LICAI/BT: A Comprehension-Based Model of Item Selection with Backtracking Capability

Muneo Kitajima and Peter G. Polson

***Abstract*—When computers are used to execute tasks, it is often necessary for the user to locate a target item in a menu or a list. For example, users of office applications select appropriate commands from a hierarchical menu to display dialog boxes and edit file or table attributes. To locate the desired information on the Web, users select the most appropriate candidate out of those presented by a search engine, and proceed through a series of hyperlinks that appear to be related to the task. However, it is commonly observed that the user does not always select the item on its first appearance; the user dismisses the item on the first visit and backtracks, and eventually selects it on a later visit. This paper describes a comprehension-based model, LICAI/BT, which is capable of simulating user's item selection including backtracking and selection-on-revisit.**

***Index Terms* — Search, Cognitive model, Backtrack, Comprehension**

I. Introduction

SELECTING an item from the interface display is one of fundamental actions necessary for performing tasks with modern graphical user interfaces. For example, the menu-based interaction style requires users to make a series of selections to get access to a dialogue box where controls for performing the task are provided. Similarly, the Web's primary control for navigation is selecting links to the next page. Users navigate to pages containing desired information by making a series of link selections.

We have developed a series of models that simulate users' action selection processes on graphical user interfaces [10][8][7]. They are applied to develop a usability inspection method called Cognitive Walkthrough for the Web [1]. These models are based on Kintsch's *construction-integration cognitive architecture* [6], which was originally developed as a model of text comprehension and has been applied to model broader human activities including action planning in HCI [11][7].

These models have successfully accounted for users' behavior such as *the label-following strategy* [14] at a coarse level (the LICAI model [8]) and errors made by skilled users [7]. However, due to the limitation of the adopted search strategy, i.e., *pure forward "best" first search*, they are not able to backtrack. Real users will redirect their search if a series of menu selections leads to a dialog that obviously has nothing to do with their task or if they reach a page that contains no relevant information or links. Rieman [16] observed how users actually implement the label-following strategy in detail, reporting that *users do not always select the correct item from a menu first time, they continue their search examining the other possibilities, but they eventually revisit the menu to select the correct item.* He found that a majority of the subjects, 90%, successfully located the correct menu, the Graph menu in Cricket Graph, eventually, when they performed the task "create a graph." However, 85% of these successful subjects did not move the cursor directly to the correct menu item and pulled it down. A half of them found the correct menu after moving the mouse cursor over other parts of the screen. The other half pulled down and examined the correct menu at some point, but then moved on to examine other menus before returning to the correct menu.

The purpose of this paper is to describe a mechanism that enables comprehension-based models to backtrack, and thus to simulate selection-on-revisit behavior. We add this mechanism to the LICAI model [8] to construct the LICAI/BT model. We show LICAI/BT successfully simulates the label following strategy at a fine level as observed by Rieman [16]. Although backtracking is trivial for the other cognitive architectures, such as Soar [12] or ACT-R [1], it is not for the comprehension-based construction-integration architecture. Soar creates a new problem space on encountering an impasse. ACT-R represents backtrack as production rules.

A. Pure forward "best" first search modeled by the construction-integration architecture

The construction-integration architecture assumes that comprehension is done by two phases; the *construction* phase that activates knowledge in long-term memory relevant to the currently processed sentence and connects the activated knowledge with the current sentence to form a network, and the *integration* phase that derives contextually appropriate meaning of the sentence as an activation pattern of the constructed network by spreading activation from the part of the network representing the reading context.

Within the construction-integration architecture, action planning is modeled as attending to an object (e.g., menu title, menu item, button, hyperlink, etc.) and acting on the object (e.g. press and hold, single clicking) in a way consistent with

Muneo Kitajima is with the National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Ibaraki, 305-8566 Japan (phone: +81-298-61-6650; fax: +81-298-61-6731; e-mail: kitajima@ni.aist.go.jp).

