Planning Units in Text Editing Behavior

Scott P. Robertson and John B. Black

Cognitive Science Program
Yale University
Box 2158 Yale Station
New Haven, CT 06520

Abstract

The organization of text editing behavior can be characterized by graph structures containing goals, subgoals, goal outcomes, and actions. Here we propose a model to represent the goals and plans of text editor users based on goal-fate analysis (Schank & Abelson, 1977). The representation captures relationships between a user's multiple goals and shows how errors can result from badly formed plans. We discuss some data from a psychological experiment which supports the hypothesis that text editing behavior is chunked into distinct plan units. The cognitive components of pause times between keystrokes were revealed by statistically removing the physical time required between keystrokes. Finally, we suggest how a system which builds goal-fate graphs from keystroke input might be useful in providing specific help information that is keyed to a user's intentions.

Introduction

Text editor users are not just hitting keys. They are engaged in a complex cognitive task that requires the formulation of goals and the execution of plans to achieve those goals. The analysis of goals and plans has been useful in many diverse applications. Some of these include story understanding (Schank & Abelson, 1977; Seifert, Robertson, & Black, 1982; Wilensky, 1978), planning and conversing about physical actions (McDermott, 1978; Miller, Galanter, & Pribram, 1960) composition and understanding of computer programs (Ehrlich & Soloway, 1982), production and understanding of natural dialogue (Allen & Perrault, 1980; Appelt, 1982; Cohen & Perrault, 1980; Robertson, Black, & Johnson, 1981), predicting affective states of actors in stories (Lehnert, 1981), and even text editing behavior (Card, Moran, & Newell, 1980, 1983).

In story understanding, comprehension is improved when specific inferences about the goals and plans of characters are made (Anderson, Spiro, & Montague, 1977; Spiro, Bruce, & Brewer, 1980). An understanding of a character's goals provides a framework within which to unite otherwise disconnected actions. Analysis of the goals and plans of text editor users would also be a useful undertaking because the acquisition of text editing skills is aided by instructional materials stessing goals and plans as opposed to procedural details (Carroll & Mack, 1983; Carroll & Mack, in press).

Card, Moran and Newell (1980, 1983) have undertaken a thorough theoretical and empirical investigation of text editing behavior within a goal-oriented, information processing framework. Their "GOMS" model consists of an organized treatment of several types of knowledge about text editing. This includes knowledge about text editing Goals, the Operators, or procedures, necessary to realize those goals, alternative Methods for goal achievement, and Selection Rules for deciding among methods. Our study supports the GOMS framework for the production of text editing behavior in a full-screen enviornment. It goes further, however, by suggesting that the outcomes of goal attempts are monitored as an aid in learning and memory of procedures.

Goal-Fate Analysis of Text Editing Behavior

In this paper we would like to propose an application in the domain of text editing behavior of the goal tracking mechanism described by Schank & Abelson (1977) for stories. The system, called *goal fate analysis*, specifically represents goal *outcomes*, and *relationships* between goals that are not part of the same plan (e.g. unexpected consequences of errors, interruptions, embedded plans, etc.). Goal-fate analysis is a representation of behavior from an *observer's* point of view. It is therefore an *understanding* model, designed in this context to characterize a known set of keystrokes in a goal-oriented framework. A computer which tried to understand the behavior of its users would, after all, be in an observer's position. The analysis assumes, however, that the goals and plans used to explain behavior are isomorphic with the goals and plans used to produce the behavior.

As an example of how text-editing goals can be represented, consider one plan for achieving the goal of changing the word *the* to *those* on a typical, full-screen text editor in "replace" mode (where typeover occurs and text is not moved automatically). The main goal, CHANGE, can be decomposed into subgoals involving DELETEing the word *the* and INSERTing the word *those*. Each of these subgoals may be further decomposed, for example INSERTing will require MAKEing-ROOM for extra letters and TYPEing the word *those*. A goal fate graph containing the hierarchically organized goals and subgoals necessary to solve the *the-->those* problem appears in Figure 1. Goals are

```
Goals & Outcomes                   Keystrokes
-----------------                  ---------
CHANGE(the-->those)
| DELETE(the)
| | MOVE-POINTER(under t) ---- 10 right-arrows
| | SUCCESS
| |
| | USE-KEY(word-delete) ----- 1 word-delete
| | SUCCESS
| SUCCESS
|
| INSERT(those)
| | MAKE-ROOM(5 spaces) ------ 5 ins-spaces
| | SUCCESS
| |
| | TYPE(those) ------------- t h o s e
| | SUCCESS
| SUCCESS
SUCCESS
```

