Knowledge Structures of Computer Programmers

Beth Adelson
Harvard University

Three research questions are addressed here: what are the knowledge structures of novice and expert programmers; what principles underlie these structures; and how is the knowledge used? The results of the two experiments described here indicate that novices form syntactically based representations of programs which are concerned with the details of how the program functions while experts form more abstract conceptually based representations which are concerned with what the program does. The results of these experiments suggest that the representation used in a task constrains performance and that new representations develop with expertise.

Experiment 1

In this experiment Novice and Expert computer programmers perform a multi-trial free recall task (MTFR). The stimulus set consists of lines of programming code which can be organized either syntactically or conceptually. The clusters found in the recall protocols of each group will suggest the nature of the knowledge structures of each group.

Method.

Subjects. Five Novice and five Expert programmers formed the two groups of subjects.

Stimuli. The organization of the stimulus set is central to this experiment. The stimuli consisted of 16 lines of code in Polymorphic Programming Language ("PPL"is a language similar to APL) which could be organized either syntactically or procedurally.Under the procedural classification the items (lines of code) can be organized into three programs: the first is a sorting routine, the second and the third are random sampling routines. The second basis for the classification of the stimulus set is syntactic, with syntax being used here the way it is used to describe natural language. Different control phrases of the computer language act as different parts of speech in that they expect certain other kinds of control phrases to precede or follow them.There are five different syntactic categories present in the stimulus set.

Procedure. Each subject saw all of the items, one at a time and then had to recall as many of the items as possible. This procedure was repeated nine times.The stimulus set was presented in a different random order on each trial. The presentation of items was not blocked either by program or by syntactic category. Neither group knew what the organization of the stimulus set was although both groups were familiar with the syntax of the language as well as the concepts behind the three programs.

Results and Discussion.

The Novices recalled significantly more than the Experts, they also had larger chunks and more consistent subjective organization. In order to look at the organization underlying these quantitative results multi-dimensional scalings and hierarchical clusterings were done on the recall protocols of each group. In Figure 1 we see the two-dimensional scaling solution

--- --- --- ---

Insert Figure 1 Here.

--- --- --- ---

of inter-item similarity for the Novice group. The points in the solution are the items, they are labelled by syntactic category. The procedural classification of each item is given in parentheses. For the Novices the items cluster by syntactic category; the Assignment statements, (the A's), cluster in the first quadrant with the Conditional IF statements (the I's) around them. The iteration or FOR statements (the F's) cluster in the second quadrant, the function Headers (the H's) cluster in the third quadrant and the RETURN statements (the R's) cluster in the fourth quadrant.

--- --- --- ---

Insert Figure 2 Here.

--- --- --- ---

The two-dimensional scaling solution for the Expert group is presented in Figure two. Here we see the same set of items, but this time they cluster by program. In the labelling of the items the first number for an item represents the program that it belongs to, the second number for an item represents its positon in the program. For example item 2.0 is the first item in program two. The items from program one, those labelled 1.0 through 1.4, are in the first quadrant. The items from program 2, those labelled 2.0 through 2.4, are in the third quadrant and the items from program three, those labelled 3.0 through 3.5, are in the fourth quadrant. The syntactic classification is given here in parentheses, it does not capture the clusters of the Experts the way it did the clusters of the Novices. The original recall protocols also showed that each of the Expert subjects recalled all of the lines from all three programs in the order in which they would have been evaluated in a running program,this suggests that the Experts are using their knowledge of the serial nature of computer programs to organize the items within a cluster. It appears that the clusters of the Expert subjects are more abstract and conceptual than the syntactic clusters of the Novice subjects. Here as in other skilled problem solving domains we find that the chunks of the experts are based on the functional principles of the skill; items are categorized as members of one procedure or another and are then organized serially within those procedural categories. The chunks of the Novices however, are not functionally based, they are syntactic in nature. In addition, the recall protocols of the Novices showed no internal organization of the items other than the grouping by syntactic category, that is within the Novice categories there was no regularity as there was in the Expert categories.

Experiment II

The results of the first experiment suggested that the organization of the Novices was syntactic while the organization of the Experts was functional. It is possible then, that when given whole programs to comprehend experts form an abstract representation that is concerned not with the specific mechanics of a given algorithm but only with *what* it is that the program does. It is also possible that a representation formed by a novice, because it is synactically based would be more concerned with *how* the specific program functioned. The second experiment presented here was designed to check out the possibility that during program comprehension novices form representations of how programs function, while experts form representations of what programs do.