Peter G. Polson is with the Institute of Cognitive Science, University of Colorado at Boulder, CO 80309-0345 USA (e-mail: ppolson@psych.colorado.edu).

the current context. The model acts on the object that it *perceives* to be closest to its current goal. *Distance* is computed by a spreading activation process that takes into account both the number of links between the goal and a screen object, and the strength of those links.

At each step of the action planning process, the model will select the *closest* object even when the available targets for action are objectively very distant from the goal. There is no notion of threshold in the model that will enable it to reject all the current alternatives and try something else. The model does pure forward best first search by selecting the *closest* object during each step of the task.

II. Comprehension-Based Model of Action Planning with Backtracking Capability

In this section, we start by reviewing a comprehension based model of exploration, LICAI, an acronym for the LInked model of Comprehension-based Action planning and Instruction taking [8], which simulates comprehension of task instructions and hints, the generation of goals, and the use of these goals to discover correct actions by exploration. We show how LICAI does pure forward best first search. We then introduce a mechanism that enables us to account for label-following at a fine level with such activities as backtracking or rejecting and then revisiting and selecting the correct menu item.

A. *Outline of LICAI [8]*

This section outlines the LICAI model, taken verbatim from [9].

1) 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 [5][6] 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 [16] demonstrated that users actually perform this transformation.

2) 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 [7] 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 [7] 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 [8] 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* [3][4][8][13][16]. 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.

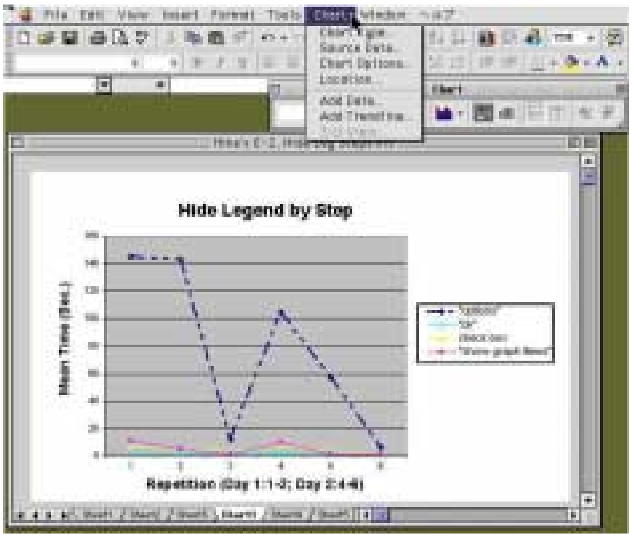

Fig. 1. Excel menu bar and Chart pulldown menu.

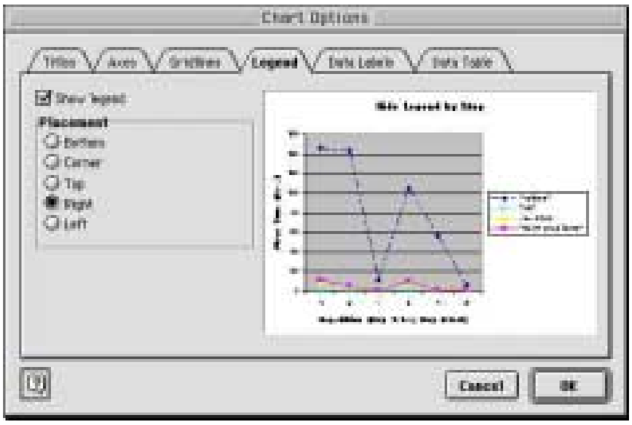

Fig. 2. Chart Options dialog box.

B. Example task and anticipated users' behavior

Let's consider a situation where a user wants to hide the legend of a chart created by Excel. The correct sequence of actions is as follows:

Interacting with menu to open the dialog box for the task;

A_1: Move the mouse cursor to the "Chart" menu-bar item and hold the mouse button down (Figure 1).

A_2: Move the mouse cursor to the "Chart Options …" pulldown menu item and release the mouse button. (The Chart Options dialog box appears.)

