PERSONALIZABLE DIRECTORIES: A CASE STUDY IN AUTOMATIC USER MODELLING

Ian H. Witten, Saul Greenberg and John Cleary

Man-Machine Systems Laboratory, Department of Computer Science
The University of Calgary, 2500 University Drive NW, Calgary, Canada T2N 1N4

Abstract

A personalized directory system is presented which demonstrates how automatic user modelling can reduce the average number of menu-based selections needed to retrieve entries from a large ordered list, such as a telephone directory. It offers a simple case study for pilot human factor experiments in automatic user modelling; it affords the possibility of a device independent menu-selection system; and it gives a technique for personalizing cheap, mass market databases. A number of difficult issues in automatic user modelling are raised, such as the user's perception of a dynamic system, the need of user control and the measurement of user satisfaction. The personalized directory system is being used as a test-bed to examine such issues, and will refine our insight into the human factors implications of user modelling.
KEYWORDS: User modelling, device independence, human factors, menus, directories

Introduction

This work examines a menu-driven selection process for retrieving entries from a large ordered list such as a telephone directory. It pre-supposes that the user possesses a machine-readable directory, say that for the city which he inhabits, which is distributed en masse and is in no way personalized. Menu-based schemes for retrieving items from the directory are devised and assessed; a full alphabetic keyboard and typing ability are not assumed. A key feature of the schemes is that they store information about accesses which have been made and so construct a user profile. This profile is used to provide rapid access to popular entries by cutting down the number of menu pages which must be scanned to retrieve these items.

This paper is concerned with a presentation system which displays a range of names as each menu item. The first-level menu divides the name space into regions, the number of which is given by a pre-defined "menu size" parameter. One of these regions is selected by the user at the first level. The system then takes the indicated range and calculates a new, second-level menu by splitting it into sub-ranges. A further selection is made, and the procedure repeats. Eventually, some of the menu items will be single names rather than regions, and when one of these is selected the search terminates.

What is novel about the scheme is that the regions are selected at each stage not to cover an approximately equal range of names, but rather to divide a probability distribution, defined on the name space, into approximately equal portions. The probability distribution is updated at the end of each selection process to reflect a "popularity" measure on the names. When the system is loaded in an unprimed state all names have equal popularity. During use, the act of selection will increase the popularity of selected names. Thus the user will be directed more quickly to names which have already been selected -- especially if they have been selected often and recently -- than to those which have not. Indeed, depending on the menu size and the pattern of selections, it may be that names sought frequently appear directly on the first-level menu; and if not, they may almost certainly appear on the second-level one.

The price to be paid for this personalization is that novel names must obviously take longer to select than they would otherwise. However, the penalty can be made very small if suitable algorithms are used for probability updating.

Importance

There are three aspects of the study which make it particularly timely and relevant to current research problems in human-computer interaction. Firstly, it provides a simple and explicit example of automatic user modelling. The extreme simplicity of the directory task -- retrieving items from an ordered list -- makes

it possible to study the user modelling problem in a lifelike situation but uncluttered by unnecessary detail. Simple human factor pilot studies can then indicate if automatic user modelling is a help or a hindrance. Ultimately we hope to apply the knowledge gained from these experiments to more complex tree-structured consumer databases such as those in videotext systems.

The second important aspect of the scheme is that it affords the possibility of a device-independent menu-selection system. To support the developing user model, menu pages need to be computed on demand from the underlying database, rather than being stored explicitly as in most menu systems. The computation can therefore take into account the size of the display and selection device at enquiry time -- an ability that is not possible with pre-stored menu pages. Several recent developments make this ability attractive. We are beginning to see a proliferation of high-resolution, full-page, display screens. VDUs with many more lines than the standard 24 are being produced, as are domestic information systems with less display capacity. Sophisticated systems now often use windows on the screen whose size is completely unknown until run time. With device-independent menu techniques, these can all be served by the same program. Moreover, the ability to work with M-way selections for arbitrary M allows the widely differing needs of handicapped people (Raitzer et al, 1976) to be accommodated easily and naturally.

Thirdly, the work seems to be particularly timely in view of current trends in the economics of information processing. Storage has become a cheap resource, but the cost of producing and maintaining individual databases is still high and increasing. The price of store filled with data is completely dominated by the expense of preparing that data. Mass-marketing techniques, which amortize the development of a resource over its many users, are already beginning to appear in the form of consumer information systems and current-affairs databases. It seems probable that methods for personalizing cheap, mass-marketed databases will have a major role to play in the future development of information technology -- especially if these methods are automatic and require no explicit action by the user.

