LICAI+: A Comprehension-Based Model of The Recall of Action Sequences

Muneo Kitajima
National Institute of
Bioscience and Human-Technology
1-1 Higashi Tsukuba Ibaraki 305, JAPAN
Tel: +81 (298) 54-6731
E-mail: kitajima@nibh.go.jp

Rodolfo Soto and Peter G. Polson
Institute of Cognitive Science
University of Colorado
Boulder, CO 80309-0345, USA
Tel: +1 (303) 492-5622
E-mail: {soto, ppolson}@psych.colorado.edu

ABSTRACT

This paper presents a model of occasional use of functions of an application by an experienced user of an environment like Windows 95 or the MacOS. We have developed a simulation model, LICAI+, that assumes that users store episodic records of correct steps discovered by exploration or told to them during training. They then use the application display and their goal as retrieval cues in attempts to recall these episodes later. The model predicts, and supporting data show, that tasks that violate the label-following strategy are not only hard to learn by exploration but also difficult to remember even if the correct steps have been previously presented.

Keywords

cognitive model, learning by exploration, label-following strategy, LICAI+

INTRODUCTION

Experienced users of an environment like Windows 95 or the MacOS are *occasional* users of many applications (e.g., a graphics package). Furthermore, many functions of a frequently used application like a word processor are only used occasionally (e.g., constructing and editing a table). Thus, a large majority of the *different* tasks undertaken by skilled users are performed infrequently (Santhanam & Wiedenbeck, 1993).

Such patterns of occasional use should constrain the design of usable computer systems. Ideally, such systems should consistently support learning by exploration. At a minimum, they should facilitate memory for action sequences learned by demonstration or by being looked up in a manual. The ease of recalling infrequently performed functions can be a major determinate of usability. This is not a novel claim. For example, the designers of the Xerox Star had very similar insights (Bewley, Roberts, Schroit, & Verplank, 1983; Smith, Irby, Kimball, Verplank, & Harslem, 1982). This paper presents a theoretical model of recall of tasks that have been done once or a few times and data supporting the model.

LICAI+ is a model of recall of occasionally used action sequences. LICAI+ assumes that users store episodic records of correct steps discovered by exploration or told to them during training. They then use the application display and their goal as retrieval cues in attempts to later recall these episodes. The resulting model of the recall process is similar to models of text recall (Wolfe & Kintsch, submitted).

LICAI+ is an extension of LICAI[1] (Kitajima & Polson, 1996; 1997) which is a model of the processes involved in comprehending task instructions and using the resulting goals to guide successful exploration. Both LICAI and LICAI+ are based on Kintsch's (1986; in press) construction-integration theory of text comprehension. LICAI+ adds to LICAI the processes involved in encoding and successfully retrieving encodings of correct actions. LICAI+ assumes that successful performance of occasionally performed tasks involves a mixture of recall of episodes of correct actions and problem solving if recall fails. The model is related to Ross' (1984) and Rickard's (1997) models of skill acquisition.

Following a general description of the LICAI+ model, we present a theoretically motivated analysis of recall of occasionally performed action sequences. Readers interested in a more detailed descriptions of the LICAI model should consult (Kitajima & Polson, 1995; 1996; 1997). In support of the LICAI+ model and our theoretical analysis we compare our simulation results with data reported by Franzke (1994; 1995) and Soto (1997). In conclusion, we describe design implications of our results. We demonstrate that *both* ease of learning by exploration and good recall are supported by similar attributes of an interface.

DESCRIPTION OF LICAI+

LICAI+ simulates skilled Mac users in an experiment where they are taught novel tasks using a new application, Cricket Graph III. The task instructions are very explicit but do not contain any information about how to perform the task. Then, at some later time ranging from several minutes to a week, they are tested for retention of these skills when given the task descriptions and the displays generated by the application as retrieval cues. Users attempt to perform each task by exploration and/or recalling an action sequence. However, hints are given by the experimenter if users cannot discover correct actions by themselves.

---

[1] LICAI is an acronym of the LInked model of Comprehension-based Action planning and Instruction taking. When LICAI is pronounced [li kai], the pronunciation represents a two-kanji Japanese word, , meaning 'comprehension.'

LICAI simulates comprehension of task instructions and hints, the generation of goals, and the use of these goals to discover correct actions by exploration. LICAI+ adds to LICAI processes that encode successful actions and retrieve them after a delay.

Goal Formation

LICAI's action planning processes contain limited capabilities to discover correct actions by exploration. These processes are controlled by *goals* generated by comprehending task instructions and hints. LICAI assumes that goal-formation is a specialized form of the normal reading process in which task specific strategies generate inferences required to guide goal formation. LICAI's goal-formation process is derived from Kintsch's (1988; in press, Chapter 10) model of word problem solving.

