

Computational Models of Information Scent-Following in a Very Large Browsable Text Collection

Peter Pirolli

Xerox PARC

3333 Coyote Hill Road

Palo Alto, CA 94304

pirolli@parc.xerox.com

ABSTRACT

An ecological-cognitive framework of analysis and a model-tracing architecture are presented and used in the analysis of data recorded from users browsing a large document collection. The users interacted with the Scatter/Gather browser, which clusters documents into groups of similar content and presents users with summaries of cluster content. Predictions made by a computational model of navigation and information foraging are matched against the observed activity.

Keywords

Information foraging, information scent, cognitive modeling, model-tracing, ACT-R, Scatter/Gather

INTRODUCTION

The Scatter/Gather browser interface studied earlier [12] was designed to meet a challenge that is encountered more and more often in a growing universe of digitally stored texts. The browser had to summarize and communicate the common content of very large collections. It was typical for the browser to attempt to communicate the sense of 100,000 documents with just four lines of text on just a small portion of the screen. Users must rely on such terse representations of content as a kind of *information scent* whose trail leads to information of interest. The computational models presented here are an attempt to evaluate and understand how those summaries work to provide information scent to users..

It has often been argued that computational models of human cognition should serve a useful role in the design and evaluation of human-computer interaction systems [10]. Such models provide a way of simulating the perception, cognition, and action of users' interaction with computer interfaces. Such computational cognitive modeling should be useful in analyzing and evaluating alternative designs.

The predictions of these models can be exquisitely detailed, but the models are typically difficult to prototype and use in an exploratory way, iteratively testing model variations against data collected from large numbers of users. Often the raw user data being matched consist of video or verbal protocols of users' interactions that must be coded by hand and compared by hand against the predictions of a

cognitive simulation. Typically the effort involved in hand coding and comparison limits the number of users that can be modeled. Other times, the simulation simply models some sort of "average" or "normative" user and those predictions are compared against the average behavior of a group of users. This paper presents a *model-tracing* methodology [2] and architecture that facilitates the development of alternative cognitive models that are matched to on-line logs of interactions from many individual users. Like some other recent developments [3, 15], the model-tracing architecture provides a means of automatic protocol analysis using protocol parsing grammars. The model-tracer extends this by providing a way to refine computational cognitive models to fit the data at hand.

Another difficulty of developing computational cognitive models is that, often, one does not want all the exquisite detail. For some interface design problems, it may be important to model every gaze and motor movement [8] and to understand performance differences that are measured in milliseconds. For longer tasks that may take on the order of hours, days, or a lifetime, a more coarse-grained analysis may be sufficient. Here, I present a level of analysis at the *knowledge level*, that may allow one to perform more abstract and approximate analysis than fine-grained cognitive modeling. The general idea is that the architecture presented here should facilitate an exploratory approach to cognitive modeling. One may start with a coarse-grained model, investigate alternative models, and refine a more fine-grained model as needed. An interface designer may trade off exquisite detail and great effort for approximate results at less effort.

This paper also presents a general framework of modeling within a theoretical approach that combines psychological explanation with the analysis of the fit of human-computer interaction to a task environment, which is a kind of ecological analysis.

To illustrate the framework, methodology, and architecture, an analysis of user browsing, studied under experimental conditions [12], is presented. The general objective for users was to find bibliographic references relevant to a set of topics using the Scatter/Gather browser. They did this by navigating through a very large text collection that had been automatically organized into

Table 1. An integrated framework for ecological and cognitive analysis of human-computer interaction.

Level	Question	Description	Analysis Elements	Examples
Adaptation	<i>Why do they do it?</i>	Rational	<ul style="list-style-type: none"> States, resources, state dynamics Constraints, affordances Feasible strategies Optimization criterion 	<ul style="list-style-type: none"> Optimal foraging theory Information foraging theory
Knowledge	<i>What do they do?</i>	Intentional	<ul style="list-style-type: none"> Environment Goals, preferences Knowledge Perception, Action 	<ul style="list-style-type: none"> Knowledge-level model-tracing
Cognitive	<i>How do they process information to do it?</i>	Mechanistic	<ul style="list-style-type: none"> Cognitive states Cognitive processes 	<ul style="list-style-type: none"> ACT*, ACT-R Soar
Biological	<i>How do they physically do it?</i>	Physical	<ul style="list-style-type: none"> Neural processes 	<ul style="list-style-type: none"> Neural models

topically related clusters. The general objective of the analysis here is to understand the navigation choices that were made every step of the way by each and every user on each and every task.