An overview of user modelling

Although the case study is an important and interesting venture in its own right, the main thrust of the present work is to illuminate some of the basic issues in user modelling. The personalizable directory is intended as a testing ground for ideas and theories, and as a vehicle for their evaluation through human factors pilot studies.

Innocent (1982) has categorized modelling strategies into three different types. The first encompasses the frequently used strategy of having a system designer listen to feedback from the user population and adjust the system to fit the current need. Unfortunately, this views the user as a canonical (typical) user which cannot be an accurate view of a highly heterogeneous community (Rich, 1983). Edmonds (1982) suggests that our knowledge about human behaviour is inadequate to portray correctly a canonical user, especially one whose needs will change over time.

The second class of user modelling strategies contains those which encourage the user himself to modify his own working environment. Many authors have referred to the need for a user to be able to tailor a system to his individual abilities, tastes, and preferences. An example is a facility for user-definable abbreviation, available in many systems. Unfortunately, there are disadvantages to this. Naive and casual users may have a difficult time learning the necessary strategy for modifying their environment -- and these are the users that may benefit from this the most. In addition, there is a trade-off between the setup overhead the user is willing to accept versus the work he wishes to accomplish (Rich, 1983). Some elegant techniques have been developed which allow the user to speed up his interactions in very natural ways; examples are the parallel/sequential trade-off of Gaines and Facey (1975), where the knowledgeable user can hasten the dialogue by entering different items of information in the same line and so avoid the need for individual prompts; and the prompt-suppression feature of Witten and Madams (1977), which allows the user to group his entries into chunks which are perceived as a whole.

Finally, there are automatic implicit modelling strategies, in which the system monitors the user's activity and automatically tries to adapt itself to his model. This is the least used, and potentially perhaps the most powerful, of the three types of strategy. One of the few examples of systems which incorporate this kind of modelling is the predict system (Witten, 1982), which predicts the user's future keystrokes on the basis of the redundancy which has been exhibited in his input so far. The work described in the present paper is a new example of the possibilities opened up by automatic implicit user modelling.

An example of a personalized telephone directory

Imagine looking up a telephone number in a directory with N entries, using a menu with M items and a M-key keypad to select the menu item. Typical values are N=250,000, for a city with a population of 600,000; and M=16 for a 16-key pad.

We will consider a presentation system which displays a range of names as each menu item; and example of the first-level menu is shown in Figure 1. This divides the name space into M regions, one of which is identified by the first-level selection. The system will then take the indicated range, split it into M equal parts, and await a further selection. Eventually, some or all of the ranges will have been narrowed down into unique entries, and the final selection will indicate which name is sought. There are some obvious problems here. For example, the beginning and end of each menu range need not be presented on the screen, and doing so -- as in the Figure -- will likely delay the user for he has more information to scan. Furthermore, name ambiguities, when the address will have to be shown to allow disambiguation, will certainly occur. If several people have the same name and initial, the system may have to resort to a menu, or sequence of menus, showing all those people, because the address field may not be listed in any specific order in the directory.

Presentation details are an important concern for human factors evaluation. A display which is difficult to scan rapidly will give the user a slow selection time per menu. If the automatic system were then measured against a static control which has a fixed and memorizable pathway, the scanning delay will be prejudicial against the personalized system. Preliminary experiments are planned to find the display which permits the directory to be scanned at maximum speed. Although these problems affect the presentation and testing details, they will not affect the algorithm for menu construction, and are thus ignored for the rest of this discussion.

```
 1   A Abode Ltd  --  Austin Cal
 2   Austin D  --  Bracal Mary
 3   Bran W  --  Church of Pentecostal
 4   Church of Prophecy  --  Dilling Ltd
 5   Dillman A  --  Frankies Flowers
 6   Frankiewicz S  --  Handfield E
 7   Handfield RS  --  James R M
 8   James R N  --  Lang J E
 9   Lang J N  --  Martin J A
10   Martin J B  --  Mt Royal College
11   Mt Royal Day Care  --  Peters Earl
12   Peters Elwood  --  Roth Energy Ltd
13   Roth F  --  Sooner Pipe Ltd
14   Soong Johg  --  True West Land Ltd
15   Truefitt C S  --  WestCan Graphics
16   WestCan Office Ltd  --  Zywoto A P
```

Figure 1. Example first-level menu

Uniform selection. With M menu items, each selection reduces the range of names to 1/M'th of its former value. Hence with N names the process will terminate after

$$k_u = \frac{\lceil \log N \rceil}{\lfloor \log M \rfloor} \quad \text{selections,}$$

or 5 with our example values.