Interacting with dialog box to performing the task;

A_3: Move the mouse cursor to the "Legend" tab in the Chart Options dialog box and click (Figure 2).

A_4: Move the mouse cursor to the "Show legend" check box and click.

A_5: Move the mouse cursor to the "OK" button and click.

For the initial part of the task, we anticipate what the user would perform by extrapolating what Rieman [16] had observed when he had his subjects perform "create a graph" task with a graphing application, Cricket Graph: users would take the following sequence of actions; pull down the Chart menu, and then look at all of the items, dismiss the menu, and search other menus, and eventually return to the Chart menu and select Chart Options …. As mentioned previously, the label following strategy is the linking process of the goal and the correct object and action. In this situation, the linking process had completed eventually but it was not obvious for the first time when the users first pulled down the Chart menu item.

For the second part of the task, users would not have any difficulty performing the task since the labels of the correct objects for the actions, "Legend" and "Show Legend," respectively, partially matches the task description, "hide legend." Thus users would easily establish the links between the goal and the correct objects, and retrieve the correct action, move the mouse cursor to the "Legend" tab and single-click, and move the mouse cursor to the "Show legend" checkbox and single-click. For the latter action, the knowledge, "HIDE is an antonym of SHOW," would be retrieved from long-term memory and reinforce the link between the goal and the correct object.

C. LICAI's simulation

Before introducing a backtracking and selection-on-revisit mechanism, we illustrate how LICAI simulates the menu selection steps in the example task. We ran a computer program that implements the LICAI model (see [7][8] for detail) to simulate the pulldown menu selection process as shown by Figure 1. The program constructed a network including the goal, screen objects, and the nodes representing the action of attending-to respective screen objects, e.g., attend-to "Chart Options pulldown menu item," and integrated the network by means of spreading activation process. The most highly activated "attend-to" node was selected as the next action to take, e.g., attend-to Chart Options.

Selecting menu item with <u>*poor*</u> *label:* When the pulldown menu in Figure 1 were input in the simulation program along with the goal, it activated the correct attend-to node associated with Chart Options pulldown menu item highest. However, its activation value, 0.216, was slightly higher than that of the similar menu, 0.201 for Chart Type, and that of the irrelevant labels, e.g., 0.176 for Add Data. Thus LICAI barely selected the correct pulldown menu item for the first time it opened the Chart pulldown menu.

It should be noted that in this simulation we assumed that critical pieces of knowledge for connecting the concept, "legend" and "chart options," were retrieved from long-term memory and helped establish links between the goal and the correct object. Otherwise, the correct menu would have been regarded as irrelevant and LICAI would have selected a wrong menu. In fact, in a simulation run without the bridging knowledge, LICAI selected Add Data pulldown menu item.

In summary, LICAI selected the correct pulldown menu item for the first time it encountered the menu item, which is not consistent with what real users would perform for the task.

Selecting menu item with *__good__* *label:* It is suggestive if we simulate a perfect matching case and compare the result with that of the poor matching case just described. In order to simulate a perfect matching case, we added an ideal but *hypothetical* correct object to the Chart pulldown menu, which had the label "Hide Legend." A simulation run showed that the activation value of the hypothetical object was 0.306, which was extremely greater than that of the real correct object and those of the rest of irrelevant objects; the activation value of Chart Options was 0.144, and that of the irrelevant object was e.g., 0.121 for the Add Data menu item. In this case LICAI selected the hypothetical correct object, which would be consistent with real users' behavior.

Summary: LICAI could perform the task successfully, not only in the good matching case but also in the poor matching case, by selecting the object closest to the goal using the label-following strategy. However, this is *not* a correct description of users' behavior as we mentioned previously. LICAI's implementation of the label-following strategy is fine for the good matching case but not for the poor matching case.

D. LICAI/BT: LICAI with backtracking capability

This section introduces a set of knowledge that enables LICAI to backtrack.

