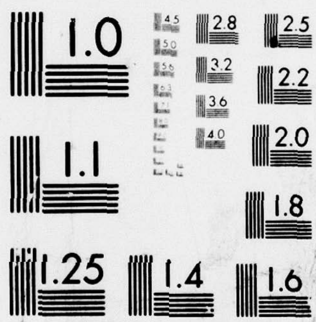
AD-A080 985 CALIFORNIA UNIV SAN DIEGO LA JOLLA CENTER FOR HUMAN --ETC F/G 5/9
COACH: A SCHEMA-BASED TUTOR.(U)
DEC 79 D R GENTNER N00014-76-C-0628
UNCLASSIFIED 7903 NL

1 OF 1
AD
A080985

END
DATE
FILMED
3-80
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12) LEVEL II

REPORT No. 7903
DECEMBER 1979

Donald R. Gentner

COACH: A SCHEMA-BASED TUTOR

CENTER FOR HUMAN INFORMATION PROCESSING
LA JOLLA, CALIFORNIA 92093

UNIVERSITY OF CALIFORNIA, SAN DIEGO

This research was supported by the Advanced Research Projects Agency and the Office of Naval Research, Personnel and Training Research Programs and was monitored by ONR under Contract N00014-76-C-0628, NR 154-387, under terms of ARPA Order No. 2284. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency, of the Office of Naval Research, or the United States Government.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: 7903	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Coach: A schema-based tutor		5. TYPE OF REPORT & PERIOD COVERED: Technical Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Donald R. Gentner		8. CONTRACT OR GRANT NUMBER(s): N00014-76-C-0628; N00014-79-C-0323
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Center for Human Information Processing, University of California, San Diego, La Jolla, CA 92093		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NR 154-387; NR 157-437
11. CONTROLLING OFFICE NAME AND ADDRESS: Personnel and Training Research Programs, Office of Naval Research (Code 458), Arlington, VA 22217		12. REPORT DATE: December 1979
		13. NUMBER OF PAGES: 20
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report): Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Automated tutor, computer-based instruction, individualized instruction, instructional theory, model of the student, representation of information, schema, semantic network, teaching.

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

The Coach system, a computer simulation of a human tutor, was constructed with the goal of obtaining a better understanding about how a tutor interprets the student's behavior, diagnoses difficulties, and gives advice to a student learning a simple computer programming language. Coach is based on a hierarchy of active schemas representing the tutor's general concepts, more specific information represented in a semantic network, and the coordination of data-driven and conceptually-guided processing which enable it to interpret behavior, recognize errors, and give advice.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF 014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

408267

Coach
A Schema-Based Tutor

Donald R. Gentner

Center for Human Information Processing
University of California, San Diego

The work reported in this paper was carried out in collaboration with Donald A. Norman, David E. Rumelhart, and Mark R. Wallen. This research was supported by the Office of Naval Research, Personnel and Training Research Programs and the Advanced Research Projects Agency, and was monitored by ONR under Contract N00014-76-C-0628, NR 154-387, under terms of ARPA Order No. 2284.

COACH: A SCHEMA-BASED TUTOR

Abstract

The Coach system, a computer simulation of a human tutor, was constructed with the goal of obtaining of a better understanding of how a tutor interprets the student's behavior, diagnoses difficulties, and gives advice. Coach gives advice to a student who is learning a simple computer programming language. Its intelligence is based on a hierarchy of active schemas which represent the tutor's general concepts, and on more specific information represented in a semantic network. The coordination of conceptually-guided and data-driven processing enables the Coach system to interpret student behavior, recognize errors, give advice to the student.

Introduction

This paper describes the Coach system, an intelligent computer-based instructional system which uses a schema representation of knowledge to interpret student actions and to give advice to the student. The Coach system gives advice to a student learning FLOW, a simple computer programming language. FLOW is similar to BASIC, with about fifteen different statements, three commands, and simple debugging aids. It is based on a language originally developed by Raskin (1974) as an introductory computer language. Earlier versions of the FLOW tutor used declarative representations of schemas (Gentner & Norman, 1977; Gentner, 1979), in contrast with the present Coach system where schemas are active processes.

The Coach system falls within the tradition of intelligent computer-assisted instructional systems originated by Carbonell (1970) and continued by such workers as Burton and Brown, Miller, and Goldstein (see the collection edited by Sleeman and Brown, 1979) The basic principles of the Coach system can be summarized briefly:

a) The Coach system is based on a set of active processes, called schemas, which interpret the student's behavior, detect errors, and generate advice to the student. The schemas form a nested hierarchy to represent generic concepts such as programming constructs, the structure of written text, teaching strategies, common errors, and individual keypresses.

b) Specific information, such as the instructional manual, the structure of the FLOW language, and the tutor's current model of the student, is represented in a semantic network.

c) Conceptually-guided processing. The tutor expands high-level schemas into their component lower-level schemas to predict and observe student behavior, solve programming problems, and give advice. When schemas are unable to find the predicted student behavior, they may suspend processing, but keep the entire structure of predictions intact for possible reactivation later.

d) Data-driven processing. Unexpected student actions activate low-level schemas which attempt to assemble themselves into structures of schemas representing errors or alternate correct actions.

e) Conceptually-guided and data-driven processes may interact in several ways. Higher-level schemas can incorporate already existing structures of schemas which have been generating by data-driven processing. Lower-level schemas can directly or indirectly activate high-level schemas which they hope will eventually incorporate them by conceptually-guided processing.

f) The sequence of processing is primarily controlled by the individual schemas which activate related higher or lower level schemas. There is also a global agenda of schemas waiting to be activated, typically containing one to three high-level schemas.

Basic structure and operation of schemas

The Coach system is implemented in Micro-Sol, a language which constructs, modifies and interprets active semantic network structures. Micro-Sol is based on Sol (Norman, Rumelhart & the LNR Research Group, 1975), but lacks the natural language capabilities of Sol. Micro-Sol currently runs on a PDP 11/45 minicomputer.

Schemas in the Coach system are used to represent concepts at many levels. At the highest level there are schemas such as those for instructional manuals and the functions of programs. There are schemas for program constructs such as loops, for common student errors, and for the statements in the FLOW language. At the lowest level there are the schemas for the individual keypresses and periods of student inactivity. Schema have arguments, which serve to distinguish the various instances of a given schema. I have used a family analogy to describe the hierarchically nested schemas: the sub-parts of a schema are referred to as the children of the schema; the schema is itself normally part of a higher-level schema known as the parent.

A schema in the Coach system corresponds to a restricted type of procedural definition in Micro-Sol. Figure 1 shows the general structure of a schema definition. When a schema becomes active, it searches for its component children by activating the corresponding schemas. If a child is absent, the schema normally suspends operation and returns to its parent with the truth value "maybe", indicating that it was unable to complete itself. If a suspended schema is later reactivated, the schema starts up again at the place where it suspended. When all its children have been found, the schema may perform some actions, such as updating the model of the student. Next the schema determines whether it was originally invoked by a higher-level parent schema. If not, i.e. when the schema is an orphan, the schema suggests some possible parents by putting their schemas on an agenda. Instead of merely suggesting possible parents, an orphaned schema which is more confident of its function may directly activate a potential parent. Finally, the schema returns to the procedure which originally activated it with a truth value of "true", indicating that the schema is complete. In the normal case where the schema has a parent, it would be returning to its parent. If a schema is completely unable to find its children, it can return to its parent with a truth value of "false".

