Flexible Execution of Cognitive Procedures

Technical Report PCG-5 (3)

Kurt VanLehn and William Ball

Departments of Psychology and Computer Science
Carnegie-Mellon University
Pittsburgh, PA l5213 U.S.A
30 June 1987

Running head: Flexible Execution of Cognitive Procedures

Acknowledgments

This research was supported by the Personnel and Training Research Programs, Psychological Sciences Division, Office of the Naval Research, under Contract No. N00014-85-C-0688. Reproduction in whole or in part is permitted for any purpose of the United States Government. Approved for public release; distribution unlimited.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

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) PCG-5	5. MONITORING ORGANIZATION REPORT NUMBER(S) PCG-5
6a. NAME OF PERFORMING ORGANIZATION Xerox Palo Alto Research Center 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Personnel and Training Research Office of Naval Research (Code 442PT)
6c. ADDRESS (City, State, and ZIP Code) 3333 Coyote Hill Road Palo Alto, CA 94304	7b. ADDRESS (City, State, and ZIP Code) 800 North Quincy Street Arlington, VA 22217-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Same as Monitoring Organization 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-85-C-0688
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	RR04206-0a	NR442a558

11. TITLE (Include Security Classification)

Flexible execution of cognitive procedures

12. PERSONAL AUTHOR(S)

VanLehn, Kurt and Ball, William

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM 6/1/85 TO 5/31/86	1987 June 30	57

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

Human problem solving, cognitive architecture, control regime

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

see reverse side

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Susan Chipman	22b. TELEPHONE (Include Area Code) (202) 696-4322 22c. OFFICE SYMBOL 442 PT

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

Abstract[1]

Several current theories of procedural knowledge hypothesize that procedures are organized as hierarchies of goals, wherein accomplishing a goal requires accomplishing all or some of its subgoals. This form of knowledge is most naturally executed with the aid of a temporary last-in-first-out stack of goals. This article presents evidence that a stack regime is not flexible enough to account for the procedural problem solving exhibited by a sample of 26 third-graders solving subtraction problems. Two alternative control regimes are investigated. One stores goals on an agenda (an unordered set) and the other stores goals in a tree. Both the agenda regime and the tree regime employ a rule-based scheduler that picks the next goal for execution. Both regimes succeed at modelling our subjects' problem solving strategies. The tree regime is able to account for data from another study as well. However, a closer examination of the fit between models and data shows that some students change their execution strategies in the midst of problem solving. This finding challenges fundamental assumptions underlying research on cognitive architectures.

[1]This is the final report on ONR Contract Number N00014-85-C-0688, which has explored mental representations of procedural knowledge and how people acquire them.

Table of Contents

1. Introduction

Much research has been devoted to uncovering the cognitive architecture that underlies human problem solving and skill acquisition. (See Newell, Laird and Rosenbloom (in press) for a review.) Many models for the cognitive architecture have been developed, including GPS (Ernst & Newell, 1969), production systems (Newell & Simon, 1972; Newell, 1978; Newell, 1973), ACT* (Anderson, 1983), applicative and-or graphs (VanLehn, 1983a; VanLehn, 1983b) and SOAR (Laird, Rosenbloom, & Newell, 1986). The theorists differ considerably on whether their models are to be taken as literal models of human cognitive architecture, or as mere notations that happen to accurately predict certain aspects of human behavior. Nonetheless, all involved would agree that people have some kind of procedural knowledge and some mechanism for turning that knowledge into action. Most investigators (but not all) assume that the mechanism is fairly simple, and that its translation of knowledge into action is fairly direct. That is, they assume that the mechanism is similar to the mechanisms used by computers to execute computer programs. This report classifies the various proposed knowledge-executing mechanisms that have been appeared in the literature, and shows which types of cognitive architectures are consistent with some new experimental evidence from the task domain of arithmetic calculation.

The classification of cognitive architectures employed here is actually one developed by computer science as a classification of program execution mechanisms. The classification is based on the features that the mechanism makes available to the programmer. The classes are called *control regimes*. For example, some programming languages (e.g., LISP, C, PASCAL) permit recursive programs, and other programming languages (e.g., FORTRAN, BASIC) do not. The execution mechanisms for the recursion-allowing languages are said to obey a *recursive control regime*, while those for FORTRAN and its ilk are said to obey a *non-recursive control regime*.

Control regimes are determined mostly by how the machine stores the temporary information that it uses to control the execution of the program, so control regimes are often named by the type of data structure used for temporary information. For instance, the control regime used by some versions of LISP is based on using a last-in-first-out stack for storing control information, so it is called a *stack regime*. A stack regime is one type of recursive control regime. Another type of recursive control regime is based on keeping control information in a randomly accessible list, called an agenda, so this control regime is called an *agenda regime*. Although the nomenclature emphasizes storage mechanisms, the classification is defined by the capabilities it allows. For instance, the agenda regime allows a type of pseudo-parallel processing called *co-routining*, whereas the stack regime does not permit co-routining.

Control regimes are classes of mechanisms, so it makes sense to ask what the control regime of the human cognitive architecture is. Knowing the control regime would tell us what capabilities (e.g., recursion, co-routining) mental programs could have. This in turn would tell us something about the initial stages of skill acquisition, where new programs are "written." For instance, suppose we knew that the human cognitive architecture obeyed a stack regime. Because a stack regime allows recursive programs, we could infer that people could learn recursive mental programs, such as the goal recursion strategy for solving the tower of Hanoi puzzle (Simon, 1975) or a top-down method of coding LISP (Anderson, Farrell, & Saurers, 1984). On the other hand, if the students' architecture obeys a non-recursive control regime, then the same training would engender a non-recursive mental program, and in particular, the program would probably have separate pieces for each level of recursion illustrated in the training.

It may seem that the control regime of the cognitive architecture is so far removed from observation that it would be impossible to ascertain its identity experimentally. However, it does have empirical consequences, and they can even be fairly direct. Suppose the training experiment just mentioned was performed for the tower of Hanoi. If the control regime is nonrecursive, the acquired program would have separate pieces for each level of recursion. This predicts that the subject could not solve problems requiring more recursive levels than the problems they received in training. This is a false prediction (Anzai & Simon, 1979). People can generalize from training on small problems to larger problems (e.g., from the 4 disk version of the tower of Hanoi to the 6 disk version). On the other hand, if the cognitive architecture obeys a recursive control regime, such as a stack regime, the acquired program could use the same pieces of knowledge for all levels of recursion, which accounts for how people can transfer their competence from the 4-disk to the 6-disk puzzle.[2] This illustrates that the control regime issue is not only an important one, but one that makes testible predictions.

---

[2]A proper argument would have to be considerably more complicated than this one. It would have to show that students acquired a recursive program and not an iterative one. See VanLehn (1983) for a proper argument in support of recursive control regimes.

This article delineates four recursive control regimes (section 1), presents an experiment (section 2), and shows that the experiment's results, when combined with other data from the same task domain, are compatible with only one of the four control regimes (section 3). However, when a simulation model based on this control regime is fit to the experiment's data (in section 4), several regularities are found that are not consistent with the usual hypothesis that a subject has just one mental program for a task. It seems instead that subjects acquire several strategies during training and switch among them during testing. The architectural implications of these findings are discussed in section 5.

2. Control Regimes

The four control regimes are all recursive ones. There is convincing argumentation that procedural knowledge is hierarchical (Simon, 1969), and recursive control regimes are the most natural control regimes for such organizations. Although there is only a little direct evidence for recursive control regimes (VanLehn, 1983c), most current accounts of cognitive architecture assume some type of recursive control regime (Anderson, 1983; Laird, Rosenbloom, & Newell, 1986; VanLehn, 1983b).

In computer science, control regimes idealize the mechanisms they describe because they do not mention the capacity limitations of the control storage. In early LISP, the stack was limited to holding a few thousand function invocations because it was in fact implemented by a table of finite size. However, the control regime is still called a stack regime, because the last-in-first-out protocol is the appropriate characterization of its behavior. In applying the control regime idea to cognitive architectures, we will continue the tradition of ignoring capacity limitations. In part, this is because the old story of seven chunks of short-term memory has developed into rich, complicated set of hypotheses (see, e.g., Zhang and Simon, 1985, or Schneider and Detweiler, 1987). For instance, there is evidence that massive training increases the apparent capacity for temporary information (Chase & Ericsson, 1982). Clearly, execution of cognitve procedures is something adults have had much practice at. Perhaps this practice has caused them to develop a large capacity memory for temporary control information. Although it is not yet clear how one should apply the more recent work on short term memory to the control component of cognitive architectures, it would clearly be naive to assume, for example, that a stack could hold at most seven goals.[3]

Unlimited capacity for control storage is also the idealization employed by current work on architectures (op cit.). Mostly, this idealization is left undefended. However, Anderson (1983) explains that when control information is forgotten due to capacity limitations, the person will reconstruct it as needed from the state of the external world. Although no one has tried to model this reconstructive process or investigate it experimentally, it seems intuitively plausible. Indeed, in the "situated action" account of procedural behavior (Suchman, 1985), the whole notion of internal, mental storage of control information is replaced by a reconstructive processes that constantly interprets the external world (the situation) in such a way as to provide roughly the same functionality as an unlimited capacity control store. Regardless of how control storage is implemented, either as mental information or interpreted situations, the issue of control regime remains. It amounts to asking what kinds of information are stored and what conventions govern its access.

The four control regimes to be investigated are delineated in subsequent paragraphs.

The deterministic stack regime. Goals are accessed according to a last-in, first-out convention. Thus, when a goal calls a subgoal, the goal's state is "pushed" onto a "stack." When the subgoal is completed, the goal is "popped" from the stack and resumes execution. The usual stack regimes (i.e., in computer languages such as PASCAL) have the added convention that the order in which subgoals are executed is fixed. Every time a goal is processed, its subgoals are executed in the same order. Usually, the order in encoded by the order in which the subgoals appear in the (written) program. We call this stack regime the *deterministic* stack regime because subgoal orders are fully determined. This control regime is used by Repair Theory (Brown & VanLehn, 1980; VanLehn, 1983a; VanLehn, 1983b), ACT* (Anderson, 1983), GRAPES (Anderson,

[3]The architecture literature uses the term "goal" for the units of modularity in procedural knowledge, so we use substitue "goal" for "function" or "procedure" when applying the control regime idea to cognition. In addition to acting as units of control modularity, many goals also describe a state that the person would like the world to be in. However, there are some notorious goals, such as "hide from Joe", that cannot be simply expressed as predicates on the state of the world. Moreover, when one tries to simulate even moderately complicated problem solving, it is frequently necessary to use such "state-less" goals to control the execution of the simulation. In this article, we will assume only that goals are units of control, and not that goals specify desired states of the world.

Farrell, & Saurers, 1984)[4] and other cognitive models.

The nondeterministic stack regime. This control regime also accesses goals according to the last-in-first-out convention. However, the subgoal order is not fixed. Instead, there is a distinct component of the program that is responsible for choosing which subgoal to execute. The knowledge encoded in this component is called a *scheduling strategy* and the component of the architecture responsible for enacting the scheduling strategy is called a *scheduler*. In the architecture literature, GPS (Ernst & Newell, 1969; Newell & Simon, 1972) was the first to employ a nondeterministic stack regime. It obeyed a nondeterministic stack regime when it was configured to perform means-ends analysis. When the current goal has several unsatisfied preconditions, GPS treats these as subgoals and uses a simple scheduler to choose which one to work on. Newell's group's most recent general problem solver, SOAR (Laird, Rosenbloom, & Newell, 1986) has a much more powerful scheduler. However, SOAR still enforces the last-in-first-out convention that characterizes means-end analysis. All subgoals must be satisfied before a goal can be popped from the stack.[5]

The tree regime. Architectures obeying a tree regime remember all goals ever invoked during the course of solving a problem and allow unrestricted access to all of them. The goal-subgoal relationships are also stored, which means that the stored information can be viewed as a tree of goals. As an illustration, suppose someone is following the goal recursion strategy for the 3-disk tower of Hanoi (i.e., to move a pyramid of N disks from peg S to peg T, move a pyramid of N-1 disks from peg S to O, move the Nth disk from S to T, then move the pyramid of N-1 disks from O to T.). The top goal in the tree (see table 1)is "move the 3 disk pyramid from peg A to peg C: Directly beneath it are the three subgoals (1) Move the 2-disk pyramid from A to B, (2) move disk 3 from A to C, and (3) move the 2-disk pyramid from B to C. Beneath the first subgoal are three more goals (1.1) move disk 1 from A to C, (1.2) move disk 2 from A to B, and (1.3) move disk 1 from C to B. After the person completes the first move, subgoal 1.1 is marked "satisfied." After the first three

---

[4]GRAPES stores goals in a tree. As will be seen shortly, storing a goal tree allows an architecture to employ a tree regime. However, GRAPES' default scheduling strategy is to search the tree depth-first, in left to right order, until it finds a pending goal, i.e., a goal that can be executed. This strategy means that GRAPES' default control regime is a deterministic stack regime. However, a GRAPES program can use special devices to edit the goal tree after it has been built, and this might allow one to implement other control regimes than the default one.

[5]SOAR has an undocumented mechanism for suspending and resuming partially completed goals, but it is rarely used (J. Laird, personal communication)

moves, goals 1.1, 1.2, 1.3 and 1 are marked satisfied. On the fourth move, goal 2 is marked done, and goal 3, move the 2-disk pyramid from B to C, is expanded producing three subgoals. Execution consists of expanding goals and marking goals satisfied. Tree structure is never deleted.

Move 3-disk pyramid from A to C

1.	Move 2-disk pyramid from A to B	
	1.1	Move 1-disk pyramid from A to C
	1.2	Move disk 2 from A to B
	1.3	Move 1 disk pyramid from C to B
2.	Move disk 3 from A to C	
3.	Move 2-disk pyramid from B to C	
	3.1	Move 1-disk pyramid from B to A
	3.2	Move disk 2 from B to C
	3.3	Move 1-disk pyramid from A to C

Table 1: Complete goal tree for 3-disk tower of Hanoi solution

In computer science, tree regimes invariably employ a scheduler that is allowed to pick any *pending* goal, where a pending goal is leaf of the tree that is not marked "satisfied." A scheduling strategy determines how the scheduler makes its choices. Consequently, a tree regime can do anything that a nondeterministic stack regime can do.

However, the tree regime permits a behavior, sometimes called co-routining or time-sharing, wherein control alternates back and forth between two or more tasks. To see how, suppose that a tree has two main goals, A and B, beneath its top node, and that A and B both have numerous subgoals. The scheduler can pick pending subgoals of A for a while, then pick subgoals of B, then go back to choosing subgoals of A. This alternation is a form of pseudo-parallel processing. It can not be done by a stack regime.

There have been no experimental tests of whether people can co-routine. In part, this is due to the types of tasks studied in the literature. Some tasks, such as the tower of Hanoi, do not permit the usual recursive solution procedures to be executed in co-routine fashion. Refering back to the tree mentioned earlier, the puzzle is constructed so that a person physically cannot work on goal 3 until goal 1 has been completed.

In lieu of experimental evidence, we can consult the intuition about whether people can co-routine. However, the intuition fails to give a clear answer. For instance, an experienced cook can alternate between chopping vegetables for a salad and basting a roast. At first glance, this apparent co-routining seems to show that the architecture obeys a tree regime. However, because we do not know what the expert cook's procedural knowledge is, we can not say with certainty that salad-making and roast-basting are adjacent goals in the meal preparation tree. Their subgoals could have been combined into one large salad/roast goal while the cook was learning how to orchestrate a meal. This salad/roast goal could be executed on a stack regime architecture, yielding the same surface behavior. Indeed, the circumstances under which beginning cooks could perform the salad/roast co-routine are unknown. They might need to use a timer, in which case a different, less powerful control regime (e.g., a stack regime with the ability to handle interrupts) suffices. The moral of this homely example is that one can not infer the control regime directly from surface behavior. One must know the structure of subjects' procedural knowledge. This important methodological prerequisite is discussed again later.

The tree regime has the odd property that satisfied goals are remembered forever. This might be approximately correct, if goal storage is implemented as episodic memory of some kind, or it could be just an idealization. The last control regime permits co-routining but store goals more economically.

The agenda regime. The agenda regime is like the tree regime, except that only the pending goals are stored. The goals are viewed as an unordered set. As an illustration, consider again the tower of Hanoi example. After the first move, the agenda is {1.2, 1.3, 2, 3}. Just after the fourth move, the agenda is {3}. On the next cycle, the scheduler picks goal 3. Processing it modifies the agenda to be {3.1, 3.2, 3.3}. Goal 3 has been removed, since it is no longer pending, and its three subgoals have been placed on the agenda. An agenda control regime supports co-routining just at the tree regime does.

Although agenda control regimes are common among current Artificial Intelligence problem solver (see Nii(1986) for a review of a particularly popular one, called the black-board architecture), no cognitive architecture has employed one. It would be an interesting direction to explore.

This completes the introduction of the four control regimes to be considered here. There are, of course, many other control regimes in computer science. For instance, we are ignoring control regimes for object-oriented programming languages. We are considering only control regimes for von Neuman style architectures because those are the ones that have been employed successfully in explaining human problem solving behavior and skill acquisition. Connectionist architectures are, so far, the only challengers, but they have a long ways to go before they can model behavior that takes longer than a minute or two. We are interested in problem solving that takes several minutes or hours to perform, so we have concentrated on von Neuman architectures.

3. The experiment

Before discussing the experiment per se, some methodological issues will be raised and dealt with.

3.1. Methodological issues

It is difficult to unequivocally determine which control regime governs the cognitive architecture because one control regime can emulate another. For instance, a nondeterministic stack regime emulates a deterministic stack regime when the scheduler employs the scheduling strategy of ordering pending goals by their order in the procedural knowledge structure. Indeed, the four control regimes under discussion happen to fall into a total order. When listed in the order (1) deterministic stack regime, (2) nondeterministic stack regime, (3) agenda regime, and (4) tree regime, each control regime can emulate the control regimes that precede it in the list. The ability of one control regime to emulate another means that the determination of control regime might have to rely on assumptions of simplicity and parsimony. If subjects are acting in such a way that the deterministic stack regime can model their behavior, then all the other control regimes can model their behavior as well and one would have to invoke parsimony and simplicity in order to argue that they are actually using a deterministic stack regime. As it turns out, this particular methodological difficulty does not arise, since the data favor the tree regime, which none of the other regimes can emulate.

A second difficulty in determining a subject's control regime is that it is not the control regime alone that determines the subject's problem solving behavior. The subject is executing some procedure or plan. As illustrated earlier, given an appropriate procedure, even the weakest control regime can act just like the strongest. So inferences about the subject's control regime are impossible without some knowledge of the procedure that they are executing.

