

Exploring Information Navigation Strategies with a Computational Model

September 8, 2000

Craig S. Miller

School of Computer Science
DePaul University
243 S. Wabash Ave.
Chicago, IL 60604 USA
+1 312 362 5085
cmiller@cs.depaul.edu

Roger W. Remington

NASA Ames Research Center
Moffett Field, CA 94035
+1 650 604 6243
rremington@mail.arc.nasa.gov

ABSTRACT

A computational model of a user navigating Web pages was used to identify factors that affect Web site usability. The model approximates a typical user searching for specified target information in architectures of varying menu depth. Search strategies, link ambiguity, and memory capacity were varied and model predictions compared to human user data. The best fits to observed data were obtained when the model assumed that users 1) used little memory capacity; 2) selected a link whenever its perceived likelihood of success exceeded a threshold; and, 3) opportunistically searched below threshold links on selected pages prior to returning to the parent page.

Keywords

Information architecture, navigation strategies, computational model

INTRODUCTION

The World Wide Web continues to revolutionize how people obtain information, buy products, and conduct business transactions. Yet many companies and organizations struggle to design Web sites that customers can easily navigate to find information or products. The identification of factors that affect the usability of the World Wide Web has become increasingly important. While many of these factors concern the graphical layout of each page in a Web site, the way in which the pages link to each other, often called the site's information architecture, plays a decisive role in the site's usability, especially for sites allowing access to large databases [10]. Our effort focuses on understanding how a site's information architecture impacts a user's ability to effectively find content in a linked information structure such as a Web site.

We develop our understanding through the construction and testing of a working computational model that simulates a user navigating through a site making choices about whether to select a given link or evaluate an alternate link on the same page. Constructing and testing a working

model not only complements empirical studies, but also offers advantages over empirical usability testing. Empirical studies are generally too expensive and time consuming to address the wide range of content, configurations, and user strategies that characterize the Web. In contrast, an implemented model can run thousands of simulated sessions in minutes. Also, empirical studies do not inherently provide explanations for their results and thus make it more difficult to determine how a given result generalizes to other circumstances, whereas a cognitive model can describe the underlying processes that produce behavior. For example, computational models have been used to highlight patterns of interactions with a browser [8] report on the accessibility of the site's content [4], demonstrate the role of link uncertainty in menu depth [5].

In this paper, we build upon the methods and model presented in Miller and Remington [5] to explore user strategies for backtracking. Like [5], the model we present here uses the threshold strategy but introduces a different strategy for exploring less probable links once any backtracking occurs. For the sake of presentation, we describe the methods and the model in its entirety, including the threshold strategy. We also show how certain design decisions are consistent with logs of users navigating a site and how the model's aggregate behavior tightly fits an empirical study addressing information architecture [6]. Finally, we experiment with the model's assumptions by exploring alternate designs and parameters in order to help identify the critical elements in the model's design.

MODELING INFORMATION NAVIGATION

Our goal is to simulate common patterns of user interaction with a web site to provide useful usability comparisons between different site architectures. A model that precisely replicates a user's navigation is not possible, nor is it necessary. Rather, a model that employs common usage patterns and simulates them with reasonable time costs can

predict and explain benefits of one design over another, such as when it is advantageous to use a two-tiered site instead of a three-tiered site.

Since completeness is not possible, process abstraction plays an important role in representing the environment and the human user. Abstraction is used principally to restrict our description of user processing, representing only their functionality. We guide abstraction and model construction with the following principles:

- The model should only perform operations that are within the physical and cognitive limitations of a human user. For example, a user can only focus upon (and evaluate) one link at a time. Also, humans have limited short-term memory and thus the model should not require an unreasonable amount of memory. In general, we choose designs that minimize memory requirements unless compelling principles or observations indicate otherwise.
- The model should make simplifying assumptions whenever they are not likely to have much impact on the model's aggregate behavior. For example the model takes a fixed amount of time to evaluate a link even though human users times are certainly variable. Since the model simulates the average user, this simplification will provide a good fit given a reasonable estimate of fixed time from human performance data.
- The model assumes that human cognition is generally rational [1, 9]. It will thus seek the most effective strategy for a given environment unless compelling evidence from human usage suggests otherwise. Given the large set of navigation strategies that can operate within reasonable physical and cognitive limitations, we examine a strategy that is most effective within known cognitive constraints.