The collection of schemas makes up a distributed intelligence system, without an overall executive procedure. The highest level process in the Coach system simply activates the next schema on the agenda. Information flow is primarily restricted to communication from parent to child (via the arguments used when invoking the child's schema) and from child to parent (a child returns with a truth value indicating whether or not it is complete). In addition, information represented in the semantic network is available to any schema. Many of the schemas maintain or have access to the model of the student, which allows them to coordinate a changing strategy as the student progresses thru the learning task. The Coach system also includes a FLOW simulator which maintains a semantic network representation of the student's terminal screen, current program, and command state.

```
define schema-1 (argument-1 argument-2)
        if 1st child is absent, suspend here and return (maybe)
        if 2nd child is absent, suspend here and return (maybe)
        if 3rd child is absent, suspend here and return (maybe)
        perform actions
        if orphaned,
                suggest possible-parent-1
                suggest possible-parent-2
        return (true).
```

Figure 1. The general structure of a schema.

Schema-based interpretation

The tutorial environment

To illustrate the operation of the schemas, I will give a series of examples showing how the Coach system analyzes a student's behavior, but first I need to describe the general instructional setup.

Figure 2 shows a typical experimental arrangement, in this case with a human tutor. On the right, a student is shown in top view, sitting at a terminal that is connected to a minicomputer where the student can enter and execute FLOW computer programs. The student has an instruction booklet that contains a general discussion of computers and computer languages, a description of the FLOW language, examples to try, and problems to solve. The minicomputer executes the student's FLOW programs and thus provides the student with feedback. The system also provides a line-at-a-time execution mode to help the student debug programs. The instructional booklet is fairly complete, and in principle the student could go through the whole booklet without help. Most students, however, have considerable difficulty in completing the instructional sequence without help. In the arrangement shown in Figure 2 there is a tutor in another room who sits in front of a CRT terminal which displays a copy of everything appearing on the student's terminal. The student and tutor can send messages to each other on a pair of interconnected terminals. There are no restrictions on the messages from the tutor, but the student's messages to the tutor are limited to "yes", "no", "ok", and "help".

We record the student's keypresses and the corresponding times for later use. Figure 3 shows an annotated portion of a student record, along with advice produced by the Coach system. The left column gives the time in seconds since the beginning of the session; the student's keypresses are to the right. When the student does not press a key for 11 seconds, the minicomputer indicates a quiet period. Although this record seems rather cryptic at first, human tutors experienced in teaching FLOW can use it to interpret student behavior almost as easily as they can interpret a copy of the student's terminal screen. The task of the Coach system is to interpret this record of keypresses in terms of the student's progress thru the instructional sequence and to give the student appropriate advice when needed. Our primary concern in this research is to learn what general knowledge, specific information, and processing capabilities a tutor must have to perform this task.

Conceptually-guided prediction

To illustrate how Coach uses conceptually-guided processing to predict and interpret student actions, I present an example in some detail, showing Coach's interpretation of the portion of the student protocol shown in Figure 3.

The example begins with Figure 4, which shows a trace of Coach's processing starting just after the student has pressed the R key to run the current program. That action has completed problem-8 and paragraph-13 in the FLOW instruction manual. The top level schema, *manual,* is now active. (The names of schemas are italicized.) The *manual* schema has a single argument, the title of the manual (in this case "The FLOW Manual"). It expects the student to complete a series of paragraphs in the instructional manual. The *manual* schema examines the node for "The FLOW Manual" in the semantic network to get the names of the paragraphs, and then activates the *paragraph* schema with the name of the next paragraph, paragraph-14, as its argument. This action is reflected in line 2 of the Coach trace in Figure 4. Figure 5 shows the definition of the *paragraph* schema, a typical schema. After first checking to see whether it contains a programming problem in addition to the text, the *paragraph* schema expects the student to complete the text and then updates the reading rate in its model of the student. Next it expects the student to complete the problem, if present. Finally, when all its children are complete, it normally returns to its parent, the *manual* schema. If it is orphaned, however, it will activate

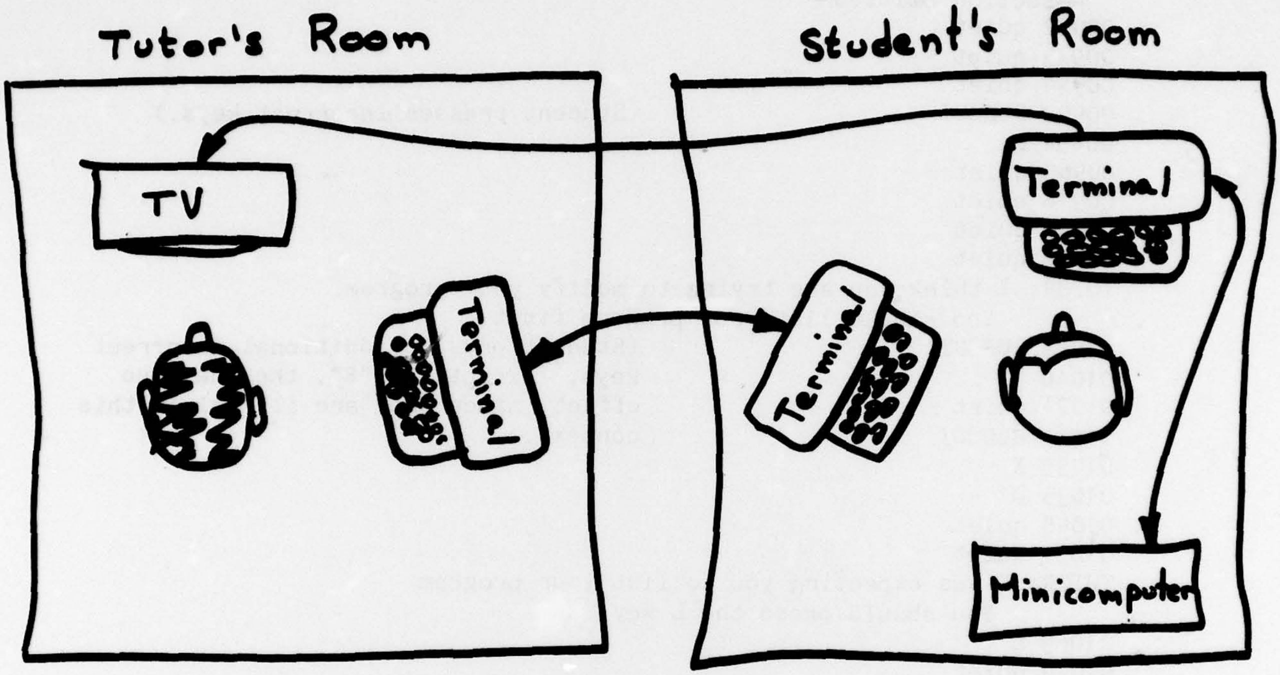


Figure 2. The FLOW instructional setting.

```
  --section omitted--
00691 quiet
00702 quiet
00713 quiet                      (Student is reading instruction manual.)
  --section omitted--
00922 quiet
00933 quiet
00944 quiet
00952 RUBOUT                     (Student presses incorrect keys.)
00954 D
00965 quiet
00976 quiet
00987 quiet
00998 quiet
TUTOR: I think you are trying to modify your program
       You should list your program first.
01003 RUBOUT                     (Student presses additional incorrect
01010 R                          keys.  Except for "R", they have no
01021 quiet                      effect, since they are illegal in this
01022 RUBOUT                     context.)
01033 X
01035 D
01046 quiet
01057 quiet
TUTOR: I was expecting you to list your program
       You should press the L key.
01069 L
01079 quiet
01090 quiet
01094 RUBOUT                     (Student goes on to solve the problem.)
01099 RUBOUT
01105 1
01107 5
01113 SPACE                      (The "SPACE" and "D" keypresses are
01124 quiet                      incorrect, but Coach does not give advice
01127 D                          since the student does not pause long
01128 SPACE                      enough.)
01139 quiet
01150 quiet
01161 quiet
01170 SPACE
01175 QUOTE
01177 SPACE
01178 QUOTE
```

