The Interaction Between Working Memory and Units of Procedural Knowledge

Thomas A. Kanarski
Donald J. Foss
University of Texas at Austin

This study focuses on the interaction between units of procedural knowledge and working memory. Evidence is provided supporting the concept that procedural knowledge is stored in memory as modular units often referred to as subroutines. The paper also gives evidence supporting the notion that more than one working memory exists in cognitive processing.

To illustrate the idea of units or modules of procedural knowledge, consider a person who must eliminate information from a computer data base system. A reasonable description of the activity will break it into a sequence of chunks (mental or behavioral units) that vary in complexity and duration. For a particular data base management system, called OMNI, one such description is: 1) find the information; 2) mark the information for later elimination; 3) eliminate the marked information. These steps can be thought of as labels. "Find the information" would be an identifier for a group or packet of instructions to get the appropriate data displayed on the CRT. For the purposes of this paper, a packet of instructions is called a unit or module of procedural knowledge.

A module of procedural knowledge may use other units. In the example, the "find the information" module may use a module called "GET" (instructions to use the "GET" command in OMNI). In turn, the "GET" module may use other modules, which use still others, and so forth, until specific motor commands are issued. The more general module, "find the information," deals with a plan of action, the level at which this study focuses.

WORKING MEMORY

An assumption made in this study is that the units are processed in working memory areas that hold instruction groups for processing on a temporary basis (Baddeley and Hitch, 1974). This centralized setup greatly reduces processing overhead (Kanarski, 1984).

Two subroutine retrieval models of working memory are considered in this paper (Sternberg, Monsell, Knoll, and Wright, 1980). For a discussion of how a limited-capacity model (Baddeley and Hitch, 1974) and a competition model (Lashley, 1951; Wickelgren, 1969) of working

memory would interact with units of knowledge see Kanarski (1984). The first subroutine retrieval model (SRM-1) loads modules or subroutines into working memory as needed. The module is processed, then the next module is found and loaded into working memory. SRM-1 has been used to predict the rapid movement sequence of speech and typing at the level of motor commands (Sternberg, et al., 1980). The data also suggest that motor command modules are subject to either rapid decay or destructive reads.

A closer look at the plan-of-action level of processing suggests that the rapid loss of information in working memory may not be efficient. Unlike motor movements, the same module implementing some portion of the overall plan is very likely to be repeated. Using the example above, a person finding several items of information in the data base which are to be deleted may repeat the same command sequence several times to find all of the items. In this case, it would be more efficient to check the contents of working memory and determine if they are needed for the next round of processing (Kanarski, 1984). This type of working memory will be referred to as the subroutine retrieval model - type 2 (SRM-2). The primary difference between the two working memory models is whether information is subject to rapid loss (as in SRM-1) or not (as in SRM-2).

By having a person perform a task for which more than one method exists, it is possible to behaviorally differentiate the two working memory models. The example of eliminating information from a data base using the OMNI data base management system is such a task. The appropriate OMNI commands are "GET" (find the information), "DELETE" (mark the information for later elimination), and "WEED" (eliminate the information). Suppose the user had several items of information to eliminate from the data base. The user could "GET" the location of each item, "DELETE" (mark) each item, and then "WEED" all of the items. This is called a short cycle method and is denoted by GET / DELETE / WEED /. Alternatively, the user could "GET" one item and "DELETE" (mark) it. This is repeated until all of the items are found and marked. Then the user could "WEED" all of the items. This is called a medium cycle method and is represented by GET DELETE / WEED /. Using the last possible method, the user could "GET" one item, "DELETE" (mark) it, and then "WEED" it. This sequence, the long cycle method, is repeated for each item. This method is denoted by GET DELETE WEED /.

Suppose that each OMNI command is represented as a module of procedural knowledge in the user's memory. After processing, a module in the SRM-1 working memory is not available for further processing. If the module is to be used again, it must be found and loaded into working memory. In terms of the OMNI task, the time for the operator to restart a DELETE command in the short cycle method should be the same as the time to start the DELETE command after finishing the GET command in both the medium and long cycle methods.

In a SRM-2 working memory, a check is made to determine if the current module is required for further processing. If it is not needed, the proper module is found and loaded into working memory. If the current module is needed, processing can simply be restarted, bypassing the search and load processes. In terms of the OMNI task, it should be faster to restart the DELETE command in the short cycle method than to start the DELETE command after finishing the GET command in either the medium or long cycle methods. Also, the time to start the DELETE command in both the medium and long cycle methods should be the same. Both methods require that a new module (DELETE) be found and loaded before processing can continue.

METHODS

The subjects were thirty undergraduates at the University of Texas at Austin selected from introductory psychology courses and those responding to a newspaper advertisement. All of the subjects had little or no computer experience and they were all were able to touch type at least 30 WPM. Expert users were not used because they would have specialized task strategies that would confound the study.

The subjects were tested individually. After a typing test, the subject was given a modified version of the OMNI manual to read. The subject was told not to memorize the manual since it would be available during the experiment. The subject then attempted seven practice tasks. These could be accomplished using only one OMNI command.