Representing a Web Site

Our model interacts with a simplified, abstract representation of a Web browser and a Web site. Each site has one root node (i.e. the top page) consisting of a list of labeled links. Each of these links leads to a separate child page. For a shallow, one-level site, these child pages are terminal pages, one of which contains the target information that the user is seeking. For deeper, multi-level sites, a child page consists of a list of links, each leading to child pages at the next level. The bottom level of all our sites consists exclusively of terminal pages, one of which is the target page. Our examples are balanced trees since we generally compare our results to studies that use balanced tree structures (e.g. [6] and [3]). However, our representation does not prevent us from running simulations on unbalanced trees, or even on structures involving multiple links to the same page and links back to parent pages.

When navigating through a site, a user must perceive link labels and gauge their relevance to the targeted information. While the evaluation of a link is a complex and interesting

process in itself, we do not model the process per se. Rather, our interest involves the consequences of different levels of perceived relevancy. As a proxy for each link label, we fix a number, which represents the user's immediately perceived likelihood that the target will be found by pursuing this link.

In an ideal situation, the user knows with certainty which link to select and pursue. Figure 1 represents such a site. The rectangles represent Web pages that contain links (underlined numbers) to child and parent pages. The numbers on links represent the perceived likelihood that the link is on the path to the target. The top page for this site contains four links where the third link, labeled with a 1.0, eventually leads to the targeted page. Of the eight terminal pages, the page represented by the filled rectangle contains the target information. In our terminology, this example site has a 4x2 architecture, where 4 is the number of links at the top-level and 2 is the number of links on each child page. For this site, the user need only follow the links labeled with a 1.0 to find the targeted page with no backtracking.

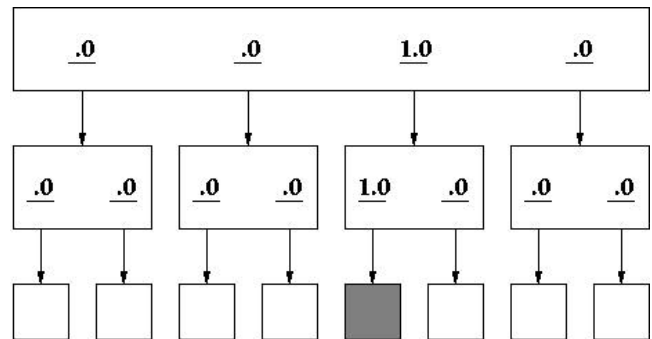


Figure 1 Site with sure path

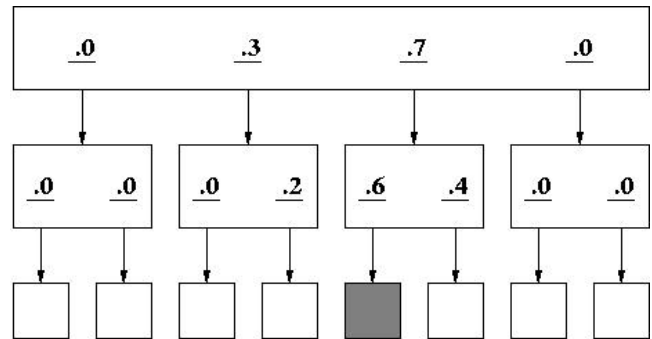


Figure 2 Site with likely path

Figure 2 shows an example of a simple two-level site with links whose relevance to the target is less certain. The top page in this figure contains four links labeled with numerical likelihood factors of .0, .3, .7 and .0 that represent the user's belief that the path associated with a given link contains the target information. As before, a user strategy that merely followed the most likely links would directly lead to the target. However, Figure 3 shows possibly the same site

with a different user for whom the meaning of the labels differs from the user in Figure 2. This user would find the target under what he or she perceives as a less plausible sequence of link selections. In this way it is possible to represent sites that differ widely in strength of association between link label and target information--from virtual certainty (links of 1.0) to complete uncertainty.

