Genetic Algorithm Can Optimize Hierarchical Menus

Shouichi Matsui
SERL, CRIEPI
2-11-1 Iwado-kita, Komae
Tokyo 201-8511, Japan
matsui@criepi.denken.or.jp

Seiji Yamada
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda
Tokyo 101-8430, Japan
seiji@nii.ac.jp

ABSTRACT
Hierarchical menus are now ubiquitous. The performance of the menu depends on many factors: structure, layout, colors and so on. There has been extensive research on novel menus, but there has been little work on improving the performance by optimizing the menu's structure. This paper proposes an algorithm based on the genetic algorithm (GA) for optimizing the performance of menus. The algorithm aims to minimize the average selection time of menu items by considering movement and decision time. We show results on a static hierarchical menu of a cellular phone where a small screen and limited input device are assumed. Our work makes several contributions: a novel mathematical optimization model for hierarchical menus; novel optimization method based on the genetic algorithm (GA).

Author Keywords
Hierarchical menu, optimization, genetic algorithm.

ACM Classification Keywords
H.5.2 [Information Interfaces and Presentation]: User Interfaces — Interaction styles, Screen design.

INTRODUCTION
Hierarchical menus are one of the primary controls for issuing commands in GUIs. The performance of the menu depends on many factors: structure, layout, colors and so on. There has been many studies on novel menus (e.g., [2, 3, 8]), but there has been little work on improving the performance of a menu by changing its structure [1, 6, 11]. There have been many studies on menu-design and menu-layout from the standpoint of the user interface. Francis et al. were the first to optimize a multi-function display, that was essentially the same as the hierarchical menu by using Simulated Annealing (SA) [6]. Liu et al. applied a visual search model to menu design [11]. They used the Guided Search (GS) model to develop menu designs. They also used an optimization algorithm to minimize the predicted search times according to predefined search frequencies of different menu items. Amant et al. showed the concepts to support the analysis of cell phone menu hierarchies [1]. They proposed a model-based evaluation of cell phone menu interaction, gathered data and evaluated three models, Fitts' law model, GOMS, and ACT-R. They concluded that the prediction by GOMS was the best among the three models. They also tried to improve menu traversal time by using a simple best-first search algorithm, and got over 30% savings in traversal time. These very simple search methods gave fairly good improvements [1, 11], therefore, we can expect further performance improvements by optimizing the structure with a better algorithm.

This paper proposes an algorithm based on the Genetic Algorithm (GA) [7] for optimizing the performance of menus. The algorithm aims to minimize the average selection time of menu items by considering movement and search/decision time. We will show results on a static hierarchical menu of a cellular phone where a small screen and limited input device are assumed.

FORMULATION OF THE PROBLEM

Overview
The optimization problem of hierarchical menus can be considered as one dealing with placing menu items on the nodes of a tree. Let us assume a tree where the maximum depth is D, the maximum number of children that a node has is W, the root is the initial state, and menu items are on nodes. An example of a hierarchical menu is shown in Fig. 1. As shown in the figure some menu items have children; i.e., some menu items have submenus. The time to select the target item is the time to traverse from the root to the target node. The problem is to minimize the average traversal time with respect to the given search frequencies of different menu items.

We cannot arbitrarily arrange the menu purely for efficiency. We must respect the semantic relationships between the items. That is, "Ringer Volume" is under the "Settings" category rather than vice versa for good reason. To cope with the difficulties of representing and reasoning about menu item semantics we introduce two metrics: *functional similarity* and *menu granularity*. Functional similarity is a metric that represents the similarity of two menu items in terms of their functions. We assume that the functional similarity takes a value between 0 and 1; 0 means that the two items have no similarity and 1 means that the two items have very high similarity. For example it is very natural to assume that "Create

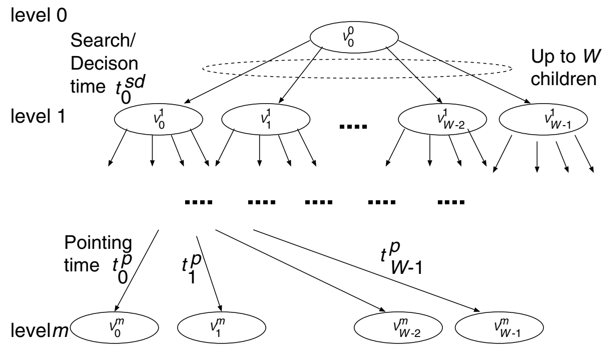

Figure 1. Tree structure of a hierarchical menu.