Kintsch's model takes as input a low-level semantic representation of problem text, the *textbase*, and processes it sentence by sentence. The result is a *problem model*. Construction of the problem model makes extensive use of comprehension schemata which elaborate the original text representation with problem domain specific inferences.

LICAI incorporates comprehension schemata that transform relevant parts of the textbase for the task instructions and hints into goals that control the action planning process. Propositions that describe actions on task objects in the textbase are recognized and further elaborated by specialized task domain schemata to generate a more complete description of a task. For example, consider a graphing task in which the user was given the instruction, Plot a variable named 'Observed' as a function of a variable named 'Serial Position.' LICAI transforms this task description into the propositional representations of two sentences. 1) Put 'Observed' on the y-axis, and 2) Put 'Serial Position' on the x-axis. The representations of the last two sentences are then transformed into *task goals* that control the action planning process. Terwilliger and Polson (1997) demonstrated that users actually perform this transformation.

In the studies described in this paper, experimenters gave hints of the form 'perform a specific action on a specified screen object' (e.g., pull-down the **Options** menu). LICAI requires that these text or verbal descriptions of an action on an object have to be transformed into a goal, a *do-it goal*, that specifies a specific object on the screen and/or legal actions on that object. Specialized comprehension schemata carry this transformation. See Kitajima and Polson (1997) for extensive descriptions of comprehension schemata.

Action Planning

The heart of LICAI is the action planning processes. LICAI assumes that successful action planning involves linking propositional representations of a goal (e.g., create a new graph), the screen object to be acted on (e.g., the **Graph** menu), and an action to be performed on that object (e.g., press and hold). The most critical of the three links is the link between the goal and the correct screen object. This link can be retrieved from memory or generated by an exploration process.

Skilled Users

Kitajima and Polson (1995) developed a version of the action planning process used by *skilled* users of an application. This model represents an arbitrary sequence of actions required to perform a task as hierarchical goal structure that is retrieved from long-term memory and used to generate the actions. A task is decomposed into a sequence of task goals. *Task goals* refer to actions (e.g., edit) on a task object (e.g., graph title). Each task goal is linked to an ordered sequence of one or more *device goals*. Each device goal specifies a unique object on the screen (e.g., the **Options** menu, the graph title) and the state of the object (e.g., highlighted) after it has been acted on. Thus, skilled users retrieve the critical links between goal and screen object from memory. However, Kitajima and Polson (1995) did not describe how such goal sequences are learned or how they are retrieved from memory.

New Users

When a *new* user of an application attempts to perform a task for the first time, Kitajima and Polson (1997) assumed that they have a task goal but not the device goals. LICAI can simulate exploration by generating the correct actions for a novel task without the device goals if the task goal can be linked to correct screen objects by LICAI's action planning processes.

A task goal is a proposition with two arguments describing a task action and a task object (e.g., hide legend). If a correct object on the screen has a label representing either one of these concepts (e.g., a menu labeled "hide"), the representation of the object will be linked to the task goal. LICAI will retrieve the correct actions (e.g., move the cursor to the object and press-and-hold) on this object from long-term memory, completing the necessary links to generate actions. We and numerous other researchers have called this linking process *the label-following strategy* (Franzke, 1994; Franzke, 1995; Kitajima & Polson, 1997; Polson & Lewis, 1990;. Rieman, Young, & Howes, 1996). Thus, the critical links can be generated to mediate successful exploration. The label-following strategy is the only method that LICAI has for learning by exploration. If there is no direct link between the task goal and the correct object, users must be given a hint.

LICAI+'s Encoding and Recall Processes

LICAI already incorporates a model of encoding and recall of goals based on the Kintsch and Welsch (1991) model of text recall. They assumed that the textbase is stored in episodic memory during the comprehension process. The strength in episodic memory of a given element of the textbase is determined by the number of cycles it stays in working memory and the activation levels it achieves during each cycle. LICAI+ generalizes this model to the encoding and recall of successful actions. LICAI+ also incorporates assumptions from the Wolfe and Kintsch

(submitted) model of story recall that enables us to compute predicted recall probabilities.

Encoding Process

LICAI+ assumes that encoding and storage of a successful action is just a special case of the comprehension process. The model "comprehends" the results of a successful action during training. A comprehension schema creates a representation of the successful action which is stored in memory during the comprehension process.