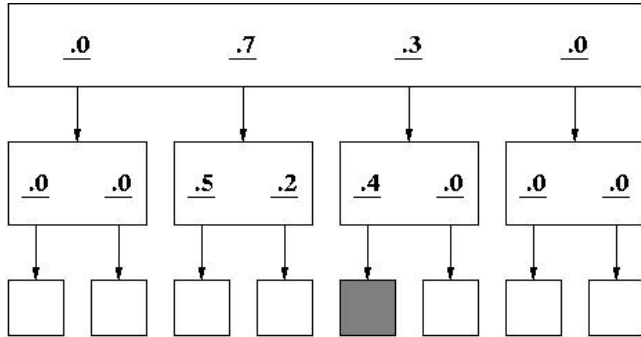


Figure 3 Site with unlikely path

Modeling the Browser and User Actions

Byrne et al. [2] found that selecting a link and pressing the Back button accounted for over 80% of the actions used for going to a new page. Consequently, we focused our modeling on the component actions underlying these behaviors. These include:

- Selecting a link
- Pressing the Back Button
- Attending to and identifying a new page
- Checking a link and evaluating its likelihood

To further simplify our model, attending to and identifying a new page can be folded into the actions of Selecting a Link and Pressing the Back Button since this action only occurs when either of these actions occur. Our revised model has three primitive actions:

- Selecting a link (and attending to and identifying a new page)
- Pressing the Back Button (and attending to and identifying a new page)
- Checking a link and evaluating its likelihood

Because of physical and cognitive limitations, only one of these actions can be performed at any one time. Fixed times are assigned to each action to account for its duration during a simulation. The model also simulates changing the color of a link when it is selected so that the modeled user can “perceive” whether the page under this link was previously visited.

Modeling Navigation Strategies

The model navigates a Web site by serially executing these three primitive actions. It also checks and evaluates links one at a time. Serial evaluation is motivated by evidence

that the human user has a single unique focus of attention that must be directed at the link for this decision.

A user may employ one of two strategies to quickly determine the path to pursue:

- The **threshold** strategy: The user immediately selects and pursues any link whose probability of success exceeds threshold.
- The **comparison** strategy: The user first evaluates a set of links and then selects the most likely of the set.

The threshold strategy is most effective if the first likely link actually leads to the targeted object. The comparison strategy is more effective only if a likely link is followed by an even more likely link that actually leads to the targeted item. Since either strategy may be effective, we examine the threshold strategy on the principle that it requires the fewest computational (cognitive) resources.

The model is neutral as to the actual order in which the links are evaluated. The design and layout of a page principally determine which links a user would evaluate first. Any understanding of how page layout and design affect the user's focus could eventually be incorporated into the model. With our current focus on the site structure, the model's representation establishes a fixed order in which links are evaluated for each run. So that the order does not systematically affect the model's predictions, our simulations randomly place the targeted item at a different terminal page for each run.

With the appearance of a new page, the model's threshold strategy first attends to the page, which, if it is a terminal page, includes checking if it contains the target information. If it does not, the model sequentially scans the links on a page selecting any link whose likelihood is equal to or above a fixed threshold (0.5 in the simulations reported below). When a page appears by selecting a link, the process of checking and scanning the page is repeated.

Once the model detects no unselected links above the threshold value, it returns to the parent page by pressing the Back button and continues scanning links on the parent page starting at the last selected link. It does not scan links it has already evaluated. Determining the last link selected places no demands on memory since the last selected link is easily detected by its color, and many browsers return the user to the location of the last selected link.

So far, for our description, the model only selects links that will probably lead to the targeted item. However, sometimes the targeted item lies behind ostensibly improbable links and, after some initial failures, human users must start selecting links even if the link labels indicate that they will probably not lead to the targeted item. The model presented in Miller & Remington [5] started selecting improbable links only after completing a full traversal of the site. We will call this the **traverse-first** strategy. However, a more effective strategy may opportunistically select improbable links at a

lower tier immediately after trying the more probable links and before returning to a higher tier in the site. We call this the **opportunistic** strategy.

Figure 4 illustrates how the opportunistic strategy may be more effective. The model scans across the top page and selects the second link (0.7). On the second level it selects the first link it encounters (0.5). After discovering that this is not the targeted item, it returns to the page on the second level. However, before returning to the top level, it temporarily reduces its threshold to 0.1, selects the second link (0.2) and finds the target on the new page. Had the targeted item been elsewhere in the site, the strategy backs up twice, returning to the top-level and resetting the threshold to the previous value (0.5).