In most cases the act of selection provides information about items that are likely to be selected in the future. As far as retrieval is concerned, this information can be modelled in terms of a non-uniform distribution over the name space. (The discussion above assumed a uniform distribution where the probability of selecting any particular name is 1/N). Instead of splitting the name space into M equal parts at each stage, it is the probability distribution that is split equally. This means, for example, that any name whose probability has grown to over 1/M will be present, in a range of its own, on the first-level menu.

Bimodal selection. Suppose you have F "friends", who have each been accessed the same number of times. The remaining entries, N-F, have never been accessed. According to some algorithm for probability updating, which will be discussed later, suppose that the probability associated with each friend is p, and that of each non-friend is q, with

$$Fp + (N-F)q = 1.$$

Now it is not the name space but the probability range which is reduced to 1/M of its former value by each menu selection. Thus to access a friend will take k_F selections, where

$$k_F = \left\lceil -\frac{\log p}{\log M} \right\rceil .$$

It will be easy to calculate the number of selections to access a non-friend. Under the reasonable assumption that F << N, and the somewhat more debatable one that Fp << 1; the expression reduces to

$$k_n = \frac{\log N - FN^{-1} + Fp}{\log M}$$

What about the value of p? Suppose that we wish to reduce the selection count for friends to half its value for a uniform distribution,

then it follows that $p = N^{-1/2}$. Plugging in numerical values, with 100 friends, leads to a selection count of just over 2 for friends and only 1% above 4.5, the uniform distribution value, for non-friends.

It is important not to let a predominance of "friend" accesses force the selection count to grow to unacceptably large values for names which have not yet been accessed. Suppose, for example, that at most $2k_u$ selections are to be necessary. This is easily achieved by ensuring that all probabilities in the distribution exceed N^{-2}.

Menu-splitting algorithms. As another part of this research project, we have investigated menu-construction algorithms for accessing items from an ordered dictionary whose frequency profile of access is specified (Witten & Cleary, 1983). The aim is to minimise the average number of selections. Despite the apparent simplicity of this problem, optimal menu construction is surprisingly difficult. If the dictionary entries are accessed uniformly, theoretical analysis leads to a selection algorithm different from the obvious one of splitting ranges into approximately equal parts at each stage. Analysis is intractable for other distributions, although the performance of menu-splitting algorithms can be bounded. The optimal menu tree can be found by searching, but this is computationally infeasible for any but the smallest problems.

Several algorithms for selecting items from an ordered dictionary with specified probabilities have been investigated. They all involve the selection of ranges of dictionary entries from a succession of menus, the range narrowing progressively until it contains just the desired item. They differ in their treatment of rounding in the menu-splitting process, and lead in general to quite different menu trees.

In the case of uniformly-distributed dictionary entries, which is amenable to theoretical treatment, the simplistic method of splitting each range into as many parts as there are menu items was found to be inferior -- often grossly inferior -- to a method which instead maximizes the utilization of leaves in the menu trees. For other distributions, theoretical analysis is intractable. Three quite different algorithms have been investigated by computer simulation with a Zipf distribution access profile. This distribution approximates word usage statistics in natural language (Zipf, 1949), and forms a good model of some more artificial phenomena (Peachey et al, 1982). The performance of the three algorithms is remarkably similar, despite the fact that our limited experience with the optimal menu tree suggests that they leave some room for improvement. Until better methods are found, however, the simple process of iteratively splitting the probability range at each stage into regions whose probabilities are as similar as possible appears to be as good as any.

Algorithm for updating probabilities

Now we need to consider algorithms for updating the probabilities when an item is retrieved. Separate methods are being investigated for bimodal and Zipf distributions. Only the former will be discussed here.

The "friends"/"non-friends" bimodal situation was introduced above, in the context of retrieval. We have designed an algorithm for updating probabilities which

- -- ensures that no entries require more than a preset maximum number of selections;
- -- when a non-friend is accessed, making him a friend, increases immediately the probability associated with him to take account of the fact that he is now much more likely to be accessed again;
- -- ensures that the probability associated with friends who are not accessed decays slowly.