The work here shares commonalities with other attempts to model the interaction of text-comprehension and action planning [9], although the general cognitive modeling and model-tracing architecture presented here extends the space of possible cognitive simulation models that can be constructed and the speed with which they can be developed and explored.

AN ECOLOGICAL-COGNITIVE APPROACH

In previous work [11], we argued for an ecological approach to the analysis of human-computer interaction and to the synthesis of new designs for interaction. Within this framework, we presented an outline of *Information Foraging Theory* as an approach to understanding human information-gathering and sense-making strategies. The general idea is that we may scientifically study human and technological adaptations to the flux of information in the cultural environment in much the same manner as biological adaptations to the flux of energy in the physical environment [14].

Our earlier emphasis was on trying to understand the degree to which observed information foraging behavior is adaptive given the environmental context in which it occurs, or how future designs may be made more adaptive. We call this *adaptation analysis*, and it involves trying to understand how human-computer interaction conforms to a priori design specifications. It is a kind of engineering analysis. We presented several optimization models as

examples of powerful tools for studying the design of human-computer interaction in information foraging.

FRAME OF ANALYSIS

Table 1 presents a summary of an overall framework for studying human-computer interaction from an ecological and cognitive perspective. I should point out that all of the pieces may be found in existing literature in recent evolutionary theory, cognitive science, and measurement, so no claim for novelty is being made. It is useful, however, to review the big picture.

Table 1 presents four levels of analysis: (1) an *adaptive analysis level* where it is assumed that the structure of behavior can be understood in terms of its adaptive fit to the structure and constraints of the environment, (2) a *knowledge level* where descriptions of observed behavior assume that it is the product of human purposes and knowledge (i.e., the observer takes an *intentional stance*), (3) a *cognitive level* where cognitive models provide mechanistic accounts for information processing activity, which ultimately map down to (4) a *biological level* where explanations are couched in terms of physical machinery.

THE SCATTER/GATHER BROWSER

The Scatter/Gather browser [5] uses the clustering of documents as the basis of a browser suitable for large numbers of documents. The clustering depends on a measure of inter-document similarity (basically: the normalized correlation of their word-frequency vectors). Clusters of documents are represented by *meta-documents* containing profiles of topical words and the most typical titles. These topical words and typical titles are also used to present users a summary of the documents in a cluster. Topical words are those that occur most frequently in a

cluster, and typical titles are those with the highest similarity to a centroid of the cluster. Together, the topical words and typical titles form a *cluster digest*.

Figure 1 presents a schematic view of the Scatter/Gather interface.¹ The document cluster digests are presented in subwindows represented by the small separate rectangles in Figure 1. An expanded view of one of the cluster digests is presented at the bottom of Figure 1. The user may *gather* clusters of interest by pointing and selecting buttons above each cluster. On command, the system will pool the documents in those clusters, then automatically *scatter* that subcollection into another set of clusters. The user may repeatedly scatter then gather clusters, moving from very large cluster collections to very small cluster collections. Eventually the user may display all the titles of documents in a cluster, then select individual documents to read.

In studies [11, 12], Scatter/Gather was applied to the 2.2 gigabyte TIPSTER text collection created for the TREC text retrieval conference [6]. This test corpus contained 742,833 full-text documents collected from the Wall Street Journal, the AP newswire, Department of Energy technical abstracts, the Federal Register, and computer articles published by Ziff-Davis. The corpus has been extensively used by the information retrieval community. Standard information retrieval tasks (queries) have been defined on it together with lists of known relevant and non-relevant Tipster documents, as judged by experts. Even if such lists of relevant documents cannot be made completely objective, the test corpus at least provides us with a common standard against which to compare performance.

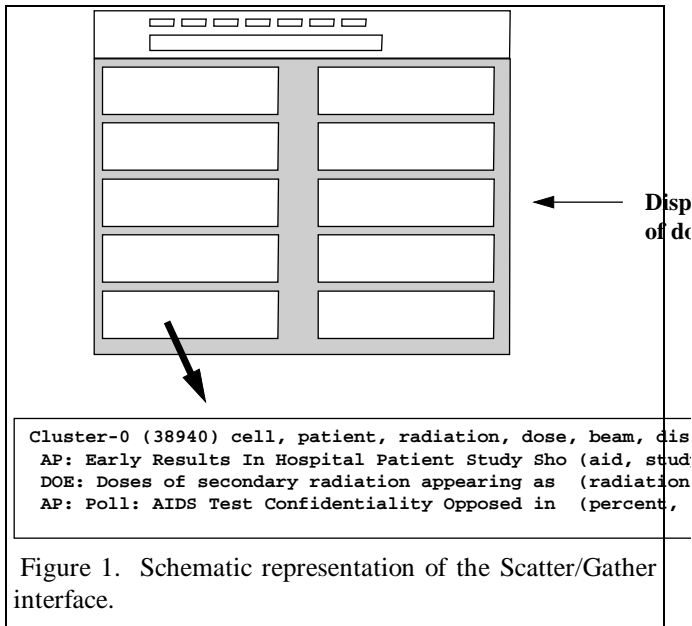


Figure 1. Schematic representation of the Scatter/Gather interface.