In order to make strong assumptions about the procedural knowledge being executed, we choose a task domain in which skill acquisition is well understood. The subjects' task is subtraction of multidigit whole numbers (e.g., 324-68). There are several comments to make about this task domain.

Unlike the classic 1970's studies of puzzle solving, where subjects are given a description of the solution state and asked to find a path to it, the subtraction task gives subjects a procedure and asks them to follow it. However, these two types of tasks are not as different as one might think. Often, the subjects in puzzle solving experiments invent partial plans and follow them. Structurally, plans and procedures are identical, and the control regimes that can be used in following them are the same. So the difference between classic puzzle solving and procedure following is only in the source of the plan/procedure being followed, and not in the way that that procedural knowledge structure is followed. Consequently, the claims presented here, supported by a procedure following task, may also hold for planning tasks. Further research would, of course, be required in order to test this purported generality.

Despite the fact that subjects are taught a procedure for subtraction, one can not assume that that specific procedure is the one that they are following because skill acquisition may not be so straightforward. For instance, it is known that some students follow buggy procedures, which are systematic and stable procedures that happen to yield incorrect answers (Brown & Burton, 1978; VanLehn, 1982). However, there has been extensive work on how subtraction procedures are acquired. A model exists that explains why some students develop bugs (VanLehn, 1983b; VanLehn, 1983a). More importantly for the purposes of this paper, this model generates a set of procedures, called *core procedures*, that are potential outcomes of instruction in subtraction. Some of the core procedures have been observed (albeit, indirectly), and others are predictions about procedures that may be observed in the future. We will assume that the subjects in the experiment reported here are following one of the 30 core procedures generated by the model when it is "taught" with the same instructional material that the subjects were taught with (see VanLehn(1983b), chapter 2). This assumption replaces the simple (and false) assumption that subjects follow the procedure that their teacher intends them to learn.

A by-product of this assumption is that when fitting a particular control regime to a subject's behavior, one must choose a core procedure from the set. Fortunately, for the data discussed below, there was never any ambiguity; only one choice was appropriate for each subject.

Another caveat to mention is that the choices of a procedure and a control regime (along with a scheduling strategy, if the control regime needs one) do not totally predict behavior. They only predict behavior as long as things go according to plan. They do not specify what happens at *impasses*, where the procedure/control regime says to do something which can not be done given the current state of the problem. Impasses have been extensively studied (Brown & VanLehn, 1980; VanLehn, 1983a; VanLehn, 1983b). In situations where subjects may not seek help, they resort to one of a small variety of heuristic *repairs*, which are local perturbations to the control state. For instance, one repair is deciding to give up on the current goal. In the stack regime, this is implemented by popping the stack. In the tree regime, is this implemented by marking the current goal and all its subgoals "non-pending." Our theory of repairs (called Repair Theory) specifies this set, although it does not indicate the circumstances under which subjects will choose one repair in preference to the other applicable repairs. This means that matching a model to a subject's behavior requires fitting a parameter, the choice of repair at each impasse. Because repairs perform such local changes to the control state, this parameter does not allow much control over the model's behavior. (In particular, one can not get one control regime to emulate another.) This is good, because it usually makes the choice of repair at each impasse unambiguous.

A last comment is that subjects do not always do what they intend. They sometimes make unintended actions, called *slips* (Norman, 1981). To deal with slips, we edit them from our data (see the discussion below), along with intended behavior that lies outside the task domain, such as rewriting answers to make them more legible.

To summarize, there are four major difficulties in determining a subject's control regime: (1) some control regimes can emulate others, so simplicity and parsimony may sometimes play a role in deciding which regime the subject has; (2) all control regimes execute a procedure, so one must choose a procedure when fitting a proposed model to a subject's behavior; (3) the control regime and procedure do not determine the subject's behavior at impasses, so their behavior at their impasses (if any) must be fit from a set of repairs; (4) slips and othere behavior from outside the task domain infest the data. None of these difficulties are insurmountable, chiefly because of extensive prior work on the particular task domain chosen for investigation.

3.2. Subjects and methods

The subjects for the experiments were drawn from three third-grade classrooms. The classrooms were pre-tested twice using a paper-and-pencil diagnostic test. We selected 33 students whose errors on the pre-tests either showed an uncommon bug or were not systematic enough to be analyzed as deriving from bugs.

These 33 subjects were tested individually in a small room adjacent to their classroom. Each student solved an individualized paper-and-pencil test whose items were designed to elicit the errors we saw on that student's pre-tests. In order to collect the exact writing actions, the test page was taped to an electronic tablet, and students filled out the test with a special pen. Equipment malfunctions caused the data from 7 students to be lost. Tablet data from each of the remaining 26 students were converted into a sequence of character-writing actions, separated by measured pauses.[6] These 26 sequences are the "protocols" used throughout this article as data.

3.3. Results: qualitative version

It is difficult to summarize protocol data in a theory-neutral way, so this section will adopt a classification that is useful for the purpose of comparing control regimes, and summarize the data in terms of that classification. A later section treats the data in a more quantitive fashion. The observed orders are classified into *standard* order, *locally nonstandard* orders, and *globally nonstandard* orders. A later section will show that all three types of orders occurred. This section merely defines the three classes, along with some other nomenclature that will be useful later.

We define a *standard* order to be an execution order that was taught to the students. Unfortunately, we do not know exactly which standard order was taught. We know the textbooks that the students were taught from, but the textbook's examples and discussions are consistent with more than one standard order. One would have to watch the teacher present examples at the

---

[6]The original purpose of the experiment was to find chronometric evidence for repair by measuring the pauses between writting actions. However, the pause data turned out to be very noisy. Long pauses seemed to be caused mostly by episodes of counting (e.g., in order to determine a number fact, such as the difference between 15 and 7), by fiddling with the pen or by other theoretically irrelevant activities. Against this high background variation in pause length, it would be difficult to measure the duration of repair episodes relative to the duration of non-repair actions. We have not undertaken a thorough analysis of the pause data.

blackboard in order to determine unambiguously which orders were taught. However, under any of the standard orders permitted by the textbook, the columns are processed in right-to-left order, and all borrowing actions for a column are completed before the answer to the column is calculated. It will be assumed that these two properties define the *standard* orders of subtraction execution.

We found that most subjects employed a standard order, but eight subjects ordered their writing actions in nonstandard ways (see table 2).[7] For instance, some students did all the borrowing first, on one right-to-left pass across the columns of a problem, then returned left-to-right filling in the column answers. Although the small sample size and and its biased selection prevents us from drawing strong conclusions about the overall population, the fact that one third of the sample used nonstandard orders indicates that phenomenon is not a trivial or idiosyncratic one.

Standard order	15
Nonstandard orders	8
No scratch marks	3
Total	26

Table 2: Number of students with standard and nonstandard orders

For the purposes of comparing control regimes, it is helpful to divide the nonstandard orders into two classes: locally nonstandard orders and globally nonstandard orders. To do so requires that some notation be introduced.

A rule-based notation is used for core procedures. Table 3 shows a core procedure. Each subscripted symbol is a goal, and each rule is a method for achieving the goal on its left side. The subscripts indicate the way a goal passes arguments to its subgoals. In the case of subtraction, the arguments always happen to be columns. For instance, the rule C_i --> A_i F_{i+1} $-_i$ means that the column processing goal, C, can be achieved for column i by achieving three subgoals: adding ten to the top digit of column i (A_i), performing the borrow-from subgoal on the next column (F_{i+1}), and taking the difference between the two digits of column i and writing it in the answer position of column i ($-_i$). Alongside each rule is a condition indicating when the rule is applicable. Thus, the rule just stated is applicable when the top digit in column i is less than the bottom digit ($T_i<B_i$).

[7]Three students used no scratch marks. Consequently, we can tell almost nothing about the order they used.

$Sub \rightarrow C_i$	For each column$_i$
$C_i \rightarrow -_i$	When $T_i \geq B_i$
$C_i \rightarrow A_i\ F_{i+1}\ -_i$	When $T_i < B_i$
$F_i \rightarrow SD_i$	When $T_i \neq 0$
$F_i \rightarrow F_{i+1}\ A_i\ SD_i$	When $T_i = 0$
$SD_i \rightarrow S_i\ D_i$	Always

Table 3: A core procedure written in a rule-based notation

This notation is designed to subordinate information which is not needed in the arguments presented below. However, it leaves intact the feature of procedures that is essential for the arguments, which is that they are hierarchies of goals. In fact, this procedure is recursive, because the borrow-from goal calls itself. Such hierarchical, recursive knowledge representations have been widely used in cognitive science for representing procedural knowledge (e.g., Newell & Simon, 1972; Anderson, 1983; Laird, Rosenbloom, & Newell, 1986). VanLehn (1983) argues from bug data to show that students in subtraction are best represented as having knowledge structures like the one in table 3.

The set of core procedures contains 30 procedures, including the one of table 3 (VanLehn, 1983b). Some core procedures are complete and correct, but different from the procedure of table 3 in that they have slightly different conventions for borrowing (e.g., for borrowing across zero, F_i calls F_{i+1} and 9_i, where 9_i is a primitive goal that changes the top digit of column i to 9). Other procedures are partial versions of a taught procedure (e.g., if borrowing from zero has not been learned, then the rule $F_i \rightarrow F_{i+1}\ A_i\ SD_i$ is missing). Other procedures have overgeneralized or overspecialized test conditions (e.g., C_i borrows when the top digit in column i is less than *or equal to* the bottom digit).

A *trace tree* is a way of displaying the history of execution of a procedure. Its leaves are the primitive, observable actions of the procedure. The nonleaf nodes are instances of nonprimitive goals. A node with its descendents corresponds to a rule: the node is the left side of the rule, instantiated, and the descendents correspond to the subgoals on the right side of the rule. Descendents of a node are arranged chronologically so that the leftmost descendent corresponds to the subgoal that was first executed, and the rightmost descendent to the last subgoal. This implies that the leaves are ordered so that the primitive actions they represented occur in the left-to-right order of the leaves. Figure 1 shows a trace tree for a standard order execution of the core procedure shown in table 3.

When a procedure is executed so that the descendents of a node have the same order as the right side of the rule, then one of the standard orders is generated. This is a consequence of the learning model, (VanLehn, 1983a; VanLehn, 1983b) which constructs core procedures by generalizing the example execution that the teacher presents.

With the notations for core procedures and trace trees introduced, the subclassifications of nonstandard orders can now be defined.

A *locally nonstandard* order produces a trace tree where the order of descendents of some node does not correspond to the right-side ordering of the corresponding rule. The trace tree in figure 2 corresponds to a nonstandard order execution, because the ordering of subgoals for borrowing does not correspond to the A_i F_{i+1} $-_i$ order of the right side of the borrowing rule of the core procedure of table 3. Indeed, none of the 30 core procedures use the order shown in the trace tree, wherein the column is answer before borrowing is completed. Thus, the protocol shown the figure, which corresponds to the leaves of the tree, can not be modelled by a standard execution of any core procedure. When we say that a person has a locally nonstandard order, we mean that there is no core procedure which allows a standard order execution, and there is at least one core procedure that allows a locally nonstandard order execution.

A *globally nonstandard* order produces a trace tree which is nonplanar, in that the branches cross. Figure 3 shows a globally nonstandard order execution for the core procedure of table 3. Notice that the borrowing actions of all columns are completed first, then the columns are answered in left-to-right order. There is no core procedure that allows this sequence of actions to be executed in a standard or locally nonstandard order, so we say that it is a globally nonstandard order.

The main points to be taken from this discussion of the data are that there are three types of orders, standard, locally nonstandard and globally nonstandardand, and that all three types occur in the data. Moreover, the orders often occur mixed together in the performance of a single student.