New Mail" and "Favorite Web Site" have low similarity and that "Create New Mail" and "Inbox of Mail" have high similarity. We use this metric to avoid placing items with low similarity on the same submenu of a node. If items with low similarity are put on the same submenu, it becomes difficult for a user to remember the menu layout. The formal definition will be given later. Menu granularity is a metric that reflects the number of submenus a node has as its descendants. We introduce this metric to avoid placing an item that has many children and an item that has no child as children of the same node. The formal definition will be given later.

Formulation

Notation

Let l be the level number, i is the ordering number in siblings, and v_i^l be a node of a tree (Fig. 1). Moreover, let $M = (V, E)$ be a tree where $V = \{v_i^l\}$ denotes the nodes, and $E = \{e_{ij}\}$ denotes the edges. We call the leaf nodes that correspond to generic functions "terminal nodes." We also assume that the selection probability of the terminal node/generic function is represented by P_i.

There are two kinds of menu item or node in the tree M. One type is terminal nodes that corresponds to generic functions, and the other is intermediate nodes. The terminal nodes cannot have children. We denote terminal nodes by g_i and intermediate nodes by s_i. We also denote the set of g_i by G and the set of s_i by S. Let I_i represent a menu item and the total number of items be N; i.e., there are $I_i (i = 1, \cdots, N)$ menu items. Items that correspond to generic functions are less than N, and some items/nodes are intermediate items/nodes that have submenu(s) as a child or children. We assume that a menu item I_i is assigned to a node v_i^l; therefore, we use I_i and v_i^l interchangeably.

Selection Time

The selection time t_i^l of a menu item/node v_i^l on the hierarchical level l can be expressed using the search/decision time t_i^{sd} and the pointing time t_i^p as $t_i^l = t_i^{sd} + t_i^p$ [4]. We also consider the time to reach level l; therefore, the whole selection time T_i of a node v_i^l on level l can be the sum from the root node to the final node Thus, the average selection time T_{avg} is $T_{avg} = \sum_{i=1}^{N} P_i T_i$.

Pointing Time and Search/Decision Time

As Silfverberg et al. [12] and Cockburn [4] reported, the pointing time t_i^p can be expressed as $t_i^p = a + b \log_2(A_i/W_i + 1)$ by using the Fitts' law.

We assume that the search/decision time t_i^{sd} at level l node that has n^l items can be expressed as reported by Cockburn et al. [4]

- For an unexperienced user, the time required for a linear search, $t_i^{sd} = b^{sd} n^l + a^{sd}$.
- For an expert, we can assume that the time t_i^{sd} obeys Hick-Hyman's law, $t_i^{sd} = b^{sd} H_i + a^{sd}$ where $H_i = \log_2(1/P_i^l)$ and P_i^l is selection probability.

Functional Similarity

Toms et al. reported the result of generating a menu hierarchy from functional descriptions using cluster analysis [13]. However, this approach is time consuming; therefore, we chose to use another one.

We represent the functional similarity of item I_x and I_y by using a function $s(I_x, I_y)$ which takes a value between 0 and 1. Let us assume that a generic function of each item I_i can be specified by some words $wl_i = \{w_1, w_2, \cdots\}$, and let $\mathbf{WL} = \bigcup_i wl_i$ be the whole words. Let us also assume that an intermediate node can be characterized by the words by which the children are specified. Let $\mathbf{x}$ be a vector in which element $x_i = 1$ iff the node x has the i-th word as its specification, And let $\mathbf{y}$ be a vector of node y. Then, the functional similarity $s(I_x, I_y)$ is defined as $s(I_x, I_y) = \frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}||\mathbf{y}|}$. Let us consider a node v_i^l that has m children. The penalty of functional similarity $P_{v_i^l}^s$ of node v_i^l is defined as $P_{v_i^l}^s = \sum_{x=0}^{m-1} \sum_{y=0}^{m-1} (1 - s(I_x, I_y))$. And the total penalty P^s is defined as $P^s = \sum_{v_i^l \in V} P_{v_i^l}^s$.

Menu Granularity

The menu granularity $g_{v_i^l}$ of a node v_i^l is defined as the total number of descendants. If node v_i^l is a terminal node, then $g_{v_i^l} = 0$. Moreover, if node v_i^l has m children ($v_j^{l+1}, j = 0, \cdots, m-1$) whose menu granularities are $g_{v_j^{l+1}}, (j = 0, \cdots, m-1)$, then $g_{v_i^l}$ is defined as $g_{v_i^l} = \sum_{j=0}^{m-1} g_{v_j^{l+1}}$. The penalty of menu granularity $P_{v_i^l}^g$ of node v_i^l is defined as $P_{v_i^l}^g = \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} \left| g_{v_i^l} - g_{v_j^l} \right|$ And the total penalty P^g is the sum over the whole nodes.

