeles, CA 90025

N. S. Sridharan
FMC Corporation
Box 580
1205 Coleman Avenue
Santa Clara, CA 95052

Dr. Marian Stearns
SRI International
333 Ravenswood Ave.
Room B-5124
Menlo Park, CA 94025

Dr. Friedrich W. Steege
Bundesministerium
  der Verteidigung
Postfach 1328
D-5300 Bonn 1
WEST GERMANY

Dr. Frederick Steinheiser
CIA-ORD
Ames Building
Washington, DC 20505

Dr. Saul Sternberg
University of Pennsylvania
Department of Psychology
3815 Walnut Street
Philadelphia, PA 19104-6196

Dr. Ronald Sternfels
Oak Ridge Assoc. Univ.
P.O. Box 117
Oak Ridge, TN 37831-0117

Dr. David E. Stone
Computer Teaching Corporation
1713 South Neil Street
Urbana, IL 61820

Dr. Patrick Suppes
Stanford University
Institute for Mathematical
  Studies in the Social Sciences
Stanford, CA 94305-4115

Dr. Perry W. Thorndyke
FMC Corporation
Central Engineering Labs
1205 Coleman Avenue, Box 580
Santa Clara, CA 95052

Dr. Sharon Tkacz
Allen Corporation
209 Madison Street
Alexandria, VA 22314

Dr. Douglas Towne
Behavioral Technology Labs
University of Southern California
250 N. Harbor Dr., Suite 309
Redondo Beach, CA 90277

Major D. D. Tucker
HQMC, Code MA, Room 4023
Washington, DC 20380

Dr. Paul T. Twohig
Army Research Institute
  ATTN: PERI-RL
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Dr. Zita E. Tyer
Department of Psychology
George Mason University
4400 University Drive
Fairfax, VA 22030

Dr. Harold P. Van Cott
Committee on Human Factors
National Academy of Sciences
2101 Constitution Avenue
Washington, DC 20418

Dr. Kurt Van Lehn
Department of Psychology
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Dr. Frank L. Vicino
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr. Jerry Vogt
Navy Personnel R&D Center
Code 51
San Diego, CA 92152-6800

Dr. Thomas A. Warm
FAA Academy AAC934D
P.O. Box 25082
Oklahoma City, OK 73125

Dr. Beth Warren
BBN Laboratories, Inc.
10 Moulton Street
Cambridge, MA 02238

Dr. Diana Wearne
Department of Educational
  Development
University of Delaware
Newark, DE 19711

Dr. Shih-sung Wen
Department of Psychology
Jackson State University
1400 J. R. Lynch Street
Jackson, MS 39217

Dr. Keith T. Wescourt
FMC Corporation
Central Engineering Labs
1205 Coleman Ave., Box 580
Santa Clara, CA 95052

Dr. Douglas Wetzel
Code 51
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr. Barbara White
School of Education
Tolman Hall, EMST
University of California
Berkeley, CA 94720

Dr. David Wilkins
University of Illinois
Department of Computer Science
1304 West Springfield Avenue
Urbana, IL 61801

Dr. Marsha R. Williams
Applic. of Advanced Technologies
National Science Foundation
SEE/MDRISE
1800 G Street, N.W., Room 635-A
Washington, DC 20550

S. H. Wilson
Code 5505
Naval Research Laboratory
Washington, DC 20375-5000

Dr. Robert A. Wisher
U.S. Army Institute for the
  Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Dr. Merlin C. Wittrock
Graduate School of Education
UCLA
Los Angeles, CA 90024

Mr. Paul T. Wohig
Army Research Institute
5001 Eisenhower Ave.
ATTN: PERI-RL
Alexandria, VA 22333-5600

Mr. Joseph Wohl
Alphatech, Inc.
2 Burlington Executive Center
111 Middlesex Turnpike
Burlington, MA 01803

Dr. Wallace Wulfeck, III
Navy Personnel R&D Center
Code 51
San Diego, CA 92152-6800

Dr. Masoud Yazdani
Dept. of Computer Science
University of Exeter
Prince of Wales Road
Exeter EX44PT
ENGLAND

Dr. Joseph L. Young
National Science Foundation
Room 320
1800 G Street, N.W.
Washington, DC 20550

Dr. Uri Zernik
General Electric
Research & Development Center
Artificial Intelligence Program
PO Box 8
Schenectady, NY 12301