Figure 3. Partial student record. The time in seconds since the beginning of the session and the corresponding keypresses are shown. Advice was generated by the Coach system.

```
   --section omitted--
Expecting paragraph paragraph-14
Expecting text text-14 true
Expecting read ( 41 ) true
Expecting pause ( 61 ) ( 246 )
Expecting quiet nil
OBSERVING           00691 quiet.
Student completed quiet ( 691 )
Expecting quiet nil
OBSERVING           00702 quiet.
Student completed quiet ( 702 )
   --section omitted--
Expecting quiet nil
OBSERVING           00933 quiet.
Student completed quiet ( 933 )
Student is still pausing after 253 seconds.
Student completed pause ( 61 ) ( 246 )
Student completed read ( 41 ) true
Student read about replace
Student read about delete
Student read about insertt
Student read about modify
Student completed text text-14 true
Student's current reading rate is 4 sec/line.
Expecting problem problem-9
Expecting modify *02512
*02512
         isainverse *02536    [ display MARY ]
         isainverse *02538    [ display SPACE ]
         isainverse *02540    [ display SMITH ]
Expecting List
Expecting L nil
OBSERVING           00944 quiet.
```

Figure 4. An excerpt of the Coach trace (Part1).

```
define "paragraph" (:name)
        {}
        if (firstnode (from :name via problemname)
                &(
                call (the firstnode :cproblem)
                call (true :problem-flag)
                ))
        firstnode (from :name via textname)
        if (absent (text (the firstnode :problem-flag) for token)
                suspend ("3" newnode the newnode maybe))
        update-reading-rate
        if (:problem-flag
                if (absent (problem (:cproblem) for token)
                        suspend ("5" newnode the newnode maybe)))
        if (orphaned (token)
                &(
                call (firstnode (from :name via contained-in) :man-name)
                push (manual (:man-name))
                ))
        return (token true).
```

Figure 5. The schema for *paragraph*.

the *manual* schema directly. In the example shown in Figure 4, as the schema *paragraph* (paragraph-14) is activated, it examines the semantic network representation of the The FLOW Manual and finds that paragraph-14 contains problem-9 and text-14, and then activates the *text* schema with the argument "text-14" as reflected in line 3 of the trace. (The second argument of *text*, "true", indicates that the paragraph also contains a problem.)

In a similar manner, the *text* schema determines from the semantic network that text-14 contains 41 lines of text, and activates the *read* schema with 41 as its argument. *Read* then checks the student model to find that student's current reading rate is 3 seconds per line, and activates the *pause* schema, expecting a pause of between 61 and 246 seconds (half and twice the estimated time). Finally the *pause* schema activates a *quiet* schema. Like the schemas for keypresses, the *quiet* schema is a bottom-level schema and it checks to see if the student has actually been quiet for an 11 second period. In this case, there was a quiet period ending at time 00691; the *quiet* schema is completed, and it returns to *pause*, its parent schema.

Before continuing, let us review briefly what has happened. Coach has used the schemas, the semantic network representation of the instructional manual, and the model of the student to predict a period of quiet while the student is reading the text in paragraph-14. When that quiet is observed, Coach assumes that the student is reading. In another context, a period of quiet could mean that the student having difficulty or working on a problem, but conceptually-guided processing has allowed Coach to interpret what otherwise would be an ambiguous event.

When the *pause* schema detects the completed *quiet* schema at line 8 of Figure 4, it determines hat the maximum expected pause length has not yet occurred and expects another period of quiet. This sequence repeats a number of times as the student continues to pause. Normally the *pause* schema would terminate when the student pressed a key, but in this case the pause runs to the full length. The *pause* and *read* schemas then return, complete. The *text* schema next examines the representation of the FLOW manual to determine what topics were covered in text-14, updates the model of the student accordingly, and returns (lines 19-23). Finally, the *paragraph* schema updates the student's reading rate and then expects the student to do problem-9.

Data-driven invocation

As long as the student's actions match Coach's expectations, conceptually-guided prediction provides a natural and efficient means of interpreting the student's keypresses. Often the student's actions do not match predictions, however, and the tutor needs a way of responding to these unexpected events. Coach uses data-driven invocation of schemas to initiate interpretation of unpredicted student actions. There are two modes of invocation: suggesting and pushing.

Figure 6 shows an example of a data-driven suggestion. This section of the trace follows immediately after that shown in Figure 4. Up until this point, the student's actions have matched Coach's expectations and the interpretation has proceeded smoothly. On line 1 of Figure 6, however, Coach is expecting an *L* and instead there is another period of quiet at time 00944. A *quiet* schema is activated, and it suggests (puts on the agenda) a schema for *unexpected-pause*. The *L* schema and all the unsatisfied parents above it suspend themselves, and the next item on the agenda, the *unexpected-pause* schema, is activated. *Unexpected-pause* preempts the orphaned *quiet* schema (line 13) and expects further quiet. (If a 60 second pause is observed, *unexpected-pause* will initiate advice to the student.) Instead of more quiet, however, the student presses the RUBOUT key. The *unexpected-pause* schema, having failed to observe an additional period of quiet, reports the 19 second pause which it did find, and then terminates.

```
OBSERVING                00944 quiet.
quiet ( 944 ) suggests unexpected-pause
L suspending
List suspending
modify suspending
problem suspending
paragraph suspending
Current agenda
agenda
        unexpected-pause        *02526
        manual  manual-02575
Student is pausing unexpectedly.
unexpected-pause preempts quiet ( 944 )
Expecting quiet nil
OBSERVING                00952 RUBOUT.
illegal key      RUBOUT ( 952 )
RUBOUT ( 952 ) suggests out-of-order *02581
Did not observe quiet nil
Student unexpectedly paused for 19 seconds.
Current agenda
agenda
        manual  manual-02575
        out-of-order      *02580
Reactivating suspended schema    paragraph-02575
Reactivating suspended schema    problem-02575
Reactivating suspended schema    modify-02575
Reactivating suspended schema    List-02575
Reactivating suspended schema    L-02575
OBSERVING                00954 D.
illegal key      D ( 954 )
D ( 954 ) suggests out-of-order *02593
L suspending
List suspending
modify suspending
problem suspending
paragraph suspending
```

Figure 6. An excerpt of the Coach trace (Part2).

In addition to suggesting possible parents by placing them on the agenda, schemas can directly activate a parent schema. This second type of data-driven invocation is called "pushing", and allows a low-level schema to directly control the flow of processing. Note that an orphaned schema cannot directly link itself to a parent schema; an orphan can only suggest or push a parent, which then when has the option of accepting or rejecting the orphaned schema.

Suspension and re-activation of schemas

In a system such as Coach, which interprets information serially, expectations that are not met at first may be fulfilled later. We need a mechanism to terminate a chain of reasoning if it runs into difficulty, with the possibility of reactivating it later when it might be more successful. In the mean-time, processing should focus on the unexpected current events.

Corresponding to the ability of human tutors to hold a set of expectations temporarily in abeyance, schemas in the Coach system can suspend themselves if they fail to find a child and allow processing to proceed on other topics. Schemas can suspend whenever they fail to find an expected child. If they are reactivated later, they resume processing where they suspended and search again for the missing child.

Two examples of the suspension and reactivation of schemas are shown in the Coach trace. The first example is in Figure 6. At the beginning of this section of the trace, Coach had developed a line of conceptually-guided prediction culminating in the *L* schema, which then observed the student. Instead of an L keypress, however, the *L* schema found a quiet period (line 2). It suspended itself and returned with the truth value of "maybe" to the *List* schema which had activated it. The *List* schema was thus incomplete, and suspended itself in turn. In a similar fashion the entire predicted chain of schemas suspends itself, until finally the top-level schema, *manual,* suspends and places itself on the global agenda. Only the top-level schema in a chain places itself on the agenda; the other suspended schemas remain linked to their respective parents.