The easiest way to satisfy the first point is to decide in advance the maximum number of selections which can be tolerated for non-friend accesses. Suppose this is μk_u, where, as before, k_u is the selection count for a uniform distribution, and μ is a constant greater than one. Then non-friend probabilities are maintained at $N^{-\mu}$ ensuring that non-friend accesses take μK_u selections (on average). Of course, an exception must be made at the beginning when there are no friends; perhaps by setting probabilities to N^{-1} initially and changing them to $N^{-\mu}$ as soon as the first person is accessed.

If there are F friends this leaves the total probability weight assigned to friends as

$$1 - (N-F)(N^{-\mu}).$$

This ensures an average selection count for friends which is significantly less than k_u if F is small compared with N.

The second condition is easily satisfied, when a new friend is made, by giving him the average probability over all existing friends

Keep a list of pointers to friends f who have already been accessed, together with a weight w(f) for each in the range [0,1].
The update algorithm will ensure that the weights sum to 1 over all friends.

Retrieval.

Use the following probabilities:

if F=0, probability for each name is $\frac{1}{N}$

otherwise,

if friend, probability is $\frac{1}{N^{\mu}} + w(f)(1 - \frac{N}{N^{\mu}})$

else probability is $\frac{1}{N^{\mu}}$.

Update.

If a new friend has been made, put him on the friend list and update F. Then

initialize his weight to $\frac{1}{F}$

decrease all other weights by $\frac{1}{F(F-1)}$.

If an old friend f has been accessed again,

$w(f) \leftarrow \alpha w(f) + 1-\alpha$

$w(g) \leftarrow \alpha w(g)$ for any other friends $g \neq f$.

In either case, check if any $w(f) \leq \frac{1}{N}$; if so

delete f from the friend list, decrementing F

increase weights of other friends g by $\frac{1}{F} w(f)$.

Figure 2. Retrieval and update algorithms

and decreasing their probabilities by the appropriate amount to ensure conservation of total probability.

The third condition requires some kind of adaptation algorithm on the probabilities, when existing friends are accessed. When a friend f is accessed, a procedure like

$p_f \leftarrow \alpha p_f + 1-\alpha$

$p_g \leftarrow \alpha p_g$ for friends $g \neq f$

appears to be suitable. This conserves probability over friends alone is equal to one; which of course will not be the case due to the non-zero probability assigned to non-friends. A simple implementation maintains values for friends alone, which sum to one; and make an appropriate adjustment to the values before they are actually used a probabilities in the retrieval algorithm. Figure 2 shows details of suitable retrieval and update algorithms.

The parameter α is a measure of how quickly friendships fade away. α should be chosen slightly less than one; the closer it is to one the slower friendships will fade.

It is necessary to check when decrementing a friend's probability that it has not become less than that of the non-friends. Indeed it may be desired to avoid gradual growth of the friend list over a long period of time. This can be done by removing friends whose probabilities have fallen below a certain value (say N^{-1}), and making appropriate adjustments to all other friends' probabilities.

Issues in the evaluation of automatic user modelling

The personalizable directory is a case study, which, while interesting in its own right, is primarily intended as a vehicle for examining problems and clarifying issues in the area of automatic user modelling. This

section discusses these problems and outlines a proposed pilot experiment which will study some of the issues raised.

How does one measure the success of a system which attempts automatically to model the user and adjust itself to his actions? If users were asked informally to evaluate a system, they would probably respond in terms of its ease of use and whether they felt "comfortable" with it. What scientific yardsticks can be used to measure this? Perhaps none exist, for although people do have measurable parameters, the notions of ease and comfort probably differ drastically from person to person. Arguments certainly flourish in coffee rooms about issues such as ease of using programming languages, different terminals, editors, and word processors.

A case which illustrates the difficulty of evaluation occurred during a human factors investigation of the predict system (Witten, 1982). An experiment showed that a number of subjects took longer to enter a certain piece of text using predict than they did without it. Nevertheless, an overwhelming majority of them claimed to like the system, and thought that it was actually speeding up entry. What conclusions should be drawn from this experiment? Should more attention be paid to the objective measure or to the subjective preference of subjects?