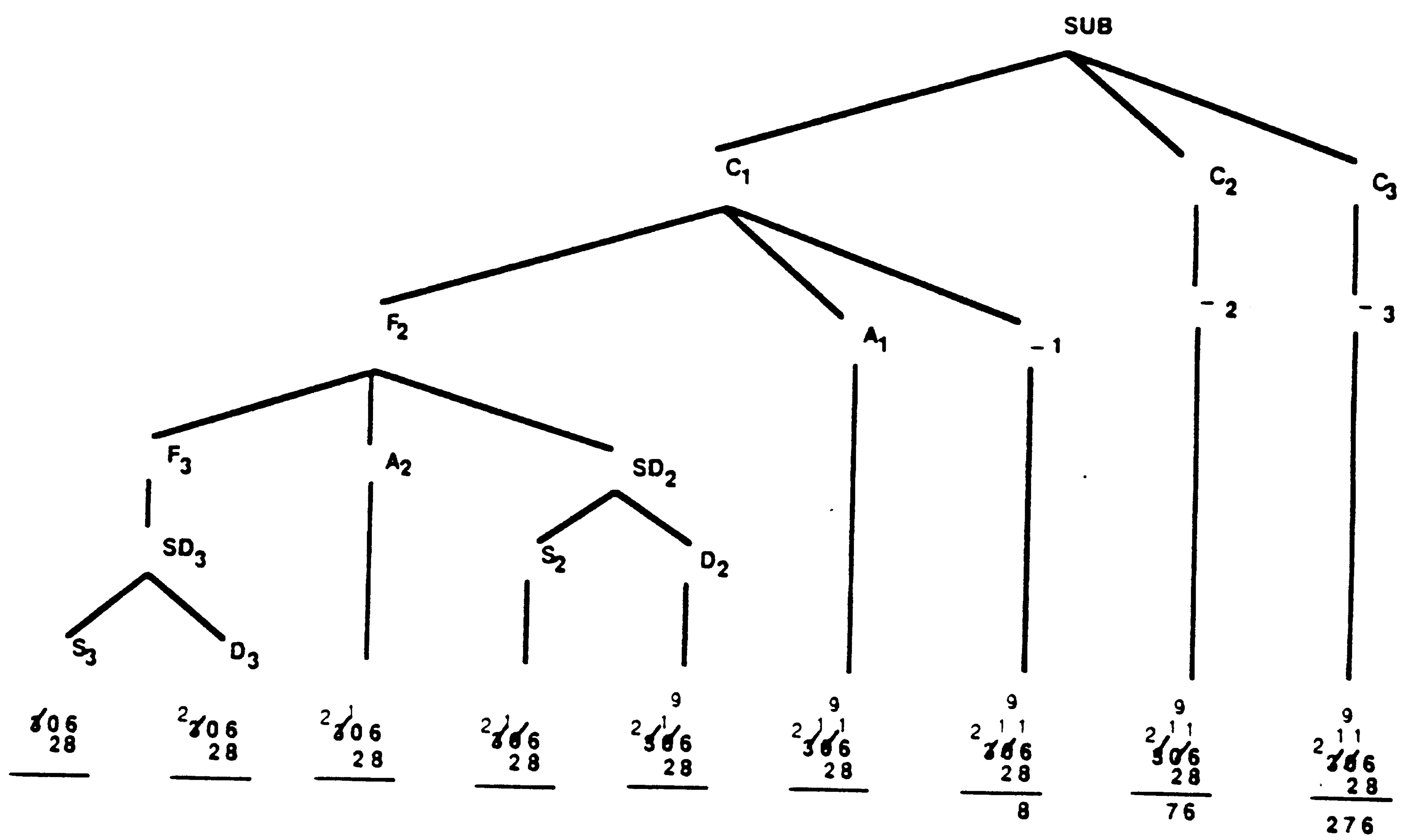


Figure 1: Trace tree for standard order execution

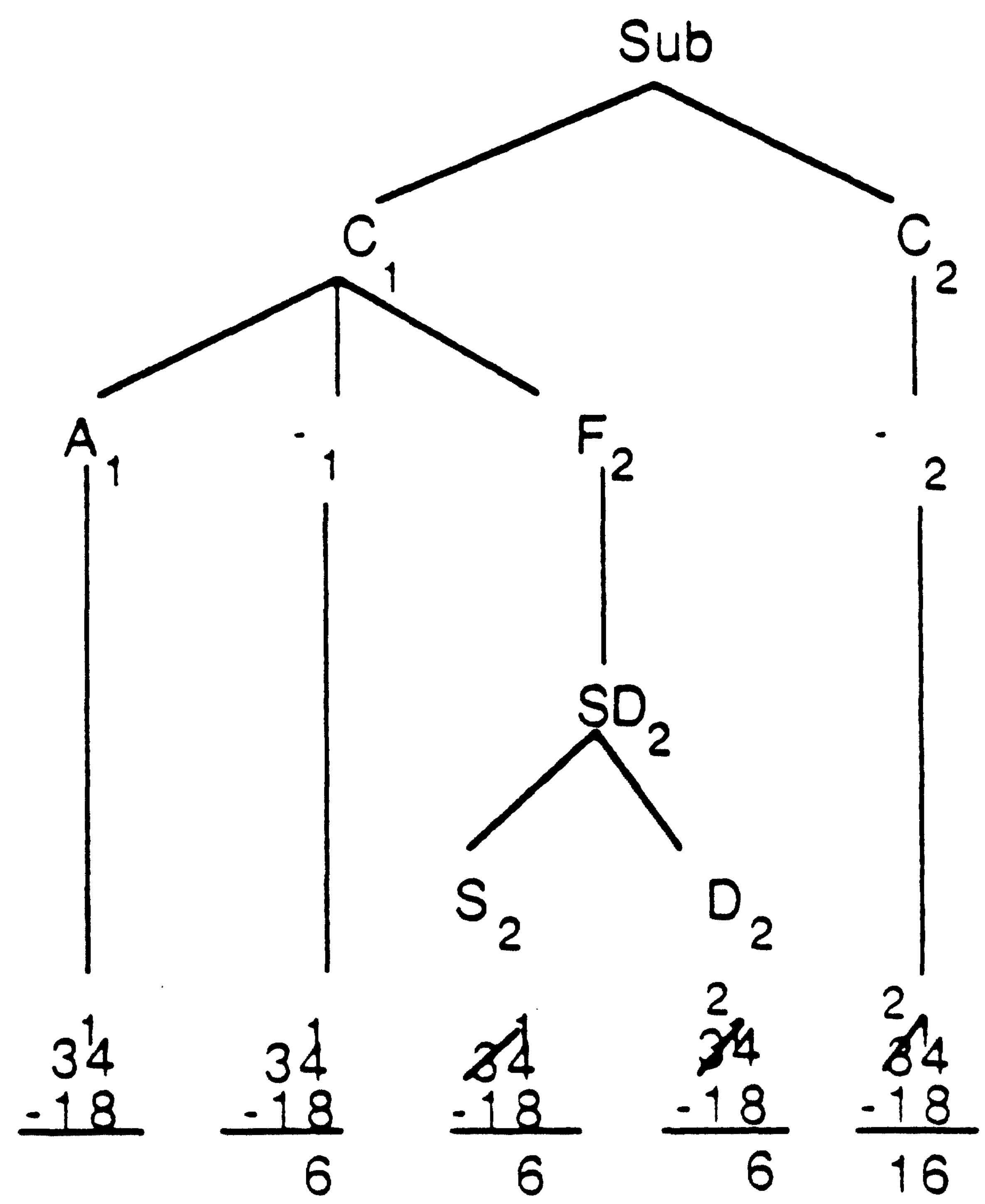


Figure 2: Trace tree for a locally nonstandard execution

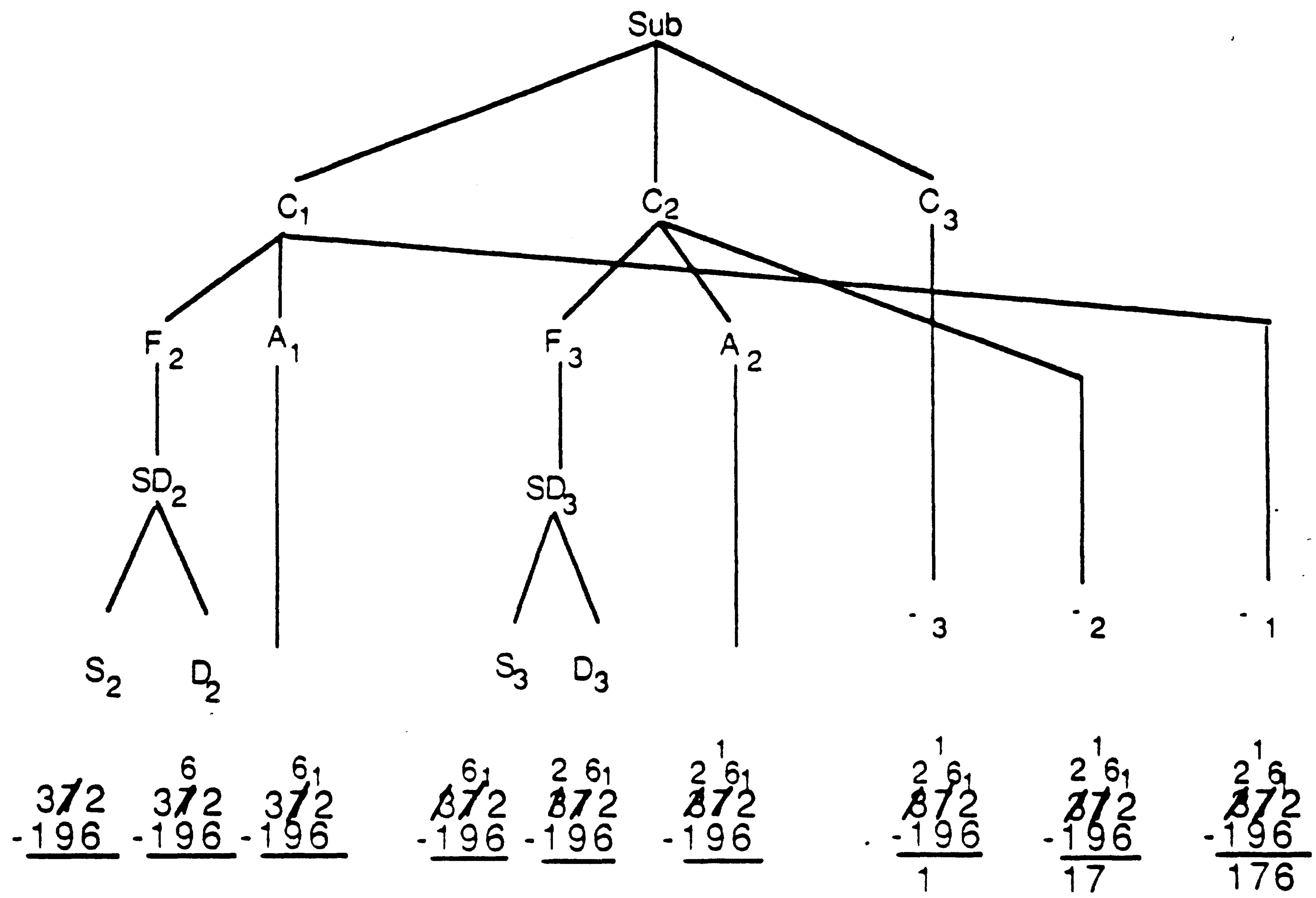


Figure 3: Trace tree for a globally nonstandard execution

4. Which control regimes are consistent with the data?

Each of the four control regimes will be considered in turn for their ability to produce nonstandard orders.

The deterministic stack regime. A deterministic stack regime is the weakest control regime. When a subgoal is being executed, the goal resides on a stack, along with an index into its list of subgoals. The index indicates which subgoal to call next when the present subgoal is finished. Figure 4 shows the stack at each cycle of interpretation during a standard order execution of the procedure of table 3 on the problem 306 − 28.

In a deterministic stack regime, the order of execution of subgoals is fixed by the contents of the procedure. The order is the order in which the subgoals were listed on the right side of the rule. However, this fixed ordering makes it impossible for the deterministic stack regime to model even the locally nonstandard orders, let alone the globally nonstandard ones.

Nondeterministic stack regime. The nondetermistic stack regime is the next step up in power from the deterministic stack regime. It does not fix the order of subgoal execution in advance, but instead employs a scheduler that chooses which subgoal to execute. Thus, when a rule is about to be executed for the first time, the scheduler chooses which subgoal from the right side to execute first. The chosen subgoal is instantiated, the current goal is pushed onto the stack, and execution of the subgoal begins. When its execution is finished, scheduler chooses one of the remaining subgoals as the second one to be executed. When all the subgoals have been executed, the goal is popped and control returns to the goal that called it.

The implementation of GPS described by Newell and Simon (1972) obeys a nondeterministic stack regime and uses a simple scheduler based on a total order of goal types. When a goal is begun or continued, the subgoals are categorized and their types are looked up in a list. The subgoal whose type occurs first in the list is chosen for execution. If two subgoals tie, because they have the same type, then the model is underdetermined at that point. It does not predict which choice the subject will make. This is why the control regime is called a *nondeterministic* stack regime. The list of goal types is a parameter to the model. It represents a scheduling strategy. Since the types are task specific, the scheduling strategy is necessarily task specific as well.

External:	$\frac{306}{28}$	$\frac{306}{28}$	$\frac{306}{28}$	$\frac{306}{28}$	$\frac{306}{28}$	$\frac{\cancel{3}06}{28}$	$\frac{\cancel{3}06}{28}$	$\frac{\overset{2}{\cancel{3}}06}{28}$
Stack:	*LOOP	*C_1 LOOP	*F_1 C_1 LOOP	*F_3 F_2 C_1 LOOP	*SD_3 F_3 F_2 C_1 LOOP	*S_3 SD_3 F_3 F_2 C_1 LOOP	*SD_3 F_3 F_2 C_1 LOOP	*D_3 SD_3 F_3 F_2 C_1 LOOP
Agenda:	*LOOP	C_3 C_2 *C_1	*F_2 A_1 - 1 C_3 C_2	SD_2 A_2 *F_3 A_1 - 1 C_3 C_2	*SD_3 SD_2 A_2 A_1 - 1 C_3 C_2	*S_3 D_3 SD_2 A_2 A_1 - 1 C_3 C_2	N O C H A N G E	*D_3 SD_2 A_2 A_1 - 1 C_3 C_2
Tree:	*LOOP	LOOP *C_1 C_2 C_3	Sub C_3 C_3 C_3 F_2 A_1 - 1	etc.				

Figure 4: Three control regimes executing. A "*" indicates the active goal.

For subtraction, a total order of goal types would merely cause emulation of the deterministic stack regime, so a slightly more powerful scheduling mechanism is used. Goal types are *partially* ordered instead of totally ordered. A scheduling strategy is notated by a set of transitive, antisymetric ordering constraints. For instance, the ordering relation $S_i>D_i$ means that slashing is preferred over decrementing given that both refer to the same column, i. This constraint will cause the scheduler to first pick S_2 off the agenda when both S_2 and D_2 are on it unless, of course, there is some other goal on the agenda that is even more highly preferred then S_2. Once S_2 has been executed and is therefore off the agenda, the scheduler is free to pick D_2. Another example is the constraint $C_i>C_{i+1}$, which means that the column processing goals will to be processed in right to left order.

The scheduler uses the partial order established by such constraints to choose one of the remaining subgoals of a goal whenever that goal is begun or continued. Because the order is partial and not total, it is possible for two goals to tie for first place even when they are of different types. For instance, suppose that $A_i>-_i$ is the only constraint relevant to the borrowing goal's subgoals, A_i, F_{i+1} and $-_i$. When the goal is begun, two goals will tie for first place, the add-ten subgoal, A_i, and the borrow-from subgoal, F_{i+1}. Thus, the model does not predict which goal will be choosen by the subject. It will be consistent with either one.

This is, of course, not the only way to notate scheduling strategies. It is merely the next step up in power from GPS's notation. Indeed, it will turn out later that this notation is not powerful enough to represent some of the regularities in the data. However, the notation does allow a precise comparision of control regimes, which is all that we need of it for the purposes of this paper.

The nondeterministic stack regime can be parameterized so that it will generate the locally nonstandard orders as well as the standard ones. For instance, in order to generate the nonstandard order illustrated in figure 2, the constraints $-_i>F_{i+1}$ and $A_i>-_i$ are used to order the subgoals of borrowing. However, there is no way to parameterize the model so that the globally nonstandard orders are generated.

The agenda regime. Another type of control regime uses an *agenda* as its goal memory. An agenda is an unordered set of instantiated goals. When a goal is executed, it is first removed from the agenda, then all its subgoals are instantiated and placed on the agenda. Thus, an agenda represents a set of pending goals, i.e., an unordered plan for all the things that need to be done to finish the problem. Figure 4 shows the agenda during execution of the procedure of figure 3.

An agenda regime requires a scheduler to choose which of the goals on the agenda to execute next. The agenda regimes in AI have tended to use very complicated schedulers (Nii, 1986). Since this scheduler is part of a comparision between control regimes, it makes sense to use the same scheduler as was used in the nondeterministic stack regime. This means that differences in generative power will be due to the control regime and not differences in the power of the scheduler.

An agenda regime allows enough flexibility to model both globally and locally nonstandard orders. The key to this flexibility is that the agenda regime can mix the execution of subgoals from two different goals, because the agenda allows access to *all* pending subgoals. The nondeterministic stack regime allows access to only the most recent goal's subgoals. As an illustration of this power, figure 5 shows the agenda at each step of a globally nonstandard execution order. The problem 345-189 is solved in two passes, with all the borrowing on the first pass and all the column differences on the second pass. Table 4 gives a scheduling strategy that will produce this order. The agenda regime is the first of the regimes to be discussed that has sufficient power to model all the observed execution orders.

The tree regime. The fourth type of control regime for recursive programs is based on keeping a tree of instantiated goals. When the procedure has finished, the tree is exactly the trace tree for the procedure on that problem. During execution, this tree is constructed incrementally. A system of markers is used to differentiate pending goals from those that have been executed. As with the agenda regime, a scheduler picks which of the pending goals to execute. Again, we will assume that the scheduler is guided by a scheduling strategy that is represented as a partial order. Figure 4 shows a few cycles of interpretation using a tree regime.

The tree regime can emulate the agenda regime because whenever the agenda always corresponds to the set of goals in the tree that are marked pending. In particular, both the tree regime and the agenda regime can handle all three types of orders. The agenda regime is slightly more elegant because it maintains the minimal temporary state needed to do its job. The extra

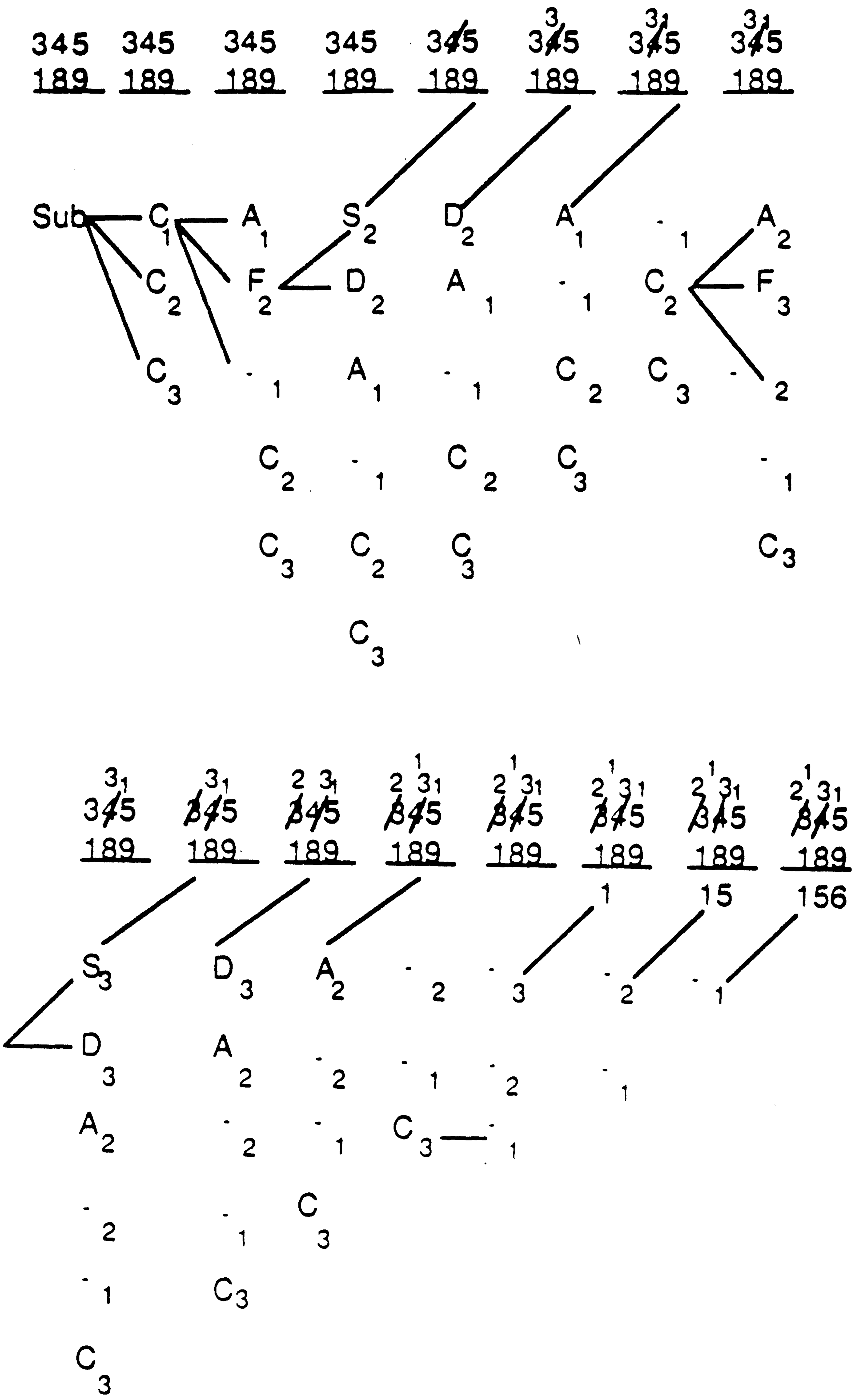


Figure 5: The two-pass strategy running on an agenda regime

$C_i > C_{i+1}$	Do columns right-to-left
$-_{i+1} > -_i$	Do column differences left-to-right
$S_i > D_i$	Do slash before decrementing
$X_{i+1} > A_i$ for X=F,S,D,A	Do all borrowing-from before add 10
$X_i > -_i$, for X=B,A,F,SD,S,D,Z, or 9	Do column differences last

Table 4: Constraints for the two-pass scheduling strategy

nodes in the tree serve no purpose as far as scheduling is concerned. However, this is not a sufficient reason to prefer the agenda regime over the tree regime. More empirical evidence is needed. Fortunately, such evidence is already at hand.

In our earlier work on repair (Brown & VanLehn, 1980; VanLehn, 1983c), it was shown that a common repair is to retreat to a super-goal of the goal where the impasse occurs and resume interpretation there. This kind of repair is called hierarchical backup. It is different from chronological backup (the kind used in (Newell & Simon, 1972)) or dependency directed backup (deKleer, 1986). It resembles a nonlocal "return" in hierarchical programming languages (e.g., the catch-throw construct in CommonLisp). In order to function properly, hierarchical backup requires that the architecture maintain either a stack or tree of goals so that hierarchical backup can easily access a supergoal of the currently active goal. With an agenda, which has only pending goals, backing up to a supergoal would require a reverse interpretation process to reconstruct the chain of supergoals that called the currently active goal. Reverse interpretation is complex because it can be non-deterministic. If a goal is called from two different supergoals, the reverse interpreter must guess which one was in fact the one that called the goal. In short, the agenda just does not have as much information as the tree, and moreover, that missing information is sometimes crucial for the backup repair. The backup evidence shows that the architecture obeys a tree regime rather than an agenda regime.

It seems that the tree-based control regime is uniquely capable of modelling both the ordering data and the repair data. We can tentatively conclude that the tree-based control regime is obeyed by the mental architecture that human procedures are executed by.

5. How well does the tree regime fit the protocol data?

So far, the argument has been based on a qualitative treatment of the protocol data. It was asserted that some subjects, on some problems, used execution orders that only the tree and agenda regímes can model. This section fits the tree regime to the protocol data in order to determine how much of the variability in student's execution orders can be captured. This will yield an intuitive assessment of the *absolute* quality of the tree regime, rather than its quality relative to the other control regimes. It will also demonstrate the instability of subject's scheduling strategies over time.

5.1. The fitting procedure

This section describes how the tree regime is fit to the data. There are three parameters that must be given values: (1) the procedure, (2) the scheduling strategy, and (3) the repairs to any impasses that occur. The parameterization method is to first choose a procedure and repairs, then fit a scheduling strategy. In all cases, the choice of procedure and repairs was quite clear cut, so it was done by hand. Table 5 shows the procedures assigned to each student.

The choice of scheduling strategy was less clear. We had to find a good fit by running the model with our best guess at a set of constraints, then examining the residuals. Residuals are places where the model and the protocol disagree. There are two kinds of residuals:

- *Underprediction*: More than one pending goal is maximal according to the constraint set, and the student's choice is among the maximal items. The model partially explains the student's choice, but falls short of exactly predicting it.
- *Misprediction*: The student's choice is not among the maximal pending goals. The model mispredicts the student's behavior.

To arrive at a numerical evaluation of the model's fit to a student's protocol, we counted the cases of underprediction and misprediction. In general, these two counts are inversely related. If the underprediction count is high, then adding ordering constraints will bring that count down. However, this usually causes the model to make some wrong choices, driving the misprediction count up.