After the *unexpected-pause* schema is activated and terminates (lines 12-19), the suspended *manual* schema comes up on the agenda and is reactivated. Reactivated schemas start out processing at the point where they had suspended. In this case the reactivated *manual* schema starts by looking for paragraph-14 (rather than paragraph-1, as a freshly activated *manual* schema would do), reactivating the suspended *paragraph* schema on line 24.

The *paragraph* schema suspended when it was unable to find a completed schema for problem-9, and it starts by reactivating the schema for problem-9. In a similar way the remaining schemas in the chain are reactivated, until finally the *L* schema observes the student. The *L* schema finds that the student has pressed the D key, and the entire sequence of schemas, from *L* thru *manual,* suspends once again (lines 32-36).

This sequence of developing a line of expectations, suspending it to attend to unexpected data, and reactivating the intact line of expectations later, might be better simulated with a parallel processing implementation of the tutor, but the suspension and reactivation of schemas gives us many of the essential features of multiple lines of interpretation within a serial processing implementation.

Solving problems

In addition to enabling Coach to follow the student's progress thru the instructional manual, another important use of conceptually-guided prediction is in the solution of FLOW programming problems. Coach interprets the student's solutions to programming problems by "solving" the problem for itself and predicting that the student will solve the problem in the same manner. The problems in the

FLOW Instructional Manual are simple enough that Coach, starting with a functional description of the computer program needed, can effectively write the program by instantiating the schemas for the functions involved. This will be illustrated by examining how Coach solves problem-9.

In problem-8, the student ran the program shown in Figure 7. The output which this program produced is shown in the lower part of the figure. (The student's name has been changed.) After describing various ways to modify a program, the instruction manual states problem-9:

> Choose one (or all) of these methods and modify your program so that it displays your name with a space between your first and last names.

Coach starts with a corresponding representation of the problem in its semantic network:

> modify to set-of (display (firstname) display ("SPACE") display (lastname)).

On line 25 of Figure 4, after concluding that the student has completed text-14, a schema for problem is activated. The *problem* schema finds the description of problem-9 in the semantic network, and activates the *modify* schema. The argument of the *modify* schema, shown on lines 27 - 30, is essentially a functional description of the required program. The proper values for firstname and lastname were obtained from the model of the student as Coach instantiated the *modify* schema. According to the definition of the *modify* schema, before a program can be modified, it must be displayed on the screen using the LIST command. Since the screen now contains the output from the previous run, Coach predicts that the student will press the L key to list the program. After the student finally lists the program, Coach continues with the solution to problem-9 in a section of the trace not shown here. The *modify* schema calculates the function of the student's current program, compares that to the desired program function, and finds that a display SPACE function must be inserted after line 10. Eventually, Coach deduces that the statement number must be changed to a number between 11 and 19, which requires the student to press the RUBOUT key. Later Coach continues the solution of the problem, expanding the schema for displaying a SPACE into a schema for a display-quoted-string statement, and finally a D keypress. After a few false starts, the student presses the D key, and Coach goes on to predict a quoted string consisting of the key sequence QUOTE, SPACE, and QUOTE.

Thus the unfolding definitions of schemas allow the Coach system to predict how the student will solve the problem. Of course even these simple problems have more than one correct solution. If the student chooses to solve the problem in a different manner, Coach has to use data-driven processing to build up a schema giving a functional description of the student's program, which can then match a predicted schema representing the intended function of the program.

Error schemas

A major function of data-driven processing in the Coach system is the recognition of errors. Coach is intended to simulate an experienced tutor and includes schemas which represent the common types of errors which students make. Schemas invoked by data-driven processing can suggest or push any schemas, including these error schemas. When they are activated, the error schemas search for their children in the normal manner. Successful completion of an error schema corresponds to the recognition of a student error. Once an error has been recognized, Coach normally does not give immediate advice, but rather allows the students to try to recover from the error on their own, and then gives advice when the students pause.

```
010 DISPLAY "MARY"
020 DISPLAY "SMITH"
030
```

```
MARYSMITH
```

Figure 7. The student's current program and the resulting output.

The Coach trace shows a good example of error interpretation. On line 15 of Figure 6, when Coach was expecting a quiet period, the student pressed the RUBOUT key. The student has just run the current program and the RUBOUT key is illegal in this context.[1] The *RUBOUT* schema, like the schemas for all other illegal keys, suggests an *out-of-order* schema (with the *RUBOUT* schema as its argument), which will eventually try to determine if the RUBOUT key was part of a predicted schema but out of order. The next item on the agenda, however, is the suspended *manual* schema, which reactivates all its children and eventually leads to the observation of a D keypress on line 29. The D key is also illegal, and the *D* schema suggests another *out-of-order* schema.

After the active schemas suspend, the next item on the agenda, at the top of Figure 8, is the *out-of-order* (RUBOUT) schema. *Out-of-order* waits for the student to pause 40 seconds before doing any investigation. In the example shown here, the student paused for 44 seconds (line 16), so *out-of-order* tries to determine if the *RUBOUT* could be a part of a predicted schema. It makes up a target set consisting of all the currently unsatisfied schemas: schemas lacking either parents or children. *Out-of-order* then refers to the semantic network representation of information about the FLOW language and, starting at RUBOUT, does a breadth-first search along the relation "part-of" looking for a schema in the target set. It eventually finds on line 34 that *RUBOUT* could be part of the expected *modify* schema, thereby concluding that the student was trying to modify the program, but neglected to LIST the program first.

Giving advice

Once the Coach system has recognized an error, it can give advice to the student. Three important issues come into focus here: 1) Should the advice be given immediately or should students be allowed an opportunity to detect and correct errors on their own? 2) At what level should the advice be phrased (e.g. Should the advice be in terms of program functions or individual keypresses)? 3) What should be done if the student does not respond to the advice?

Coach decides when to give advice based on its model of the student. Advice is given relatively soon when the student is unfamiliar with the concept at issue. When the student has successfully used the relevant concept earlier, advice is delayed to allow the student to debug the program without help. Each schema in the Coach system corresponds to a concept in the FLOW instructional course. Coach maintains a semantic network representation of the student's experience with each of the schemas. Originally schemas are not associated with the student model. When the student reads about a concept in the instructional manual, Coach notes in the student model that that student "read about" the corresponding schema. When the student successfully completes the schema for the first time, it is noted as "used", and when the student completes the schema a second time Coach notes that the student "mastered" the schema.

