# USING SPREADING ACTIVATION TO IDENTIFY RELEVANT HELP

Adele Howe*
Timothy W. Finin
Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104

**Abstract:** *On-line assistance programs should have the ability to fulfill complex requests for information. We have built an assistance program for the Franz Lisp programming language in which users can enter multiple keyword queries in an unstructured form. The keywords are mapped into the semantic network database, and spreading activation is used to determine the object to be retrieved.*

*A many-to-many mapping between keywords and topics permits familiar words to refer to potentially unfamiliar and diverse topics; for example in Franz Lisp, the keyword 'add' is associated with CONS, APPEND and PLUS. A weighting scheme assigns a value for relatedness between keywords and objects, making ADD most closely related to PLUS. Activation is directed by assigning weights to the topics and to the classes of links between objects.*

## Introduction

On-line assistance programs are frequently included in interactive systems to make them easier to use. Assistance is generally initiated by an explicit request from the user, who enters a command like *help* or *man* [8]. One of the most common and frustrating problems with most conventional help systems is a variation on the old dictionary-lookup problem:

> *How can I look a word up in the dictionary to discover its spelling if I don't know how to spell it?*

Users of computer systems are often faced with a need to learn about some aspect of the system but do not know what information to ask for or exactly how to ask for it.

There are several possible approaches to this problem. One approach is to index information in the Help database by function terms that the user will have a good chance of knowing [8]. Another is to endow the help system with a model of its users which can be used to predict what information the user will need [4]. Still another approach is to develop a help system which the user can easily explore in a top-down manner to find the information he needs [1]. The information retrieval field [7] has a wide set of stategies for identifying possibly relevant items from a large data base.

In this research, we have followed the general approach of providing the user with a network of chunks of help text connected by a variety of syntactic and semantic links. The user can explore the network to seek the answer to a particular question or more generally browse through the network to discover new facts. One of the primary problems with such a network based help system is providing the user with a mechanism to find an appropriate place in the network from which to begin his exploration. Asking the user to employ a top-down search strategy from a root help node places a large burden on him when the network is large. We have provided a *keyword* access system which the user can use to identify relevant starting places in the network.

In some keyword access systems, the user describes the desired information by specifying a set of unique terms for commands or objects in the system. Accessing information using unique keywords, however, presupposes that the user knows the keywords and their corresponding topics in the system. To request information about the function that adds an atom to the front of a list in Lisp, the user would need to know that it is called *cons*.

Alternatively, using many keywords to refer to objects in the system causes confusion about what the user wants to know. For example, one can *add* numbers to numbers (e.g. Plus) or *add* atoms to lists (e.g., Cons). If the user asks for help about *add*, how does the help program determine what type of adding is of interest? We propose a technique for determining responses in help programs given a network database of help information and at least two keywords as input from the user.

## Description of HOW?

Our help facility, HOW?, provides textual information as descriptions, examples, and errors about a subset of the language and programming environment of Franz Lisp. The information is organized in a network, where the nodes are textual descriptions and the named links (*subtopic*, *related topic*, *supertopic*, *example-of*, *errors-from*, *details-about*) are the relations between the concepts that the descriptions represent. The system is entered from Lisp via the function HELP along with any number of keywords. Further information is accessed by choosing from a menu of associated topics, executing designated help commands, or entering a list of one or more keywords in an unstructured form.

## Translation of User Request to Program Response

Using a string of words to describe a topic seems to be the most natural method for a human. Given at least two words that refer to concepts in the database and the network configuration, a node can be selected for presentation by using spreading activation [2].

Spreading activation theory models human memory retrieval as activation energy spreading from input nodes across links in a semantic network to an intersection [2].

* Address is now ITT Advanced Technology Center, Shelton, CT 06484

Applied in the help network database, this theory provides the basis for retrieving information from the database in response to complex, multiple word queries.

Abstractly, our version of this technique involves spread-to-limit from a starting point of a list of keywords, which refer to nodes. In spread-to-limit, the activation is divided among the initial nodes, multiplied by an attenuator or *spread-decay* value and spread to adjacent nodes (and spread to their adjacent nodes and so on) until the activation is below the *spread-limit*, a threshold value that defines negligible activation levels.

Description of the Algorithm

The algorithm relies on the weighting of the keyword types and the meaning of the links for distributing the activation. Keywords can be of three types, based on how closely they describe their topic. Synonomous or unique keys are given a weight of three, keys that describe a topic very well a weight of two, and secondary or loosely descriptive keys a weight of one. Each type of link (supertopic, subtopic, related-topic, details-of, errors-of, examples-of) also has a weight associated with it. These weights are determined by a combination of the designer's intuition of their relative importance and empirical testing.

The parallel search is simulated by maintaining a queue of active nodes (nodes with activation to spread further) and maintaining two activation levels per node. The two levels are *temp-level* and *activation-level*. *Temp-level* is the level of activation that the topic has received since the last time it spread activation to other nodes. *Activation-level* is the total amount of activation that has been accumulated by it during the current retrieval.