¹This interface was developed by Marti Hearst at Xerox PARC.

In the retrieval tasks we studied, participants were asked to find as many documents as possible, as efficiently as possible, relevant to retrieval tasks of three levels of difficulty: Hard, Medium, and Easy. The difficulties were determined by the number of relevant documents identified by experts in the Scatter/Gather Tipster corpus for each task, for the purposes of evaluation studies by the TREC community. The Hard tasks had $M = 56$ expert-identified documents, the Medium had $M = 303$ expert-identified documents, and the Easy had $M = 865$ documents.

For each task, the user read a task query (a topic description typically taking up one paragraph). They then browsed through the Scatter/Gather browsing seeking document citations that appeared relevant to the query they had read.

Summary of the Adaptationist Analysis

Due to space constraints, only the gist of the adaptationist analysis of user interaction with Scatter/Gather can be given here. One may think of the user as an “information predator” whose aim it is to select “information prey” and to maximize the rate of gain of information relevant to their task. A mathematical model for this kind of information diet problem was presented in Pirolli and Card [11]. The optimal solution to that model predicts that users somehow rank the profitabilities of the information prey (clusters) where, in this case, the profitability is proportional to the proportion of relevant documents in a cluster (*cluster precision*) divided by the time it will take to process the cluster. The optimal *cluster diet* can be constructed by selecting clusters in order of descending profitability up to a threshold. That threshold is at the point where the overall average rate of finding relevant documents will be decreased by adding the next lower ranked cluster. For unreported data from the study in Pirolli et al. [12], it can be shown that even a coarse, approximate model of the cluster diet at the first level of Scatter/Gather clustering will predict the observation that users select more clusters as the difficulty of query decreases from Hard to Medium to Easy, and the predicted number of clusters selected is within about 25% of the observed.

This adaptationist analysis proposes the rationale (the why) of user behavior. The knowledge-level model below is intended to explore explanations of what and how users judge cluster precision (profitability) from the cluster digest texts presented on their screens. Given what is presented on the screen what do they choose to do and how does such choice seem to be made?

Model-tracing Architecture

Overview

The model-tracing architecture is built upon the ACT-R production system [1], which is proposed as a general theory of human cognition. ACT-R consists of a

production memory and a *working memory*. The working memory basically models the information being attended to, and information that has been recalled (*activated*) from long-term declarative memory. The production memory contains production rules which are patterns of the general form *Condition* → *Action*. ACT-R operates on a basic *match-execute cycle*. During the match phase, the condition part of the production rules patterns are matched against information in working memory. Those that match are then ranked by an evaluation function, the best match is selected, and its action pattern is executed during the execution phase. Actions specify updates to working memory, setting of goals, and actions to be performed in the world.

Figure 2 presents a general overview of how the model-tracing architecture was used. The raw observed data were the logged interaction protocols obtained in the Pirolli et al [12] study. These were preprocessed into two files used by the production system model: (1) a long-term memory (LTM) file that contained a specification of a *spreading activation network* to represent words and inter-word memory associations in users’ long-term memory and (2) an interword correlation (IWC) representation used to approximate users’ conception of word synonymy [13]. An LTM and IWC file was produced separately for each log file, and each log file captured an individual user working on one of their individual tasks. The LTM and IWC files were loaded into the production system model and used to govern its dynamical behavior. In particular, these were used by *evaluation functions* to determine which production rules were best to execute given (a) the particular clusters being viewed, (b) the query being worked on, and (c) the interword correlations or spreading activation between the query and clusters.

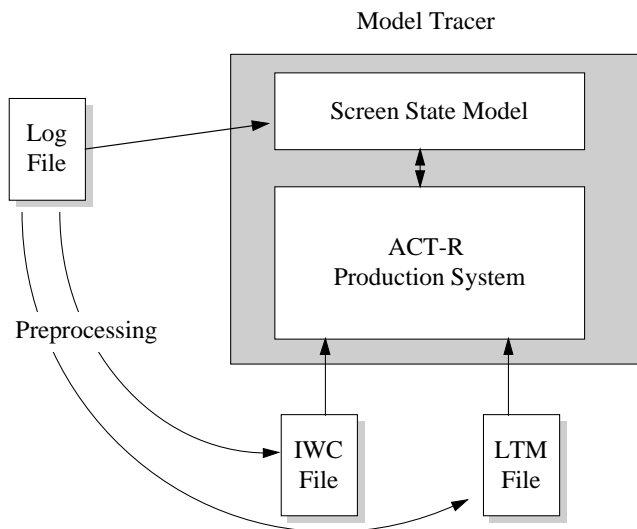


Figure 2. Flow of information in the model tracer.