We can see how this information in the student model is used by continuing the analysis of the "out of order" problem. On line 34 of Figure 8 Coach has inferred that the RUBOUT key was part of an attempt to modify the program, but was out of order. The advice will therefore be given at the level of the *modify* schema. As indicated in the next line, Coach finds from the model of the student that the student has read about modifying programs, but has never done any modification. Based on this information, the *out-of-order* schema decides to give advice immediately. (It had already allowed a 40 second pause before attempting to interpret the RUBOUT key.) Coach would have waited for an

----

1. Only the R, W, and L keys, for the RUN, WALK, and LIST commands are legal at this point. Syntactically illegal keys have no effect in the FLOW system. The illegal key is displayed briefly on the terminal screen and then erased as the terminal beeps.

```
Current agenda
agenda
          out-of-order        *02580
          out-of-order        *02592
          manual   manual-02605
Out-of-order ( RUBOUT ) waiting for 40 seconds
Expecting pause ( 40 ) ( 40 )
Expecting quiet nil
OBSERVING           00965 quiet.
Student completed quiet ( 965 )
  --section omitted--
Expecting quiet nil
OBSERVING           00998 quiet.
Student completed quiet ( 998 )
Student is still pausing after 44 seconds.
Student completed pause ( 40 ) ( 40 )
targetset is
*02627
          isainverse          unexpected-pause
          isainverse          L
          isainverse          List
          isainverse          modify
          isainverse          problem
          isainverse          paragraph
          isainverse          manual
bfsearch starting at RUBOUT
searching along part-of
bfsearch checking character-delete
bfsearch checking change-statement-number
bfsearch checking insert
bfsearch checking delete
bfsearch checking append
bfsearch checking modify
bfsearch found modify
Out-of-order found student read about modify
Therefore will give help immediately
TUTOR     I think you are trying to modify your program
         You should list your program first.
```

Figure 8. An excerpt of the Coach trace (Part 3).

additional 20 second pause before giving advice if the student had actually used the *modify* schema successfully on some previous occasion, or waited for a 40 second pause if the student had mastered the *modify* schema. The teaching strategy here is to give students a chance to work out their own problems if they seem to know the required concepts, but to give help early when they are dealing with new concepts.

The actual advice which Coach gives is generated fairly simply. *Out-of-order* has assumed that the problem is with the *modify* schema. It examines the suspended *modify* schema and finds that *modify* is currently searching for a *List* schema. *Out-of-order* has an advice frame which looks like:

"I think you are trying to <schema>.
You should <current child of schema> first."

The bracketed arguments in the advice frame are filled in with phrases associated with the corresponding schemas, and Coach produces the advice shown in lines 37 and 38.

Whenever Coach gives advice to perform a specific task, in this case to list the program, it invokes a *monitor* to ensure that the student performs that task correctly. In this case, the *monitor* checks the model of the student, finds that the student has used the LIST command once, and decides to allow a 20 second pause before giving further advice. *Monitor* essentially narrows Coach's focus of attention. As long as the student does not change the state of the FLOW system, Coach looks only for completion of the *List* schema or the 20 second pause. Other non-critical information is ignored. The level of Coach's attention has also changed. Coach originally thought the problem was at the level of the *modify* schema, and advised the student to list the program. With this particular student, however, that advice was not effective because the student had forgotten how to list the program. As we have seen, Coach's attention has now shifted to the *List* schema, and subsequent advice at that level led the student out of difficulty.

Finding children

So far, I have not described in detail how a schema looks for its children. In most of the cases we've examined, the parent schema instantiates a new schema for its children and activates the schema. But activation of a new schema is the last resort after the parent has made three other attempts to find its child. In order to take advantage of data-driven and suspended processing, the parent schema must first check to see if a suitable child already exists before activating an entirely new schema.

Schemas search for their children using the "absent" procedure, which instantiates a schema for the child and then tries to find an available schema which "matches" the new instance. A schema is said to match the instance if it has the same name, and the same or more constrained arguments. The search for a suitable child proceeds in four steps.

First, the "absent" procedure checks for a matching orphan (i.e. a schema without a parent). For example, on line 13 of Figure 6, the *unexpected-pause* schema finds an orphaned *quiet* schema which it preempts. Preempting an orphaned schema can have side effects, if that orphan has suggested other schemas. On lines 16 and 17 in Figure 9 an unexpected *D* schema suggested schemas for two different FLOW statements which it could be part of (display-quoted-string and display-variable). Later on line 33, when another *display-quoted-string* schema, (which had been a conceptually-guided prediction), preempts the *D* schema, the support for the two suggested schemas collapses and they are removed from the agenda. This is consistent with the general rule that a schema may have only one parent at a time.

Second, if an appropriate orphan is not found, the parent schema is checked to see if it has previously activated and suspended this child. If so, the suspended schema is reactivated. There are many

```
   --section omitted--
Student completed change-statement-number ( 11 ) ( 19 )
Expecting display SPACE
Expecting display-quoted-string anything SPACE
Expecting D nil
OBSERVING       01113 SPACE.
   --section omitted--
Expecting quiet nil
OBSERVING       01127 D.
currentline
        contains *00602     [ 0 nil ]
        contains *02802     [ 1 ( 1105 ) ]
        contains *02808     [ 5 ( 1107 ) ]
        contains *02853     [ D ( 1127 ) ]
The current state is now DISPLAY
D ( 1127 ) suggests display-quoted-string nil
D ( 1127 ) suggests display-variable nil
Did not observe quiet nil
Student paused for 14 seconds.
Did not observe pause ( 40 ) ( 40 )
Current agenda
agenda
        manual  manual-02841
        display-quoted-string    *02854
        display-variable         *02855
Reactivating suspended schema   paragraph-02840
Reactivating suspended schema   problem-02838
Reactivating suspended schema   modify-02836
Reactivating suspended schema   insertf-02834
Reactivating suspended schema   insertt-02832
Reactivating suspended schema   display-02830
Reactivating suspended schema   display-quoted-string-02828
display-quoted-string ( 15 ) SPACE preempts D ( 1127 )
display-variable nil collapses.
display-quoted-string nil collapses.
Expecting qstring SPACE
Expecting QUOTE nil
OBSERVING       01128 SPACE.
   --section omitted--
Student completed QUOTE ( 1175 )
Expecting SPACE anything
OBSERVING       01177 SPACE.
currentline
        contains *00602     [ 0 nil ]
        contains *02802     [ 1 ( 1105 ) ]
        contains *02808     [ 5 ( 1107 ) ]
        contains *02853     [ D ( 1127 ) ]
        contains *02871     [ QUOTE ( 1175 ) ]
        contains *02944     [ SPACE ( 1177 ) ]
Student has used SPACE once.
Student completed SPACE ( 1177 )
Expecting QUOTE nil
OBSERVING       01178 QUOTE.
```

Figure 9. An excerpt of the Coach trace (Part 4).

instances of the reactivation of a child in the Coach trace. For example in lines 24 - 28 of Figure 6, a series of suspended schemas are reactivated and reactivate their children in turn. A schema has access only to suspended schemas for children which it originally activated.

Third, if there was no suspended child, "absent" looks for a matching schema which has been suggested and placed on the global agenda. If found, the schema is taken off the agenda and activated directly rather than having to wait its turn. This happens relatively infrequently, and there are no examples of "taking a suggestion" in this section of the Coach trace.

Fourth, the newly instantiated schema itself is activated. This is the normal case as conceptually-guided predictions are generated, for example in lines 2 - 6 of Figure 4. In all cases the new child is linked to the parent schema. The child will be unlinked later if it does not successfully complete itself.

Discussion

The original intent in building the Coach system was to explore how a human tutor interprets the student's behavior and gives advice, rather than to build a complete instructional system. In terms of a working CAI system, there are a number of limitations in the present implementation of the Coach system. The schemas in the current system cover only the material presented in the first half of the FLOW instruction manual (about 30 minutes of instructional time.) In addition, the current set of error schemas are sufficient for only a portion of the errors which we have seen in students. These limitations primarily reflect the fact that Coach is currently implemented on a minicomputer with limited memory size. There is no reason in principle that the number of schemas could not be enlarged to handle a wider range of material. At this point it is not clear what the ultimate limits of this approach are. A more serious source of limitations is that Coach predicts the student's solution of programming problems by solving the problems itself. With more complex programming problems, this would not be possible; we would be faced with all the difficulties of automatic programming.