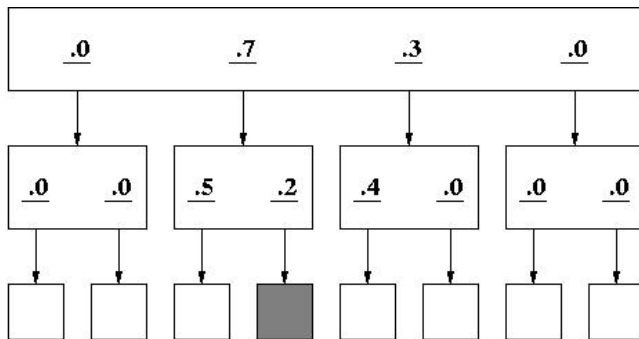


Figure 4 Site showing opportunistic strategy

The opportunistic strategy is a more effective strategy than the traverse-first strategy. First of all, opportunistic strategy explores the less probable links when the cost of doing so is minimal, that is, when the less probable links are immediately available. Secondly, it implicitly takes into account the positive evaluation of the parent link, which had indicated that the targeted item was probably under one of the links of the current page.

We further qualify when the opportunistic strategy is used. In some cases, a user may scan a page of links and determine that not even one of these links have the remote possibility of leading to the targeted item (defined as a likelihood factor of less than 0.1). In this case, our model assumes that the user has the memory to support the realization that rescanning the page would be futile. Instead of employing the opportunistic strategy, the model returns to the parent page. This memory of knowing that the page has nothing worthwhile only lasts as long as the model remains on the current page. Thus, if the model leaves the page and then returns to this same page, the model must assume that the page may be worth rescanning and the opportunistic strategy is employed. This qualification is also consistent with our design principles in that it contributes to an effective strategy while minimizing memory resources.

While the opportunistic strategy is consistent with our design principle that endorses an effective strategy, it does

require more memory. If the opportunistic strategy fails to find the targeted item with its opportunistic selections, it must reset the link selection threshold to the previous value upon returning to the upper level. Resetting a value requires storing the old value before reducing it. Storing and recalling one or two values reasonably fall within the limits of human cognition, but storing and recalling an arbitrary number of values does not. For this reason, our model allows us to fix a limit on the number of previous threshold values it can recall. We initially set this number to one, but later in this paper we will explore the impact of being able to store and recall additional values.

Part of our reason for adopting the opportunistic strategy in place of the traverse-first strategy was our examination of usage logs. Users were asked to search for a coffee grinder in a Web site whose organization mirrored a popular discount department store. Figure 5 shows a partial site map. While the coffee grinder is located under Small Electronics, most of the users first looked under Houseware and searched for the item in a manner consistent to the opportunistic strategy (although here the opportunistic strategy is not the most effective). In particular, 7 out of 11 users selected Houseware before trying Small Electronics (the remaining 4 users went directly to the target and thus their actions cannot support or refute the opportunistic strategy). Of these 7 users, 4 users initially skipped Cookware to select Kitchen Gadgets. Of course, server logs do not directly reveal how users evaluated the links, but skipping a link indicates that a user believed that it does not likely lead to the targeted item. After discovering that the coffee grinder is not under Kitchen Gadgets, these 4 users went back and selected Cookware to see if the coffee grinder was in this category. Consistent with the opportunistic strategy, these users tried a less probable link before returning to the top page.

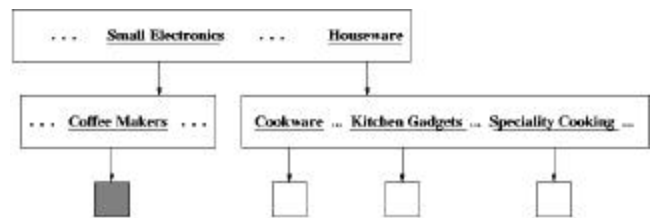


Figure 5 Example showing opportunistic strategy

Of the remaining 3 users in the group of 7 who selected Houseware, one of them also ostensibly used the opportunistic strategy. This person first selected Cookware and then skipped a few links before selecting Kitchen Gadgets. After discovering that the coffee grinder was not under Kitchen Gadgets, this user went back and selected the skipped links (i.e. Specialty Cooking and Cooking Utensils) to see if the coffee grinder was in this category.