To model-trace a log file, the ACT-R production system is initialized with (a) production rules for the task and (b) the portion of long-term word memory that will be used (from the LTM file). The model-tracer opens an action stream that delivers the sequence of user actions that were observed in the log file, and it uses this to maintain a model of the Scatter/Gather screen state. Changes in screen state are “perceived” by the ACT-R production system, which means that working memory elements are created and added to working memory when they “appear” on the screen in the screen state model.

The model-tracer runs the ACT-R production system for one cycle at a time, catching it just as it has prioritized the productions that match to the current goal and state of working memory. At this point, the ACT-R production system has made its predictions about the best-matching production as well as an overall ranking of the matching productions. From this sorted *conflict set*, the production that actually matches the protocol is selected by the model-tracer for execution, duly noting how it ranked among the productions picked and ranked by ACT-R. Following production execution, the model-tracer reads the next action from the log file, updates the Scatter/Gather screen state model, and ACT-R updates its working memory in accordance with any “perceived” screen changes.

Events, Actions, and Model-trace Matching

Figure 3 presents the relationship among three streams of occurrences that are used in the model-tracer. The raw data log files contained recordings of button presses and releases (from the user using a mouse) and screen updates involving new windows being created or deleted. Each record was time stamped, though they were not necessarily written out in a time-ordered manner.

A finite-state recognizer [7] with regular language parsing rules is used to parse the raw log file records into *<event, interface-object>* elements that are delivered to an *event stream* in a time-ordered way and also pushed onto a stack. These event elements are codings of *physical events*. *Actions*, on the other hand, are codings of users’ *intentions*. That is, we assume that when a user presses a button labeled “Cluster 1” (the physical event) that they intended to “choose Cluster 1 to better satisfy my goals” (an intention). This distinction between physical descriptions of behavior and intentional descriptions of actions is common in many writings in the social, behavioral, and cognitive sciences. Elements from the event stream, the event stack, and the current system state are parsed using an augmented transition network [4]. The action parser contains transition rules that specify how events map onto actions and how they change the system state.

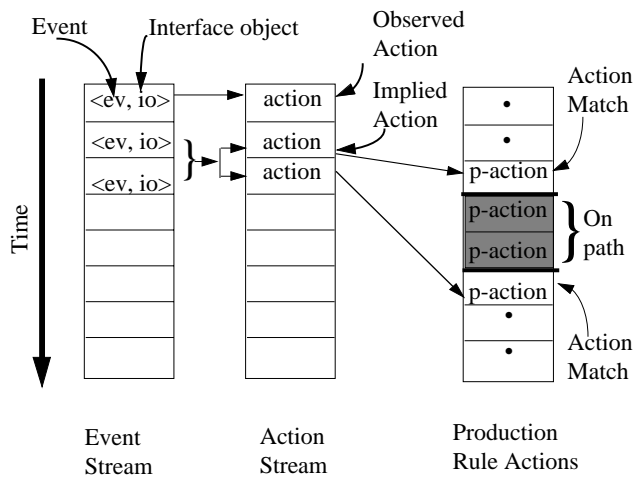


Figure 3. The parse of log files to event, action, and production rule streams.

Occasionally it is necessary to infer an action. In the Scatter/Gather example, this occurs when one event associated with one window is followed by another event associated with another window. In this case, a transition rule is needed to infer that a shift of attention has occurred prior to the observed event. The mapping of two events onto two actions in Figure 3 captures this kind of transition.

The model-tracer has to match the activity of the production system model against the observed stream of actions. The stream of production system actions (*p-actions*) is indicated on the right side of Figure 3. Often, the production system model must take several steps before emitting an action that could match a log file action. For instance, before selecting a cluster, the production system model must “look at” that cluster on the screen, which is an action that is unrecorded in the log files. Consequently, the model-tracer distinguishes between *action matches*, which match a *p-action* to log file action, and *on-path*

matches, which match *p-actions* that are consistent with the action stream but not observed. In addition, *default matches* occur when there is neither an action match nor a path match, but the production system has suggested a best match *p-action*. To specify the way that productions match log file actions one defines *match-patterns* for individual production rules, which specify the kind of match (action-match, on-path, or default) and tests of the production and the log file action.