The functioning of schemas in the Coach system does not properly simulate our models of human thought in some areas. For instance, only one schema is active at a time. A more attractive model of human perception is one in which many schemas are simultaneously active, competing for the data and substructures, and interacting with each other to give alternate perspectives on the environment. Some of the assumptions and restrictions built into the Coach system are based more on intuition than psychological theory. For example, in the Coach system, schemas can have only one parent at a given time. This was intended to correspond to the idea that we cannot interpret data (such as a Necker cube) from two different perspectives simultaneously. But one could also argue the opposite position, that two high level schemas could share a lower level structure. Related to these objections is the problem that there is no good way to evaluate a system of this type. It is encouraging that the system works over some limited range of environments, but other approaches, such as production systems, also perform satisfactorily and there is no straightforward way to determine which is a "better" simulation of the human tutor.

Nonetheless the Coach system has a number of significant features. It is a well specified description of a schema-based system for interpreting complex events. Specific information, including a model of the student, is represented in a semantic network database. Generic knowledge is represented with active schemas which search for their children, modify the internal knowledge base, and perform actions in the external world. Schemas may be referred to only by their names and arguments. Schemas which do not have a parent may suggest or invoke possible parents, thus influencing their own interpretation.

Interpretation of events is based on the interaction of conceptually-guided and data-driven processing. These two processing modes interface when a schema searches for its children by examining orphaned and suggested schemas before instantiating new schemas. Errors and unexpected-events are interpreted in the same manner as correct or expected events: the corresponding schemas are activated by either conceptually-guided prediction or data-driven invocation and "interpret" lower-level schemas by incorporating acceptable ones into their structure. The level of advice given to the student derives naturally from the fact that intelligence and processing are distributed among schemas at all levels: the schema which detects an error gives advice at its own level. Thus a schema-based intelligence gives the Coach system many of the surface behaviors and underlying processes which human tutors appear to have.

References

Carbonell, J. R. AI in CAI: an artificial-intelligence approach to computer-assisted instruction. *IEEE Transactions on Man-Machine Systems,* 1970, *MMS-11,* 190-202.

Gentner, D. R. Toward an intelligent automated tutor. In H. F. O'Neil, Jr. (Ed.) *Procedures for Instructional Systems Development* New York: Academic Press, 1979.

Gentner, D. R. & Norman, D. A. *The FLOW tutor: Schemas for tutoring* (Tech. Rept.). La Jolla, Calif.: University of California, San Diego, Center for Human Information Processing, 1977.

Norman, D. A., Rumelhart, D. E., & the LNR Research Group. *Explorations in cognition.* San Francisco: Freeman, 1975

Raskin, J. Flow language for computer programming. *Computers and the Humanities,* 1974, *8,* 231-237.

Sleeman, D. H., & Brown, J. S. (Eds.). Intelligent tutoring systems. *International Journal of Man-Machine Studies,* 1979, *11,* 1-156.

Navy

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

1 Dr. Robert Blanchard
Navy Personnel R&D Center
Managment Support Department
San Diego, CA 92151

1 Dr. Jack R. Borsting
Provost & Academic Dean
U.S. Naval Postgraduate School
Monterey, CA 93940

1 Dr. Robert Breaux
Code N-71
NAVTRAEQUIPCEN
Orlando, FL 32813

1 Mr. James S. Duva
Chief, Human Factors Laboratory
Naval Training Equipment Center
(Code N-215)
Orlando, Florida 32813

1 DR. PAT FEDERICO
NAVY PERSONNEL R&D CENTER
SAN DIEGO, CA 92152

1 Dr. John Ford
Navy Personnel R&D Center
San Diego, CA 92152

1 LT Steven D. Harris, MSC, USN
Code 6021
Naval Air Development Center
Warminster, Pennsylvania 18974

1 Dr. Norman J. Kerr
Chief of Naval Technical Training
Naval Air Station Memphis (75)
Millington, TN 38054

1 Dr. Leonard Kroeker
Navy Personnel R&D Center
San Diego, CA 92152

1 Library
Navy Personnel R&D Center
San Diego, CA 92152

6 Commanding Officer
Naval Research Laboratory
Code 2627
Washington, DC 20390

1 JOHN OLSEN
CHIEF OF NAVAL EDUCATION &
TRAINING SUPPORT
PENSACOLA, FL 32509

1 Psychologist
ONR Branch Office
495 Summer Street
Boston, MA 02210

1 Psychologist
ONR Branch Office
536 S. Clark Street
Chicago, IL 60605

1 Office of Naval Research
Code 200
Arlington, VA 22217

1 Office of Naval Research
Code 437
800 N. Quincy SStreet
Arlington, VA 22217

5 Personnel & Training Research Programs
(Code 458)
Office of Naval Research
Arlington, VA 22217

1 Psychologist
OFFICE OF NAVAL RESEARCH BRANCH
223 OLD MARYLEBONE ROAD
LONDON, NW, 15TH ENGLAND

1 Psychologist
ONR Branch Office
1030 East Green Street
Pasadena, CA 91101

1 Dr. Alfred F. Smode
Training Analysis & Evaluation Group
(TAEG)
Dept. of the Navy
Orlando, FL 32813

1 Dr. Richard Sorensen
Navy Personnel R&D Cente
San Diego, CA 92152

1 CHAIRMAN, LEADERSHIP & LAW DEPT.
DIV. OF PROFESSIONAL DEVELOPMMENT
U.S. NAVAL ACADEMYY
ANNAPOLIS, MD 21402

1 Dr. William L. Maloy
Principal Civilian Advisor for
Education and Training
Naval Training Command, Code 00A
Pensacola, FL 32508

1 Dr. Kneale Marshall
Scientific Advisor to DCNO(MPT)
OP01T
Washington DC 20370

1 CAPT Richard L. Martin
USS Francis Marion (LPA-Z49)
FPO New York, NY 09501

2 Dr. James McGrath
Navy Personnel R&D Center
Code 306
San Diego, CA 92152

1 CDR. MERCER
CNET LIAISON OFFICER
AFHRL/FLYING TRAINING DIV.
WILLIAMS AFB, AZ 85224

1 Dr. George Moeller
Head, Human Facors Branch
Naval Submarine Medical Research
Groton, CN 06340

1 Dr William Montague
Navy Personnel R&D Center
San Diego, CA 92152

1 Commanding Officer
Naval Health Research
Center
Attn: Library
San Diego, CA 92152

1 Naval Medical R&D Command
Code 44
National Naval Medical Center
Bethesda, MD 20014

1 Scientific Director
Office of Naval Research
Scientific Liaison Group/Tokyo
American Embassy
APO San Francisco, CA 96503

1 Office of the Chief of Naval Operat
Research, Development, and Studies
(OP-102)
Washington, DC 20350

1 DR. RICHARD A. POLLAK
ACADEMIC COMPUTING CENTER
U.S. NAVAL ACADEMY
ANNAPOLIS, MD 21402

1 Dr. Gary Poock
Operations Research Department
Naval Postgraduate School
Monterey, CA 93940

1 Dr. Bernard Rimland
Navy Personnel R&D Center
San Diego, CA 92152

1 Mr. Arnold Rubenstein
Naval Personnel Support Technology
Naval Material Command (08T244)
Room 1044, Crystal Plaza #5
2221 Jefferson Davis Highway
Arlington, VA 20360