In general, we preferred to minimize mispredictions. A model that predicts that the students sometimes act on guesses is preferable to a model that predicts that they have a scheduling

<u>Angela</u>
$C_i \rightarrow -_i$
$C_i \rightarrow F_{i+1}\ A_i\ -_i$
$F_i \rightarrow SD_i$
$F_i \rightarrow F_{i+1}\ A_i\ SD_i$

<u>Janine and Tanya</u>
$C_i \rightarrow -_i$
$C_i \rightarrow A_i\ F_{i+1}\ -_i$
$F_i \rightarrow SD_i$
$F_i \rightarrow 9_i\ F_{i+1}$

<u>Robby</u>
$C_i \rightarrow -_i$
$C_i \rightarrow A_i\ F_{i+1}\ -_i$
$F_i \rightarrow SD_i$

<u>Trina</u>
$C_i \rightarrow -_i$
$C_i \rightarrow A_i\ F_{i+1}\ -_i$
$F_i \rightarrow SD_i$
$F_i \rightarrow A_i\ F_{i+1}$

<u>Hilda</u>
$C_i \rightarrow -_i$
$C_i \rightarrow F_{i+1}\ A_i\ -_i$
$F_i \rightarrow S_i\ D_i$
$F_i \rightarrow F_{i+1}$

<u>Paul and Pete</u>
$C_i \rightarrow -_i$
$C_i \rightarrow F_{i+1}\ A_i\ -_i$
$F_i \rightarrow S_i\ D_i$
$F_i \rightarrow F_{i+1}\ A_i\ S_i\ D_i$

<u>Primitives</u>
$-_i$ Take difference in column i
A_i Add ten to top of column i
S_i Slash top digit of column i
D_i Decrement top digit of column i
SD_i Slash and decrement column i
9_i Change top digit of column i to 9

Table 5: Procedures for the Nonstandard order students.
Hilda, Trina and Robby have bugs. The others students are bug-free.

strategy that they do not obey.[8]

Table 6 shows the best fitting models for the eight nonstandard order students. The scheduling strategies for all the students except Tanya included ordering relations needed for standard procedures to be executed in such a way that they produce a correct answer. We call these the *base* set of relations. Table 7 lists them. They are included by reference in the figure 6. Tanya's constraint set included some, but not all, of the base relations.

Table 6 shows that the number of underdetermined choices is high. It ranges from 8% to 51% of the total number of choices made during the protocol, with a mean of 33%. This is not good. Taken literally, it means that the students are guessing one-third of the time.

[8]There is a second reason for avoiding mispredictions, which is that mispredictions are harder to count than underpredictions. When a model has been forced to take a choice that it would not have chosen itself, it is frequently the case that there will be a second choice that it must be forced to make. The first forced choice pushes the model off its preferred track and the second forced choice pushes it back on. It is not clear whether to count this as one case of misprediction or two. When the model makes frequent mispredictions, the forced choices can interfer with each other, making the counting of mispredictions a very messy business.

Subject	Miss	Under	Total	Scheduling strategy
Angela	0%	46.3%	138	$A_{i+1}>A_i$, $F_{i+1}>A_i$, $SD_{i+1}>A_i$, $F_{i+j}>-_i$, $A_{i+j}>-_i$, base
Hilda	0%	16.3%	123	$F_{i+1}>A_i$, $S_{i+j}>A_i$, $-_i>C_{i+j}$, base
Janine	0%	42.6%	155	$A_i>F_{i+1}$, $A_{i+j}>-_i$, $SD_{i+j}>-_i$, $SD_{i+j}>C_i$, $9_i>F_{i+1}$, $9_{i+j}>-_i$, base
Paul	0%	19.6%	158	$A_{i+1}>A_i$, $F_{i+1}>A_i$, $S_{i+1}>A_i$, $A_i>C_{i+1}$, $-_i>C_{i+j}$, base
Pete	0%	51.0%	100	$F_{i+1}>A_i$, $S_{i+1}>A_i$, $-_i>-_{i+1}$, base
Robby	0%	43.4%	145	$SD_{i+1}>A_i$, $A_{i+j}>-_i$, $F_{i+j}>-_i$, $SD_{i+j}>-_i$, base
Tanya	0%	7.9%	114	$SD_{i+j}>A_i$, $9_i>F_{i+1}$, $X_i>-_i$ for X=F, SD, A, S, or 9, $X_{i+j}>-_i$ for X=F, SD, A, S, or 9.
Trina	1.4%	44.7%	143	$A_{i+j}>-_i$, $SD_{i+j}>-_i$, $SD_{i+j}>A_i$, base

Table 6: Best fitting agenda models for the eight nonstandard order students.
Miss = Percentage of total choices mispredicted by the model.
Under = Percentage of total choice underpredicted by the model.
Total = Number of agenda choices total.

$C_i>C_{i+1}$	Subtract columns from right to left
$X_i>C_i$ for X=F, S, D, SD, 9, A	Change column before testing T>B
$A_i>-_i$	Avoid subtracting larger from smaller
$S_i>D_i$	Slash before decrement
$A_i>SD_i$	Avoid decrementing 0 during BFZ
$A_i>S_i$	Add10 before slash during BFZ
$A_i>D_i$	Add10 before decrement during BFZ

Table 7: The base set of relations

When we examined the students' supposed guesses, we discovered an underlying pattern. Some students use more than one execution strategy. To illustrate it, we'll consider one student, Paul, in detail. A facsimile of Paul's test sheet is included as figure 6. His protocols are given in table 8.

Before going over Paul's protocol in detail, it should be mentioned that the protocol data for all the students has been edited in order to remove actions that can not be accounted for by the simple core procedures we used. For instance, in problem 1, Paul actually did $-_1 -_2 -_1 -_2 -_3$. He rewrote his answers to the units and the tens columns, probably because he thought that they were illegible (they looked illegible to us, too.). The simple core procedures produced by the learning model can not represent these extra actions, so they were removed from the data. The appendix presents the raw data alongside the edited data, and explains why each edit was made. We feel that the cleaned up data remains adequate for testing the fit of the tree regime.

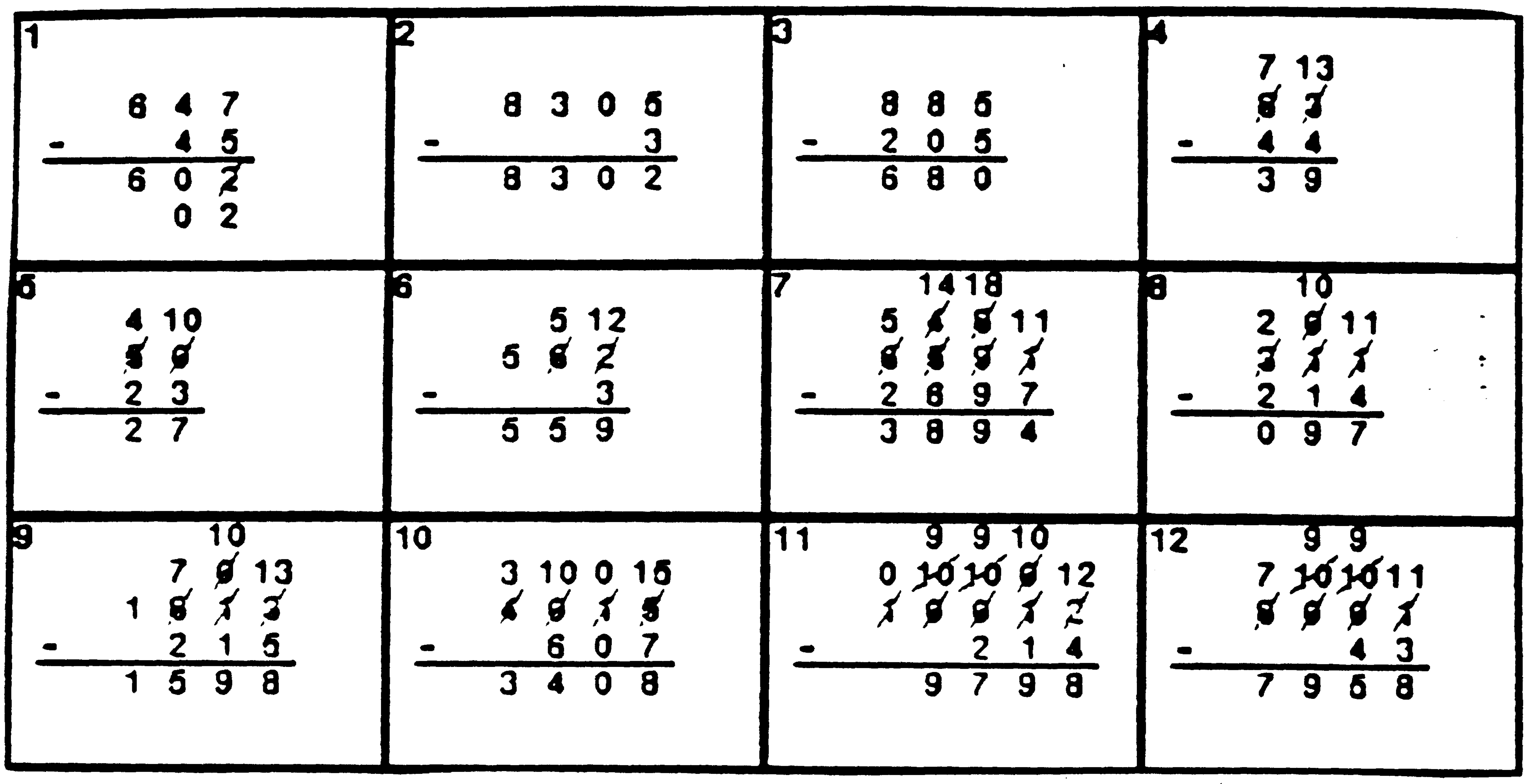


Figure 6: Paul's test

On some problems (problems 12 and most of problem 11), Paul consistently decrements before adding ten during borrowing. On other problems (problems 4, 5, 6, 8 and 9), he adds ten and subtracts the column before decrementing. On problems 7 and 10, he sometimes decrements first and sometimes adds ten first. The remaining problems (problems 1 through 3) do not require any borrowing, so we can not tell what scheduling strategy he was using for them. Consequently, the largest constraint set that avoids mispredictions is one that does not take a stand on how the decrement operation is ordered with respect to the other borrowing operations. This five-member constraint set is the one shown in in table 6. It exactly predicts Paul's performance only on the

1. $-_1\ -_2\ -_3$
2. $-_1\ -_2\ -_3\ -_4$
3. $-_1\ -_2\ -_3$
4. $S_2\ A_1\ -_1\ D_2\ -_2$
5. $S_2\ A_1\ -_1\ D_2\ -_2$
6. $S_2\ A_1\ -_1\ D_2\ -_2\ -_3$
7. $S_2\ A_1\ -_1\ D_2\ S_3\ A_2\ D_3\ -_2\ S_4\ A_3\ -_3\ D_4\ -_4$
8. $S_2\ A_1\ -_1\ D_2\ S_3\ A_2\ -_2\ D_3\ -_3$
9. $S_2\ A_1\ -_1\ D_2\ S_3\ A_2\ -_2\ D_3\ -_3\ -_4$
10. $S_2\ A_1\ -_1\ D_2\ -_2\ S_4\ A_3\ D_4\ -_3\ -_4$
11. $S_2\ A_1\ -_1\ D_2\ S_5\ D_5\ A_4\ S_4\ D_4\ A_3\ S_3\ D_3\ A_2\ -_2\ -_3\ -_4\ -_5$
12. $S_4\ D_4\ A_3\ S_3\ D_3\ A_2\ S_2\ D_2\ A_1\ -_1\ -_2\ -_3\ -_4$

Table 8: Paul's protocols

three problems that do not require borrowing. However, Paul's choices on the test can be almost exactly predicted if the model employs one constraint set (i.e., the original five plus $-_i > D_{i+1}$) on some problems (problems 4,5,6,8 and 9, and parts of problems 7,10 and 11), a different constraint set (i.e., the original five plus $D_{i+1} > A_i$) on others (problem 12 and the remainder of problem 11), and a third constraint set on two columns in the middle of problems 7 and 10.

Paul is *not* alternating randomly among the possible legal orderings of borrowing operations, as the scheduling strategy of figure 6 predicts. There are several more permutations of borrowing operations than the three that Paul uses.[9] Paul has definite preferences about which orders to do borrowing, and these constraints sets capture them exactly.

We found similar patterns for six of the eight students with nonstandard orders. Table 9 shows the results of fitting the model to minimize underprediction by using multiple constraint sets. The appendix presents the strategies fit to each student.

In three cases (Hilda, Paul and Tanya), we found sets of constraints that would yield an exact match, indicating that the students were alternating among multiple scheduling strategies. In

[9]There are three orders that Paul does not use: (1) add ten, slash, decrement, difference; (2) add ten, slash, difference, decrement; (3) add ten, difference, slash, decrement.

Subject	Sets	Miss	Under
Angela	1	3.6%	1.4%
Hilda	3	0%	0%
Janine	4	0%	7.1%
Paul	3	0%	0%
Pete	3	0%	5%
Robby	3	0%	5.2%
Tanya	2	0%	0%
Trina	1	8.4%	0%

Table 9: Fits, allowing multiple scheduling strategies
Sets = number of constraint sets.
Miss = Percentage of agenda choices mispredicted.
Under = Percentage of agenda choices underpredicted.

three other cases (Janine, Pete and Robby), the use of multiple strategies instead of one helped the fit, but did not yield an exact match. In the remaining two cases (Angela and Trina), using two or more strategies did not help the fit much at all.

Fit with one scheduling strategy	1
Fit with two or three scheduling strategies	6
Poor fit	1
Total	8

Table 10: Summary of the fit of the agenda model

The overall fit of the model is summarized in table 10. It shows two basic findings. First, good fits were obtained for all the students except Trina. We judge that the model's fit to Trina is a "poor" fit, although it is not as bad as the fit of, say, the deterministic stack regime to her protocol. We just can not see any pattern in Trina's performance. The second basic finding is that 75% of the nonstandard order students seem to be using multiple scheduling strategies.

We conclude that the tree regime allows an excellent fit to the data, but that it fits well only at the cost of adding a mystery: what causes students to shift strategies? The next section presents our speculations on this issue.

6. Multiple strategies: what, when and why?

At this time, we do not have an explanation of why some students have multiple strategies and what causes them to shift among them. However, there are some interesting aspects to the multiple-strategy data that hint at the underlying causes.

The first question to ask is what sort of strategies tend to appear together in one student's behavior. There is no way to answer such a clustering question in a principled manner, so a heuristic, intuitive classification will have to suffice. The eight subjects can be classified into two groups of four each. The first group (Hilda, Paul, Robby, and Tanya) have strategies that differ only in the way they order the subgoals of borrowing, whereas the second group (Angela, Pete, Janine and Trina) have strategies that differ in the way they order the column subgoals. For example, Tanya is in the first, "borrow variations" group because all her strategies are similar. She always does all her borrowing before any column is answered. She borrows from right to left, then she answers columns from right to left. However, she uses two strategies for borrowing. Sometimes she adds ten then borrows-from, and other times she borrows-from then adds ten. So her strategies are minor variations of each other.

We conjecture that students in the "borrow variations" group actually have a single, uniform strategy, but our representation for strategies can not express that strategy. In fact, we found uniform strategies for two of the students, Paul and Tanya. Tanya's strategy can be expressed as: if i=1 then $A_i>F_i$, else if i=2 then $F_i>A_i$. Paul's strategy (except for problems 7 and 10) seems to be: if borrowing from zero, then $D_{i+1}>A_i$, else $-_i>D_{i+1}$. These strategies are conditional on the state of the problem solving, so the partial order cannot express them.

We have developed a more expressive representation for scheduling strategies as well as a program that will automatically fit a strategy, expressed in this representation, to the subject's behavior (VanLehn & Garlick, 1987). With this tool, we hope to discover precise, uniform strategies for all the students in the borrow variations group, which is a first step towards explaining what students are doing and why.

For instance, having noted that Paul's strategy is "if borrowing from zero, then $D_{i+1}>A_i$, else $-_i>D_{i+1}$," one can immediately see a compelling intuitive explanation. Paul does a standard order on borrow-from-zero problems because he thinks those problems are hard and he would increase his chances of getting them right by using the teacher's strategy. This implies that he knows two distinct strategies, his and the teacher's, and uses meta-cognitive reasoning to select one. This explanation is consistent with a phenomenon that has been occasionally observed but has not yet received systematic investigation. Some students seem to have two or more distinct procedures. For instance, one student we interviewed answered a whole test by taking the absolute difference in every column, even those requiring borrowing. Afterwards, the interviewer asked her if she knew about borrowing. She said she did, and showed us by answering two borrowing problems correctly. Resnick and Omanson (1986) observed several subjects who seem to have multiple procedures. Only a little prompting (their "prohibition" condition) sufficed to make them switch from a buggy procedure to a correct one. Paul is similar to these students in that he seems to know two distinct strategies, believes they both give correct answers, but have different resource or accuracy characteristics.

If this conjectured multiplicity of procedures and strategies withstands empirical testing, then one of the central assumptions of cognitive modelling must be modified. It has always been assumed that subjects in skill acquisition and problem solving experiments have just one knowledge structure, but these findings indicate there might be several, with a "big switch" that selects one or the other.

Although the patterns of behavior in the borrow variations group may soon yield to explanations with important theoretical implications, the patterns of behavior in the other group (Angela, Janine, Pete and Trina) are more difficult to understand. Intuitively, it appears that all four students start out with a standard order strategy at the beginning of the test, then become increasing nonstandard towards the end. All four ended with a strategy that involves answering columns in a "wild" order, i.e., one that is neither right-to-left nor left-to-right. One possible explanation for this behavior is that they become increasingly confident as the test progresses and begin to show off their skills. There are, of course, other equally plausible explanations. Considerable empirical and theoretical work will be needed in order to understand and differentiate such explanations.

6.1. Execution of procedures as search

This section is speculative. It introduces a generalization of the tree-based control regime that makes intuitive sense and connects the results presented earlier with the existing litterature on puzzle solving. However, the results do *not* conclusively support this model over a simpler tree-based model. This conjecture should be understood as an outline for future research.

Under the tree regime, two different mechanisms are searching the goal tree -- one is the scheduler, and the other is the mechanism that does repairs. Both search for a goal to perform next. We conjecture that they are the same cognitive process because the student is trying to solve the same problem. The student's problem is "which goal should I do next?" and it should be solved in such a way that the resulting solutions should look on paper like the standard procedure had been executed in a standard order. As both scheduling and repair seem to respect this constraint, it seems likely that they result from the same process, rather than being two different processes, as they are under the tree regime.

The proposed process is like classical problem solving, except that the problem to be solved is not at the level of the task, but is *meta* to the task. In particular, the problem is not to solve a subtraction problem but to find a goal in the subtraction procedure to execute.