Figure 1:
CHANGE plan for a typical full-screen text editor. Goals and their outcomes are in capital letters. Each level of indentation represents a set of subgoals. Keystrokes are organized by the higher level planning units.

represented in capital letters, subgoals are indented under major goals and must all be achieved in order to achieve the major goal. Keystrokes are in small letters on the right. The keystrokes run from top to bottom in time, and at any level of indentation, goals that are above must be achieved before the goals below them are initiated.

The goal-fate graph for this task explicitly represents the outcomes of goals. We propose that when a text editor user comes to the end of a plan unit, the plan must be evaluated for success before the next planning unit is initiated (Card, Moran, & Newell, 1983). The usefulness of representing goal outcomes is apparent when we consider an error in the CHANGE plan just discussed. Figure 2 shows the goal-fate representation of a case in which the user did not create enough room before inserting the word *those*. A FAILURE is assigned to the TYPE goal under the first INSERT goal, and this outcome initiates a new INSERT goal and its associated subgoals (i.e. MAKE-ROOM and TYPE). Once the new INSERT goal has been achieved, the original INSERT goal is also assigned a SUCCESS outcome. Note that a number of complex goal relationships besides goal-subgoal relationships are apparent in the representation. The second INSERT goal, for example, arose because of a failure in the TYPE portion of the old INSERT goal. Note also that this analysis provides a representation that can be used to evaluate the success and effectiveness of a plan. Nested goals that are achieved in the same manner (e.g. the nested INSERTs both achieved by MAKE-SPACE+TYPE) can be combined into a single MAKE-SPACE+TYPE plan in future INSERT tasks.

Pause Times as Evidence for Plan Structures

A planning model of text editing behavior predicts that boundaries of major goals will be characterized by increased processing because goal evaluation and initiation is occuring. The goal just attempted must be evaluated and, if this evaluation succeeds, the next goal must be initiated. Failures, of course, must also be evaluated at plan boundaries and problem solving initiated if necessary. These evaluations will require a small amount of time which should be observable in slight pauses before keystrokes that initiate major plan sequences. We turn now to an experiment that addresses this hypothesis.

Subjects. Twenty-six subjects participated in a study of text editing behavior designed to test the pause time hypothesis. The subjects were Yale undergraduates who had no prior experience with computer text editing. They were paid $4.00 each to spend an hour learning how to change single words in sentences presented one at a time in an experimental text editing task.

Apparatus. A special editor and task presentation program was designed and implemented (in the 'C' programming language) on a PDP-11/40. The program presented text editing tasks to subjects and collected keystrokes and times between keystrokes during editing. Each task was presented as a pair of sentences displayed in the center of the CRT screen. Subjects were instructed that the first sentence should be changed so that it looked like the second sentence. This always involved modifying a single word. After viewing a sentence pair, the subjects pressed a button to begin editing the first sentence of the pair. At this time, the sentence to be edited appeared between two dotted lines, and the second sentence disappeared. A ***pointer***, positioned initially under the first letter, was imbedded in the bottom line and could be moved to indicate the current letter. After changing the sentence, the subjects pressed another button to end the task and display the next sentence pair.

The editor itself was as simple as possible. Eight keys on the number pad were utilized as special function keys. The function keys were labelled with a 4 letter mnemonic. Their labels and functions are shown in Table 1. The function of each key was explained to subjects before the experiment

```
Goals & Outcomes                   Keystrokes
-----------------                  ---------
CHANGE(the-->those)
| DELETE(the)
| | MOVE-POINTER(under t) ---- 10 right-arrows
| | SUCCESS
| |
| | USE-KEY(word-delete) ----- 1 word-delete
| | SUCCESS
| SUCCESS
|
| INSERT(those)
| | MAKE-ROOM(4 spaces) ------ 4 ins-spaces
| | SUCCESS
| |
| | TYPE(those) ------------- t h o s
| | FAILURE(no space)
| | | |
| | | INSERT(e)
| | | | MAKE-ROOM(1 space) -- 1 ins-spaces
| | | | SUCCESS
| | | |
| | | | TYPE(e) ------------- e
| | | | SUCCESS
| | | SUCCESS
| | SUCCESS
| SUCCESS
SUCCESS
```