Scatter/Gather Screen Model and User Action Types

All screen objects in the screen model are represented as Common Lisp objects. Actions are also represented as Common Lisp objects. The system state records the start time of the task and a stack of window objects representing the overlapping windows on the screen. The two main types of windows of interest are (1) the Scatter/Gather windows, which present the cluster summaries, and (2) titles display windows which present the titles of all the documents in a cluster. Each cluster summary consists of topic words and typical titles plus profile words (typics).

The “perception” of screen state by the production system model occurs by transducing screen state objects into ACT-R working memory elements. To do this, ACT-R was augmented with *transduction rules*, which specify how Scatter/Gather screen model objects are mapped to working memory elements when the production action LOOK-AT is executed by a production rule.

Production Model of Scatter/Gather Users

The model of Scatter/Gather users consists of 15 production rules. These are glossed in Table 2. On the left of the arrow are mnemonic names for the productions and the conditions for matching working memory. The right side of the arrows are the actions of the production rules. Some productions are annotated with a “(2)” to indicate that there are actually two copies of the productions for the two different types of window.

Evaluation Functions

At each cycle of the production system simulation, ACT-R selects production rules that match the current state of working memory. These matched rules are known as the conflict set. Such matched productions (or *production instantiations*) in the conflict set are ranked by some evaluation function and the highest ranked (best match) production is selected for execution. The ACT-R architecture comes with a theoretically motivated evaluation function [1]. Unfortunately, this one seems to be ill-suited, in any straightforward formulation, to the Scatter/Gather data because it assumes tasks with goals with fixed net values (Anderson 1994, personal communication). Consequently, some exploration of possible evaluation functions seemed necessary as a step towards specifying a cognitive model.

Three kinds of evaluation function were formulated for the cluster-selecting production, **select-relevant-cluster**. The condition of this production matches to words in the query task and words in a cluster summary. Each evaluation function embodied a different approach to the computation of the correspondence (degree of information scent) between the query and cluster:

- *Simple word overlap*. The value of a cluster selecting production is the size of the intersection set of words in the task query and the words in a cluster summary. This served as a base-line evaluation function.
- *Interword correlation*. The values of cluster selecting productions are sums of the interword correlations between each query word compared to each cluster summary word.
- *Strength of activation*. The values of cluster selecting productions are the sum of the activations of all query words. Query words receive activation spread from the cluster summary when it receives attention.

Interword Correlations

It is an hypothesis that interword correlations capture the topical or semantic similarity between two words, and they have been used to automatically compute a thesaurus for use in information retrieval [13]. The aim is to construct and use a word by word matrix, where each element of the matrix records the number of times two words cooccur in a window of a certain size (i.e., a 40 word window) for the text corpus under investigation. Because of computational limits a reduced matrix representation is used [13]. The similarity of two words is just the normalized correlation (the cosine of the angle) between two vector rows in the matrix. This is an assessment of the degree to which two words seem to occur in the same kinds of surrounding text.

Activation Strengths

Spreading activation theories of human memory generally predict how a resource called *activation* is spread from cognitive structures that reside in a focus of attention.

Table 2. ACT-R production rules to model Scatter/Gather

Notice-new -window (2)	
New window on screen	→ Attend to it & set goal to process it
Attend-to -window (2)	
Attend to window	→ Look at window
Unattend-to-screen (2)	
Goal is to process a window & different window has appeared	→ Pop the goal
Shift-attention	
Another window is present	→ Attend to that window
Process-clusters	
Goal is to process SG window	→ Set goal to process clusters
Process-next-cluster	
Goal is to process SG window & one is unprocessed	→ Set goal to process next cluster
Look-at-next-cluster	
Goal is to process next cluster	→ Look at cluster & pop the goal & set goal to process cluster elements
Look-at-cluster-elements	
Goal is to look at cluster elements	→ Look at topics and typics & pop the goal
Select-relevant-cluster	
Goal is to process SG window & there is a query & there is an unselected cluster	→ Select the cluster
Deselect-relevant-cluster	
Goal is to process SG window & there is a query & there is a selected cluster	→ Deselect the cluster
Do-Scatter/Gather	
Goal is to process SG window and some clusters have been selected	→ Scatter/Gather the window
Do-display titles	
Goal is to process SG window and some clusters have been selected	→ Display the titles in the window

Activation may be interpreted metaphorically as a kind of mental energy that drives cognitive processing. The spread of activation from one cognitive structure to another is

determined by some network representation where interstructure links weight the rate of activation flow (analogous to pipes or wires with specific flow capacities). Spreading activation theories are usually interpreted as predicting that more activated structures will receive more favorable processing.