Only 2 of the 7 showed no evidence of following the opportunistic strategy. These users selected only one link

in the Houseware page (Kitchen Gadgets for one user and Speciality Cooking for the other user) before returning to the top page and selecting Small Electronics to successfully find the Coffee Grinder. Even though their actions do not support the opportunistic strategy, their actions are not necessarily inconsistent with it. It is possible that these users judged the remaining links in the Houseware page to be so unlikely that their evaluation did not even meet a lower threshold that triggers the opportunistic strategy.

Simulation Parameters

Miller and Remington [5] established reasonable constants that estimate the time costs of link evaluation and link selection. Using a range of time costs, the model's performance was compared to results from menu selection studies. The model produced close fits when the link evaluation cost was set to 250ms and link selection cost was set to 500ms.

The comparison to menu selection results assumes ideal links. That is, the model need only follow a '1' to successfully find its target page without any backtracking. While this assumption was appropriate for simulating the menu selection studies where little backtracking occurred, it does not model situations, which include many Web sites, where users frequently select the wrong links and need to backtrack. Miller and Remington also presented a systematic method to simulate label ambiguity, which we review here.

To assign likelihood factors to the links, the ideal link values (1, 0) are perturbed with noise according to the formula below:

$$g * n + v$$

In this formula, g is a number chosen randomly from a standard normal gaussian distribution (mean=0, stdev=1); n is the noise factor multiplier (equivalent to increasing the variance of the normal distribution); and v is the original likelihood value (0 or 1). Since this formula occasionally produces a number outside the range from zero to one, our algorithm may repeatedly invoke the formula for a link until it generates a number in this range. The noise factor n thus models the level of label ambiguity in the site.

SIMULATIONS

Whenever a new model is proposed, it is helpful if its behavior is compared to the same studies as previous models to determine if its predictions are at least as accurate. Using the same threshold strategy, our model is equivalent to that presented in Miller and Remington [5] as long as no backtracking occurs. It produces the same predictions for the menu selection studies where minimal backtracking occurred.

Larson and Czerwinski [3] reported results of Web navigation where participants often backtracked before finding a targeted item. Using two two-tiered architectures (16x32 and 32x16) and a three-tiered architecture (8x8x8),

their participants navigated three Web sites which were otherwise comparable. Participants took longer to find items in the three-tiered site (58 seconds on average) than the two-tiered sites (36 seconds for the 16x32 site and 46 seconds for the 32x16 site). The difference between the three-tiered site compared to each of the two-tiered site was significant.

Simulation results using the threshold strategy with the traverse-first strategy [5] produced results that are consistent with the human users study in that simulations using the 8x8x8 site predicted slower search times than simulations using both of the two-tier sites. These results occurred when the noise factor was greater than 0.2. However, when the noise factor was lower, the simulations predicted that the three-tiered 8x8x8 site produces faster search times than both of the two-tiered sites.

Simulations of the Opportunistic Strategy

We conducted simulations using the threshold strategy for link selection with the opportunistic strategy for backtracking. Sites were constructed by randomly placing the target item at one of the terminal pages and assigning a value of 1.0 to links leading to the targeted item, 0 for all other links. Link values were then perturbed by gaussian noise as described above. Unlike [5], the noise was not applied the bottom level that leads to the terminal pages. While not necessarily plausible for all Web sites, these sites corresponds more closely to the sites used by Larson and Czerwinski since their participants could clearly tell whether the link's label matched the text of the targeted item. Figure 6 shows a sample 4x2x2 architecture generated with a noise factor of .3.

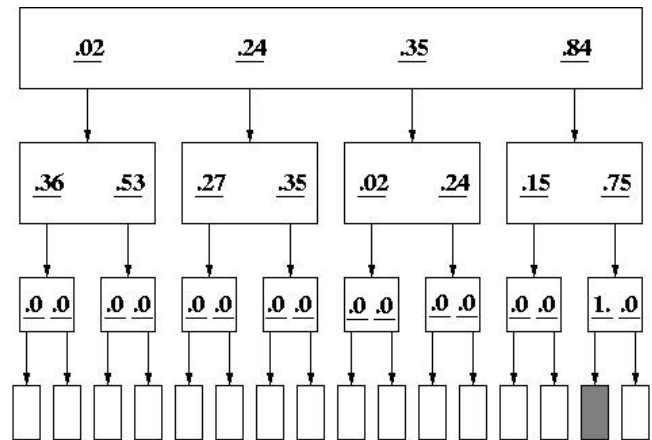


Figure 6 Site with no label noise on the bottom level

Simulations were run 10,000 times for each site architecture (8x8x8, 16x32, and 32x16). Simulated time costs were 250ms for evaluating a link, 500ms for selecting a link, and 500ms for return to the previous page (pressing the back button). In keeping with Larson and Czerwinski [3] any run lasting more than 300 seconds was terminated and assigned a duration of 300 seconds.