The subject was then given seven experimental tasks. The tasks fell into one of three types and there were at least two possible methods to accomplish each task type. One task was to eliminate five items from the data base. The methods are described earlier in this paper. Another task was to add five items to the data base and maintain the alphabetical order of the data base. The commands are ADD (add an item to the end of the data base) and ORDER (get the data base in alphabetical order). This task has a short cycle method (ADD / ORDER /) and a long cycle method (ADD ORDER /). The last task was to change information of five items in the data base. The commands are GET (find the item) and CHANGE (change information in the item). This task also has a short cycle method (GET / CHANGE /) and a long cycle method (GET CHANGE /).

The instructions for each task stated how the task was to be accomplished without explicitly stating which commands were to be used. Thus, each subject performed each task type using the short, medium, and long cycle methods. That is, cycle length was the within-subject variable. The experimenter did not give the subject any help unless there was an equipment failure or the subject deviated from the task

instructions. The subjects were not told about either the experimental hypothesis or whether a task was practice or experimental.

The subjects were randomly assigned to one of two groups depending on the order that the tasks were presented. The tasks were interweaved as not to have a task of a particular type follow a task of the same type.

The commands of interest were DELETE, ADD, and CHANGE. Each of these commands occured in only one task type and were used the same number of times in each task. The dependent measure was the mean times before the third, fourth, and fifth use of a command of interest. The first and second times were not used because pilot studies indicated that there were large practice effects influencing the measure. By the third use of a command with a task, none of the subjects referred to either the manual or the task instructions.

RESULTS

Three ANOVAs were performed, one for each command of interest. If the last three times to the next use of a command could not be properly extracted, that subject's data were thrown out for that command only. This occurred once in each analysis.

In each ANOVA, the time to the next use of the command in the short cycle method was significantly less than that time in the long cycle method (see Table 1).

TASK	CYCLE LENGTH	MEAN TIME TO THE NEXT USE OF COMMAND (SECONDS)	F	DF	P
ADD	SHORT	2.65	5.86	1,27	< .05
	LONG	3.34			
CHANGE	SHORT	3.80	12.27	1,27	< .05
	LONG	4.88			
DELETE	SHORT	3.50	6.65	1,54	< .05
	MEDIUM	4.45			
	LONG	4.24			

TABLE 1

The order main effects and the order by cycle length interactions were not significant in any of the analyses. A planned comparison showed that the time to the next command in the short cycle method of the DELETE task was significantly less than both the medium and long cycle methods ($F = 13.70$, $p < .05$). Also, the medium and long cycle method times were not significantly different ($F < 1.0$).

DISCUSSION

The data clearly support the subroutine retrieval model in which the contents of working memory are not subject to rapid loss (SRM-2). The faster times to start commands of interest in the short cycle methods over the long cycle methods indicate that the current contents of working memory are checked. If the match is successful, as it would be in a short cycle method, processing the current contents simply recurs. If the match is not successful, as in the medium and long cycle methods, the proper module must be found and loaded into working memory before processing can continue.

SRM-2 also predicted that there would be no difference in the times to the next use of the DELETE command between the medium and long cycle methods. In both cases, the DELETE module was not in working memory. The same amount of time, on the average, was spent searching for and loading the module in both of the methods.

The assumption that procedural knowledge is packaged into modules simplifies computational theories of cognition. The results of this study were also predicted under this assumption. For a discussion of how non-modular procedural knowledge interacts with working memory see Kanarski (1984).

The data from this study and from Sternberg, et al. (1980) suggest that there exist at least two working memories. One is responsible for processing motor command modules. This working memory acts like SRM-1, that is, there is a rapid loss of information to prevent it from interfering with new information coming into working memory. The other working memory processes information at the plan-of-action level. This working memory retains data for possible continued processing. That there may be two, possibly more, working memories that have different properties is not unreasonable. Considering the tremendous replication of neural structures and localization of function in the brain, many working memory areas, each with specialized processing capbilities, are certainly possible (Rosenzweig and Leiman, 1982).

REFERENCES

Baddeley, A.D. and Hitch, G. (1974). Working memory. In G.H. Bower (Ed.), The psychology of learning and motivation, advances in research and theory, volume 8.

Kanarski, T.A. (1984). Units of knowledge and working memory: Support for a subroutine retrieval model of working memory. Unpublished master's thesis, University of Texas, Austin, Tx.

Lashley, K.S. (1951). The problem of serial order in behavior. In L.A. Jeffress (Ed.), Cerebral mechanisms in behavior.

Sternberg, S., Monsell, S., Knoll, R.L., and Wright, C.E. (1980). The latency and duration of rapid movement sequences: Comparisons of speech and typewriting. In R.A. Cole (Ed), Perception and production of fluent speech. Hillsdale, NJ: Lawrence Erlbaum Associates.

Wickelgren, W.A. (1969). Context-sensitive coding, associative memory, and serial order in (speech) behavior. Psychological Review, 76, 1-15.