The simulations here used an evaluation function that rated cluster matching productions based on the activation of the task query when a cluster summary and its words were in the focus of attention. The activation computation was based on that of ACT-R (though not actually computed by the ACT-R architecture). The activation of a query word i is

$$A_i = B_i + \sum_j W_j S_{ji} \quad (1)$$

where B_i is the base-level activation of i , S_{ji} is the association strength between cluster word j and query word i , and W_j is the base level activation of cluster word j . Equation 1 is a Bayesian prediction of the relevance of one word in the context of other words. A_i in Equation 1 is interpreted as reflecting the log posterior odds that i is relevant, B_i is the log prior odds of i being relevant, and S_{ji} reflects the log likelihood ratios that i is relevant given that it occurs in the context of word j .

The spreading activation network in the LTM files of Figure 2 are based on the following equations used to derive the values in Equation 1. B_i reflects the log prior odds so

$$B_i = \ln\left(\frac{\Pr(i)}{\Pr(\bar{i})} \psi\right) \quad (2)$$

where $\Pr(i)$ is the probability of word i occurring in the text word sequence, $\Pr(\bar{i})$ is the probability $1 - \Pr(i)$ that the word will not occur, and ψ is a normalizing constant used to yield positive values. These probabilities are computed from the raw frequency statistics of word i in the total number of word sequences for the particular corpus. W_j is the analogous value for each word j .

S_{ji} reflects the log likelihood ratio

$$s_{ji} = \ln\left(\frac{\Pr(j|i)}{\Pr(j|\bar{i})} \psi\right) \quad (3)$$

where $\Pr(j|i)$ is the conditional probability of word j occurring in the context of word i and $\Pr(j|\bar{i})$ is the conditional probability of word j occurring in a context that does not contain word i . Again, these conditional probabilities are computed from the raw concurrence frequencies available for the text database.

One nagging question is how well the spreading activation networks computed for each subject actually reflect their memory structure for words. Such an assumption would seem more justified if the text corpus were known to be representative of each users' past experience with text. We do not really know this for sure, although the sheer size of the corpus might be expected to mean that its word statistics were more reflective of the world of text than a smaller sample. The assumption might also seem justified if we knew that the amount of experience with the corpus that was observed in our studies was enough to enable the users to learn the underlying corpus statistics. Again we do not know this for sure, although analyses reported in Pirolli et al [12] suggest that people are learning quite a bit about the text corpus. Future work will be required to test the derived spreading activation networks against known word association statistics and to test the sensitivity of the model to deviations from the norms.

Results

Figures 4, 5, and 6 present results from the runs of the three kinds of evaluation functions. The log files used for comparison were from all $N = 8$ of the Scatter/Gather participants from Pirolli et al. [12] working on the Medium difficulty queries from the last half of the study when users had gained some experience with the system. In these data, there are 151 observed cluster-selecting actions. On each cycle, when clusters could be selected by productions, the ACT-R production system used the given evaluation function to rank the selection productions.

The model-tracer recorded the rank of the cluster-selection productions in the conflict set that actually matched the observations of users. A histogram of the rank of the productions that matched the observed actions is presented in the histograms in Figures 4, 5, and 6. The histograms can be interpreted as reflecting the probability that the predicted actions match the observed actions.

Each evaluation function appears to provide a good prediction of which clusters were selected by users. Particularly surprising is the finding that simple word overlap between the cluster digests presented on the screen and the query read by users, does a such a good job of predicting which clusters are selected for the higher ranked cluster-selection productions. One problem with this kind of evaluation, however, is that a very large percentage (56%) of clusters have zero overlap. Indeed, close inspection of Figure 4 reveals that the lower ranked cluster selection productions are more randomly selected (these are typically the just zero word overlap productions).

The interword correlations (Figure 5) or strength of activations (Figure 6) between query words and cluster digest words provide an nonzero evaluation of cluster-query comparisons for all clusters. Again, these evaluation functions also seem to provide good prediction of the clusters selected by Scatter/Gather users. From the histograms in Figures 4, 5, and 6 it is possible to compute the χ^2 statistics for the distribution of predicted actions from purely randomly selected. All the evaluation functions were significantly better than random at the $p < .0005$ level (for word overlap, $\chi^2(8) = 58.71$; interword correlations $\chi^2(8) = 39.29$, spreading activation, $\chi^2(8) = 31.00$). This first round of comparison suggests that either the interword correlation or spreading activation evaluation functions could be simply enhanced by giving extra weight to simple word overlaps.

One of the predictions of the information diet model discussed earlier is that the threshold for what is included in the diet increases as the environment of information becomes enriched. As more “good stuff” becomes available, one is more likely to ignore more “junk.” One’s “information diet” becomes richer but narrower.