Figure 7 shows the calculated mean times of the simulation runs. The 8x8x8 architecture produced faster times at low levels of noise but slower times at noise levels above 0.2. At these higher noise levels the results are consistent with the human users. At noise levels of 0.4 and higher, the simulations ran faster with the 16x32 architecture than the 32x16 architecture. This difference was also noted in the study with human users, albeit not reported as statistically significant.

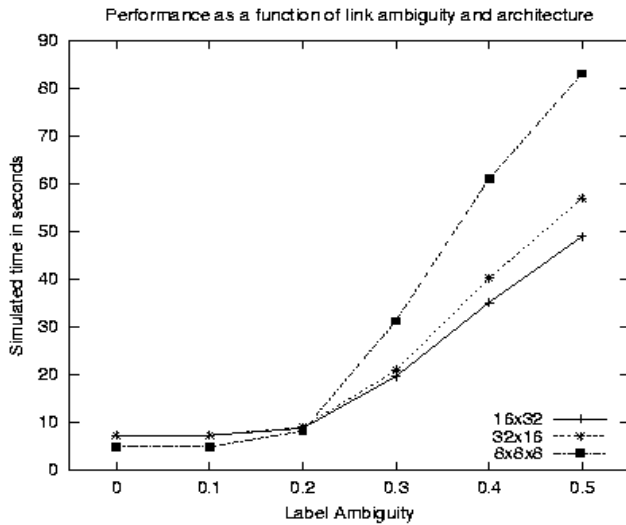


Figure 7 Simulating threshold and opportunistic strategies

At a noise level of 0.4, the simulation results closely match the human results in absolute terms: 62s (compare to 58s for humans) for 8x8x8, 43s (compare to 46s) for 32x16, and 35s (compare to 36s). It appears that the 0.4 serves a good parameter estimate describing the amount of label ambiguity in the sites used by Larson and Czerwinski.

Impact of Time Costs

While changing the time costs (250ms for link evaluations and 500ms for link selection and returning to the previous page) will affect absolute simulation times, it is less clear if different time costs will change which architecture produces the fastest times. For example, one may wonder if the 8x8x8 architectures would still produce the slowest times if the link selection cost were double, which may occur for a slower internet connection.

To explore the impact of time costs, we look at the number of link evaluations, link selections and page returns. If independent counts of these actions correlate with the aggregate simulation time, we conclude that varying the time costs have minimal impact on the relative performance of the different architectures. For example, if the 8x8x8 requires more evaluations, more selections and more returns than the other architectures, we know that 8x8x8 will produce slower search times regardless of the time costs.

Looking at the number of evaluations, selections and returns, we see that the 8x8x8 architecture required more of each action (173, 17, and 19 respectively) at the 0.4 noise level than the 16x32 (125, 3, and 5) and the 32x16 (134, 6, and 8). Further experimentation reveals that this relationship holds across all but the lowest noise levels (0.2 and less). We conclude that changing the time costs have no effect on the relative comparisons provided that the noise factor is at least 0.3.

Impact of Memory Capacity

Recall that the opportunistic strategy requires the model to store and retrieve threshold values so that the previous threshold can be reinstated upon returning to a parent page. So far, our simulations have assumed that only one threshold value can be restored. Thus, if the model returned to the top level of a three-tier architecture, it would no longer be able to recall the previous threshold and would simply leave the threshold at its current state.

Because this limited memory capacity only hinders performance in a three-tiered site (e.g. 8x8x8), we ran simulations where the memory capacity could hold the additional threshold value so that the previous value could be reinstated when navigating through a three-tiered site. Figure 8 shows the results using the same scale as Figure 7. While we see that the extra memory capacity improves the performance of the 8x8x8 architecture, its navigation is still slower than the two-tiered architectures.