As a species of (meta-) problem solving, one would expect the cognitive process to have some of the attributes found in ordinary, base-level problem solving. In particular, just as some subjects alternate among search strategies while solving puzzles (Newell & Simon, 1972), one would expect to find subjects switching among search strategies while doing (meta-level problem solving. As mentioned earlier, most of the nonstandard order subjects do shift among strategies. This does not explain why they switch strategies, nor where they got the strategies that they switch among. However, it is somehow comforting that the same familiar mysteries appear in both meta-level and base-level problem solving.

This view of procedure execution seems consistent with observations by Suchman and Wynn (1984) in their study of office procedures. They studied clerks in a customer service office. They found that much of the daily work of the clerks was not simply following the prescribed office procedures (i.e., the ones found in the procedure manuals of the corporation). Although some tasks were accomplished according to standard procedure, much of the time went into handling cases where the prescribed procedure could not be followed exactly. In such cases, the clerks may do some complicated problem solving so that it will appear that the prescribed procedure has been followed. Suchman and Wynn recorded the following account of a clerk who attempted to get a customer to pay a bill when the bill is incorrect:

> Okay, you call the buyer, the buyer says, um, the reason why I'm not paying this is, I said I would pay *twenty* dollars and seventy-*three* cents for a carton, not twenty-four dollars and seventy-two cents, which you bill me on this five thousand dollar shipment of paper. So then you say, that's all I need to know, let me get back with you. You get back, you go through your billing system, you try to find out, you know, how it (pause): In the meantime, let's suppose time is running out and you do not have time to get a billing adjustment through. So you got to sit there and think, *How can I get this person to pay this invoice?* It's wrong, they got the wrong PO, they billed them wrong, accounts payable doesn't want to do anything with it. So you call them back up and say, I'm not asking you to pay something that is not due. What I want you to do is pay (pause) according to your PO. Pay the invoice short, okay? Then he says, I will not pay that invoice short because I've had too many problems with that. Unless I get a typed invoice from you specifically. So you sit there and think, I can't go through the billing system, it's too late. I can type them an invoice. Set the system going through the billing system at the same time. Coordinate that so when he pays the check short, there will be a balance on the account. When the credit issues through I'll have the billing department hold that credit, deliver that credit to me, not deliver it to the customer cause the customer will wonder why am I getting the credit if they think they're already gonna receive a bill, right? Then I would just clean up their account later. But in the meantime....(Suchman & Wynn, 1984, pg. 34)

This episode dramatically illustrates how complicated the problem solving of procedure following can be. This clerk is clearly an expert at it. Our current conjecture is that nonstandard orders and repair, which we have observed in our studies of subtraction, are just simple forms of this type of problem solving.

References

Anderson, J. R. (1983). *The Architecture of Cognition.* Cambridge, MA: Harvard.

Anderson, J. R., Farrell, R., & Saurers, R. (1984). Learning to program in LISP. *Cognitive Science, 8*, 87-129.

Anzai, Y. & Simon, H.A. (1979). The theory of learning by doing. *Psychological Review, 86*, 124-140.

Brown, J. S., & Burton, R. B. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science, 2*, 155-192.

Brown, J. S. & VanLehn, K. (1980). Repair Theory: A generative theory of bugs in procedural skills. *Cognitive Science, 4*, 379-426.

Chase, W.G. & Ericsson, K.A. (1982). Skill and working memory. In G.H. Bower (Ed.), *The psychology of learning and motivation, Vol. 16.* New York: Academic Press.

deKleer, J. (1986). An assumption-based truth maintenance system. *Artifical Intelligence, 28*, 127-162.

Ernst, G.W. & Newell, A. (1969). *GPS: A Case Study in Generality and Problem Solving.* New York, NY: Academic Press.

Laird, J. E., Rosenbloom, P. S., and Newell, A. (1986). Chunking in Soar: The anatomy of a general learning mechanism. *Machine Learning, 1*(1), 11-46.

Newell, A. (1973). Production systems: Models of control structures. In W. C. Chase (Ed.), *Visual Information Processing.* New York: Academic.

Newell, A. (1978). Harpy, production systems and human cognition. In R. Cole (Ed.), *Perception and Production of Fluent Speech.* New York: Erlbaum.

Newell, A. & Simon, H. A. (1972). *Human Problem Solving.* Englewood Cliffs, NJ: Prentice-Hall.

Newell, A., Laird, J.E. & Rosenbloom, P. (1988). Symbolic Architectures. In Posner, M. (Ed.), *Foundations of Cognitive Science.* Cambridge, MKA: MIT Press. In preparation.

Nii, P. (1986). The blackboard model of problem solving. *AI Magazine, 7*(2), 38-53.

Norman, D. A. (1981). Categorization of action slips. *Psychological Review, 88*, 1-15.

Resnick, L. B. & Omanson. S. F. (1987). Learning to understand arithmetic. In R. Glaser (Ed.), *Advances in Instructional Psychology.* Hillsdale, NJ: Erlbaum.

Schneider, W. & Detweiler, M. (1987). A connectionist/control architecture for working memory. In G.H. Bower (Ed.), *The psychology of learning and motivation, Vol. 21.* New York, NY:

Academic Press.

Simon, H.A. (1969). *The Sciences of the Artificial.* Cambridge, MA: MIT Press.

Simon, H.A. (1975). The functional equivalence of problem solving skills. *Cognitive Psychology, 7,* 268-288. Reprinted in H.A. Simon, Models of Thought, Yale University Press, 1979.

Suchman, L. (1985). *Plans and Situated Actions* (Technical report). Xerox Palo Alto Research Center.

Suchman, L. & Wynn, E. (1984). Procedures and problems in the office. *Office: Technology and People, 2,* 133-154.

VanLehn, K. (1982). Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills. *The Journal of Mathematical Behavior, 3*(2), 3-71.

VanLehn, K. (1983). Human skill acquisition: Theory, model and psychological validation. In *Proceedings of AAAI-83.* Los Altos, CA: Morgan Kaufman,

VanLehn, K. (1983). *Felicity conditions for human skill acquisition: Validating an AI-based theory* (Tech. Report CIS-21). Xerox Palo Alto Research Center.

VanLehn, K. (1983). The representation of procedures in repair theory. In H. P. Ginsberg (Ed.), *The Development of Mathematical Thinking.* Hillsdale, NJ: Erlbaum.

VanLehn, K. & Garlick, S. (1987). Cirrus: an automated protocol analysis tool. In Langley, P. (Ed.), *Proceedings of the fourth Machine Learning Workshop.* Los Altos, CA: Morgan-Kaufman.

Zhang, G. & Simon, H.A. (1985). STM capacity for Chinese words and idioms: Chunking and acoustical loop hypotheses. *Memory and Cognition, 13*(3), 193-201.

Appendix:

This appendix presents the protocols of each of the eight subjects exhibiting a nonstandard order, and our analyses of them. There are eight sections, one for each subject. Each section has three subsections: 1) the subject's protocol, annotated to indicate our idealizations of it; 2) the constraint sets for each scheduling strategy; and 3) a figure showing how each strategy fares on predicting the subject's agenda choices. The figures require some explanation. The large tick marks indicate agenda selections. The vertical stripe beneath a small tick mark is black if the strategy correctly predicts the subject's agenda selection, and white if it does not. In cases where the subject seems to use multiple strategies, the figure also shows a bar, labelled "Union," that shows the best fit one can obtain by assuming the subject switches strategies.

Protocol and Analyses for Angela

$\begin{array}{r} 562 \\ -\ \ 3 \\ \hline \end{array}$ $SD_2, A_1, -_1, -_2, -_3$

$\begin{array}{r} 742 \\ -\ 136 \\ \hline \end{array}$ $SD_2, A_1, -_1, -_2, -_3$

$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array}$ $SD_2, A_1, -_1, -_2$

$\begin{array}{r} 8305 \\ -\ \ \ 3 \\ \hline \end{array}$ [$SD_3, A_2, SD_2, A_1, -_1$, <write 1 in column 2>],

$-_2, -_3, -_4$

• We count it a slip that she borrows starting in column 1 with 5 over 3; so the idealized protocol does $-_1, -_2, -_3, -_4$.

$\begin{array}{r} 106 \\ -\ 70 \\ \hline \end{array}$ $SD_3, A_2, -_2, -_1, -_3$

$\begin{array}{r} 716 \\ -\ 598 \\ \hline \end{array}$ $SD_2, A_1, -_1$, [$-_2$], $-_3$

• She does a $-_2$ instead of initiating a borrow; the idealized protocol does the borrow.

$\begin{array}{r} 1564 \\ -\ 887 \\ \hline \end{array}$ $SD_2, A_1, -_1, SD_3, A_2, -_2, SD_4, A_3, -_3, -_4$

$\begin{array}{r} 6591 \\ -\ 2697 \\ \hline \end{array}$ $SD_2, A_1, -_1$, [$-_2$], $SD_4, A_3, -_3, -_4$

• We count it a slip that she fails to borrow in column 2 with 8 over 9; the idealized protocol inserts [SD_3 , A_2] prior to $-_2$.

$\begin{array}{r} 311 \\ -\ 214 \\ \hline \end{array}$ [$SD_3, A_2, -_2$, [A_3], $-_3$, [<crossout ans $_3$>], $SD_2, A_1, -_1$]

• We do not model this problem at all, because of a slip she makes in doing a Borrow-From-Zero over the 1 in column 2 (a slip she does not make in the earlier problem: 716 - 598).

102 - 34	$SD_3, A_2, [-_2, SD_2, A_1, -_1], -_3$

• She inserts $[-_2]$ before the sequence $[SD_2, A_1, -_1]$, instead of after that sequence (as the idealized protocol handles it).

9007 - 6880	$-_1, SD_4, A_3, [-_3, SD_3, A_2, -_2], -_4$

• She inserts $[-_3]$ before $[SD_3, A_2, -_2]$ instead of following. Idealized protocol uses $[SD_3, A_2, -_2, -_3]$ for the bracketed sequence.

702 - 108	$SD_3, A_2, SD_2, A_1, -_1, -_2, -_3$

CONSTRAINTS

Base:

$C_i > C_{i+1}$

$X_i > C_i$ $\qquad X = F, SD, A$

$A_i > SD_i$

$A_i > -_i$

$A_{i+1} > A_i$ $\qquad$ - Needed for Borrow-From-Zero

Common:

Since we only have one fully constrained set for Angela, common = standard order.

Standard:

$F_i > X_i$ $\qquad X = C, F, A, -, SD$

$SD_{i+1} > SD_i$

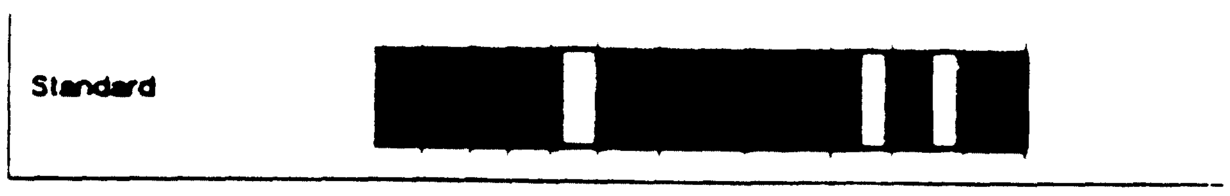
$A_i > C_{i+1}$

$-_i > C_{i+j}$

$SD_{i+1} > A_i$

$F_{i+1} > A_i$

COMPARISON OF AGENDA SELECTIONS: ANGELA



Protocol and analyses for Hilda

$\begin{array}{r} 647 \\ -\ \ 45 \\ \hline \end{array}$ $-_1, -_2, -_3$

$\begin{array}{r} 885 \\ -\ 405 \\ \hline \end{array}$ $-_1, -_2, -_3$

$\begin{array}{r} 83 \\ -\ 44 \\ \hline \end{array}$ $S_2, A_1, D_2, -_1, -_2$

$\begin{array}{r} 8305 \\ -\ \ \ \ 3 \\ \hline \end{array}$ $-_1, -_2, -_3, -_4$

$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array}$ $S_2, D_2, A_1, -_1, -_2$

$\begin{array}{r} 562 \\ -\ \ \ 3 \\ \hline \end{array}$ $S_2, A_1, D_2, -_1, -_2, -_3$

$\begin{array}{r} 742 \\ -\ 136 \\ \hline \end{array}$ $S_2, A_1, -_1, D_2, -_2, -_3$

$\begin{array}{r} 106 \\ -\ \ 70 \\ \hline \end{array}$ $-_1, S_3, D_3, A_2, -_2, -_3$

$\begin{array}{r} 9007 \\ -\ 6880 \\ \hline \end{array}$ $-_1, S_4, A_2, -_2, D_4$ *

- The idealized protocol finishes answering columns 3 and 4. There is an impasse on trying to decrement 4, because it is already decremented. She does a Quit repair. We model it as a Force repair.

$\begin{array}{r} 4015 \\ -\ \ 607 \\ \hline \end{array}$ $S_2, D_2, A_1, -_1, -_2, S_4, A_3, D_4, -_3, -_4$

$\begin{array}{r} 702 \\ -\ 108 \\ \hline \end{array}$ $[S_2], S_3, A_1, D_3, -_1, -_2, -_3$

- We count as a slip her slash of column 2.

$\begin{array}{r} 2006 \\ -\ \ \ 42 \\ \hline \end{array}$ $-_1$, [crossout ans $_1$ and redo], $[S_3]$, S_4, A_2, D_4, $-_2$ *

- The idealized protocol finishes answering columns 3 and 4.

CONSTRAINTS

Base:

$C_i > C_{i+1}$

$X_i > C_i$ $\quad X = -, F, A, S, D$

$S_i > D_i$

$A_i > -_i$

Common:

$-_i > C_{i+j}$

$S_{i+j} > A_i$

$F_{i+j} > A_i$

$X_i > C_{i+j}$ $\quad X = F, A, S, D$

Standard:

$D_{i+j} > A_i$

Weave:

$A_i > D_{i+j}$

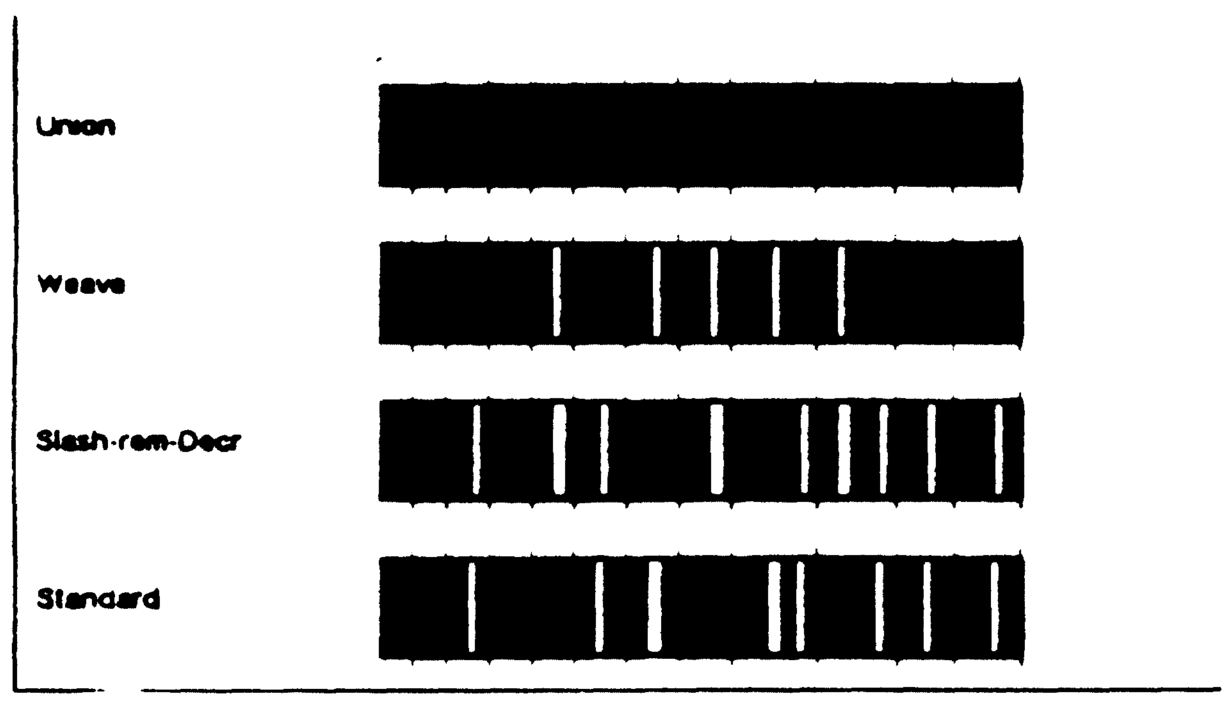
$D_{i+j} > -_i$

Slash-Reminds-to Decrement:

$A_i > D_{i+j}$

$-_i > D_{i+j}$

COMPARISON OF AGENDA SELECTIONS: HILDA



Protocol and analyses for Janine

$$\begin{array}{r} 83 \\ -\ 44 \\ \hline \end{array}$$ A_1 , SD_2, $-_1$, $-_2$

$$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array}$$ A_1 , SD_2, $-_1$, $-_2$

$$\begin{array}{r} 742 \\ -\ 136 \\ \hline \end{array}$$ A_1 , $-_1$, SD_2, $-_2$, $-_3$

$$\begin{array}{r} 106 \\ -\ 70 \\ \hline \end{array}$$ $-_1$, A_2 , SD_3 , $-_2$ *

• Janine does not do the - in column 3, since it is just a matter of bringing down a 0. The ideal protocol does the - .

$$\begin{array}{r} 716 \\ -\ 598 \\ \hline \end{array}$$ A_1, SD_2, A_2, SD_3, $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 1564 \\ -\ 887 \\ \hline \end{array}$$ A_1 , SD_2, A_2, SD_3, A_3, SD_4, $-_1$, $-_2$, $-_3$ *

• This is another instance of a neglected - in the last column. The ideal protocol does the - .

$$\begin{array}{r} 102 \\ -\ 39 \\ \hline \end{array}$$ A_1 , $-_1$, [correct ans_1], $S9_2$, $-_2$, SD_3 *

• Another neglected - in the last column. The ideal protocol does the - .

$$\begin{array}{r} 9007 \\ -\ 6880 \\ \hline \end{array}$$ $-_1$, A_2 , $S9_3$, SD_4, $-_2$, $-_3$, $-_4$

$$\begin{array}{r} 702 \\ -\ 108 \\ \hline \end{array}$$ A_1 , $-_1$,* , 9_2 , $-_2$, SD_3 , $-_3$, [rewrite 6 over column 3]

• She inexplicably rewrites the Decrement over Top 3. Also, she neglects to slash Top 2 before writing 9.

$$\begin{array}{r} 2006 \\ -\ 42 \\ \hline \end{array}$$ $-_1$, A_2, $-_2$, [9_3 , S_3], SD_4, $-_3$, $-_4$

• She does the 9, then the slash. The ideal protocol reverses the order.

$$\begin{array}{r} 10012 \\ -\ \ \ 214 \\ \hline \end{array}$$ A_1 , SD_2, A_2, $S9_3$, $S9_4$, SD_5, $-_1$, $-_2$, $-_3$, $-_4$ *

• This is another instance of a neglected - in the last column. The ideal protocol executes the - .

$$\begin{array}{r} 8001 \\ -\ \ \ 43 \\ \hline \end{array}$$ A_1, $S9_2$, $S9_3$, SD_4, $-_4$, $-_3$, $-_2$, $-_1$

$$\begin{array}{r} 401 \\ -\ 206 \\ \hline \end{array}$$ A_1 , $S9_2$, $-_1$, $-_2$, SD_3, $-_3$

<u>CONSTRAINTS</u>

Base:

$X_i > C_i$ $\qquad$ X = F, A, SD, - , S9

$A_i > -_i$

$C_i > C_{i+1}$

Common:

$S9_i > F_i$

$SD_{i+j} > -_i$

$A_i > F_{i+1}$

Prepfirst (Do everything except - on every column, then come back and do all the - s in order from right to left):

**$A_{i+j} > -_i$

*$F_i > C_i$

**$F_{i+j} > -_i$

$-_i > -_{i+j}$

$C > -$

*$SD_i > C_i$

* Needed because - no longer eliminates C.