There are two forms of this encoding. The first includes the task goal, the object acted on, and results of the action if the label-following strategy can discover the correct action. The second case is defined by the failure of the label-following strategy. The experimenter gives a hint which is transformed into a do-it goal by the instruction comprehension processes. A do-it goal specifies an action on a screen object (e.g., Pull-down the **Options** menu). The do-it goal is included in the encoding of the successful action in this second case.

LICAI+'s goal formation, action planning, encoding, and retrieval processes are implemented as special cases of Kintsch's (1988; in press) construction-integration theory of text comprehension. Each process is modeled by one or more iterations of a general construction-integration cycle.

The following is a description of the encoding and recall cycles. See Kitajima and Polson (1997) for detailed descriptions of the remaining processes.

The construction phase of the encoding process generates a network of propositions that contains the following representations:

1) the task goal,
2) the do-it goal (if a hint was given),
3) the acted-on object,
4) its label (if the acted-on object is labeled),
5) salient changes in the display state caused by the action (e.g., menu dropped),
6) the display caused by the action (e.g., a pull-down menu),
7) a special encoding node that links the nodes 1, 2, 3, 4, and 5 with the strengths defined by an analyst.

In addition, the fundamental linking mechanism assumed by the construction-integration theory, the argument overlap mechanism, is applied to connect any two propositions in the network sharing arguments. Figure 1 illustrates a network generated for encoding a step of pulling down the **Legend** menu. This action caused a pull-down menu to appear with menu items, **Hide**, **Show**, **Move**, and **Arrange**.

The integration phase of the encoding process is performed using a spreading activation process. The nodes in the network can be partitioned into sources of activation, targets of activation, and links between sources and targets. In the encoding process, the representations of screen objects, the task goal, and the do-it goal serve as sources of activation. In Figure 1, these nodes are shaded. The encoding node is the target.

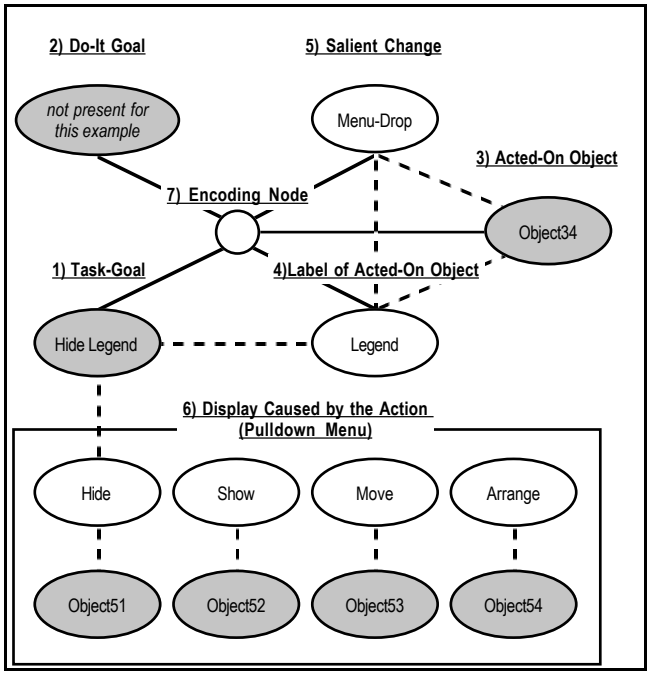

Figure 1. A diagram showing the propositional network generated by the construction subprocess in the encoding process. The dotted lines represent the argument overlap links. The solid lines connecting nodes, 1 through 5, with the encoding node, 7, are special links defining the encoding process.

The results of the integration of the network are stored in episodic memory.

At the end of training, episodic memory contains the nodes representing the textbase for the task instructions and hints, and the nodes participated in encoding processes for the correct steps. The strengths of links between these nodes are determined by the pattern of activation levels achieved in respective integration processes for text comprehension and encoding.

Recall Process

The recall process of LICAI+ assumes that users employ the task goal and the current display representation as retrieval cues. The recall process retrieves nodes in episodic memory that are linked to these cues. Nodes from episodic memory are sampled with replacement until the model retrieves an encoding of a step or retrieves a do-it goal (i.e., the action planning representation of a hint).

The predicted sampling distribution for retrieving nodes from episodic memory for a given set of retrieval cues is calculated by using a sampling probability matrix. This matrix is a fully interconnected matrix generated from the original episodic memory network. Following Wolfe and Kintsch (submitted), the sampling probability matrix is generated by two steps: 1) dividing each link strength in the episodic memory network by the maximum link strength, 2) for any two nodes linked by an indirect path, assigning the product of the strength values of the link segments in the path to their link strength.