The strategy used to look at this possibility was to have both groups form and use both types of representations and then to look at the resulting performance in each situation.

Method

Subjects. The subjects were a group of Novice and a group of Expert programmers.
Stimuli. The stimuli consisted of eight PPL programs with two types of flow charts and two types of questions for each of the eight programs. The flow charts consisted of Low Detail flow charts which described what the program did and High Detail flow charts which described how the program functioned. The questions consisted of Low Detail questions which asked a question about what the program did and High Detail questions which asked a question about how the program functioned.
Procedure. In order to encourage subjects to form a representation at one level of abstraction or another subjects were first shown a flow chart of a program at a given level of abstraction. They were then shown the actual program along with a question about the program. After seeing the program itself, along with the question the subjects answered the question . Level of detail of the flow chart was combined with level of detail of the question to form four conditions for each group of subjects; two "congruent" conditions and two "non-congruent" conditions. In the two congruent conditions subjects saw either a Low Detail flow chart and then a Low Detail question or a High Detail flow chart and then a High Detail question. In the two "non-congruent" conditions subjects saw either a Low Detail flow chart and then a High Detail question or a High Detail flow chart and then a Low Detail question.

Two times were recorded: Comprehension time, that is the time it took the subject to say that s/he understood the flow chart well enough to go on and study the program and the question together and Question time, that is the time interval between seeing the program with the question and being able to write down the answer.

A "Code Only" control conditon was included in which subjects saw the program itself immediately and then saw the same program again with the question.

Results and Discussion

I. Comprehension Time

(Interval between seeing the flow chart and understanding it)

Comparing the results of the Low Detail, High Detail and Code Only conditions, subjects understood the Low Detail flow charts more quickly than the High Detail flow charts and the High Detail flow charts more quickly than the code alone. The flow charts do seem to aid comprehension and it appears that they do so by organizing the information for the subjects rather than by just reducing the information since the High Detail flow charts, which had more units on the average than the code were understood more quickly than the code alone in the Code Only control condition (Units are boxes in the flow chart or single lines in the code.)

II. Question Time (Interval between seeing the question and being able to answer it)

A. Non-congruent Conditons (here the level of detail of the flow chart and the question do not match)

The results of the non-congruent conditions give us information about each group's most natural level of abstraction. The rationale behind this is that the flow charts prepare the subjects to represent the code at one level of abstraction or another. If the level then turns out to be inappropriate for answering the question subjects will be substantially slowed down if the appropriate level is also not the level which they usually use. On the other hand if the appropriate level is the level which they usually use they should still be able to quickly form a representation at that level.

--- --- --- ---

Insert Figure 3 Here.