Objective Function

The problem is to minimize the following objective function:

$$f = T_{avg} + \alpha P^s + \beta P^g, \tag{1}$$

where α and β are constants that control the preference of functional similarity and menu granularity.

Optimization Problem

When menu items that are placed as the children of a node V are given, the placement that minimizes the average pointing time is straightforward. Therefore, the problem is to find the best assignment of menu items to nodes of a tree that minimizes Equation (1).

GENETIC ALGORITHM

Genetic Algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems. GA is a meta-heuristic algorithm that uses techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover [7].

Basic Strategy and Chromosome Representation

Previous studies showed that breadth was preferable to depth [5, 10, 14]. Because the previous studies showed that breadth was preferable to depth, we use a kind of breadth-first search algorithm as the core of the proposed GA.

An algorithm that places a menu items I_i one by one on a usable node can find a good solution. For this reason, we use an algorithm that assign I_i to a node as follows:

1. A chromosome of the GA is a sequence of I_i; i.e., a chromosome can be represented as a permutation of numbers.
2. According to the permutation, assign a menu items I_i one by one to a usable node that has the smallest node number.
3. If a generic function is assigned to a node then a node cannot have children, and the node below the assigned node is marked as *unusable*.

Other Configurations of GA

We use a crossover operator that does not generate an invalid chromosome. As described above, a chromosome is a permutation of numbers; therefore, we use crossover operators that are developed for the representation. We use the swap mutation as the mutation operator. Randomly chosen gene at position p and q are swapped. The crossover and mutation operators do not generate invalid chromosomes; i.e., offsprings are always valid permutations. We use a steady state GA, the population size is 100, the mutation rate is one swap per chromosome. The tournament selection of size two is used as the selection method.

NUMERICAL EXPERIMENTS

The target is a cellular phone that is used by one of the authors. The phone [9] has 24 keys as shown in Fig. 2.

Experimental Data

Pointing Time and Decision Time

The index of difficulty for 24×24 key pairs were calculated as follows. We measured the relative coordinates of the center (x, y) of each key, and measured the width and height of each key. We calculated the index of difficulty to an accuracy of one digit after the decimal point. This gave us 28 groups of indexes of difficulty.

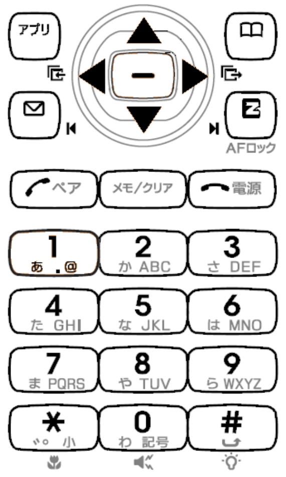

Figure 2. The key layout of the target cellular phone.

We measured the pointing time of one-handed thumb users for the above 28 groups by recording the tone produced by each key press [1]. Unpaid volunteers participated in the experiment. We prepared 28 tasks corresponding to the 28 groups. We got $t_i^p = 192 + 63\log_2(A_i/W_i + 1)$ (ms) for predicting the pointing time, and the equation is very similar to the one reported by Silfvergerg et al.[12][1] Menu selections are done by pressing keys, therefore we assume that moving to the next level starts from the key that reached the current node. Although the target phone has the ability to select a menu item by pressing a key that is prefixed to item title (shortcut key), we assumed that all selections were done by cursor movements.

The target of this experiments was an expert; therefore, we used $t_i^{sd} = 80\log_2(n^l) + 240$ (ms) [4][2];

Usage Frequency Data

We gathered usage frequency data as follows. The first author recorded the daily usage of each function for two months, and we generated the usage frequency data from the record. There were 129 terminal nodes in the data.

Functional Similarity

We assigned three to five words to each generic function according to the users' manual of the target phone [9].

Results

We conducted the following experiments. Because GA is a stochastic algorithm, we conducted 50 runs for every test cases, and the results shown in Table 1 and Table 2 are averages over 50 runs. The two parameters for weights were set to $\alpha = 10.0$ and $\beta = 1.0$ in Table 1.

case 1 Typical Usage: This experiment was conducted to assess the typical improvement by the GA. The maximum width W was 16.

[1] $t_i^p = 176 + 64\log_2(A_i/W_i + 1)$ (ms).

[2] The equation is derived from experiments conducted for a computer display, and is not for a cellular phone.

Case	T_{ave}(ms)	(%)	P^s	P^g
Original	3331	0.0	454	793
Local Move	2813	15.6	454	793
Case 1 ($W = 16$)	2066	38.0	721	1299
Case 2 ($W = 12$)	2034	38.9	539	871
Case 2 ($W = 9$)	1981	40.5	392	336
Case 2 ($W = 6$)	2224	33.2	274	203

Table 1. Improvements in average selection time.