Figure 2:
Goal-fate representation of an error and recovery during execution of a CHANGE plan.

```
Label          Function
-----          --------
STRT...Start the editing session.
MOVL...Move the pointer left one space.
MOVR...Move the pointer right one space.
FNSH...End the editing session. Also leave typing mode.
PSHR...Push the current letter right one space (all adjacent
       letters to the right are also pushed, adjacent spaces
       to the right are closed).
PSHL...Push the current letter left one space (all adjacent
       letters to the left are also pushed, adjacent spaces
       to the left are closed).
INSR...Enter typing mode to type letters on the screen.
REMV...Remove current letter from the screen (no fill).
```

Table 1:
Key functions and labels for the editor used in the pause time experiment.

began and they were shown an example of each key's effect on the display. Subjects were free to employ any strategy they wished using these functions and none were suggested by the instructions. The first six trials for each subject were practice trials involving deleting and inserting words. The next thirty trials were *learning* trials during which subjects solved problems involving changing words. After the learning trials, the subjects repeated the thirty change tasks in *test* trials.

Tasks and Dependent Measures. Subjects were presented with three different kinds of editing tasks involving changing one word of a sentence into another word. The tasks differed on the lengths of the words to be changed relative to the lengths of their replacement words. In this paper, however, we will be concerned only with those tasks in which each word to be changed was the same size as its replacement word. Each subject perfomed ten such "same length" tasks.

All keystrokes and times between keystrokes were collected. Since critical comparisons were to be made on times which occurred between different pairs of keys, the raw times could not be used. Instead, estimates of the times required to hit all 64 different combinations of two function keys (order and hitting the same key twice are counted) were obtained from an exercise in which subjects hit two keys in sequence after reading a pair of key labels displayed on the screen. The subjects performed this task twice during their editing session, running through all pairs (presented randomly) twice each time. Thus, each subject provided a mean time for each key combination based on four separate observations. The mean times for each key pair were averaged over subjects and these times were taken as estimates of the relative times required to move between and hit two keys in succession. A regression of the key-pair estimates on the raw scores was perfomed on each subject's data. Each observed keystroke's predictor value was the key-pair estimate for that keystroke and the prior observed keystroke. Residuals from the regression represent the deviation of each raw score from its *predicted* value based on the overall linear association between the key-pair estimates and the raw scores. Residual scores are large to the extent that additional, cognitive processes are contributing to raw score times over and above the physical time required to go between the particular key pairs (see Haberlant (1980) and Haberlant, Berian, & Sandson (1980) for application of this technique to the study of cognitive processes apparent in reading times of sentences in short stories). Typing times between letters and times between letters and function keys were not estimated and were not included in the regression or subsequent analyses.

Common Plans. For each task type, subjects pursued a number of different strategies. The strategies were characterized by different combinations of a few basic subplans. Three common subplans in the same length tasks were OVERTYPE, REMOVE, and INSERT. OVERTYPE is defined as an instance of replacing old letters or words with new ones by typing over them. It consists of pointer movement to the old word, entering typing mode, typing, and leaving typing mode. This plan is shown in the top portion of Figure 3. REMOVE is defined as an instance of deleting letters from the screen. It consists of pointer movement to the old word and sequential instances of letter deletion and pointer movement to the next letter. This subplan is shown in the bottom portion of Figure 3. INSERT is defined as an instance of typing letters into a blank space. It consists of pointer movement to the space, entering typing mode, typing, and leaving typing mode. This plan is also shown in the bottom portion of Figure 3. REMOVE and INSERT were always combined as one approach to the goal of changing a word, while OVERTYPE accomplishes the same goal in one step.