Any nodes that are directly linked with the retrieval cues in the sampling probability matrix are retrievable. The

probability of retrieving a retrievable node in a single memory sampling trial is proportional to its relative link strengths with the retrieval cues.

Sampling is with replacement, and sampling terminates on retrieval of one of the step encodings or a do-it goal. These assumptions enable us to calculate the recall probability distribution for step encodings and do-it goals (recall targets).

Action Planning After Recall

LICAI+ attempts to act using the retrieved step encoding or the hint. If the step encoding or the hint generates the correct action, the model successfully recalls the current step. However, there are no explicit order cues in the encoding of each step, so the model can retrieve steps out of order or retrieve hints that don't apply to the current display. In this case, the retrieval process fails, and the model has to explore the interface again as on the training trial. The exploration will succeed in performing the correct action if the label-following strategy works for this step.

AN ANALYSIS OF RECALL OF OCCASIONALLY PERFORMED TASKS

The basic claim of LICAI+ is that how a step in a task is learned, by exploration or with hints, determines how that step is encoded and retrieved. Thus, we distinguish between label-following (LF) steps or tasks, and non-label-following (NLF) steps or tasks where the label-following strategy fails for lack of linking shared concepts.

Franzke (1994; 1995) and many others have shown that LF steps are rapidly discovered and "accurately" recalled. However, it is hard to distinguish between rediscovery and recall of a step after one training trial because both recall and discovery processes can have similar latency distributions.

Soto (1997), in an analysis of a large number of different graphing tasks using Cricket Graph III, showed that NLF tasks have some LF steps, usually toward the end of their action sequences. The task 'hide legend' is a good example. The first two steps (pull-down the **Options** menu, and select **Show Graph Items...**) are NLF steps. No menu label matches the task goal. The third step (clear the check box labeled by Legend) is an LF step. The last step (click OK) is a highly over-learned action that closes a dialog box and terminates the action sequence.

Rodriguez (1997) and Soto (1997) found that the first NLF step in the hide legend task is the source of the difficulties that users have with this task. Almost all users required a hint to complete the first step. Franzke (1994; 1995) found a highly significant interaction for number of hints between number of targets (screen objects) for possible actions on the screen and LF versus NLF steps. There are many targets for possible actions on the first step of any task. Thus, we would expect first steps to be especially problematic. Once users are given the hint "pull-down the **Options** menu" in the hide legend task, there are only 7 menu items on that menu.

We have used two versions of the hide legend task in the simulation described in the following sections. The first version was a simulation of performing the hide legend task using Cricket Graph III, Version 1.5.3 described above. We will refer to this as the NLF scenario. The other version of the simulated task used a hypothetical version of Cricket Graph III that added a **Legend** menu to the menu bar. The items on this menu were **Show**, **Hide**, **Move**, and **Arrange**. This version of the hide legend task requires two steps (select **Hide** from the **Legend** menu) using this hypothetical interface. We will refer to this simulation as the LF scenario. Our discussion will focus on recall of the first step for each of the two versions.

SIMULATION

A Mathematica program was developed implementing processes incorporated in LICAI+ and simulating responses from Cricket Graph III for correct actions in the hide legend task. Training was simulated by assuming that each step was performed correctly with hints given for the first NLF step. The following processes are simulated for the training: the comprehension process that generates goals and comprehends hints, storage in episodic memory during comprehension, retrieval of goals from episodic memory, and action planning, encoding of successful actions, and storage in episodic memory.

Representations of the task instructions, hints, and interface displays were coded and input to the simulation. The simulation also incorporated extensive knowledge about the basic Macintosh interface conventions for each screen object. For example, the **Options** menu item affords pull-down, and the **Options** menu item causes menu-selection, and so on. Other knowledge about actions, including moving and dragging the mouse pointer, and single- and double-clicking the mouse button, etc., was incorporated into the model.

Simulation of Training

Training on each of the scenarios for the hide legend task was simulated in several encoding conditions as described below. At the end of training, episodic memory included nodes representing the task instructions, the hint (for the NLF scenario), the acted-on object and its label for each step, and the display generated by the application. The link strengths of nodes in episodic memory are proportional to the activation level of these nodes obtained in the encoding cycle.

Encoding Bias

In encoding cycles, we manipulated the relative strengths of the links between the rest of the network and the links between the network and the task and do-it goals. The motivation for such manipulations is a fundamental property of the action planning process. The action planning process will *not* work unless the links between the current task, or do-it goal, and the rest of the network are much stronger than the rest of the links in the

network. These strong links cause a goal to dominate the integration subprocess. This subprocess selects the object to be acted on and the action to be performed on each step of the task. Manipulating relative strengths of the links between the goal and the rest of the network enables us to explore the hypothesis that the goal may dominate *both* action planning and encoding processes.