α	β	T_{ave}(ms)	(%)	P^s	P^g
10.0	1.0	1981	40.5	392	336
20.0	1.0	2034	38.9	386	398
5.0	1.0	1957	41.2	401	307
0.0	0.0	1874	43.7	569	556

Table 2. Effect of weights.

case 2 Limited Breadth: Although breadth is preferable to depth, pressing a far key or pressing a "Down" key many times is sometimes tedious. This experiment was conducted to see the effect of limiting the breadth. In this case, we set the breadth (W) to 12, 9, and 6.

In Table 1, "Local Move" shows the results of a local modification that places menu items according to their frequency; i.e., the most frequently used item is placed as the top item, and so on. As the table shows, the proposed algorithm can generate menu with shorter average selection time. Moreover, limiting the number of usable keys gave us better menus. This is partly because the search/decision time is proportional to $\log_2(n)$, where n is the number of items. As the number of items increases, the search/decision time increase; therefore, the average selection time increase. Limiting the number of keys to 6 gave a longer selection time, and smaller penalties.

Effects of weights

We introduced two weights for penalties of functional similarity and of menu granularity. Table 2 shows the results of different weight settings for the case $W = 9$. As Table 2 shows α and β to non-zero gave similar results. Setting them to zero gave a shorter selection time, but the penalties were larger.

DISCUSSION AND FUTURE WORK

The experiments show that the proposed algorithm can generate better menu hierarchies for the target phone. Because our targets are not limited to cellular phones, and the preliminary results are promising, we will apply the algorithm to wider varieties of targets.

In this paper, we focused on a static menu as the target; adaptive/dynamic menu (e.g., [2, 3, 8]) that changes menu contents depending on usage will be a future target.

The data used in the experiments, especially selection frequency data, were limited. Therefore we should gather a wider variety of usage data and use that to confirm the effectiveness of the proposed method.

CONCLUSION

We proposed a GA-based algorithm for minimizing the average selection time of menu items that consider the movement time and the decision time. The preliminary results showed that the algorithm can generate a better menu structure. The target of the proposed algorithm is not limited to cellular phones, and can be applied to other kinds of menus.

REFERENCES

1. Amant, St., Horton,T.E. and Ritter, F.E. Model-based evaluation of cell phone menu interaction, *Proc. CHI 2004*, ACM Press (2004), 343–350.
2. Ahlström, D. Modeling and improving selection in cascading pull-down menus using Fitts' law, the steering law and force fields, *Proc. CHI 2005*, ACM Press (2005), 61–70.
3. Beck, J., Han, S.H., and Park, J. Presenting a submenu window for menu search on a cellular phone, *JHCI*, 20,3 (2006), 233–245.
4. Cockburn, A., Gutwin, G., and Greenberg, S. A predictive model of menu performance, *Proc. CHI 2007*, ACM Press (2007), 627–636.
5. Schultz Jr., E.E. and Curran, P.S. Menu structure and ordering of menu selection: independent or interactive effects?, *SIGCHI Bull.*, 18,2 (1986), 69–71.
6. Francis, G. Designing multifunction displays: an optimization approach, *Int. J. of Cognitive Ergonomics*, 4,2 (2000), 107–124.
7. Goldberg, D.E. *Genetic algorithm*, Addison-Wesley (1989).
8. Findlater, L. and McGrenere,J. A comparison of static, adaptive, and adaptable menus, *Proc. CHI 2004*, ACM Press (2004), 89–96.
9. KDDI. Manual for CASIO W43CA, http://www.au.kddi.com/torisetsu/pdf/w43ca/w43ca_torisetsu.pdf (2004).
10. Kiger, J.I. The depth/breadth trade-off in the design of menu-driven user interfaces, *Int. J. Man-Mach. Stud.*, 20,2 (1984), 201–213.
11. Liu, B. Francis, G., and Salvendy, G. Applying models of visual search to menu design, *Int. J. Human-Computer Studies*, 56 (2002), 307–330.
12. Silfverberg, M., MacKenzie, I.S., and Kauppinen, T. Predicting text entry speed on mobile phones, *Proc. CHI 2000*, ACM Press (2000), 9–16.
13. Toms, M.L., Cummings-Hill, M.A., Curry, D.G., and Cone, S.M. Using cluster analysis for deriving menu structures for automotive mobile multimedia applications, *SAE Technical Paper Series*, 2001-01-0359 (2001), SAE.
14. Zaphiris, P., Kurniawan, S.H., and Ellis, R.D. Age related difference and the depth vs. breadth tradeoffs in hierarchical online information systems, *Proc. User Interfaces for All, LNCS 2615*, (2003), 23–42.