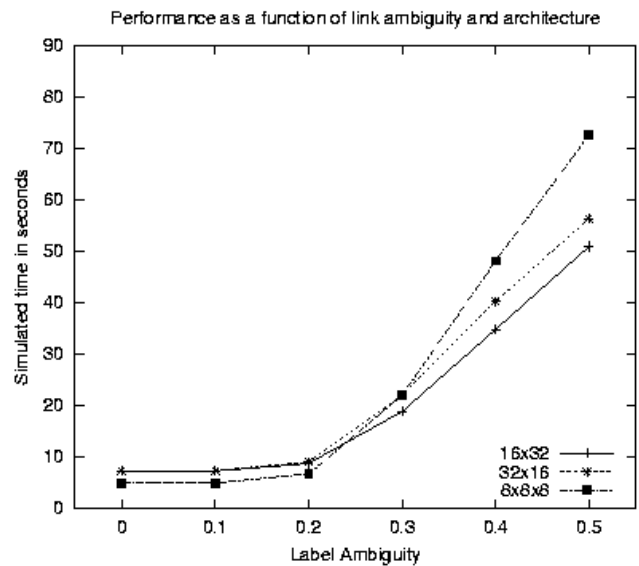


Figure 8 Results using a larger memory capacity

Impact of Bottom-Level Noise

The above results were from simulations where the bottom level of links have unambiguous labels. While this corresponds to the sites constructed for the Larson and Czerwinski study, this assumption does not hold for many

real Web sites. In particular, people often do not search for a specific item, but need to visit the target page before realizing they have found what they are looking for. To explore the consequences of having ambiguous links at the bottom level, we ran simulations where the noise-producing process was applied to every label in the site.

Figure 9 shows the results. Not surprisingly, the addition of noise on the bottom level increased the absolute times for all architecture (note that a larger range for the y-axis is required to plot the results). More importantly, the three-tiered 8x8x8 architecture this time produced the fastest results for all noise levels.

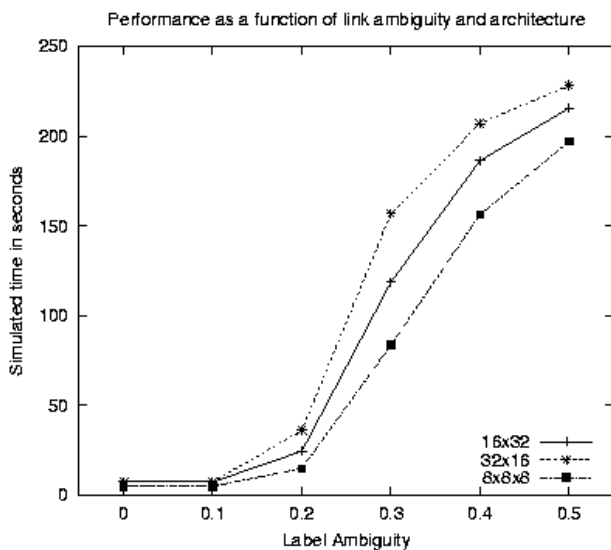


Figure 9 Results with noise applied to every link label

DISCUSSION

Consistent with Miller and Remington [5] we have shown that a simple model of a web user can provide an excellent account of user behavior and reveal important factors underlying web usage. Like [5] the model shows that link ambiguity interacts with information architecture (menu depth here) to determine usability. As link ambiguity decreases, better performance is found from architectures with deep structures that minimize the number of links searched. This is in keeping with information theory and has been observed in human data. As link ambiguity increases, the model shows performance degradations for architectures with deeper structures. The same pattern is characteristic of human users. However, the preference for shallow hierarchies is observed only when the bottom layer of labels is unambiguous. Thus, the results of Larson and Czerwinski [3] may not generalize to large numbers of real Web pages with ambiguity at all levels. Once again, one of the advantages of modeling is that it makes it possible to explore a larger parameter space than is possible with realistic empirical tests.

We are currently trying to determine why ambiguity (noise) at the lowest level alters the results so significantly. The reason probably involves the observation that the two-tiered architectures have more links at the lowest level than the three-tiered architecture. Adding noise to the lowest level may then degrade performance more for the two-tiered architectures since noise is added to more of their links. Additional simulations with noise selectively added to different levels may further qualify the advantages of shallow architectures over deeper architectures.