Encoding processes have been simulated under three conditions. In task goal biased encoding condition (TG), we generated a network by multiplying by a factor of 4 the strengths of links from the task goal. The strengths of the links from the do-it goal were not changed. In Figure 1, three links from the task goal (hide legend) are strengthend by a factor of 4. In do-it goal biased encoding condition (DIG), the strengths of the links from the do-it goal were multiplied by a factor of 4, and those from the task goal remained unchanged. In the neutral encoding condition (N), no multiplication factor was applied. The NLF scenario was simulated using the TG, DIG, and N conditions. The LF scenario was simulated for the TG and N conditions since hints are not required and there is no do-it goal for the LF scenario.

Simulation of Recall

The recall cues are the task instruction and the representation of task goals used in the action planning process in training trial, and the initial display for the first step. In each simulation, nodes in the episodic memory that match the representations of the cues were identified, and then the probability distribution of retrieving the recall targets were calculated. The recall targets were two encoding nodes for the LF scenario, and the do-it goal and four encoding nodes for the NLF scenario.

Recall after LF training

The probabilities of recalling the encoding of the first step for the LF scenario for TG and N bias conditions are given in Table 1. In the LF scenario, the encodings of the first and second steps are linked to the task goal. In the TG condition, the probabilities of recalling the encoding for each of the two steps was nearly equal since the task goal dominated the encoding process, reducing the influence of the application display. Thus, the model retrieved the representation of the first step a little more than 50% of the time. In the remainder, the model retrieved representation of the second step blocking the successful retrieval of the first step.

Correct performance of both steps is mediated by the same task goal, and the encodings are linked strongly to the common task goal in the TG condition. One implication of these results is that the encoding of a multi-step LF task will not reliably be retrieved by the combinations of task goal and display cues on each step. Thus, correct performance will depend on a mixture of successful recall and the label-following strategy. However, by lessening the biasing on the task goal in the N encoding condition, the display cues made a much stronger contribution to the encoding process and significantly increased the probability of correctly recalling the encoding of each step.

Table 1. Probabilities of recalling the do-it goal or the encoding of first step for the LF and NLF scenarios. TG, N, and DIG stand for task goal biased, neutral, and do-it goal biased encoding condition, respectively.

	LF Scenario		*NLF Scenario*		
	TG	*N*	*TG*	*N*	*DIG*
Probability of recalling the do-it goal	N/A	N/A	.027	.253	.618
Probability of recalling first step encoding	.551	.736	.251	.446	.177
Total	.551	.736	.278	.698	.795
Predicted Hints	N/A	N/A	.722	.302	.205

Recall after NLF training

The probabilities of recalling the encoding for the first step and the do-it goal for the NLF scenario in the TG, DIG, and N bias conditions are given in Table 1. For the NLF scenario, the row labeled *Total* gives the probability of correctly performing the first step. LICAI+ cannot perform the first step without recalling the encoding or the do-it goal. The entries for Predicted Hints are, 1–*Total*.

Manipulation in the NLF scenario of the bias has a huge impact on recall performance. In the TG biasing condition, the probability of recalling the do-it goal is small. The task goal dominates the encoding process and the do-it goal has very weak, indirect links to the task goal. The task goal does have links to all four encodings of each step. The probabilities of recalling each step encoding are almost equal, .251, .227, .180, and .315, respectively.

In the N encoding condition, both the recall probabilities for the do-it goal and the first step encoding increased compared with the TG encoding condition. The reason is the same as the LF case. The display cues become more effective in recall process. Included in these cues is the label for the **Options** menu which is directly linked to the do-it goal. Thus, the initial display is a more effective retrieval cue for both the encoding of the first step and the do-it goal.

On the other hand, in the DIG condition, all links involving the concept Option are very strong. This enhances the effectiveness of the representation of the **Options** menu as a retrieval cue and strengthens the representation of the do-it goal in episodic memory, making it easier to retrieve.

COMPARISONS WITH USER PERFORMANCE

Franzke (1994) and Soto (1997) have done studies relevant to evaluating LICAI+'s recall predictions. For NLF steps, the model predicts that users will require a hint to successfully perform the step if they fail to recall the correct step encoding or hint. We used the best available measure of recall, proportion of subject

Table 2. Proportion of times at least one hint was required for steps categorized by link type, training (exploration) and recall trial (short or long delay). From Franzke (1994).