1) Hypothetical device goal

As we have seen in the previous simulation, there was a huge difference between the patterns of activation values for the attend-to nodes depending on the quality of labels associated with the correct object. In the poor label case, the superiority of the activation value of the correct object was very small, whereas in the perfect matching case, it was tremendously huge. We assume that the user starts exploration of the interface for an object with a perfect matching label. However, it is not sure to be present but is hypothetical.

The target object can be regarded as a kind of device goal as described in the skilled user model [7]. It is associated with a hypothetical screen object, <Object-X>, having a label that matches the current task goal. We assume that the user performs action selection processes with the guidance of a task goal and a hypothetical device goal retrieved from long-term memory, which is defined by:

> *Hypothetical Device Goal (HDG):* Expects to see a hypothetical screen object <Object-X> that has the labels identical with the representation of the current task goal. The <Object-X> should be found on the screen with varying degree of strengths of belief – we denote it as W_b.

2) Backtracking skill

The hypothetical screen object can be attended-to as real objects. However, we assume that the hypothetical object is associated with skills, i.e., procedural knowledge, necessary to perform backtracking, e.g., close a dialog box, close a pulldown menu and select another menu, and so on. Thus, if it is attended-to, the user will perform one of these actions suitable for the particular situation.

3) Belief strength

We assume that the degree of belief of finding the hypothetical object should vary depending on the situation of interaction. For example, when selecting an item from menu-bar, it is unlikely to find a good matching label, thus the belief should be weak. On the other hand, when selecting a pulldown menu to issue a command, it is anticipated to find a good matching label. The stronger the belief, the more likely the hypothetical object is attended-to and the model backtracks. Thus, the model's behavior should vary according to the assignment of belief values.

4) Strategic control of belief strength

It is necessary to have knowledge to assign a belief value to each action planning step. We assume that the strengths of belief should be controlled strategically. For example, when a user is searching for a command in a two-layered hierarchical menu, he/she would anticipate that the degree of consistency of the representations of the objects with the task goal should become larger as the deeper the objects lay in the menu hierarchy. Thus, a strategy that might be effective for discovering the correct object for the command in this circumstance without issuing wrong commands would be stated as follows:

> *Strategy for menu item selection:* Start with a small W_b for menu-bar item selection, followed by a large W_b for pulldown menu item selection. If backtracking occurs, relax the belief strength and perform pulldown menu item selection with a moderate W_b.

E. Simulation of the example task

This section shows LICAI/BT's simulation of the pulldown menu selection process. The following is assumed as the strategy for menu item selection:

1. Select menu-bar item, e.g., Chart, with $W_b = 0$. Since the menu-bar item cannot be the target object, there should be no belief of finding the target item at the men-bar level.
2. Select pulldown-menu item with $W_b = 4$, representing strong belief of finding target object at that level.
3. If there is a real target object that has almost perfect matching with the current task goal, the model would select it. Otherwise, the model would select <Object-X>. This is associated with the action of moving the mouse pointer to the next menu-bar item, e.g., Window, and a new pulldown menu is opened. The model continues on with the same belief strength.
4. When all the menu-bar items are examined and no real object is selected, the model relaxes the strength of the belief, $W_b = 2$, and repeats the same procedure.