As for Web search strategies, combining threshold-based selection with opportunistic search strategies improved the fits to observed data from those reported in [5]. The simulated times are now very close to actual observed times for 0.4 noise level. This also corresponds to the observed behavior of several users in the department store study reported above. We recognize the need for converging methods to independently determine link ambiguity and are exploring theoretical and empirical methods of estimating actual values.

To make time predictions, our model assumes plausible time costs for link evaluation, link selection and returning to the previous page. By noting the actual counts for these operations, our simulations help us understand what happens when the link selection time is significantly longer, as would be the case for a slow internet connection. We found, however, that the time costs have no effect on the relative comparisons provided that the noise factor is at least 0.3. This suggests that a slower internet connection does not impact the relative advantage of shallow architectures as long as they have significant amount of link ambiguity, at least for the case where no noise is present at the bottom level.

Our simulations also contributed to understanding how human memory impacts effective navigation. Increasing the model's memory capacity improved performance for the deep (8x8x8) structure but left the other two architectures largely unaffected. This suggests that memory is more useful in keeping track of site architecture than in searching within a page. Since searching a page is facilitated by visual cues (e.g., changes in the color of previously selected links) users can avoid reliance on memory. Visual cues are typically not present to remind users of the names and locations of previous links. The interaction of information architecture with memory capacity indicates further that simple heuristics for representing capacity (e.g., Miller's capacity limit of 7 ± 2 [7]) are insufficient to capture memory phenomena of importance. Instead, it is necessary to examine how the structure of information sites provides aids to memory. Our analysis contrasts with previous advice suggesting that the number of links per page should be limited to 10 [10] (see [3] for a discussion based on experimental results).

We have shown that a simple model of a user interacting with a simplified Web page can reveal important factors that affect usability and can support the investigation of the interactions between those factors across a wide range of conditions. What we have presented is not a comprehensive model of Web navigation. No attempt is made to account for how people scan a page, or evaluate link labels or images. By abstracting these processes, and representing only their functionality, the model can focus on understanding how information architecture affects the navigation process. As an approximation of user navigation through a Web site the model can account for a vast range of human behaviors by varying likelihood factors in its site representations. We have shown that the model provides a good approximation of the behavior of the common (modal) user. By varying parameters it should be possible to extend the model to account for alternate strategies.

ACKNOWLEDGMENTS

We thank Gillian Jetson for running the user navigation study, whose user logs are described in this paper.

REFERENCES

1. Anderson, J.R. *The Adaptive Character of Thought*. Erlbaum, Hillsdale, NJ, 1990.
2. Byrne, M.D., John, B.E., Wehrle, N.S., & Crow, D.C. The tangled web we wove: A taskonomy of WWW use, in *Proceedings of CHI'99 Human Factors in Computing Systems* (1999), ACM press.
3. Larson, K. and Czerwinski, M. Web page design: Implications of memory, structure, and scent for information retrieval, in *Proceedings of CHI'98 Human Factors in Computing Systems* (1998), ACM press.
4. Lynch, G., Palmiter, S., and Tilt, C. The max model: A standard web site user model, in *Proceedings of the 5th Annual Human Factors and the Web Conference* (1999).
5. Miller, C.S. and Remington, R.W. A computational model of Web navigation: Exploring interactions between hierarchical depth and link ambiguity, in the proceedings of *The 6th Conference on Human Factors and the Web* (Austin TX, June 2000).
6. Miller, D. P. The depth/breadth tradeoff in hierarchical computer menus, in *Proceedings of the Human Factors Society* (1981), 296-300.
7. Miller, G. The Magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review* 63, 2 (1956), 81-97.
8. Peck, V. A. & John, B. E. . Browser Soar: A computational model of a highly interactive task. In *Proceedings of CHI'92 Human Factors in Computing Systems* (1992), ACM press, 165-172.
9. Pirolli, P and Card, S. Information foraging. *Psychological Review* 106, 4 (Oct 1999), 643-675.
10. Rosenfeld, R. and Morville, P. *Information Architecture for the World Wide Web*. O'Reilly & Associates, Sebastopol, CA 1998.