** Needed because C is processed before - to its right is done.

Prepfirst/LR (Same as Prepfirst, but answers from left to right):

Same constraints as Prepfirst, except:

$-_{i+j} > -_i$ instead of $-_i > -_{i+j}$

Reverse (Does A, then SD, then - when processing Borrow-From):

$- > C$

$S9_{i+j} > -i$

$F_i > -i$

$SD_{i+j} > -i$ Have to do any to the left because of the Write 9 Borrow-From-Zero.

$F_{i+j} > -i$ Have to do any to the left because of the Write 9 Borrow-From-Zero.

Onepass (Does the A and the - before proceeding to the Borrow-From):

$- > C$

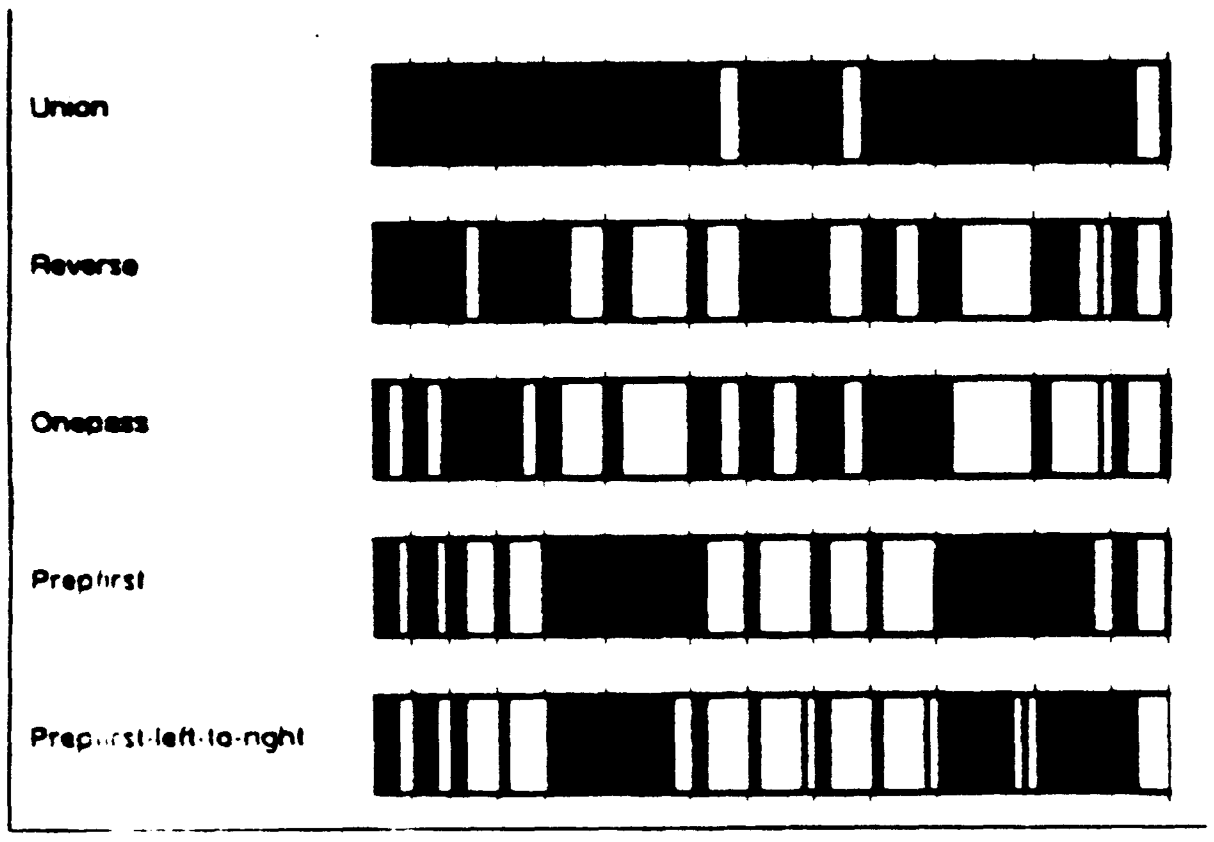
$S9_{i+j} > -i$

$-i > F_i$ Notice that this replaces the last three in the Reverse set.

$\left.\begin{array}{r} F_i > C_i \\ SD_{i+1} > C_i \end{array}\right\}$ Needed since the Cs are no longer always eliminated by - s.

NOTE that none of the foregoing constraint sets handles problems 7, 9, and 13, in which she interrupts her Borrow-From-Zero procedure to process a column.

COMPARISON OF AGENDA SELECTIONS: JANINE



Protocol and analyses for Paul

$\begin{array}{r} 647 \\ -\ 45 \\ \hline \end{array}$ $^-1$, $^-2$, [$^-1$, $^-2$], $^-3$

- Paul "stutters" in rewriting his answer in columns 1 and 2.

$\begin{array}{r} 8305 \\ -\ 3 \\ \hline \end{array}$ $^-1$, $^-2$, $^-3$, $^-4$

$\begin{array}{r} 885 \\ -\ 205 \\ \hline \end{array}$ $^-1$, $^-2$, $^-3$

$\begin{array}{r} 83 \\ -\ 44 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, $^-2$

$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, $^-2$

$\begin{array}{r} 562 \\ -\ 3 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, $^-2$, $^-3$

$\begin{array}{r} 6591 \\ -\ 2697 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, S_3, A_2, D_3, $^-2$, S_4, A_3, $^-3$, D_4, $^-4$

$\begin{array}{r} 311 \\ -\ 214 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, S_3, A_2, $^-2$, D_3, $^-3$

$\begin{array}{r} 1813 \\ -\ 215 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, S_3, A_2, $^-2$, D_3, $^-3$, $^-4$

$\begin{array}{r} 4015 \\ -\ 607 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, $^-2$, S_4, A_3, D_4, $^-3$, $^-4$

$\begin{array}{r} 10012 \\ -\ 214 \\ \hline \end{array}$ S_2, A_1, $^-1$, D_2, S_5, D_5, A_4, S_4, D_4, A_3, S_3, D_3, A_2, $^-2$, $^-3$, $^-4$, [$^-5$]

- Paul does not write the column 5 "0" in the answer row.

$$\begin{array}{r} 8001 \\ -\ \ \ 43 \\ \hline \end{array} \quad S_4, D_4, A_3, S_3, D_3, A_2, S_2, D_2, A_1, -_1, -_2, -_3, -_4$$

CONSTRAINTS

Base:

$C_i > C_{i+1}$
$X_i > C_i$ $\quad X = F, S, D, A$
$S_i > D_i$
$A_i > -_i$
$A_i > S_i$
$A_{i+1} > A_i$

Common:

$-_i > C_{i+j}$
$F_{i+1} > A_i$
$S_{i+1} > A_i$

Slash-Reminds-to Decrement (As in standard borrow, process the Borrow-From goal and its Slash subgoal, but then shift back to do Add10 and Diff, using the slash mark to "remind" that the column needs to be decremented):

$A_i > D_{i+1}$
$-_i > D_{i+1}$
$D_i > D_{i+1}$
$C_i > D_{i+1}$ } Needed for Borrow-From-Zero (which he doesn't do)

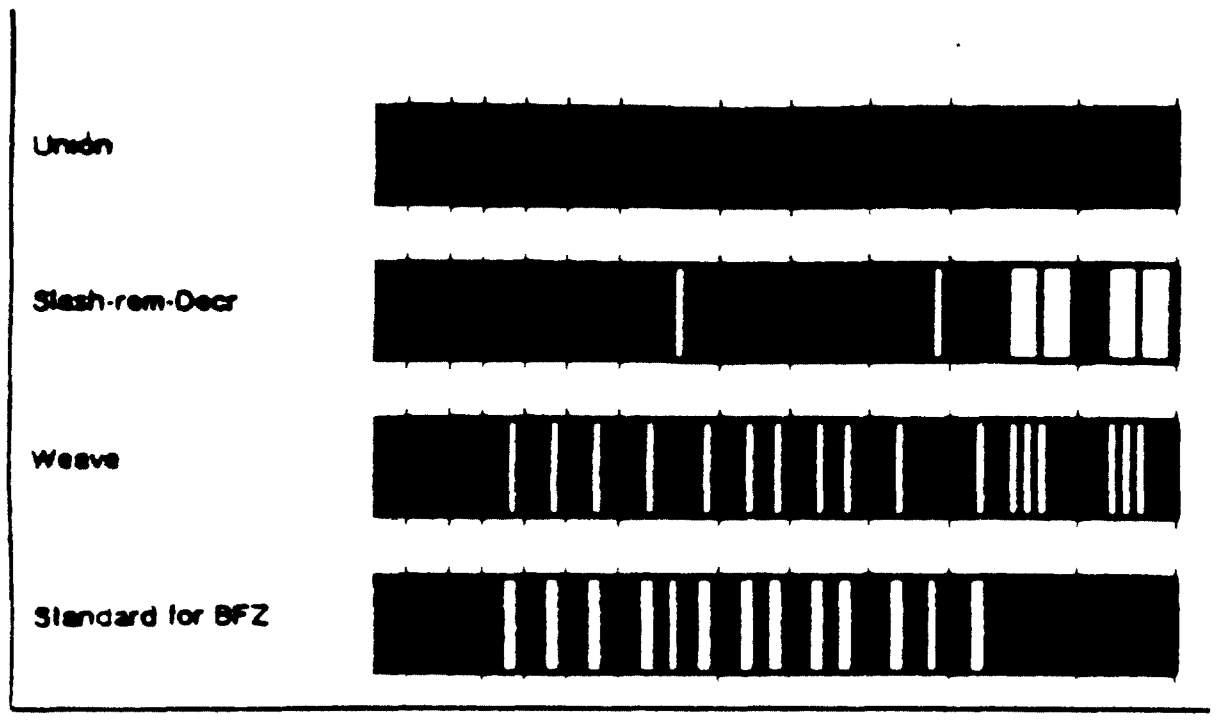
Weave (As in standard borrow, process Borrow-From goal and its Slash subgoal first, but then shift back to do the Add10, then shift columns again to finish the Borrow-From's other subgoal, Decr, before doing the Diff):

$A_i > D_{i+1}$
$D_{i+1} > -_i$
$D_{i+1} > S_i$

Standard: Paul uses the standard order when he does a Borrow-From-Zero..

$D_{i+1} > A_i$
$A_i > C_{i+1}$

COMPARISON OF AGENDA SELECTIONS: PAUL



Protocol and analyses for Pete

$$\begin{array}{r} 647 \\ -\ 45 \\ \hline \end{array}$$

$^{-}1$, $^{-}2$, $^{-}3$ [rewrite ans $_2$]

- He rewrites his answer to column 2. The model does not.

$$\begin{array}{r} 885 \\ -\ 205 \\ \hline \end{array}$$

$^{-}1$, $^{-}2$, $^{-}3$

$$\begin{array}{r} 83 \\ -\ 44 \\ \hline \end{array}$$

S_2, A_1, D_2, $^{-}1$, $^{-}2$

$$\begin{array}{r} 8305 \\ -\ 3 \\ \hline \end{array}$$

$^{-}1$, $^{-}2$, $^{-}3$, $^{-}4$

$$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array}$$

S_2, A_1, $^{-}1$, D_2, $^{-}2$

$$\begin{array}{r} 562 \\ -\ 3 \\ \hline \end{array}$$

S_2, A_1, D_2, $^{-}1$, $^{-}2$, $^{-}3$

$$\begin{array}{r} 742 \\ -\ 136 \\ \hline \end{array}$$

$^{-}1$, $^{-}2$, $^{-}3$

- Takes absolute differences instead of borrowing.

$$\begin{array}{r} 106 \\ -\ 70 \\ \hline \end{array}$$

S_3, A_2, D_2, S_2, A_1

- Borrows when he shouldn't.

$$\begin{array}{r} 106 \\ -\ 70 \\ \hline \end{array}$$

S_3, A_2, D_2, A_1, $^{-}1$, $^{-}2$

- Borrows when he shouldn't.

$$\begin{array}{r} 311 \\ -\ 214 \\ \hline \end{array}$$

S_3, D_3, A_2, S_2, D_2, A_1, $^{-}1$, $^{-}2$, $^{-}3$

- Borrows from "Zero" even though it's a 1.

We excluded these from the idealized protocol.

$$\begin{array}{r} 6591 \\ -\ 2697 \\ \hline \end{array} \quad S_2, A_1, D_2, S_3, A_2, D_3, S_4, D_4, A_3, -_1, -_2, -_3, -_4$$

$$\begin{array}{r} 1564 \\ -\ 887 \\ \hline \end{array} \quad S_2, A_1, D_2, S_3, A_2, D_3, S_4, D_4, A_3, -_1, -_2, -_3 \quad \star$$

- He does not do the final column Diff when it's just a zero.

$$\begin{array}{r} 716 \\ -\ 598 \\ \hline \end{array} \quad S_2, A_1, [S_3, D_3, D_2], A_2, -_1, -_2, -_3$$

- Ideal protocol substitutes for his sequence $[S_3, D_3, D_2]$ the sequence $[D_2, S_3, D_3]$.

<u>CONSTRAINTS</u>

Base:

$C_i > C_{i+1}$
$X_i > C_i$ $\qquad X = F, A, S, D$
$S_i > D_i$
$A_i > -_i$

Common:

$F_{i+1} > A_i$
$S_{i+1} > A_i$
$A_i > D_{i+1}$

Weave :

$D_{i+1} > -_i$
$-_i > C_{i+1}$

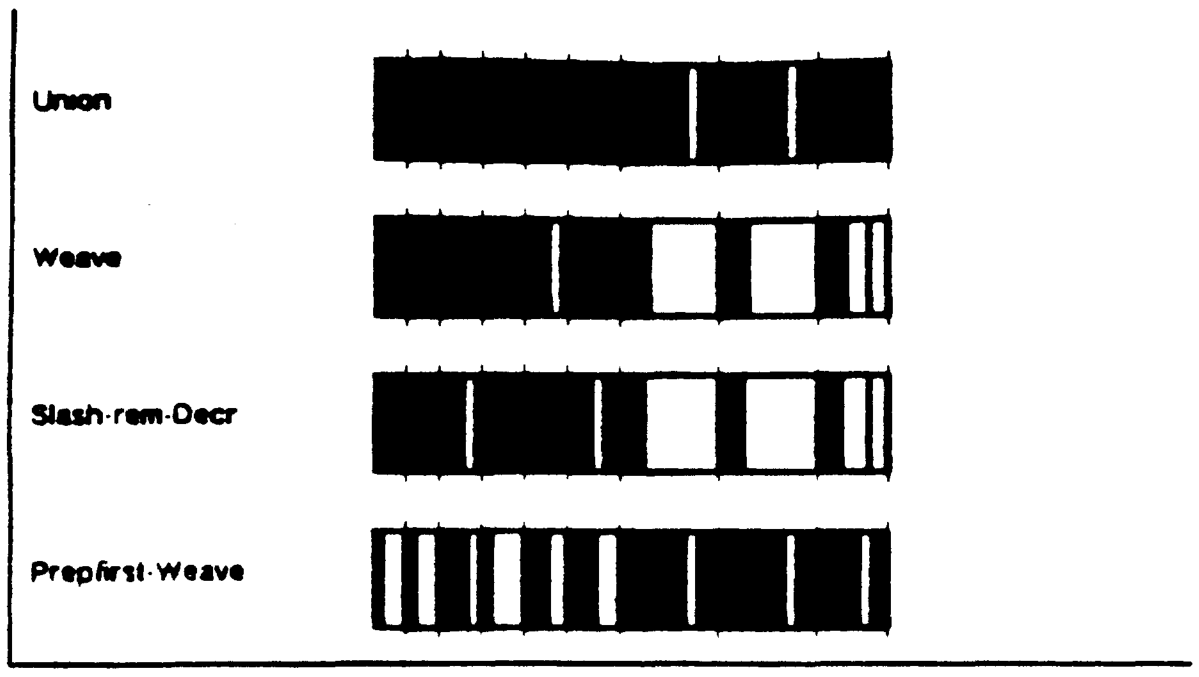
Slash-Reminds-to-Decrement:

$-_i > D_{i+1}$
$-_i > C_{i+1}$

Prepfirst-Weave (Does all top processing in Weave-like manner before writing any answers):

$-_i > -_{i+j}$
$X > -$ $\qquad X = C, F, A, S, D$

COMPARISON OF AGENDA SELECTIONS: PETE



Protocol and analyses for Robby

$$\begin{array}{r} 885 \\ -\ 205 \\ \hline \end{array}$$ $-_1, -_2, -_3$

$$\begin{array}{r} 8305 \\ -\ \ \ \ \ 3 \\ \hline \end{array}$$ $-_1, -_2, -_3, -_4$

$$\begin{array}{r} 83 \\ -\ 44 \\ \hline \end{array}$$ $\mathrm{SD}_2, \mathrm{A}_1, -_1, -_2$

$$\begin{array}{r} 907 \\ -\ 607 \\ \hline \end{array}$$ $-_1, -_2, \{-_3\}$

- He math-slips at column 3: 9 - 6 = 4.

$$\begin{array}{r} 106 \\ -\ 70 \\ \hline \end{array}$$ $-_1$, A_2, SD_3, $-_2$, ["correct" ans $_2$], $-_3$

- He "corrects" his answer in column 2. The ideal protocol sequence leaves it alone.

$$\begin{array}{r} 6591 \\ -\ 2697 \\ \hline \end{array}$$ $\mathrm{A}_1, \mathrm{SD}_2, [-_2], \mathrm{A}_3, \mathrm{SD}_4, -_4, -_3, -_1$

- He does 8 - 9 = 1. Ideal protocol does Borrow procedure, so for $[-_2]$, the ideal does $[\mathrm{A}_2, -_2, \mathrm{SD}_3]$.

$$\begin{array}{r} 108 \\ -\ 60 \\ \hline \end{array}$$ $-_1$, A_2, SD_3, $-_3$, $-_2$, [correct ans $_2$]

- He math-slips at column 2 answer and corrects. Ideal gets it right.

$$\begin{array}{r} 1236 \\ -\ 497 \\ \hline \end{array}$$ $\mathrm{A}_1, \mathrm{S}\star_2, -_1, \mathrm{A}_2, \mathrm{S}\star_3, -_2, \mathrm{A}_3, -_3, \star, -_4$

- He does not write his decrement in column 2 or column 3, and fails to Borrow from column 4. Ideal adds these steps.

$$\begin{array}{r} 1813 \\ -\ 215 \\ \hline \end{array}$$ $\mathrm{A}_1, \{-_1\}, \star, -_2, \mathrm{SD}_3, -_3, -_4$

- Math-slip at column 1. Robby does not write his SD_2 or A_2. Ideal protocol does.

$$\begin{array}{r} 102 \\ -\ 39 \\ \hline \end{array}$$ $\mathrm{A}_1, \{-_1\}, \mathrm{A}_2, \mathrm{SD}_3, -_2, -_3$

- Math-slip in column 1: 12 - 9 = 4.

$$\begin{array}{r} 9007 \\ -\ 6880 \\ \hline \end{array} \quad -_1, A_2, -_2, A_3, -_3, [SD_4], -_4$$

• He decrements 9 to 7. Possibly he accumulates decrements, but we've modelled his Borrow-From-Zero as a No-Operation Borrow-From-Zero, so we don't catch this, and decrement 9 to 8.

$$\begin{array}{r} 4015 \\ -\ 607 \\ \hline \end{array} \quad A_1, -_1, * \ D_2, -_2, A_3, SD_4, -_3, -_4$$

• He does not write the Slash in column 2. Ideal protocol does.

$$\begin{array}{r} 104 \\ -\ 27 \\ \hline \end{array} \quad [S_3 *], A_1, -_1, A_2, -_2, -_3$$

• He does not write the decrement in column 3. Ideal protocol does. Also, we can't get his order, given the No-Operation Borrow-From-Zero.