Link Type	*Training*	*Short Delay*	*Long Delay*
Exact Match	.07	.00	.14
Synonym	.08	.02	.18
Inference	.42	.07	.29
No Link	.88	.05	.60

Table 3. Observed proportions of tasks requiring at least one hint as a function of task type and training and delay. From Soto (1997).

	Session 1		*Session 2*	
Task Type	*Training*	*Short Delay*	*Long Delay*	*Short Delay*
LF/C	.01	.00	.00	.00
LF/U	.19	N/A	.12	N/A
PL/C	.84	.26	.46	.11
PL/U	.58	N/A	.29	N/A

requiring a hint on a task or step. However, this variable does not provide an unambiguous measure for evaluating the recall predictions for LF steps and tasks. Both successful recall and the label-following strategy can generate correct actions within 10 seconds.

For LF steps and tasks, LICAI+ predicts that no hints should be required during training or on recall trials. However, Rieman (1996) and Rieman, Young, and Howes (1996) found that users will explore an interface before taking the initial correct action predicted by the label-following strategy. This initial exploratory behavior can lead to long latencies and hints on LF steps that are outside the scope of LICAI+.

Description of Available Experimental Data

We first present experimental data from Franzke (1994) and Soto (1997) focusing on the proportion of hints required on training and recall trials.

Description of Franzke (1994)

Franzke (1994) had four groups of 20 participants create a graph and then perform 9 editing tasks on the graph using one of four graphics applications, Cricket Graph I or III, or one of two versions of EXCEL. During training, participants did the task by exploration, receiving hints when necessary. Half the participants in each group were tested for retention after a 5 minute delay (short delay), and the remainder were tested after a 7 day delay (long delay).

Franzke classified each step in each task into one of four categories according to the relationship between the task goal for each step given in her instructions and the label of the object to be acted on for that step. Her exact match and synonym categories are examples of LF steps. In her third category an inference is required to link the correct object and the task goal. In the fourth category (no link) there is no meaningful link between the screen object and task goal. The latter two categories are both examples of NLF steps.

The results relevant to LICAI+ from Franzke's (1994) experiment are shown in Table 2. The table shows the proportion of times that at least one hint was required on a step, with the steps categorized by link type, training (exploration) and recall trial (short or long delay).

Description of Soto (1997)

Soto (1997) performed a study replicating and extending Franzke's results. Soto's 19 participants were trained on a series of 33 graph editing tasks using Cricket Graph III and were tested for retention after a 2 or a 7 day delay. All participants were experienced Macintosh users who had not used a graphing application. Editing tasks were carried out on three types of graphs: histograms, pie charts, and bar charts. The 11 histogram editing tasks and the first of the 11 bar and pie chart editing tasks were used as warm-up tasks, and these data are not included in the results described below.

Four out of the 10 experimental pie and bar chart editing tasks were unique (U) to that graph type and occurred once during training and testing. An example is "stand out a pie slice." Six of the tasks were common (C) to both graph types and occurred twice during training and recall sessions. An example is 'hide legend.' The delay between the two presentations of the common tasks averaged about 7 minutes. In Soto's data analysis, the second occurrence of a common task was treated as a recall trial with a short delay. His participants had no trouble recognizing the second occurrence even with a change in graph type.

Soto classified his editing tasks into three categories. Label-following (LF) tasks required acting on objects whose labels were semantically related to the goal. Thus, all steps in these tasks were equivalent to Franzke's direct match and synonym step types. Direct-manipulation (DM) tasks required acting on the task object (e.g. pie slice) mentioned in the task goal. These data are not discussed as it is beyond the scope of this version of LICAI+. Poorly-labeled (PL) tasks did not support either label-following or direct-manipulation violating the label-following strategy. Occasionally, a task supported label following as well as direct manipulation (e.g., 'Change the graph title to "Year of Production"'). For this reason, the tasks were classified based on the method used by the subject, rather than on a priori criteria.

Soto's analysis is by task rather than by the step level. The typical PL task has one or two initial NLF steps. Soto's findings and Franzke's (1994) results suggest that the initial NLF step has the largest impact on users' performance. Previously, we summarized Franzke's result showing that there is an interaction for the number of hints needed between LF versus NLF and the number of possible targets for action on a screen. The difficulty of

NLF steps increases dramatically as a function of the number of targets.

Comparison With LICAI+'s Predictions

Training Performance

LICAI+ predicts perfect performance for both training and recall trials at all delays for LF steps. If we use the proportion of users requiring hints as our measure, a large majority of Franzke's (1994) results (shown in Table 2) and Soto's (1997) findings (shown in Table 3) support this prediction. The largest deviation that we know of is in the data from LF/U, Soto's condition where 19% of the participants required hints on the training trial.