--- --- --- ---

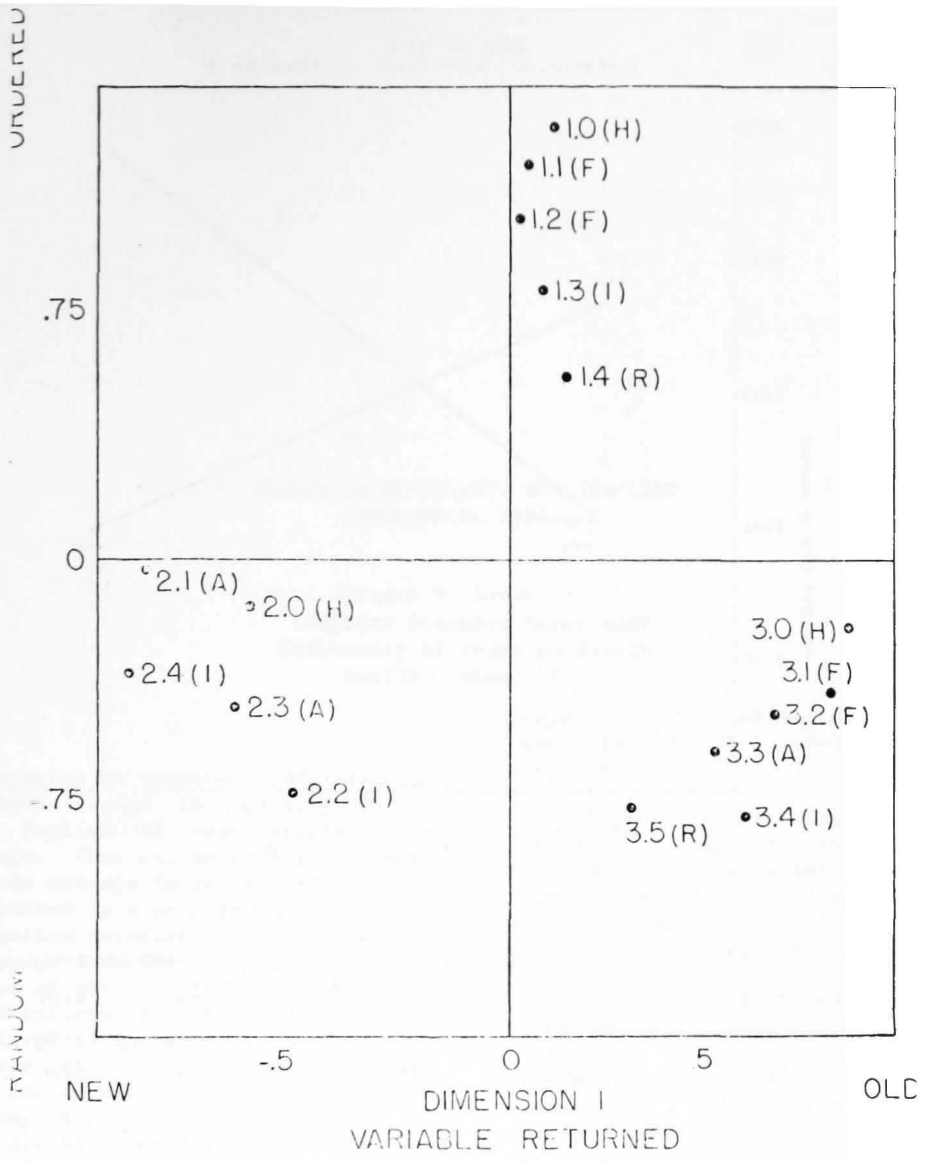
Looking at the left hand side of Figure 3 we see that the Novices are actually faster than the Experts when answering a High Detail question although we see on the right hand side that the Experts are faster than the Novices when answering a Low Detail question. That the Novices are faster than the Experts when having to answer a High Detail question indicates that the Novices are used to forming High Detail, that is non-abstract, specific representations of how programs function. That the Experts are faster than the Novices when having to answer a Low Detail question indicates that the Experts are used to forming Low Detail, that is abstract representations of what a program is doing. The error rates for the non-congruent and Code Only conditions both show the same interaction, Novices do well on High Detail questions while Experts do well on Low Detail questions.

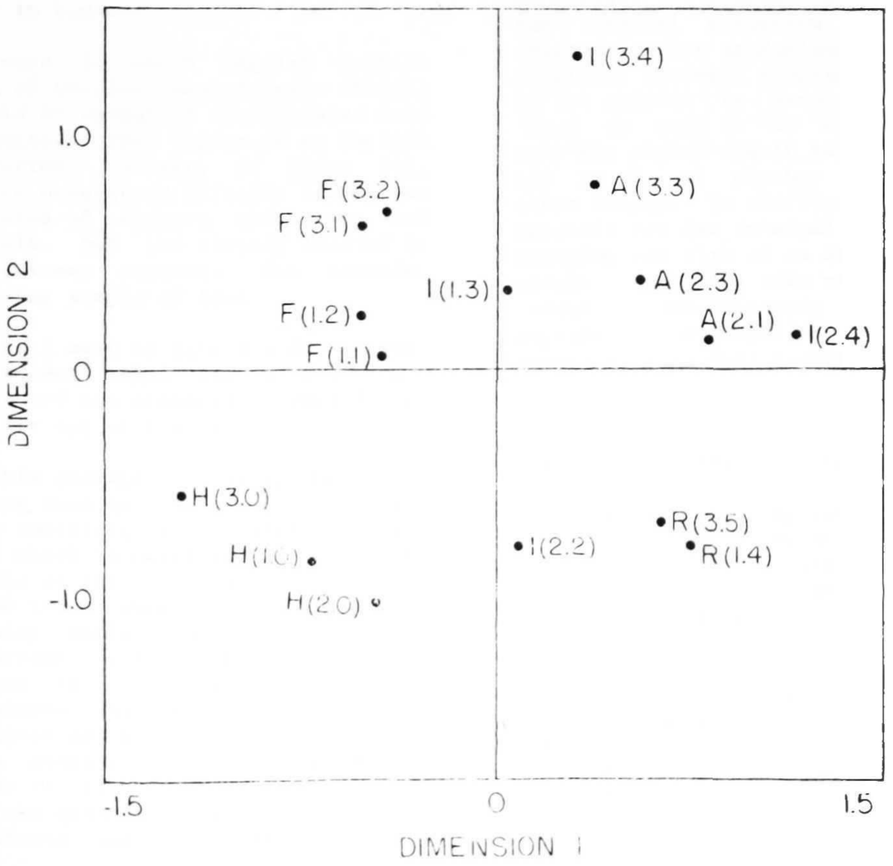
B. Congruent Conditons (here the level of detail of the flow chart and the question match)