Figure 7 suggests that this phenomena is occurring in Scatter/Gather. The data come from a model-tracing run using the spreading activation evaluation function. All of the Easy, Medium, and Hard queries for the $N = 8$ participants were analyzed to provide a broader sample of “information diets.” Each of the $N = 533$ points in the figure represents a cluster-selecting production that matched the log files. Each point plots an estimate of the *richness* of the available information and an estimate of the *threshold* for selecting clusters. The richness estimate is the average activation across all the clusters in the window on which the cluster was selected. The threshold is the maximum activation of the clusters that *were not* selected. This maximum could be considered a lower-bound estimate of the threshold for the information diet in the context of the clusters available on the screen. Clusters below that threshold were not selected. Figure 7 shows a very strong positive relation between estimated thresholds and the overall strength of activation of all the clusters on the screen.

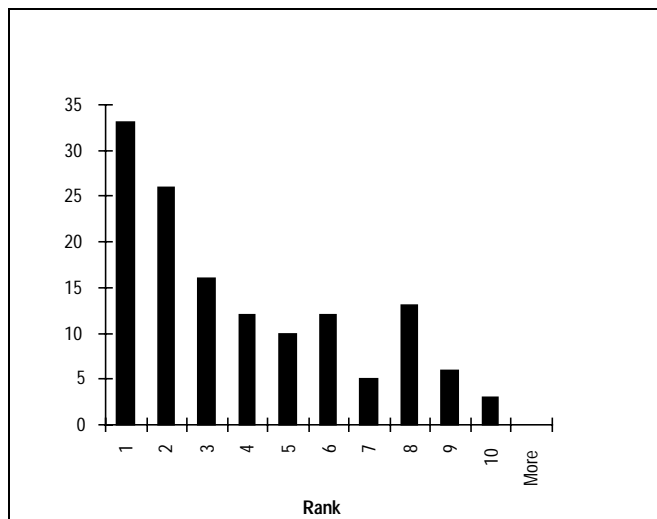


Figure 4. Match of observed actions to cluster-selection productions ranked by *word overlap*.

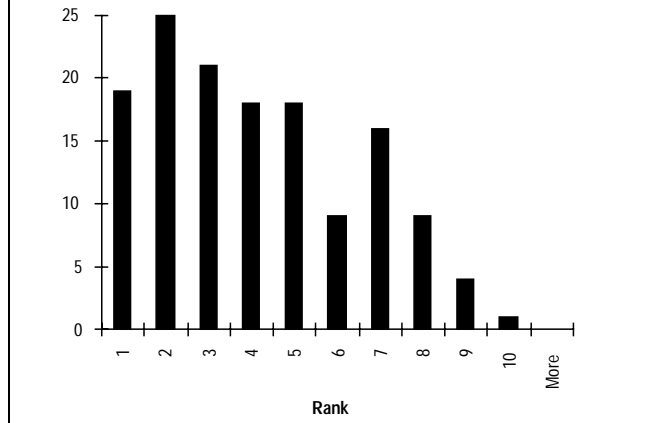


Figure 5. Match of observed actions to cluster-selection productions ranked by *interword correlation*.

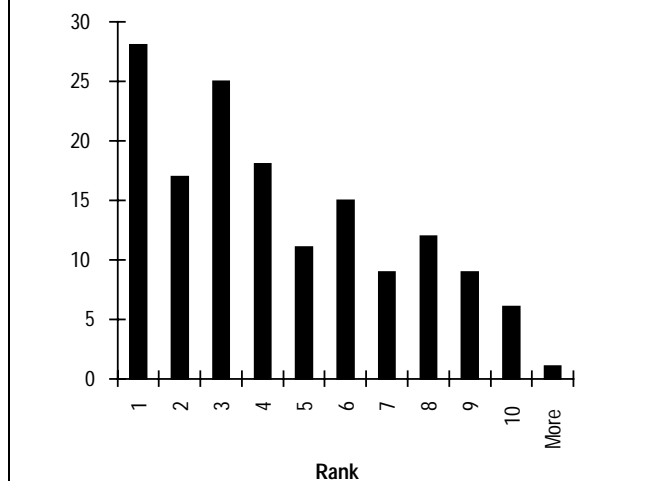


Figure 6. Match of observed actions to cluster-selection productions ranked by *spreading activation*.