In the same length task, 12 subjects (46%) used the OVERTYPE plan during the learning trials and 13 subjects (50%) used the REMOVE+INSERT plan (the remaining subject used an unusually complex plan not discussed here). During the test phase, 18 subjects (69%) used the OVERTYPE plan while only 7 (27%) stayed with the REMOVE+INSERT plan. Analysis of keystroke time residuals was perfomed separately for the 12 OVERTYPE subjects and 7 REMOVE+INSERT subjects who repeated their initial plan in the test trials.

Pause Time Results. The OVERTYPE plan has one subplan boundary within it for which residuals could be calculated--when typing mode is entered (see Figure 3). We expected that the time between the last MOVR keystroke in the MOVE-POINTER sequence and the INSR keystroke would be much longer than predicted by the physical distance

```
Goals                                    Keystrokes
-----                                    ----------
                  Plan 1:
CHANGE(word1->word2)
   OVERTYPE(word1->word2)
      MOVE-POINTER(to first letter)     MOVR, MOVR...MOVR
     *ENTER-TYPING-MODE                 INSR
      TYPE(word)                        w o r d
      LEAVE-TYPING-MODE                 FNSH

                  Plan 2:
CHANGE(word1->word2)
   REMOVE(word1)
      MOVE-POINTER(to first letter)     MOVR, MOVR...MOVR
     *DELETE(word1)                     REMV, MOVR, REMV, MOVR..
   INSERT(word2)                                    .
     *MOVE-POINTER(to first space)      MOVL, MOVL...MOVL
     *ENTER-TYPING-MODE                 INSR
      TYPE(word)                        w o r d
      LEAVE-TYPING-MODE                 FNSH
```

Figure 3:
Alternative plans for changing a word in the same length task. Plan boundaries analysed for increased keystroke time are indicated by asterisks.

	Keystroke		
Trials	MOVR	*INSR	Mean
Learning	-0.473	3.407	1.467
Test	-0.542	1.750	0.604
Mean	-0.508	2.579	1.036

Table 2:
Mean residuals for the MOVR and INSR keystrokes in the OVERTYPE plan. INSR is the keystroke at the plan boundary.

between MOVR and INSR. In Table 2 the mean residuals for this keystroke are compared with the mean residuals for the prior keystroke (MOVR) for both the learning and test trials. Analysis of variance shows the mean residual for INSR to be significantly larger than the mean residual for MOVR, $F(1,11) = 46.95$, $p<.001$. There is also a significant decrease in the mean residuals from learning to test time, $F(1,11) = 10.89$, $p<.01$, indicating that there was more cognitive processing involved in the task during learning than at test time. Interestingly, the interaction was also significant, $F(1,11) = 16.33$, $p<.001$, suggesting that the reduction in cognitive load from learning time to test time was greater for the keystroke at the subplan boundary than for the prior keystroke.

The REMOVE+INSERT plan provides a more interesting analysis because it allows for a greater number of comparisons and because it allows comparison of the same keystroke pair, MOVR-REMV, at both a subplan boundary and within a subplan unit (see Figure 3). In the REMOVE portion of this plan, it was predicted that the initial REMV keystroke would be longer than the keystroke that precedes it (MOVR) because it is the first keystroke in the DELETE sequence. It should also be longer than subsequent keystrokes within DELETE, including subsequent REMV's. Table 3 shows the mean residuals for the sequence of keystrokes around the REMOVE boundary. As predicted, the initial REMV keystroke produced a much higher residual than surrounding keystrokes. Analysis of variance on this data shows the differences among keystrokes to be significant, $F(5,30) = 21.78$, $p<.001$, the test trial residuals to be lower than the learning trial residuals, $F(1,6) = 14.71$, $p<.01$, and a significant interaction, $F(5,30) = 4.38$, $p<.001$. The interaction appears again to be due to an attenuated decrease in processing load from learning to test trials at the subplan boundary.