Figure 3 illustrates the network constructed for the

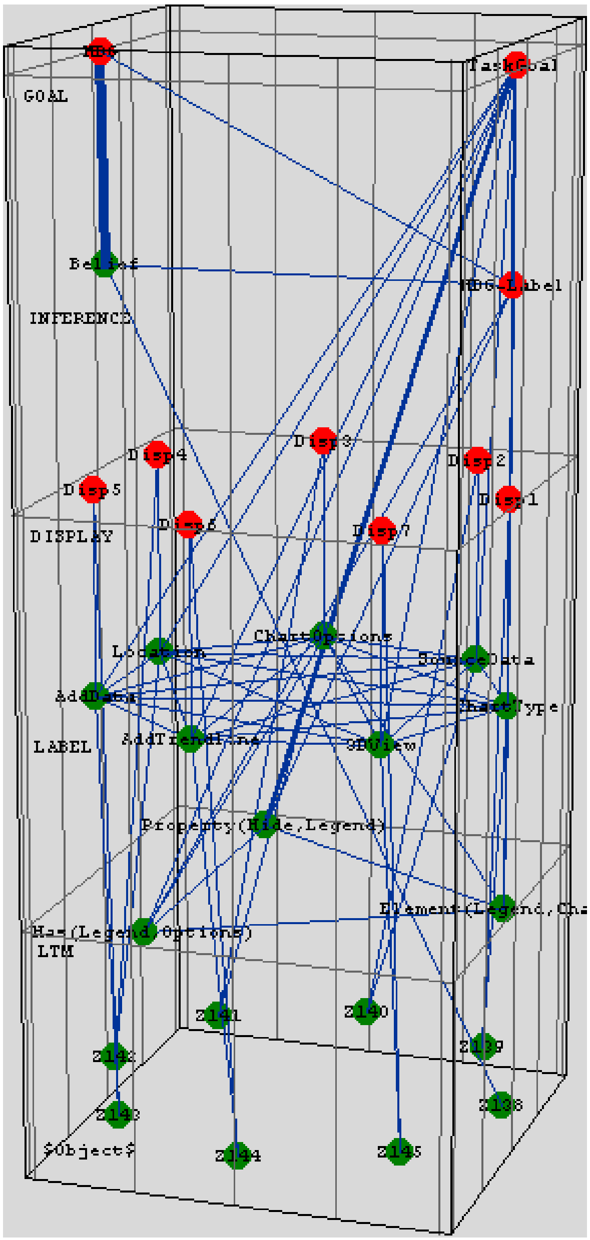

Figure 3. The constructed network for the hide legend task with the belief strength of 4.

simulation. The strength of belief was set to 4. At the top layer, the task goal and the hypothetical device goal, HDG: (*expect-to-see* DISPLAY-OBJECT), are represented. The hypothetical device goal is linked to two nodes at the second layer that represent associated features of the hypothetical device goal, HDG_Label: (*is-labeled-by* DISPLAY-OBJECT HIDE LEGEND) and Belief: (*believe-to-find-display-object-as* <Object-X>), respectively. HDG and HDG_Label are connected with Belief by links with strength W_b. The nodes at the third layer represent the seven pulldown menu items, and each is linked to a node that represents its label at the fourth layer. The label nodes are connected each other and the task goal considering the degree of semantic similarity between them. The nodes at the fifth layer represent knowledge retrieved from long-term memory for elaborating the concepts related to the task, i.e., Legend, Options, and Chart. The nodes that represents "attend-to a real object" or "attend-to a hypothetical object" are located at the bottom layer. Each attend-to node is labeled with an identification number Znnn provided by the simulation program. In the figure, Z138 represents "attend-to <Object-X>," and Z141, the correct action, "attend-to Chart Options." The thickness of line is proportional to the link strength. The nodes with red color serve as activation sources.

Figure 4 shows the activation value of each attend-to node finally obtained as a function of the strength of belief, ranged from 1 to 4. The rightmost plot corresponds to a case where the pulldown menu item that exactly matching the representation of the task goal be included in the pulldown menu. As can be seen in the figure, the perfectly matching "Hide Legend" pulldown menu would have been selected, if it should have existed, even when a strong belief value, W_b=4, had been used.

In Figure 4, the open circles and the filled circles denote the activation values of the nodes representing "attend-to Chart Options" and "attend-to <Object-X>," respectively. As can be read from the figure, when the model first examined the pulldown menu items with a strong belief value, W_b=4, "attend-to <Object-X>" activated highest, meaning that the pulldown menu would be dismissed. Though we do not show in this paper the results for the other pulldown menus, we had the same results.

After scanning all the pulldown menus, LICAI/BT relaxed the strength of belief to W_b=2, and starting with the Chart pulldown menu, it selected "attend-to Chart Options."