The model makes equally strong training performance predictions for tasks and steps that do not support the label-following strategy (NLF tasks). LICAI+ predicts that these tasks and steps cannot be learned by exploration without hints or information looked up in a manual or help system. However, this prediction for NLF tasks is not sound. The observed proportions of tasks or steps requiring at least one hint ranges from less than .5 to .9 in different conditions of the Franzke and the Soto data.

However, the pattern of deviations in both the Franzke and the Soto data is instructive and supports the claim that the LF-NLF distinction is a useful design heuristic. LICAI+ makes incorrect predictions for learning by exploration in NLF tasks because of the model's simple exploration process. First, the model cannot perform exploratory activities like pulling down a menu to see if any items on that menu link to the tasks goal. Experienced Macintosh users carefully explore menus (Rieman, 1996) and act upon matching labels uncovered during such explorations.

Second, users seem to be able to use elimination strategies when dealing with a small number of screen objects like the items on a menu. For example, when participants are given the hint to pull down the **Options** menu in the hide legend task, they correctly select **Show Graph Items...** by a process of elimination. The other items on this menu are more specific and clearly have nothing to do with the hide legend task. LICAI+ can perform this step if it is given the knowledge that 'show is the opposite of hide' and that 'the legend is a graph item.'

The above arguments suggest that an interesting test of the model would be to consider NLF tasks in which the first two steps violate the label-following strategy. 'Hide legend' is such a task. Rodriguez (1997) shows that 100% of his subjects required hints to be able to perform this task. Franzke (1994) found that approximately 90% of the participants required hints for steps where there was no link between the task goal and the correct object's label.

Recall at Short Delays for NLF Tasks

LICAI+ predicts that successful performance on recall trials is possible only when users retrieve a hint or an encoding of a step from episodic memory. However, the model does not make predictions about the effects of delay. We have assumed that LICAI+'s recall predictions apply to delays of one or more days.

Franzke's (1994) and Soto's (1997) results show that immediate recall of NLF steps is quite good. Franzke (1994) found that about 90% of NLF steps can be recalled after a 5 minute delay (see Table 2). About 75% of Soto's PL tasks were performed correctly, without a hint, after a short delay (See Table 3).

Recall at Long Delays for NLF Tasks and Steps

LICAI+ predicts that successful recall performance can vary from .722, to .205 as a function of the encoding bias for NLF tasks and steps. Franzke's and Soto's results at long delays are hard to interpret because of the results from training trials for NLF tasks. Users' learning by exploration is better than that predicted by LICAI+. Thus, contrary to the predictions of the model, users will be able to discover the correct action on a recall trial even if they fail to recall a hint or encoding of the step.

We reanalyzed both Franzke's no link and inference steps at the long delay shown in Table 2 and Soto's recall data from his PL conditions shown in Table 3 at the long delay. We made the assumption that the probability of requiring hints on recall trials, $P_{require\_hint}$, is just the probability of failing to recall a hint or step encoding, P_{fail_recall} , times the probability of failing to discover the correct action by exploration, $P_{fail\_exploration}$, assuming that the two events are independent. If we assume that $P_{fail\_exploration}$ estimated by the probability of requiring hints on the training trial, P_{fail_recall} can be estimated by $P_{fail\_recall} = P_{require\_hint}/P_{fail\_exploration}$.

The estimated values of P_{fail_recall} for Franzke's no link steps is .68, and .69 for the inference steps. These values are close to the predicted value for the TG condition shown in Table 1.

The estimated values of P_{fail_recall} for Soto's poorly labeled tasks at a long delay is .50 for the unique tasks and .55 for the common tasks. These results suggest that the task goal has a strong influence on the encoding process but that it is not as strong as the 4:1 bias assumed in computing the predictions for the TG conditions shown in Table 1.

CONCLUSIONS AND IMPLICATIONS FOR PRACTICE

We have asserted that most users are occasional users of many applications, and they routinely use only a small fraction of the functionality of their frequently used applications. A model of routine cognitive skill is not a good description of users' actual patterns of use. The action sequences for occasionally performed tasks are generated by a mixture of recall of previous episodes of use and of problem solving processes that attempt to reconstruct missing action knowledge. Performance of these tasks is more like the reconstructive processes involved in recalling a story rather than the execution of a rule-based representation of a routine cognitive skill.