The results of the Non-congruent Conditions suggest that Novices do represent how a program functions while Experts represent what a program does. Although it seems difficult for both groups to switch from their preferred levels of representation when they have been encouraged to use them, it is still possible that both groups can form effective representations at their non-preferred levels if they are given the proper aid. This is what was done in the congruent conditons. The results of these conditons showed that both Novices and Experts answered both High and Low Detail questions equally quickly suggesting that they can be somewhat flexible in their encoding.

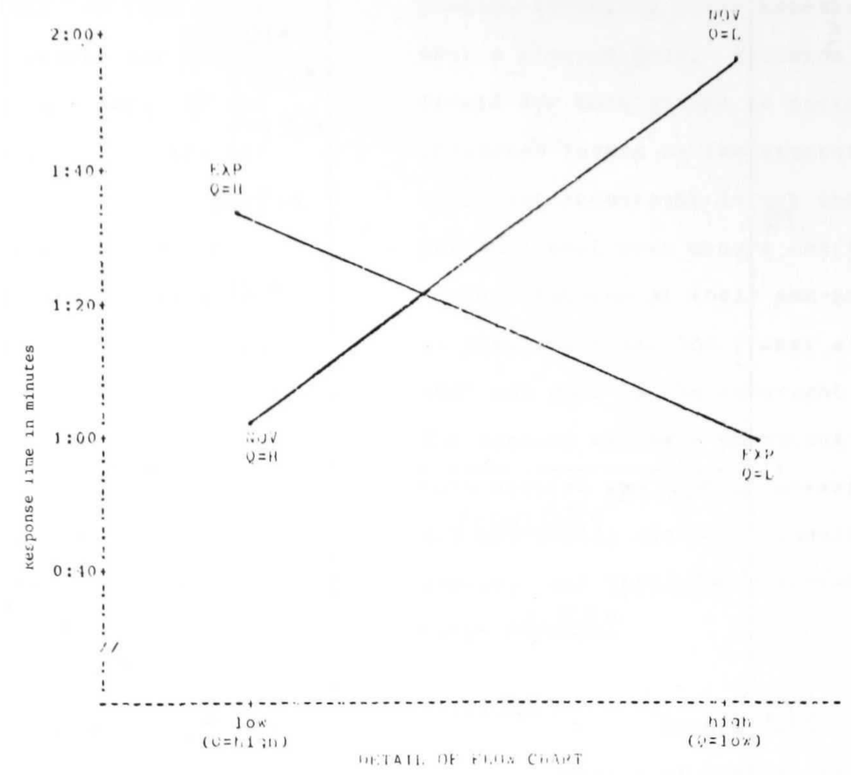
Conclusions

The results of the two experiments reported here suggest that Novice and Expert computer programmers represent information about programs differently. The representations of the Experts is more abstract and is based on what the program does, while the representation of the Novices is more detailed and is based on how the program functions. These differences also influence the utilization of the information being represented.


ORDERED
RANDOM
1.0 (H)
1.1 (F)
1.2 (F)
1.3 (I)
1.4 (R)
2.1 (A)
2.0 (H)
2.4 (I)
2.3 (A)
2.2 (I)
3.0 (H)
3.1 (F)
3.2 (F)
3.3 (A)
3.4 (I)
3.5 (R)
.75
0
-.75
-.5
0
.5
NEW
OLD
DIMENSION I
VARIABLE RETURNED


I (3.4)
F (3.2)
A (3.3)
F (3.1)
I (1.3)
A (2.3)
F (1.2)
A (2.1)
I (2.4)
F (1.1)
H (3.0)
R (3.5)
H (1.0)
I (2.2)
R (1.4)
H (2.0)
1.0
0
-1.0
-1.5
0
1.5
DIMENSION 2
DIMENSION I

QUESTION TIME
(QUESTION AND FLOW CHART NOT CONGRUENT)



MEANS

	CONGRUENT		NOT CONGRUENT	
FLOW CHART =	LOW	HIGH	LOW	HIGH
NOVICE	1:11	1:22	1:01	1:55
EXPERT	1:03	1:10	1:32	1:01