Finally, the INSERT portion of the REMOVE+INSERT plan provides a chance to look at two subplan boundaries in the same analysis. Residuals could be calculated both for the MOVL keystroke which initiates pointer movement after the REMOVE portion of the plan, and for the later INSR keystroke. (see Figure 3). Table 4 shows the mean residuals for these two critical keystrokes and their surrounding keystrokes. Both boundary residuals are high relative to the other means. Again, analysis of variance shows the differences among keystrokes to be significant, $F(4,24) = 17.49$, $p<.001$, and a significant interaction, $F(4,24) = 4.88$, $p<.01$. There was no effect of the trials factor $F(1,6) = 1.93$, because of the peculiar nature of the interaction. While the keystroke residual at the beginning of typing mode dropped, as usual, from learning to test trials, the keystroke residual at the initiation of pointer movement actually increased. One hypothesis is that during the learning phase subjects do not have pointer movement and entry into insert mode integrated into a single plan unit. Entry into typing mode in this phase is at a *major* boundary and requires considerable cognitive processing to initiate. The relative size of the INSR residual when compared to the boundary MOVL residual during the learning phase tends to bear out that INSR is at a major boundary. By test time, however, typing mode entry has been subordinated to a higher order plan unit which involves pointer movement at the beginning. This makes the initial MOVL keystroke a more significant boundary marker. A comparison of the sizes of residuals for the boundary MOVL and INSR keystrokes in the test phase shows that their relative importance has reversed. Thus the initiation of pointer movement involves more cognitive processing during the test phase because it has become the initial action in the important INSERT subplan. The INSR residual drops dramatically in the test phase not only because the decision making process is smoother, but also because it has become part of a higher level plan already initiated.

These data lend support to the notion of plan units in text editing behavior. The interactions also suggest some interesting refinements to classic views of learning by isolating "practice effects" at planning boundaries and by providing some evidence for plan restructuring. If supported by further

```
                              REMOVE
                                |
              +-----------------+-----------------+
              |                                   |
         MOVE-POINTER                          DELETE
              |                                   |
          ---+---+          +--------+---------+---------+---------+
                 |          |        |         |         |         |
```

Trials	MOVR	*REMV	MOVR	REMV	MOVR	REMV	Mean
Learning	-0.219	2.651	0.479	-0.091	-0.253	-0.180	0.398
Test	-0.077	1.525	-0.436	-0.359	-0.326	-0.200	0.021
Mean	-0.148	2.088	0.021	-0.225	-0.290	-0.190	0.209

Table 3:
Mean residuals for the keystrokes in the REMOVE portion of the REMOVE+INSERT plan. The first REMV keystroke is at the plan boundary.

```
  REMOVE                                    INSERT
    |                                         |
 ---+----+                    +---------+-----+----------
         |                    |               |
       DELETE           MOVE-POINTER   ENTER-TYPING-MODE
         |                    |               |
      --+---+          +--------++. .----+    |
            |          |       |       |      |
Trials     REMV      *MOVL    MOVL ... MOVL  *INSR    Mean
------     -----     -----    -----   -----  -----   -----
```

Trials	REMV	*MOVL	MOVL ... MOVL	*INSR	Mean
Learning	-0.007	1.976	-0.485...-0.485	4.412	1.082
Test	-0.300	3.336	-0.191...-0.358	1.711	0.839
Mean	-0.153	2.655	-0.338...-0.421	3.061	0.961

Table 4:
Mean residuals for the keystroke sequence in the INSERT portion of the REMOVE+INSERT plan. The first MOVL keystroke and the INSR keystroke are at plan boundaries.

analysis and research, these results would indicate that learning involves building plan units to fit a task and then refining the evaluation and initiation phases at plan boundaries (Neves & Anderson, 1981).

Usefulness of Plan Analysis

Plan analysis is useful to researchers in understanding the cognitive mechanisms that guide procedural behavior. Goal fate analysis specifies the relations among goals and between goals and their outcomes. Performance analysis based on goal-fate graphs would yield more detailed explanations of the behavior of text editor users than simple measures of overall success at a task or overall task time. Further, knowing how errors occur and how text editor user's plans look would be very useful in helping users to sharpen their skills.

The ultimate use of goal-fate analysis, or plan analysis of any kind for that matter, would be as a tool in understanding the conceptual components of errors generated by new users. A plan recognition system which monitored keystroke input and built goal-fate representations would be in a position to generate useful help information when learners got into trouble. By specifying bugs in plans or simplifications of overly-complex plans, a goal-tracker could provide a very helpful training function. The task of plan recognition would be greatly simplified by combining training exercises, in which the goals are known, with a simple goal tracking mechanism that could build error representations like the one in Figure 2 and generate help information from them (e.g. "You need to make another space" if the user was stuck after the initial failure, or "It would be simpler to make all of your spaces in one step" after the entire episode was understood). It is even conceivable that knowledge of a user's preferred plans could be kept after being inferred during simplified learning trials to be used in later, more unconstrained contexts.