The process is started by extracting the keywords from the user request and setting the initial activation levels of the appropriate topics. The starting network activation of 1.0 units is divided evenly among the input keywords (words from the request that correspond to nodes in the network). The initial keyword activation level is then distributed among the topics, referred to by the keyword, by summing their weights (three, two, or one) and distributing the keyword's initial weight between the topics by percentage of total weight. A symbolic form of the equation used appears in figure 1. The topics are added to the active queue, and normal cycling is begun.

---

Given: KEYS= $K_1$, $K_2$, $K_3$...$K_M$
which map to nodes $N_1$, $N_2$, $N_3$...$N_S$
with values V of {3, 2 or 1}

$$activation/node = \frac{V_j}{\sum V} * W_i$$

where $V_j$ = value associated with each node
$W_i$ = *activation/key* = $\frac{1}{M}$

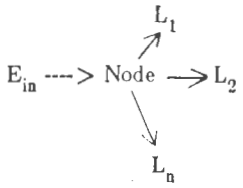
**Figure 1:** Formula for Initial Activation Levels

---

In a normal cycle, the next node on the queue is read. Its *temp-level* is multiplied by an attenuation factor. The attenuation factor is much like the resistance of the links to having energy spread through them; a weak link, such as *errors-from*, has a high resistance and so allows less energy to pass. The attenuation factor reduces the influence of the initial activation over time/distance. If the *temp-level* times the attenuator is less than the *spread-limit*, then no activation will be spread.

The attenuated activation is divided among the associated nodes according to their percentage of the total weight. The weights of the links are summed, and the activation to be spread to each is the input activation multiplied by the weight of the link and divided by the sum of the weights. This amount is added to both the activation-level and the temp-level for the associated nodes, which are pushed onto the queue. Finally, the node just processed is removed from the queue. This formula is in figure 2.

---




where
A = attenuation factor
W = weight of each link
$E_{in}$ = temp-level for the node

$$E_{out} \text{ for each } L_i = \frac{W_i}{\sum W} * E_{in} * A$$

where $(E_{in} * A) >$ *spread-limit*

**Figure 2:** Formula for Spreading Activation During Cycling

---

Cycling continues until there no nodes are left on the queue. The *spread-activation* function returns the list of topics, that have been activated, sorted according to their activation-levels. The highest ranking candidate on the list is the topic to be retrieved, with other topics offered to the user as alternatives. A more complex alternative selection scheme would involve choosing alternatives relative to the highest. For example, if the first topic's activation is considerably higher than any other's, then only this topic would be offered. The alternates are printed on the screen for the user's reference and can be accessed by use of a built-in command.

Results and Conclusions

The technique has been tested with a series of multiple keyword requests. Because the system discards irrelevant keywords, pseudo-natural language input is possible, but not necessary. Figure 3 demonstrates a few test cases input to the spreading activation functions. The requests were chosen from the portion of the database that is the most complete.

One benefit of this technique is that a request which seems appropriate to a number of related topics will usually suggest a suitable general node from which the user can explore. This stems from the organization of the network along *supertopic* and *subtopic* links.

```
How do I add an atom to a list?
highest ranking candidate: APPEND
alternatives: APPEND1 CONS List-data-type CAR/CDR

How do I add an atom to the front of a list?
highest ranking candidate: CONS
alternatives: APPEND1 CAR/CDR APPEND List-data-type

How do I add an atom to the back of a list?
highest ranking candidate: APPEND1
alternatives: CONS CAR/CDR APPEND List-data-type

How do I add two lists together?
highest ranking candidate: APPEND
alternatives: APPEND1 CONS List-data-type CAR/CDR
```

**Figure 3:** Spreading Activation Test Cases

The use of spreading activation with the weighted keyword scheme shows promise as a method for processing complex queries to a help program without developing a natural language interface. Because the database structure and keyword set impact the result of spreading activation searches, the database must currently be hand coded. Current research is investigating ways of automating the database construction process to confront this issue.

References


[1] Acksyn, Robert M., McCracken, Donald L.
*The ZOG Project.*
Technical Report, Dept. of Computer Science, Carnegie-Mellon University, June, 1982.

[2] Collins, Allan M., Loftus, Elizabeth F.
A Spreading-Activation Theory of Semantic Processing.
*Psychological Review* 82(6), 1975.

[3] Findler, Nicholas V., Ed.
*Associative Networks: Representation and Use of Knowledge by Computers.*
Academic Press, N.Y.,N.Y., 1979.

[4] Finin, T.
Providing Help and Advice in Task Oriented Systems.
In *Proc. 8th Int'l. Joint Conf. on Art. Intelligence.* IJCAI, August, 1983.

[5] Foderaro, John K., Sklower, Keith L.
*The FRANZ LISP Manual*
University of California, 1981.

[6] Howe, Adele.
*HOW? A Customizable, Associative Network Based Help Facility.*
Technical Report MS-CIS-83-14, Computer and Information Science, Univ. of Pennsylvania, 1983.

[7] Salton, Gerard, Ed.
*The SMART Retrieval System: Experiments in Automatic Document Processing.*
Prentice Hall, Englewood Cliffs, N.J., 1971.

[8] Sondheimer, Norman K., Relles, Nathan.
Human Factors and User Assistance in Interactive Computing Systems: An Introduction.
*IEEE Transactions on Systems, Man and Cybernetics* SMC-12(2), March/April, 1982.