1 Dr. Worth Scanland
Chief of Naval Education and Train
Code N-5
NAS, Pensacola, FL 32508

1 A. A. SJOHOLM
TECH. SUPPORT, CODE 201
NAVY PERSONNEL R& D CENTER
SAN DIEGO, CA 92152

1 Mr. Robert Smith
Office of Chief of Naval Operation
OP-987E
Washington, DC 20350

ARMY

1 Technical Director
U. S. Army Research Institute for
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

1 HQ USAREUE & 7th Army
ODCSOPS
USAAREUE Director of GED
APO New York 09403

1 DR. RALPH DUSEK
U.S. ARMY RESEARCH INSTITUTE
5001 EISENHOWER AVENUE
ALEXANDRIA, VA 22333

1 Dr. Michael Kaplan
U.S. ARMY RESEARCH INSTITUTE
5001 EISENHOWER AVENUE
ALEXANDRIA, VA 22333

1 Dr. Milton S. Katz
Individual Training & Skill
Evaluation Technical Area
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Beatrice J. Farr
Army Research Institute (PERI-OK)
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Harold F. O'Neil, Jr.
Attn: PERI-OK
Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Robert Sasmor
U. S. Army Research Institute for
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

1 Dr. Frederick Steinheiser
U. S. Army Reserch Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Marines

1 H. William Greenup
Education Advisor (E031)
Education Center, MCDEC
Quantico, VA 22134

1 DR. A.L. SLAFKOSKY
SCIENTIFIC ADVISOR (CODE RD-1)
HQ, U.S. MARINE CORPS
WASHINGTON, DC 20380

Air Force

1 Air Force Human Resources Lab
AFHRL/PED
Brooks AFB, TX 78235

1 Air University Library
AUL/LSE 76/443
Maxwell AFB, AL 36112

1 Dr. Earl A. Alluisi
HQ, AFHRL (AFSC)
Brooks AFB, TX 78235

1 DR. T. E. COTTERMAN
AFHRL/ASR
WRIGHT PATTERSON AFB
OHIO 45433

1 Dr. Genevieve Haddad
Program Manager
Life Sciences Directorate
AFOSR
Bolling AFB, DC 20332

1 Dr. Donald E. Meyer
U.S. Air Force
ATC/XPTD
Randolph AFB, TX 78148

1 Dr. Ross L. Morgan (AFHRL/ASR)
Wright -Patterson AFB
Ohio 45433

1 Dr. Marty Rockway (AFHRL/TT)
Lowry AFB
Colorado 80230

2 Faculty Development Division
Headquarters Sheppard Technical
Training Center (ATC)
Sheppard AFB, TX 76311

1 Jack A. Thorpe, Maj., USAF
Naval War College
Providence, RI 02846

1 Brian K. Waters, LCOL, USAF
Air University
Maxwell AFB
Montgomery, AL 36112

Coast Guard

1 Mr. Richard Lanterman
PSYCHOLOGICAL RESEARCH (G-P-1/62)
U.S. COAST GUARD HQ
WASHINGTON, DC 20590

Other DoD

1 Dr. Craig I. Fields
ADVANCED RESEARCH PROJECTS AGENCY
1400 WILSON BLVD.
ARLINGTON, VA 22209

12 Defense Documentation Center
Cameron Station, Bldg. 5
Alexandria, VA 22314
Attn: TC

1 Dr. Dexter Fletcher
ADVANCED RESEARCH PROJECTS AGENCY
1400 WILSON BLVD.
ARLINGTON, VA 22209

1 Military Assistant for Training an
Personnel Technology
Office of the Under Secretary of
for Research & Engineering
Room 3D129, The Pentagon
Washington, DC 20301

Civil Govt

1 Dr. Susan Chipman
Basic Skills Program
National Institute of Education
1200 19th Street NW
Washington, DC 20208

1 Mr. James M. Ferstl
Bureau of Training
U.S. Civil Service Commission
Washington, D.C. 20415

1 Dr. Joseph I. Lipson
Division of Science Education
Room W-638
National Science Foundation
Washington, DC 20550

1 Dr. Joseph Markowitz
Office of Research and Development
Central Intelligence Agency
Washington, DC 20205

1 Dr. John Mays
National Institute of Education
1200 19th Street NW
Washington, DC 20208

1 William J. McLaurin
Rm. 301, Internal Revenue Service
2221 Jefferson Davis Highway
Arlington, VA 22202

1 Dr. Arthur Melmed
National Intitute of Education
1200 19th Street NW
Washington, DC 20208

1 Dr. Andrew R. Molnar
Science Education Dev.
and Research
National Science Foundation
Washington, DC 20550

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

1 Dr. Frank Withrow
U. S. Office of Education
400 6th Street SW
Washington, DC 20202

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

Non-Govt

1 Dr. John R. Anderson
Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

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

1 DR. MICHAEL ATWOOD
SCIENCE APPLICATIONS INSTITUTE
40 DENVER TECH. CENTER WEST
7935 E. PRENTICE AVENUE
ENGLEWOOD, CO 80110

1 Dr. R. A. Avner
University of Illinois
Computer-Based Educational Research Lal
Urbana, IL 61801

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

1 Dr. Patricia Baggett
Department of Psychology
University of Denver
University Park
Denver, CO 80208

1 Mr Avron Barr
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. John Bergan
School of Education
University of Arizona
Tuscon AZ 85721

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

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

1 Dr. Meredith P. Crawford
American Psychological Association
1200 17th Street, N.W.
Washington, DC 20036

1 Dr. Hubert Dreyfus
Department of Philosophy
University of California
Berkely, CA 94720

1 ERIC Facility-Acquisitions
4833 Rugby Avenue
Bethesda, MD 20014

1 MAJOR I. N. EVONIC
CANADIAN FORCES PERS. APPLIED RESEARCH
1107 AVENUE ROAD
TORONTO, ONTARIO, CANADA

1 Dr. Ed Feigenbaum
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. Richard L. Ferguson
The American College Testing Program
P.O. Box 168
Iowa City, IA 52240