The full trace of the simulation including backtracking and selection-on-revisit acvtivities is summarized as follows:

1. Attended-to Chart menu-bar item with W_b=0, moved the mouse cursor to it, and single-clicked the left mouse button;
2. Attended-to <Object-X> with W_b=4, single-clicked the left mouse button to close the pulldown menu;
3. Repeated the same sequence for the rest of the menu bar items;
4. Attended-to Chart menu-bar item again with W_b=0, moved the mouse cursor to it, and single-clicked the left mouse button;
5. Attended-to Chart Options pulldown menu item with a reduced belief value, W_b=2, moved the mouse pointer to the attended-to object, and single-clicked the mouse button;

III. CONCLUSION

We presented in this paper that LICAI/BT can simulate the label following strategy as real users implement it in real situations by incorporating a hypothetical device goal and a set of strategic knowledge to utilize it into LICAI [8]. LICAI has only coarsely simulated the label-following strategy by adopting pure forward best first search strategy. The hypothetical device goal and its implementation knowledge are crucial to implement the label following strategy appropriately in various situations.

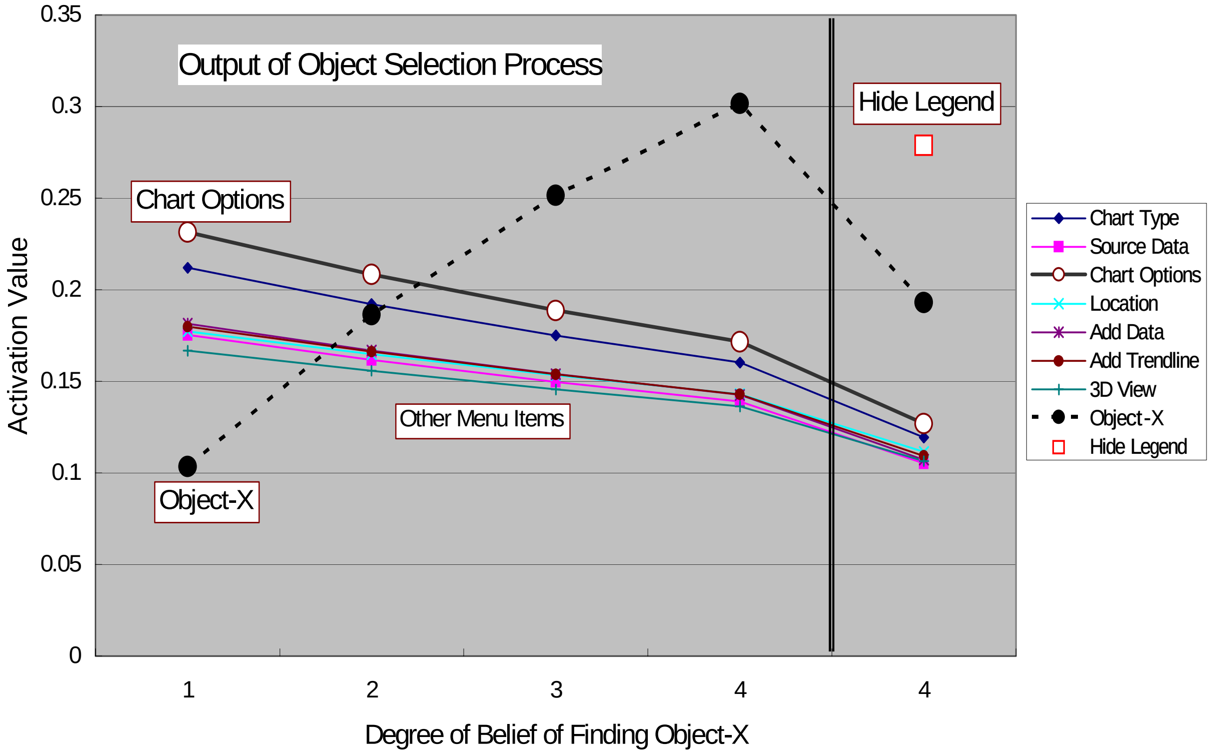

Figure 4. Simulation results.