LICAI+ is a model of occasional users. This model suggests the partitioning of all steps executed in

performing a task into two categories: steps that support the label-following strategy and those that do not. Steps and tasks that support the label-following strategy can be performed by exploration. We know that users have strong preferences for learning by exploration (Carroll, 1990; Rieman, 1996), which the label-following strategy supports.

Experienced users can make effective use of manuals (Rieman, 1996) to perform tasks that are not supported by the label-following strategy. However, users will have continued trouble with steps not supported by label following (NLF steps). These steps once correctly performed with the assistance of hints are difficult to remember over long delays (2 or more days). We estimate that the probability of recall failure is at least .5.

The data from the short delay recall conditions also suggests a possible limitation of empirical usability tests. Test users will have trouble with the initial versions of common tasks that don't support the label-following strategy. Second and third versions of these tasks that are given to test-takers later in a session will be performed correctly, and evaluators may incorrectly infer that there are no problems with the interface for these later versions.

In summary, the theoretical and empirical results presented in this paper and in numerous other studies demonstrate the wide applicability of the label-following strategy. It supports rapid learning of all kinds of applications, not just walk-up-and-use applications like automated teller machines. We have shown in this paper that label following is also a major contributor to the usability of occasionally performed tasks.

REFERENCES

Bewley, W.L., Roberts, T.L., Schroit, D., & Verplank, W.L. (1983). *Human Factors Testing in the Design of Xerox's 8010 'Star' Office Workstation: Case Study D: The Star, the Lisa, and the Macintosh.*

Carroll, J.M. (1990). *The Nuremberg funnel: Designing minimalist instruction for practical computer skills*. Cambridge, MA: MIT Press.

Franzke, M. (1994). *Exploration, acquisition, and retention of skill with display-based systems*. Unpublished Dissertation, University of Colorado, Boulder, Department of Psychology.

Franzke, M. (1995). Turning research into practice: Characteristics of display-based interaction. *Proceedings of human factors in computing systems CHI '95* (pp. 421–428). New York: ACM.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction-integration model. *Psychological Review*, **95**, 163–182.

Kintsch, W. (in press). *Comprehension: A paradigm for cognition.* Cambridge University Press.

Kintsch, W., & Welsch, D.M. (1991). The construction-integration model: A framework for studying memory for text. In W. E. Hockley and S. Lewan-dowsky (Eds.), *Relating theory and data: Essays on human memory* (pp. 367–385). Hillsdale, NJ: Erlbaum.

Kitajima, M., & Polson, P.G. (1995). A comprehension-based model of correct performance and errors in skilled, display-based human-computer interaction. *International Journal of Human-Computer Studies*, **43**, 65–99.

Kitajima, M., & Polson, P.G. (1996). A comprehension-based model of exploration. *Proceedings of human factors in computing systems CHI '96* (pp. 324–331). New York: ACM.

Kitajima, M., & Polson, P.G. (1997). A comprehension-based model of exploration. *Human-Computer Interaction*, **12**, 345–389.

Polson, P.G., & Lewis, C. (1990). Theory-based design for easily learned interfaces. *Human-Computer Interaction*, **5**, 191–220.

Rickard, T.C. (1997). Bending the power law: A CMPL theory of strategy shifts and the automatization of cognitive skills. *Journal of Experimental Psychology: General*, **126**, 288–311.

Rieman, J. (1996). A field study of exploratory learning strategies. *ACM Transactions on Computer-Human Interaction*, **3**, 189–218.

Rieman, J., Young, R.M., & Howes, A. (1996). A dual space model of iteratively deepening exploratory learning. *International Journal of Human-Computer Studies*, **44**, 743–775.

Rodriguez, M. (1997). A detailed analysis of exploratory behavior of new users of a graphics application. Institute of Cognitive Science Technical Report. University of Colorado, Boulder.

Ross, B. (1984). Remindings and their effects in learning a cognitive skill. *Cognitive Psychology*, **16**, 371–416.

Santhanam, R., & Wiedenbeck, S. (1993). Neither novice nor expert: the discretionary user of software. *International Journal of Man-Machine Studies*, **38**, 201–229.

Smith, D.C., Irby, C., Kimball, R., Verplank, W.L., & Harslem, E. (1982). *Designing the Star User Interface: Case Study D: The Star, the Lisa, and the Macintosh*.

Soto, R. (1997). Properties of Tasks that Determine Success in Learning By Exploration and Recall. Unpublished Master Theses.

Terwilliger, R.B., & Polson, P.G. (1997). Relationships between users' and interfaces' task representations. Proceedings of human factors in computing systems CHI '97. New York: ACM.

Wolfe, M., & Kintsch, W. (submitted). An overview of the construction-integration model.