Conclusion

Text editor users chunk behavior into distinct planning units. At plan boundaries, active goals are evaluated for their success and new goals are initiated. This is evident in pause time data from real text editing sessions. The goal-fate analysis of text editing behavior provides a useful and potentially practical way of understanding errors and relationships between sets of actions. This kind of analysis could also be used to help new users organize their behavior in a sensible, goal-directed manner.

Acknowledgements and Notes

This research is funded by a grant to the Yale Cognitive Science Program from the System Development Foundation. The research was initiated at the IBM, Thomas J. Watson Research center where John Carroll and Robert Mack deserve special thanks. We would also like to thank Mara Miller and Brian Reiser for their comments on the paper. The first author is now at the Department of Psychology, Catholic University of America, Washington, D.C., 20064.

References

Allen, J., & Perrault, C.R. Analyzing intention in utterances. *Artificial Intelligence*, 1980, **15**, 143-178.

Anderson, R.C., & Spiro, R.J. *Schooling and the acquisition of knowledge.* Hillsdale, N.J.: Erlbaum, 1977.

Appelt, D.E. Planning Natural-Language Utterances. *Proceedings of the National Conference on Artificial Intelligence*, 1982, 59-62.

Card, S.K., Moran, T.P., & Newell, A. Computer text editing: An information-processing analysis of a routine cognitive skill. *Cognitive Psychology*, 1980, **12**, 32-74.

Card, S.K., Moran, T.P., & Newell, A. *The psychology of human-computer interaction.* Hillsdale, N.J.: Erlbaum, 1983.

Carroll, J.M. & Mack, R. Actively learning to use a word processor. In Cooper W. (Ed.), *Cognitive aspects of skilled typewriting.* New York: Springer-Verlag, 1983.

Carroll, J.M. & Mack, R. Actively learning to use a word processor: By doing, by thinking, and by knowing. J.C. Thomas & M. Schneider (Eds.), *Human factors in computers.* in press.

Cohen, P., & Perrault, C.R. Elements of a plan based theory of speech acts. *Cognitive Science*, 1979, **3**, 177-212.

Ehrlich, K., & Soloway, E. *An empirical investigation of the tacit plan knowledge in programming.* Technical Report #236, Department of Computer Science, Yale University, 1982.

Haberlandt, K. Story grammar and reading time of story constituents. *Poetics*, 1980, **9**, 99-116.

Haberlandt, K., Berian, C., & Sandson, J. The episode schema in story processing. *Journal of Verbal Learning and Verbal Behavior*, 1980, **19**, 635-650.

Lehnert, W.G. Plot units and narrative summarization. *Cognitive Science*, 1981, **5**, 293-331.

McDermott, D. Planning and acting. *Cognitive Science*, 1978, **2**, 71-109.

Neves, D.M., & Anderson, J.R. Knowledge compilation: Mechanisms for the automatization of cognitive skills. In J.R. Anderson (Ed.), *Cognitive skills and their acquisition*, Hillsdale, N.J.: Erlbaum, 1981.

Robertson, S.P., Black, J.B., & Johnson, P.N. Intention and topic in conversation. *Cognition and Brain Theory*, 1981, **4**, 303-326.

Schank, R.C., & Abelson, R.P. *Scripts, plans, goals, and understanding.* Hillsdale, N.J.: Erlbaum, 1977.

Spiro, R.J., Bruce, B.C., & Brewer, W.F. *Theoretical issues in reading comprehension: Perspectives from cognitive psychology, linguistics, artificial intelligence, and education.* Hillsdale, N.J.: Erlbaum, 1980.

Wilensky, R. *Understanding goal-based stories.* Technical Report #140, Department of Computer Science, Yale University, 1978.

Wilensky, R. Talking to UNIX in English: An overview of UC. *Proceedings of the National Conference on Artificial Intelligence*, 1982, 103-106.

Winograd, T. *Understanding natural language.* New York: Academic Press, 1972.