The idealizations at problems 11 and 13 point to problems with our model of Robby at the core procedure level, i.e. giving him a No-Operation Borrow-From-Zero.

CONSTRAINTS

Base:

$C_i > C_{i+1}$

$X_i > C_i$ $\quad X = A, F, SD$

$A_i > -_i$

Reverse (When processing a Borrow goal, do A, then F, then -):

$A_i > F_{i+1}$

$A_i > SD_{i+1}$

$F_{i+1} > -_i$

$SD_{i+1} > -_i$

Onepass (Do A and - before doing F):

$A_i > F_{i+1}$

$A_i > SD_{i+1}$

$-_i > F_{i+1}$

$-_i > SD_{i+1}$

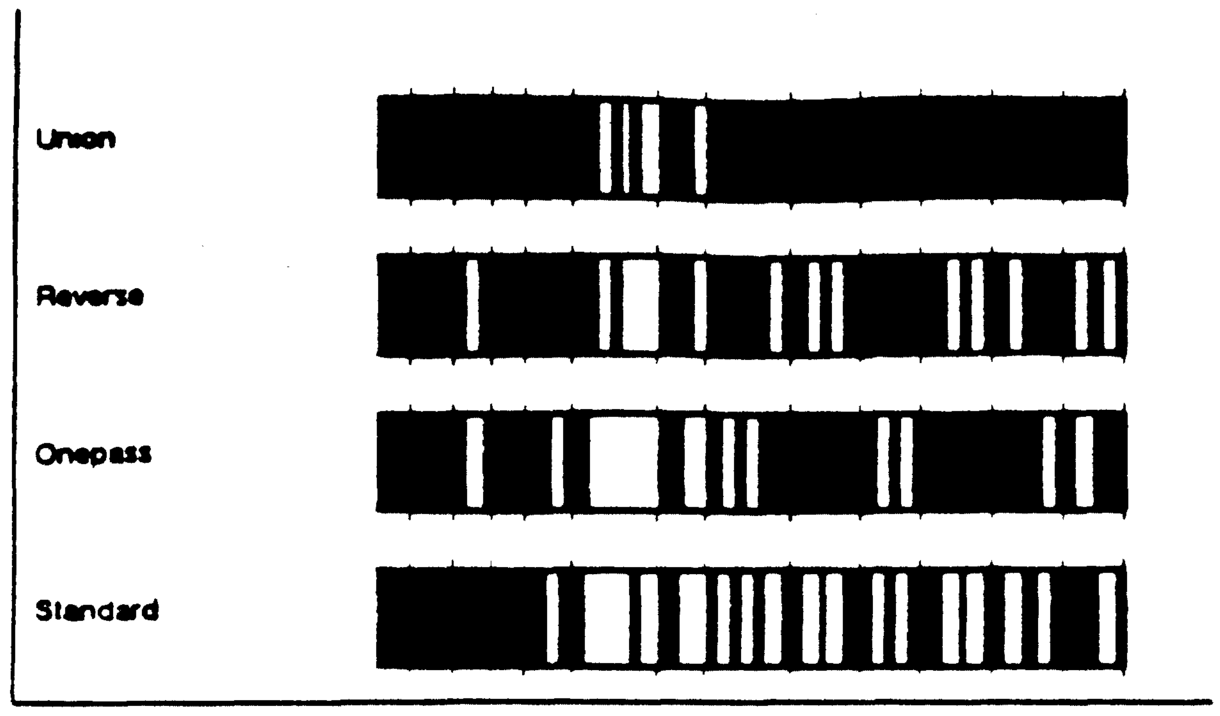
Standard: **(NB: He only uses this order on problem 3)**

$F_{i+1} > A_i$

$SD_{i+1} > A_i$

$A_i > C_{i+1}$

COMPARISON OF AGENDA SELECTIONS: ROBBY



Protocol and analyses for Tanya

$$\begin{array}{r} 647 \\ -\ \ 45 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 885 \\ -\ 205 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 83 \\ -\ 44 \\ \hline \end{array}$$ A_1, SD_2, $-_1$, $-_2$

$$\begin{array}{r} 8305 \\ -\ \ \ \ 3 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$, $-_4$

$$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array}$$ $-_1$, $-_2$

$$\begin{array}{r} 106 \\ -\ \ 70 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 716 \\ -\ 598 \\ \hline \end{array}$$ A_1, SD_2, SD_3, A_2, $-_1$, $-_2$, [rewrite ans $_2$], $-_3$

- She rewrites her initially incorrect answer in column 2. Ideal does not.

$$\begin{array}{r} 311 \\ -\ 214 \\ \hline \end{array}$$ A_1, SD_2, SD_3, A_2, $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 102 \\ -\ \ 39 \\ \hline \end{array}$$ A_1, 9_2, SD_3, $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 9007 \\ -\ 6880 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$, $-_4$

$$\begin{array}{r} 4015 \\ -\ \ 607 \\ \hline \end{array}$$ A_1, SD_2, [A_2, 9_3, $\star$], $-_1$, $-_2$, $-_3$, $-_4$

- She slips and does a borrow originating in column 2. Ideal does not.

$$\begin{array}{r} 702 \\ -\ 108 \\ \hline \end{array}$$ A_1 , 9_2 , SD_3, $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 205 \\ -\ 30 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$

$$\begin{array}{r} 100 \\ -\ 60 \\ \hline \end{array}$$ $-_1$, $-_2$, $-_3$

<u>CONSTRAINTS</u>

Base:

$A_i > -_i$

$X_i > C_i$ X = A, F, 9, SD (i.e., for the same column)

$C_i > C_{i+1}$

Common = Nailed

$X_i > -_j$ $X \neq -$

$-_i > -_{i+1}$

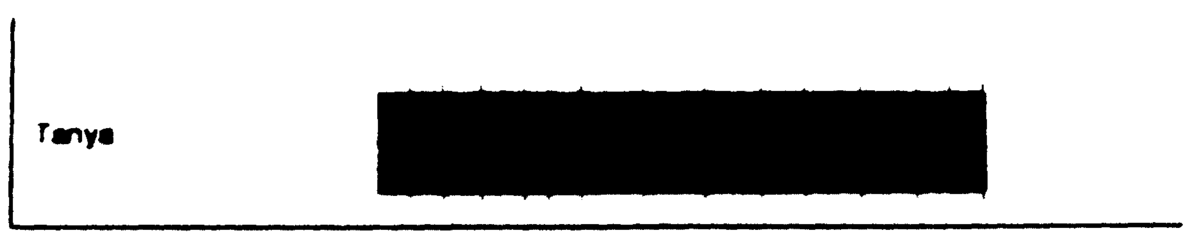
$X_i > C_j$ X = A, F, 9, SD (i.e., for any column)

$9_i > F_{i+1}$

$\left.\begin{array}{l} A_1 > F_2 \\ F_2 > A_1 \end{array}\right\}$ Note the completely regular flip-flop she does.

$SD_{i+1} > A_i$

COMPARISON OF AGENDA SELECTIONS: TANYA



Protocol and analyses for Trina

$$\begin{array}{r} 50 \\ -\ 23 \\ \hline \end{array} \qquad SD_2, A_1, -_1, -_2$$

$$\begin{array}{r} 562 \\ -\ \ \ 3 \\ \hline \end{array} \qquad A_1, -_1, SD_2, -_2, -_3$$

$$\begin{array}{r} 742 \\ -\ 136 \\ \hline \end{array} \qquad A_1, -_1, SD_2, -_2, -_3$$

$$\begin{array}{r} 106 \\ -\ \ 70 \\ \hline \end{array} \qquad -_1, \text{[corrects ans }_1\text{]}, A_2, -_2, SD_3, \star$$

- She corrects her column 1 answer; she does not write 0 in column 3.

$$\begin{array}{r} 716 \\ -\ 598 \\ \hline \end{array} \qquad A_1, -_1, SD_2, SD_3, A_2, -_2, -_3$$

$$\begin{array}{r} 102 \\ -\ \ 39 \\ \hline \end{array} \qquad A_1, -_1, A_2, SD_3, -_2, -_3$$

$$\begin{array}{r} 9007 \\ -\ 6880 \\ \hline \end{array} \qquad -_1, A_2, A_3, SD_3, -_2, -_3$$

$$\begin{array}{r} 4015 \\ -\ \ 607 \\ \hline \end{array} \qquad A_1, -_1, SD_2, -_2, A_3, -_3, SD_4, -_4$$

$$\begin{array}{r} 702 \\ -\ 108 \\ \hline \end{array} \qquad [SD_2], A_1, -_1, A_2, SD_3, -_3, -_2$$

- She does a weird SD of 0 in column 2, which the ideal protocol does not do.

$$\begin{array}{r} 2006 \\ -\ \ \ 42 \\ \hline \end{array} \qquad -_1, A_2, A_3, SD_4, -_4, -_3, -_2$$

$$\begin{array}{r} 10012 \\ -\ \ \ 214 \\ \hline \end{array} \qquad A_1, -_1, SD_2, A_3, A_4, SD_5, A_2, -_2, -_3, -_4, -_5$$

8001 - 43	**A_1, $-_1$, [S_2], A_3, SD_4, [$-_4$, $-_2$, A_2], [write 0 in column 2]**

• She does a Slash in column 2, and a write 0 over her 10 at some point. We also did not model the [$-_4$, $-_2$, A_2] sequence, using instead [A_2, $-_2$, $-_3$, $-_4$], after the order in the previous problem, 10012 - 214.

CONSTRAINTS

Base:

$A_i > -_i$
$X_i > C_i$ X = A, F, SD (for same column)
$C_i > C_{i+1}$

Common:

$X_i > C_j$ X = A, F, SD (for all columns)
$-_i > C_i$

Onepass (When processing a Borrow goal, do A and - before doing the F):

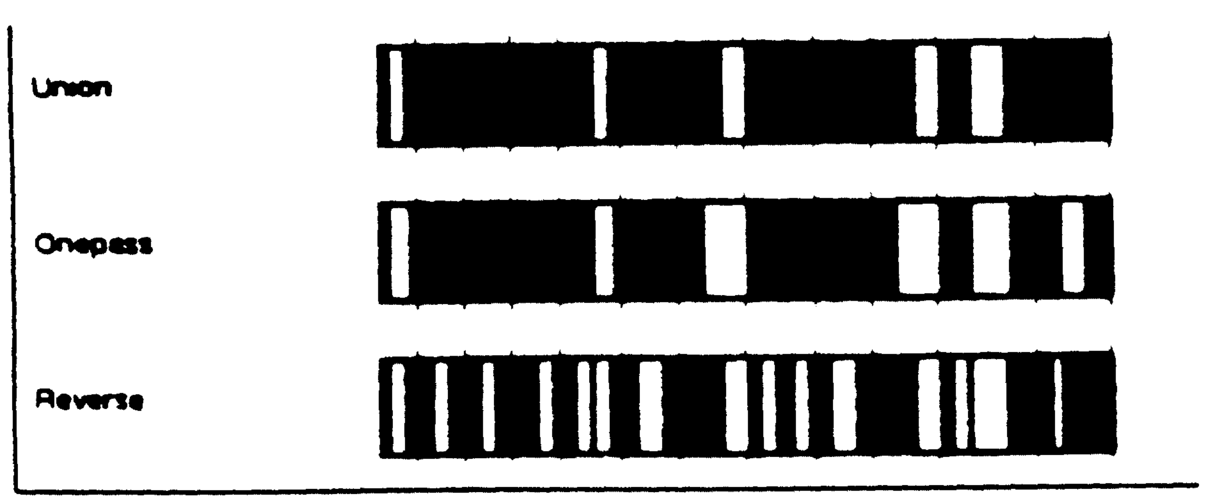
$A_i > F_{i+1}$
$-_i > F_{i+1}$

Reverse (When processing a Borrow goal, do A, then F, then come back for the -):

$F_{i+1} > -_i$
$SD_{i+1} > -_i$
$\left.\begin{array}{l} A_{i+j} > -_i \\ F_{i+j} > -_i \\ SD_{i+j} > -_i \end{array}\right\}$ Needed to handle her Borrow-From -Zero procedure

---

COMPARISON OF AGENDA SELECTIONS: TRINA



Dr. Phillip L. Ackerman
University of Minnesota
Department of Psychology
Minneapolis, MN 55455

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

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

Dr. Ed Aiken
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr. Robert Aiken
Temple University
School of Business Administration
Department of Computer and Information Sciences
Philadelphia, PA 19122

Dr. James Algina
University of Florida
Gainesville, FL 32605

Dr. John Allen
Department of Psychology
George Mason University
4400 University Drive
Fairfax, VA 22030

Dr. William E. Alley
AFHRL/MOT
Brooks AFB, TX 78235

Dr. John R. Anderson
Department of Psychology
Carnegie-Mellon University
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. Steve Andriole
George Mason University
School of Information Technology & Engineering
4400 University Drive
Fairfax, VA 22030

Technical Director, ARI
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Alan Baddeley
Medical Research Council
Applied Psychology Unit
15 Chaucer Road
Cambridge CB2 2EF
ENGLAND

Dr. Patricia Baggett
University of Colorado
Department of Psychology
Box 345
Boulder, CO 80309

Dr. Eva L. Baker
UCLA Center for the Study of Evaluation
145 Moore Hall
University of California
Los Angeles, CA 90024

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

Dr. Isaac Bejar
Educational Testing Service
Princeton, NJ 08450

Leo Beltracchi
United States Nuclear Regulatory Commission
Washington DC 20555

Dr. Mark H. Bickhard
University of Texas
EDB 504 ED Psych
Austin, TX 78712

Dr. John Black
Teachers College
Columbia University
525 West 121st Street
New York, NY 10027

Dr. Arthur S. Blaiwes
Code N711
Naval Training Systems Center
Orlando, FL 32813

Dr. Robert Blanchard
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr. R. Darrell Bock
University of Chicago
NORC
6030 South Ellis
Chicago, IL 60637

Dr. Jeff Bonar
Learning R&D Center
University of Pittsburgh
Pittsburgh, PA 15260

Dr. Richard Braby
NTSC Code 10
Orlando, FL 32751

Dr. Jomills H. Braddock II
Center for the Social Organization of Schools
The Johns Hopkins University
3505 North Charles Street
Baltimore, MD 21218

Dr. Robert Breaux
Code N-095R
Naval Training Systems Center
Orlando, FL 32813

Dr. Ann Brown
Center for the Study of Reading
University of Illinois
51 Gerty Drive
Champaign, IL 61280

Commanding Officer
CAPT Lorin W. Brown
NROTC Unit
Illinois Institute of Technology
3300 S. Federal Street
Chicago, IL 60616-3793

Dr. John S. Brown
XEROX Palo Alto Research Center
3333 Coyote Road
Palo Alto, CA 94304

Dr. John Bruer
The James S. McDonnell Foundation
University Club Tower, Suite 1610
1034 South Brentwood Blvd
St. Louis, MO 63117

Dr. Bruce Buchanan
Computer Science Department
Stanford University
Stanford, CA 94305

Dr. Patricia A. Butler
OERI
555 New Jersey Ave. NW
Washington, DC 20208

Dr. Tom Cafferty
Dept. of Psychology
University of South Carolina
Columbia, SC 29208

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

Joanne Capper
Center for Research into Practice
1718 Connecticut Ave., N.W.
Washington, DC 20009

Dr. Susan Carey
Harvard Graduate School of Education
337 Gutman Library
Appian Way
Cambridge, MA 02138

Dr. Pat Carpenter
Carnegie-Mellon University
Department of Psychology
Pittsburgh, PA 15213

Dr. John M. Carroll
IBM Watson Research Center
User Interface Institute
P.O. Box 218
Yorktown Heights, NY 10598

LCDR Robert Carter
Office of the Chief
of Naval Operations
OP-01B
Pentagon
Washington, DC 20350-2000

Dr. Alphonse Chapanis
8415 Bellona Lane
Suite 210
Buxton Towers
Baltimore, MD 21204

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

Dr. Paul R. Chatelier
OUSDRE
Pentagon
Washington, DC 20350-2000

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

Dr. L. J. Chmura
Computer Science and Systems
Code 7590
Information Technology Division
Naval Research Laboratory
Washington, DC 20375