A somewhat different area of concern is that users will always construct their own mental model of systems with which they interact (Gained and Facey, 1975). Most existing software interfaces are quite static, so that a time-invariant model will suffice. However, automatic user modelling presents the user with a continuously changing system. Presumably it is easier for the user to model a static system than a dynamic one, and adaptive techniques may cause him untold confusion by confounding his predictions about its behaviour. On the other hand, people react in the real world to a constantly changing environment, usually with some success. It seems likely that users will be able to construct satisfactory mental models of adaptive interfaces provided that they possess consistent properties which allow users to anticipate, or at least rationalize, the changes.

Yet another issue is who has control over the dialogue. Usually, either the system or the user directs the communication. Recent studies (Gained and Shaw, 1983) have recommended that the user be in control, although the system may provide him with some overall direction. Does a constantly changing environment detract from user control? If so, mechanisms may have to be added to allow him to explicitly alter the system's model of him.

Simple human factor pilot studies are planned to begin to address some of the issues mentioned above. Initially, we plan to measure parameters such as keystroke time and error rate on the personalizable directory system. As a control, a non-adapting menu-based retrieval system will be used. This will allow several hypotheses to be investigated, such as

-- adaptation does/does not affect the time taken to make individual menu selections
-- adaptation does/does not affect the mean access time to items
-- adaptation does/does not affect error rates.

Perhaps more interesting than these measurable issues, however, will be users' informal reactions to the changing environment in which they find themselves.

Summary

We have demonstrated how automatic user modelling in a sequential directory can reduce the average number of selections needed to retrieve entries. The method described has the important advantage of being applicable to displays of any size, and, indeed, displays whose size changes within the retrieval process. This is useful in a dynamic window-based computer interface. There is room for research into practical selection algorithms which achieve the theoretical, entropy-based, minimum number of selections without excessive searching (Witten & Cleary, 1983). However, simple methods come quite close to realizing the full potential of the method for reducing selections.

The selection method requires an estimate to be available of the frequency profile with which items are accessed. Such estimates are complicated by the fact that the distribution changes with time, and that the information available is somewhat sparse in relation to the large number of possible directory accesses. We have given an example of a plausible estimation technique for a bimodal distribution of friends and non-friends.

This simple personalizable directory scheme raises a number of difficult issues in automatic user modelling. These include the user's perception of a dynamic system, the need of user control, and the measurement of user satisfaction. We are presently using the directory scheme as a test-bed to examine such issues through simple

pilot experiments. Afterwards, it is planned to expand the system to include explicit modelling techniques. Through these studies we hope to gain some insight into the human factors implications of automatic user modelling, and to develop meaningful guidelines to supplant previous wholly intuitive design methodologies.

References

Edmonds, E. (1982) "The man-computer interface: a note on concepts and design", Int. J. Man-Machine Studies, 16 (3) 231-236, April.

Gaines, B. R. and Facey, P. V. (1975) "Some experience in interactive system development and application", Proc Institute of Electrical and Electronic Engineers, 63 (6) 894-911, June.

Gaines, B. and Shaw, M. L. (1983) "Dialog Engineering" in Computers and People series, edited by Coombs and Simon. Academic Press, London, in press.

Peachey, J. B., Bunt, R. B., and Colbourn, C. J. (1982) "Bradford-Zipf phenomena in computer systems", Proc Canadian Information Processing Society National Conf, 155-161, Saskatoon, Saskatchewan, May.

Innocent, P. R. (1982) "Towards self-adaptive interface systems", Int. J. Man-Machine Studies, 16 (3) 287-299, April.

Raitzer, G. A., Vanderhelden, G. C., and Holt, C. S. (1976) "Interfacing computers for the physically handicapped -- a review of international approaches", Proc AFIPS National Computer Conference, 209-216.

Rich, E. (1983) "Users are individuals: individualizing user models", Int. J. Man-Machine Studies, 18.

Witten, I. H. and Madams, P. H. C. (1977) "The Telephone Enquiry Service: a man-machine system using synthetic speech", Int. J. Man-Machine Studies, 9 (4) 449-464, July. Reprinted in Larson, J. A. (ed.) (1982), End user facilities in the 1980's. IEEE Computer Society Press, NY, pp. 73-88.

Witten, I. H. (1982) "An interactive computer terminal interface which predicts user entries", Proc IEE Conference on Man-Machine Interaction, 1-5, Manchester, England, July.

Witten, I. H. and Cleary, J. (1983) "On frequency-based menu-splitting algorithms", Research report, Department of Computer Science, University of Calgary.

Zipf, G. K. (1949) Human behaviour and the principle of least effort. Addison-Wesley, Ontario.