Users' behavior is guided by task goals and *hypothetical* device goals for new situations as simulated by LICAI/BT; whereas it is guided by task goals and *real* device goals for skilled performance as modeled by [7]. We see a beautiful parallel in these models in terms of the knowledge to be used to guide performing by exploration and skilled performance, which is unique to comprehension-based models.

References

[1] J. R. Anderson and C. Libiere. The Atomic Components of Thought. Lawrence Erlbaum Associates, 1998.

[2] M. H. Blackmon, P. G. Polson, M. Kitajima, and C. Lewis, "Cognitive Walkthrough for the Web," in *Proceedings of CHI2002 conference*, pp. 463-470, ACM Press, New York, 2002.

[3] M. Franzke, Exploration, acquisition, and retention of skill with display-based systems, Unpublished Dissertation, University of Colorado, Boulder, Department of Psychology, 1994.

[4] M. Franzke, "Turning research into practice: Characteristics of display-based interaction," In *Proceedings of CHI'95 conference*, pp. 421–428, New York: ACM Press, New York, 1995.

[5] W. Kintsch, "The role of knowledge in discourse comprehension: A construction-integration model," Psychological Review, 95, 163–182. 1988.

[6] W. Kintsch, Comprehension: A Paradigm for Cognition, Chapter 10, Cambridge University Press, 1998.

[7] M. Kitajima and P. G. Polson, "A comprehension-based model of correct performance and errors in skilled, display-based human-computer interaction," *International Journal of Human-Computer Studies*, Vol. 43, pp. 65-99, 1995.

[8] M. Kitajima and P. G. Polson, "A comprehension-based model of exploration," *Human-Computer Interaction*, vol. 12, No. 4, pp. 345-389, 1997.

[9] M. Kitajima, R. Soto, and P. G. Polson, "LICAI+: A comprehension-based model of the recall of action sequences," 2nd European Conference on Cognitive Modelling, pp. 82-89, 1998.

[10] M. Kitajima, M. H. Blackmon, and P. G. Polson, "A Comprehension-based Model of Web Navigation and Its Application to Web Usability Analysis," in *People and Computers XIV - Usability or Else! (Proceedings of HCI 2000)*, S. McDonald, Y. Waern & G. Cockton (eds.), Springer, pp.357-373, 2000.

[11] S. M. Mannes and W. Kintsch, "Routine computing tasks: Planning as understanding," *Cognitive Science*, Vol. 15, pp. 305–342, 1991.

[12] A. Newell, Unified theories of cognition. Cambridge, MA: Harvard University Press, 1990.

[13] P. G. Polson and C. Lewis, "Theory-based design for easily learned interfaces," *Human-Computer Interaction*, 5, pp. 191–220, 1990.

[14] P. G. Polson, C. Lewis, J. Rieman, and C. Wharton, Cognitive walkthroughs: A method for theory-based evaluation of user interfaces, *International Journal of Man-Machine Studies*, Vol. 36, pp. 741-773, 1992.

[15] J. Rieman, "Learning Strategies and Exploratory Behavior of Interactive Computer Users," Ph. D. dissertation, Technical Report CU-CS-723-94, Department of Computer Science, University of Colorado, 1994

[16] J. Rieman, "A field study of exploratory learning strategies," *ACM Transactions on Computer-Human Interaction*, 3, pp. 189–218, 1996.

[17] J. Rieman, R. M. Young, and A. Howes, "A dual space model of iteratively deepening exploratory learning," *International Journal of Human-Computer Studies*, Vol. 44, pp. 743–775, 1996.

[18] R. B. Terwilliger and P. G. Polson, "Relationships between users' and interfaces' task representations," in *Proceedings of CHI'97 conference*, pp. 99-106, ACM Press, New York, 1997.