Mr. Raymond E. Christal
AFHRL/MOE
Brooks AFB, TX 78235

Dr. William Clancey
Stanford University
Knowledge Systems Laboratory
701 Welch Road, Bldg. C
Palo Alto, CA 94304

Dr. Charles Clifton
Tobin Hall
Department of Psychology
University of
Massachusetts
Amherst, MA 01003

Dr. Allan M. Collins
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

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

Dr. Lynn A. Cooper
Learning R&D Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

LT Judy Crookshanks
Chief of Naval Operations
OP-112G5
Washington, DC 20370-2000

Phil Cunniff
Commanding Officer, Code 7522
Naval Undersea Warfare Engineering
Keyport, WA 98345

Dr. Cary Czichon
Intelligent Instructional Systems
Texas Instruments AI Lab
P.O. Box 660245
Dallas, TX 75266

Brian Dallman
3400 TTW/TTGXS
Lowry AFB, CO 80230-5000

Dr. Natalie Dehn
Department of Computer and
Information Science
University of Oregon
Eugene, OR 97403

Dr. Gerald F. DeJong
Artificial Intelligence Group
Coordinated Science Laboratory
University of Illinois
Urbana, IL 61801

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

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

Dr. Andrea di Sessa
University of California
School of Education
Tolman Hall
Berkeley, CA 94720

Dr. R. K. Dismukes
Associate Director for Life Sciences
AFOSR
Bolling AFB
Washington, DC 20332

Dr. Stephanie Doan
Code 6021
Naval Air Development Center
Warminster, PA 18974-5000

Defense Technical
Information Center
Cameron Station, Bldg 5
Alexandria, VA 22314
Attn: TC
(12 Copies)

Dr. Thomas M. Duffy
Communications Design Center
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Dr. Richard Duran
University of California
Santa Barbara, CA 93106

Edward E. Eddowes
CNATRA N301
Naval Air Station
Corpus Christi, TX 78419

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

Dr. Jeffrey Elman
University of California,
San Diego
Department of Linguistics, C-008
La Jolla, CA 92093

Dr. Susan Embretson
University of Kansas
Psychology Department
426 Fraser
Lawrence, KS 66045

Dr. Randy Engle
Department of Psychology
University of South Carolina
Columbia, SC 29208

Dr. William Epstein
University of Wisconsin
W. J. Brogden Psychology Bldg
1202 W. Johnson Street
Madison, WI 53706

ERIC Facility-Acquisitions
4833 Rugby Avenue
Bethesda, MD 20014

Dr. K. Anders Ericsson
University of Colorado
Department of Psychology
Boulder, CO 80309

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

Dr. Marshall J. Farr
Farr-Sight Co.
2520 North Vernon Street
Arlington, VA 22207

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

Mr. Wallace Feurzeig
Educational Technology
Bolt Beranek & Newman
10 Moulton St.
Cambridge, MA 02238

Dr. Gerhard Fischer
University of Colorado
Department of Computer Science
Boulder, CO 80309

J. D. Fletcher
9931 Corsica Street
Vienna VA 22180

Dr. Linda Flower
Carnegie-Mellon University
Department of English
Pittsburgh, PA 15213

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
McGill University
3700 McTavish Street
Montreal, Quebec H3A 1Y2
CANADA

Dr. John R. Frederiksen
Bolt Beranek & Newman
50 Moulton Street
Cambridge, MA 02138

Dr. Michael Genesereth
Stanford University
Computer Science Department
Stanford, CA 94305

Dr. Dedre Gentner
University of Illinois
Department of Psychology
603 E. Daniel St.
Champaign, IL 61820

Lee Gladwin
Route 3 -- Box 225
Winchester, VA 22601

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

Dr. Arthur M. Glenberg
University of Wisconsin
W. J. Brogden Psychology Bldg.
1202 W. Johnson Street
Madison, WI 53706

Dr. Marvin D. Glock
13 Stone Hall
Cornell University
Ithaca, NY 14853

Dr. Sam Glucksberg
Department of Psychology
Princeton University
Princeton, NJ 08540

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

Dr. Susan Goldman
University of California
Santa Barbara, CA 93106

Dr. Daniel Gopher
Industrial Engineering
& Management
TECHNION
Haifa 32000
ISRAEL

Dr. Sherrie Gott
AFHRL/MODJ
Brooks AFB, TX 78235

Jordan Grafman, Ph.D.
2021 Lyttonsville Road
Silver Spring, MD 20910

Dr. Wayne Gray
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Bert Green
Johns Hopkins University
Department of Psychology
Charles & 34th Street
Baltimore, MD 21218

Dr. James G. Greeno
University of California
Berkeley, CA 94720

Prof. Edward Haertel
School of Education
Stanford University
Stanford, CA 94305

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

Janice Hart
Office of the Chief
of Naval Operations
OP-11HD
Department of the Navy
Washington, D.C. 20350-2000

Mr. William Hartung
PEAM Product Manager
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

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

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

Dr. Barbara Hayes-Roth
Department of Computer Science
Stanford University
Stanford, CA 95305

Dr. Joan I. Heller
505 Haddon Road
Oakland, CA 94606

Dr. Shelly Heller
Department of Electrical Engi-
neering & Computer Science
George Washington University
Washington DC 20052

Dr. Jim Hollan
Intelligent Systems Group
Institute for
Cognitive Science (C-015)
UCSD
La Jolla, CA 92093

Dr. Melissa Holland
Army Research Institute for the
Behavioral and Social Science
5001 Eisenhower Avenue
Alexandria, VA 22333

Ms. Julia S. Hough
Lawrence Erlbaum Associates
6012 Greene Street
Philadelphia, PA 19144

Dr. James Howard
Dept. of Psychology
Human Performance Laboratory
Catholic University of
America
Washington DC 20064

Dr. Earl Hunt
Department of Psychology
University of Washington
Seattle, WA 98105

Dr. Ed Hutchins
Intelligent Systems Group
Institute for
Cognitive Science (C-015)
UCSD
La Jolla, CA 92093

Dr. Dillon Inouye
WICAT Education Institute
Provo, UT 84057

Dr. Alice Isen
Department of Psychology
University of Maryland
Catonsville, MD 21228

Dr. R. J. K. Jacob
Computer Science and Systems
Code 7590
Information Technology Division
Naval Research Laboratory
Washington, DC 20375

Dr. Zachary Jacobson
Bureau of Management Consulting
365 Laurier Avenue West
Ottawa, Ontario K1A 0S5
CANADA

Dr. Robert Jannarone
Department of Psychology
University of South Carolina
Columbia, SC 29208

Dr. Claude Janvier
Directeur, CIRADE
Universite du Quebec a Montreal
P.O. Box 8888, St. "A"
Montreal, Quebec H3C 3P8
CANADA

Dr. Robin Jeffries
Hewlett-Packard Laboratories
P.O. Box 10490
Palo Alto, CA 94303-0971

Margaret Jerome
c/o Dr. Peter Chandler
83, The Drive
Hove
Sussex
UNITED KINGDOM

Dr. Douglas H. Jones
Thatcher Jones Associates
P.O. Box 6640
10 Trafalgar Court
Lawrenceville, NJ 08648

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. Milton S. Katz
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Dennis Kibler
University of California
Department of Information
and Computer Science
Irvine, CA 92717

Dr. David Kieras
University of Michigan
Technical Communication
College of Engineering
1223 E. Engineering Building
Ann Arbor, MI 48109

Dr. Peter Kincaid
Training Analysis
& Evaluation Group
Department of the Navy
Orlando, FL 32813

Dr. Paula Kirk
Oakridge Associated Universities
University Programs Division
P.O. Box 117
Oakridge, TN 37831-0117

Dr. David Klahr
Carnegie-Mellon University
Department of Psychology
Schenley Park
Pittsburgh, PA 15213

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

Dr. Kenneth Kotovsky
Department of Psychology
Community College of
Allegheny County
800 Allegheny Avenue
Pittsburgh, PA 15233

Dr. Benjamin Kuipers
University of Texas at Austin
Department of Computer Sciences
T.S. Painter Hall 3.28
Austin, TX 78712

Dr. Pat Langley
University of California
Department of Information
and Computer Science
Irvine, CA 92717

M. Diane Langston
Communications Design Center
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Dr. Jill Larkin
Carnegie-Mellon University
Department of Psychology
Pittsburgh, PA 15213

Dr. R. W. Lawler
ARI 6 S 10
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

Dr. Alan M. Lesgold
Learning Research and
Development Center
University of Pittsburgh
Pittsburgh, PA 15260

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

Dr. Clayton Lewis
University of Colorado
Department of Computer Science
Campus Box 430
Boulder, CO 80309

Library
Naval War College
Newport, RI 02940

Library
Naval Training Systems Center
Orlando, FL 32813

Dr. Charlotte Linde
Structural Semantics
P.O. Box 707
Palo Alto, CA 94320

Dr. Marcia C. Linn
Lawrence Hall of Science
University of California
Berkeley, CA 94720

Dr. Frederic M. Lord
Educational Testing Service
Princeton, NJ 08541

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

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

Dr. Jay McClelland
Department of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

Dr. Joe McLachlan
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr. James S. McMichael
Navy Personnel Research
and Development Center
Code 05
San Diego, CA 92152

Dr. Barbara Means
Human Resources
Research Organization
1100 South Washington
Alexandria, VA 22314

Dr. Arthur Melmed
U. S. Department of Education
724 Brown
Washington, DC 20208

Dr. George A. Miller
Department of Psychology
Green Hall
Princeton University
Princeton, NJ 08540

Dr. James R. Miller
MCC
9430 Research Blvd.
Echelon Building #1, Suite 231
Austin, TX 78759

Dr. Mark Miller
Computer*Thought Corporation
1721 West Plano Parkway
Plano, TX 75075

Dr. Andrew R. Molnar
Scientific and Engineering
Personnel and Education
National Science Foundation
Washington, DC 20550

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

Dr. Randy Mumaw
Program Manager
Training Research Division
HumRRO
1100 S. Washington
Alexandria, VA 22314

Dr. Allen Munro
Behavioral Technology
Laboratories - USC
1845 S. Elena Ave., 4th Floor
Redondo Beach, CA 90277

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

Dr. Richard E. Nisbett
University of Michigan
Institute for Social Research
Room 5261
Ann Arbor, MI 48109

Dr. Mary Jo Nissen
University of Minnesota
N218 Elliott Hall
Minneapolis, MN 55455

Dr. A. F. Norcio
Computer Science and Systems
Code 7590
Information Technology Division
Naval Research Laboratory
Washington, DC 20375

Dr. Donald A. Norman
Institute for Cognitive
Science C-015
University of California, San Diego
La Jolla, California 92093

Director, Training Laboratory
NPRDC (Code 05)
San Diego, CA 92152-6800

Director, Manpower and Personnel
Laboratory,
NPRDC (Code 06)
San Diego, CA 92152-6800

Director, Human Factors
& Organizational Systems Lab,
NPRDC (Code 07)
San Diego, CA 92152-6800

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

Technical Director
Navy Personnel R&D Center
San Diego, CA 92152-6800

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. Michael Oberlin
Naval Training Systems Center
Code 711
Orlando, FL 32813-7100

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

Director, Research Programs,
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

Office of Naval Research,
Code 1133
800 N. Quincy Street
Arlington, VA 22217-5000

Office of Naval Research,
Code 1142PS
800 N. Quincy Street
Arlington, VA 22217-5000

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

Office of Naval Research,
Code 11R
800 N. Quincy Street
Arlington, VA 22217-5000

Director, Technology Programs
Office of Naval Research
Code 12
800 North Quincy Street
Arlington, VA 22217-5000

Office of Naval Research,
Code 125
800 N. Quincy Street
Arlington, VA 22217-5000

Psychologist
Office of Naval Research
Branch Office, London
Box 39
FPO New York, NY 09510

Special Assistant for Marine
Corps Matters,
ONR Code 00MC
800 N. Quincy St.
Arlington, VA 22217-5000

Psychologist
Office of Naval Research
Liaison Office, Far East
APO San Francisco, CA 96503

Office of Naval Research,
Resident Representative,
UCSD
University of California,
San Diego
La Jolla, CA 92093-0001

Assistant for Planning MANTR,
OP 01B6
Washington, DC 20370

Assistant for MPT Research, Development and Studies
OP 01B7
Washington, DC 20370

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

CDR R. T. Parlette
Chief of Naval Operations
OP-112G
Washington, DC 20370-2000

Dr. James Paulson
Department of Psychology
Portland State University
P. O. Box 751
Portland, OR 97207

Dr. Douglas Pearse
DCIEM
Box 2000
Downsview, Ontario
CANADA

Dr. James W. Pellegrino
University of California, Santa Barbara
Department of Psychology
Santa Barbara, CA 93106

Dr. Virginia E. Pendergrass
Code 711
Naval Training Systems Center
Orlando, FL 32813-7100

Dr. Nancy Pennington
University of Chicago
Graduate School of Business
1101 E. 58th St.
Chicago, IL 60637

Military Assistant for Training and Personnel Technology,
OUSD (R & E)
Room 3D129, The Pentagon
Washington, DC 20301-3080

Dr. Ray Perez
ARI (PERI-II)
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. David N. Perkins
Educational Technology Center
337 Gutman Library
Appian Way
Cambridge, MA 02138

Dr. Steven Pinker
Department of Psychology
E10-018
M. I. T.
Cambridge, MA 02139

Dr. Tjeerd Plomp
Twente University of Technology
Department of Education
P. O. Box 217
7500 AE ENSCHEDE
THE NETHERLANDS

Dr. Martha Polson
Department of Psychology
Campus Box 346
University of Colorado
Boulder, CO 80309

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

Dr. Michael I. Posner
Department of Neurology
Washington University Medical School
St. Louis, MO 63110

Dr. Joseph Psotka
ATTN: PERI-1C
Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

Dr. Mark D. Reckase
ACT
P. O. Box 168
Iowa City, IA 52243

Dr. Lynne Reder
Department of Psychology
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Dr. Wesley Regian
AFHRL/MOD
Brooks AFB, TX 78235

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

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

Dr. Gil Ricard
Mail Stop C04-14
Grumman Aerospace Corp.
Bethpage, NY 11714

Mark Richer
1041 Lake Street
San Francisco, CA 94118

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

Dr. Andrew M. Rose
American Institutes for Research
1055 Thomas Jefferson St., NW
Washington, DC 20007

Dr. David Rumelhart
Center for Human Information Processing
Univ. of California
La Jolla, CA 92093

Dr. James F. Sanford
Department of Psychology
George Mason University
4400 University Drive
Fairfax, VA 22030

Dr. Walter Schneider
Learning R&D Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15260

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

Dr. Janet Schofield
Learning R&D Center
University of Pittsburgh
Pittsburgh, PA 15260

Karen A. Schriver
Department of English
Carnegie-Mellon University
Pittsburgh, PA 15213

Dr. Marc Sebrechts
Department of Psychology
Wesleyan University
Middletown, CT 06475

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

Dr. Colleen M. Seifert
Intelligent Systems Group
Institute for Cognitive Science (C-015)
UCSD
La Jolla, CA 92093

Dr. Ramsay W. Selden
Assessment Center
CCSSO
Suite 379
400 N. Capitol, NW
Washington, DC 20001

Dr. Sylvia A. S. Shafto
Department of
Computer Science
Towson State University
Towson, MD 21204

Dr. Ben Shneiderman
Dept. of Computer Science
University of Maryland
College Park, MD 20742

Dr. Lee Shulman
Stanford University
1040 Cathcart Way
Stanford, CA 94305

Dr. Randall Shumaker
Naval Research Laboratory
Code 7510
4555 Overlook Avenue, S W
Washington, DC 20375-5000

Dr. Valerie Shute
AFHRL/MOE
Brooks AFB, TX 78235

Dr Robert S Siegler
Carnegie-Mellon University
Department of Psychology
Schenley Park
Pittsburgh, PA 15213

Dr Zita M Simutis
Instructional Technology
Systems Area
ARI
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. H. Wallace Sinaiko
Manpower Research
and Advisory Services
Smithsonian Institution
801 North Pitt Street
Alexandria, VA 22314

Dr. Derek Sleeman
Dept. of Computing Science
King's College
Old Aberdeen
AB9 2UB
UNITED KINGDOM

Dr Richard E. Snow
Department of Psychology
Stanford University
Stanford, CA 94306

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

Dr Kathryn T Spoehr
Brown University
Department of Psychology
Providence, RI 02912

James J Staszewski
Research Associate
Carnegie-Mellon University
Department of Psychology
Schenley Park
Pittsburgh, PA 15213

Dr Robert Sternberg
Department of Psychology
Yale University
Box 11A, Yale Station
New Haven, CT 06520

Dr Albert Stevens
Bolt Beranek & Newman, Inc
10 Moulton St
Cambridge, MA 02238

Dr Paul J Sticha
Senior Staff Scientist
Training Research Division
HumRRO
1100 S Washington
Alexandria, VA 22314

Dr Thomas Sticht
Navy Personnel R&D Center
San Diego, CA 92152-6800

Dr John Tangney
AFOSR/NL
Bolling AFB, DC 20332

Dr. Kikumi Tatsuoka
CERL
252 Engineering Research
Laboratory
Urbana, IL 61801

Dr. Robert P Taylor
Teachers College
Columbia University
New York, NY 10027

Dr. Perry W Thorndyke
FMC Corporation
Central Engineering Labs
1185 Coleman Avenue, Box 580
Santa Clara, CA 95052

Dr Sharon Tkacz
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Dr. Douglas Towne
Behavioral Technology Labs
1845 S. Elena Ave
Redondo Beach, CA 90277

Dr. Paul Twohig
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

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

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

Dr. Beth Warren
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

Dr. Barbara White
Bolt Beranek & Newman, Inc
10 Moulton Street
Cambridge, MA 02238

LCDR Cory deGroot Whitehead
Chief of Naval Operations
OP-112G1
Washington, DC 20370-2000

Dr. Heather Wild
Naval Air Development
Center
Code 6021
Warminster, PA 18974-5000

Dr William Clancey
Stanford University
Knowledge Systems Laboratory
701 Welch Road, Bldg C
Palo Alto, CA 94304

Dr Michael Williams
IntelliCorp
1975 El Camino Real West
Mountain View, CA 94040-2216

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

Dr Martin F Wiskoff
Navy Personnel R & D Center
San Diego, CA 92152-6800

Dr Dan Wolz
AFHRL/MOE
Brooks AFB, TX 78235

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

Dr Joe Yasatuke
AFHRL/LRT
Lowry AFB, CO 80230

Dr Joseph L Young
Memory & Cognitive
Processes
National Science Foundation
Washington, DC 20550

Dr Steven Zornetzer
Office of Naval Research
Code 114
800 N Quincy St
Arlington, VA 22217-5000