DISCUSSION

This paper is an installment on a promissory note made in our earlier paper on information foraging [11]: That adaptation analysis will complement and guide models of human cognition. Here some small steps were taken in that direction by (a) developing a framework for analysis that spans levels from adaptationist analysis to cognitive analysis, (b) describing a model-tracing architecture for the intermediate levels of such analysis, (c) using this architecture to analyze the interaction protocols of users of a complex browsing system for a very large-scale text database. This paper does not go the full distance and fully propose a fine-grained mechanistic cognitive model for the data. That remains for future work.

The model presented here and the more general architecture could be used to evaluate other methods of providing information scent to the users. The cluster summaries used in the Scatter/Gather interface have a very prominent impact on user behavior, which validates the intuitions of its designers. To move beyond design by good intuition, however, requires some psychological insights as to how the gist of large collections can be communicated effectively in small amounts of space and time. Interestingly, there seems to be very little basic psychological research that is relevant here, and so there is much room for fundamental research on cognition as well in this area.

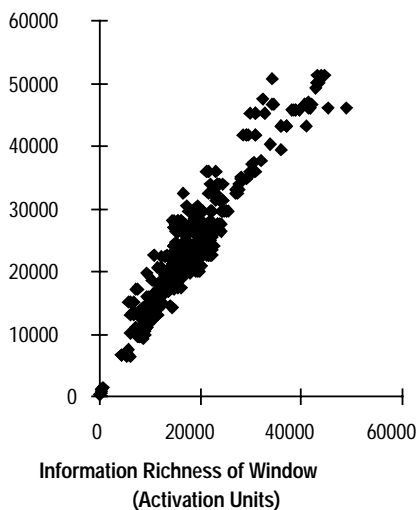


Figure 7. Estimated threshold for cluster selection as a function of overall cluster activation.

ACKNOWLEDGMENTS

This work is supported by an Office of Naval Research grant No. N00014-96-C-0097. Thanks to Christian Lebiere who provided valuable advice and solutions to problems arising in the modifications of ACT-R. Thanks

to Hinrich Schutze making the interword correlations available and for advice on their use.

REFERENCES

1. Anderson, J.R. (1993). *Rules of the mind* Hillsdale, NJ: Lawrence Erlbaum Associates.
2. Anderson, J.R., C.F. Boyle, A. Corbett, and M.W. Lewis. (1990). Cognitive modelling and intelligent tutoring. *Artificial Intelligence*, 42, 7-49.
3. Beard, D.V. (1996). Quick and dirty GOMS: A case study of computed tomography interpretation. *Human-Computer Interaction*, 11, 157-180.
4. Charniak, E. and D. McDermott. (1985). *An introduction to artificial intelligence* Reading, MA: Addison-Wesley.
5. Cutting, D.R., D.R. Karger, J.O. Pedersen, and J.W. Tukey. (1992). Scatter/gather: A cluster-based approach to browsing large document collections. In Proceedings of the *SIGIR '92* (pp. 318-329), ACM Press.
6. Harman, D. (1993). Overview of the first text retrieval conference. In Proceedings of the *16th Annual International ACM/SIGIR Conference* (pp. 36-38), Pittsburgh, PA: ACM Press.
7. Hopcroft, J.E. and J.D. Ullman. (1969). *Formal languages and their relation to automata* Reading, MA: Addison-Wesley.
8. Kieras, D.E., S.D. Wood, and D.E. Meyer. (1995). Predictive engineering models using the EPIC architecture for a high-performance task. In Proceedings of the *Human Factors in Computing, CHI '95 Conference* (pp. 11-18), Denver, CO: ACM Press.
9. Kitajima, M. and P. Polson. (1996). A comprehension-based model of exploration. In Proceedings of the *Human Factors in Computing, CHI '96 Conference* (pp. 324-331), Vancouver, BC, Canada: ACM Press.
10. Newell, A. (1990). *Unified theories of cognition* Cambridge, MA: Harvard University Press.
11. Pirolli, P. and S.K. Card. (1995). Information foraging in information access environments. In Proceedings of the *Conference on Human Factors in Computing, CHI '95* (pp. 51-58), Denver, CO: ACM Press.
12. Pirolli, P., P. Schank, M. Hearst, and C. Diehl. (1996). Scatter/Gather browsing communicates the topic structure of a very large text collection. In Proceedings of the *Conference on Human Factors in Computing Systems, CHI-96*. Vancouver, BC: ACM Press.
13. Schuetze, H. (1992). Dimensions of meaning. In Proceedings of the *Supercomputing '92* (pp. 787-796), Minneapolis, MN.

14. Smith, E.A. and B. Winterhalder, ed. *Evolutionary ecology and human behavior*. de Gruyter, New York, 1992.

15. Smith, J.B., D.K. Smith, and E. Kupstas. (1993). Automated protocol analysis. *Human-Computer Interaction*, 8, 101-145.