1 Mr. Wallace Feurzeig
Bolt Beranek & Newman, Inc.
50 Moulton St.
Cambridge, MA 02138

1 Dr. Victor Fields
Dept. of Psychology
Montgomery College
Rockville, MD 20850

1 Mr. Richards J. Heuer, Jr.
27585 Via Sereno
Carmel, CA 92923

1 Dr. Dustin H. Heuston
Wicat, Inc
Box 986 Orem, UT 84057

1 Dr. Nicholas A. Bond
Dept. of Psychology
Sacramento State College
600 Jay Street
Sacramento, CA 95819

1 Dr. Lyle Bourne
Department of Psychology
University of Colorado
Boulder, CO 80302

1 Dr. Kenneth Bowles
Institute for Information Sciences
University of California at San Di
La Jolla, CA 92037

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

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

1 DR. C. VICTOR BUNDERSON
WICAT INC.
UNIVERSITY PLAZA, SUITE 10
1160 SO. STATE ST.
OREM, UT 84057

1 Dr. Anthony Cancelli
School of Education
University of Arizona
Tuscon, AZ 85721

1 Dr. William Chase
Department of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

1 Dr. Edwin A. Fleishman
Advanced Research Resources Organ.
Suite 900
4330 East West Highway
Washington, DC 20014

1 DR. ROBERT GLASER
LRDC
UNIVERSITY OF PITTSBURGH
3939 O'HARA STREET
PITTSBURGH, PA 15213

1 Dr. Marvin D. Glock
Department of Education
Stone Hall
Cornell University
Ithaca, NY 14853

1 Dr. Ira Goldstein
XEROX Palo Alto Research Center
3333 Coyote Road
Palo Alto, CA 94304

1 DR. JAMES G. GREENO
LRDC
UNIVERSITY OF PITTSBURGH
3939 O'HARA STREET
PITTSBURGH, PA 15213

1 Dr. Ron Hambleton
School of Education
University of Massechusetts
Amherst, MA 01002

1 Dr. Harold Hawkins
Department of Psychology
University of Oregon
Eugene OR 97403

1 Dr. Barbara Hayes-Roth
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

1 Dr. Frederick Hayes-Roth
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

1 Dr. Stephen Kosslyn
Harvard University
Department of Psychology
33 Kirkland Street
Cambridge, MA 02138

Mr. Marlin Kroger
1117 Via Goleta
Palos Verdes Estates, CA 90274

1 Library
HumRRO/Western Division
27857 Berwick Drive
Carmel, CA 93921

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

1 Dr. Wilson A. Judd
McDonnell-Douglas
Astronautics Co. East
Lowry AFB
Denver, CO 80230

1 Dr. Walter Kintsch
Department of Psychology
University of Colorado
Boulder, CO 80302

1 Dr. David Kieras
Department of Psychology
University of Arizona
Tuscon, AZ 85721

1 Dr. Kenneth Klivington
Alfred P. Sloan Foundation
630 Fifth Avenue
New York, NY 10020

1 Dr. Mazie Knerr
Litton-Mellonics
Box 1286
Springfield, VA 22151

1 Dr. Stuart Milner
Department of Education
George Mason University
4400 Fairfax Drive
Fairfax, VA 22030

1 Dr. Allen Munro
Univ. of So. California
Behavioral Technology Labs
3717 South Hope Street
Los Angeles, CA 90007

1 Dr. Jesse Orlansky
Institute for Defense Analysis
400 Army Navy Drive
Arlington, VA 22202

1 Dr. Seymour A. Papert
Massachusetts Institute of Technolog
Artificial Intelligence Lab
545 Technology Square
Cambridge, MA 02139

1 Dr. James A. Paulson
Portland State University
P.O. Box 751
Portland, OR 97207

1 MR. LUIGI PETRULLO
2431 N. EDGEWOOD STREET
ARLINGTON, VA 22207

1 DR. PETER POLSON
DEPT. OF PSYCHOLOGY
UNIVERSITY OF COLORADO
BOULDER, CO 80302

1 DR. DIANE M. RAMSEY-KLEE
R-K RESEARCH & SYSTEM DESIGN
3947 RIDGEMONT DRIVE
MALIBU, CA 90265

1 DR. ROBERT J. SEIDEL
INSTRUCTIONAL TECHNOLOGY GROUP
HUMRRO
300 N. WASHINGTON ST.
ALEXANDRIA, VA 22314

1 Dr. Robert Smith
Department of Computer Science
Rutgers University
New Brunswick, NJ 08903

1 Dr. Richard Snow
School of Education
Stanford University
Stanford, CA 94305

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

1 LCOL. C.R.J. LAFLEUR
PERSONNEL APPLIED RESEARCH
NATIONAL DEFENSE HQS
101 COLONEL BY DRIVE
OTTAWA, CANADA K1A 0K2

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

1 Dr. Alan Lesgold
Learning R&D Center
University of Pittsburgh
Pittsburgh, PA 15260

1 Dr. Robert Linn
College of Education
University of Illinois
Urbana, IL 61801

1 Dr. Mark Miller
Systems and Information Sciences L
Central Research Laboratories
TEXAS INSTRUMENTS, INC.
Mail Station 5
Post Office Box 5936
Dallas, TX 75222

1 Dr. Richard B. Millward
Dept. of Psychology
Hunter Lab.
Brown University
Providence, RI 82912

1 MIN. RET. M. RAUCH
P II 4
BUNDESMINISTERIUM DER VERTEIDIGUNG
POSTFACH 161
53 BONN 1, GERMANY

1 Dr. Peter B. Read
Social Science Research Council
605 Third Avenue
New York, NY 10016

1 Dr. Mark D. Reckase
Educational Psychology Dept.
University of Missouri-Columbia
12 Hill Hall
Columbia, MO 65201

1 Dr. Fred Reif
SESAME
c/o Physics Department
University of California
Berkely, CA 94720

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

1 Dr. Ernst Z. Rothkopf
Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974

1 Dr. David Rumelhart
Center for Human Information
Processing C-009
UC San Diego
La Jolla, CA 92093

1 PROF. FUMIKO SAMEJIMA
DEPT. OF PSYCHOLOGY
UNIVERSITY OF TENNESSEE
KNOXVILLE, TN 37916

1 Dr. Allen Schoenfeld
Department of Mathematics
Hamilton College
Clinton, NY 13323

1 Dr. John Thomas
IBM Thomas J. Watson Research Cent
P.O. Box 218
Yorktown Heights, NY 10598

1 DR. PERRY THORNDYKE
THE RAND CORPORATION
1700 MAIN STREET
SANTA MONICA, CA 90406

1 Dr. Douglas Towne
Univ. of So. California
Behavioral Technology Labs
3717 South Hope Street
Los Angeles, CA 90007

1 DR. ALBERT STEVENS
BOLT BERANEK & NEWMAN, INC.
50 MOULTON STREET
CAMBRIDGE, MA 02138

1 Dr. Thomas Sticht
HumRRO
300 N. Washington Street
Alexandria, VA 22314

1 Dr. David Stone
ED 236
SUNY, Albany
Albany, NY 12222

1 DR. PATRICK SUPPES
INSTITUTE FOR MATHEMATICAL STUDIES IN
THE SOCIAL SCIENCES
STANFORD UNIVERSITY
STANFORD, CA 94305

1 Dr. Kikumi Tatsuoka
Computer Based Education Research
Laboratory
252 Engineering Research Laboratory
University of Illinois
Urbana, IL 61801

1 Col. Frank Hart's address is now
Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

1 DR. GERSHON WELTMAN
PERCEPTRONICS INC.
6271 VARIEL AVE.
WOODLAND HILLS, CA 91367

1 DR. JOHN D. FOLLEY JR.
APPLIED SCIENCES ASSOCIATES INC
BOX 158
VALENCIA, PA 16059

1 DR. KAY INABA
21116 VANOWEN ST
CANOGA PARK, CA 91303

1 Shelly L. Wilson
Navtraequipcen, Code N-71
Orlando, Fla. 32813

1 Larry Barsalon
Dept. of Psychology
Stanford University
Stanford, CA 94305

1 Dr. J. Uhlaner
Perceptronics, Inc.
6271 Variel Avenue
Woodland Hills, CA 91364

1 Dr. Benton J. Underwood
Dept. of Psychology
Northwestern University
Evanston, IL 60201

1 Dr. Phyllis Weaver
Graduate School of Education
Harvard University
200 Larsen Hall, Appian Way
Cambridge, MA 02138

1 Dr. David J. Weiss
N660 Elliott Hall
University of Minnesota
75 E. River Road
Minneapolis, MN 55455

1 Dr. William B. Whitten, II
Department of Psychology
SUNY, Albany
1400 Washington Avenue
Albany, NY 12222

1 Dr. Karl Zinn
Center for research on Learning
and Teaching
University of Michigan
Ann Arbor, MI 48104

1 Pat Langley
Psychology Dept.
CMU
Pittsburgh, PA 15213

1 Dr. S. A. Cerri
Italiaans Seminarium
Universiteit van Amsterdam
Single 132-134
Amsterdam

1 Mr. A. Finelli, Librarian
Universite de Paris VI
Institut de Programmation
Bibliotheque
4 Place Jussieu, Tour 55,
75230 Paris CEDEX 05
FRANCE

1 Erik de Corte
Professor of Educational Psychology
Department of Psychology
Vesaliusstraat 2
B-3000 Leuvein
BELGIUM

1 Elliot Soloway
Department of Computer Sciences
University of Massachusetts
Amherst, Mass. 01003

1 Mike Cohen
Teaching and Instruction Program
National Institute of Education
1200 19th Street NW
Washington, DC 20208

1 Tom Baranowski
Dept. of Community Medicine
West Virginia University Medical Center
3110 MacCorkle Ave., SE
Charleston, WV 25304

1 B. Rogoff
Department of Psychology
University of Utah
Salt Lake City, Utah 84112