One-Level Menu Optimization: How to Make Conventional Things Better

Evgeny IVANKO¹,²

¹ Institute of Mathematics and Mechanics, Ural Branch, Russian Academy of Sciences, S.Kovalevskaya str. 16, 620990 Ekaterinburg, Russia
² Ural Federal University, Lenin ave. 51, 620000 Ekaterinburg, Russia

Keywords: Menu optimization, HCI, access time, convenience, balance, UPGMA, Fitt's law.

Abstract. In this article, I address the problem of 1-level horizontal menu access time minimization while preserving the menu’s conventional appearance. The proposed mathematical formalization shows high computational complexity even for the considered simplified problem. As an alternative, a heuristic is proposed to produce feasible solutions in polynomial time; 2 results of the application of the heuristic are presented.

Introduction

Menu design is a complex interdisciplinary problem with a wide range of applications as we all want to have convenient, efficient and good-looking menus at hand. Menus design was born at the crossroads of such fields as cognitive science, mathematical optimization, and graphical design. Presently, there are increasing attempts to combine convenience and efficiency of menus with the help of mathematics [1–5, 9, 10, 12, 15, 17, 20, 23, 25, 32]. It proved to be a hard problem due to the two main reasons: 1) the detailed models of user, where psychophysiology meets biomechanics, are relatively weakly studied [6–8, 13, 16, 19, 21, 24, 26–31]; 2) the balance between minimization of the expected access time and preservation of the accustomed appearance is ambiguous [11]. In this article, I will address the second issue.

One-level menu may be formalized as an ordered sequence of “tabs”, where each tab is an ordered sequence of items (commands). In this article, I will talk only about the most common horizontal menus (see Fig.1), however, all the results may be directly extended to the case of vertical menus. The idea of optimization of one-level horizontal menus is based on placing the most frequently used menu items (or commands) top-left since, in accordance with cognitive science models, it minimizes the access time [1]. Whereas the “optimization” criterion tells us to put the most frequently used items at the most readily accessible places, the “appearance” criterion requires each item to be easy to find in context (it should lie within a reasonably named menu tab and should stay together with similar items).

Formulism in optimization here will lead to the situation, where e.g. “Save” and “Close” items are situated far from each other (since “Save” is used frequently and will stay top-left whereas “Close” will be pushed out to down-right by more frequently accessed items). To avoid this “overoptimization”, the arrangement of the items is replaced with the arrangement of unsplittable clusters of the items (Section 2). However, unsplittable clusters is not a panacea. The “appearance” criterion reveals itself in a more complex problem of constructing semantically consistent tabs (in
other words, the tabs that can be easily named). The leftmost tab, constructed by straightforward optimization techniques, may e.g. contain two frequently used groups “Open, Save, Save As” and “Cut, Copy, Paste”, which not only breaks the conventional user expectations but also makes it hard to give this tab a head. Section 3 is devoted to the development of the heuristic that conducts optimization taking into account the above described specific issues.

**Mathematical statement of simplified problem**

The mathematical statements in the current and next sections are given using standard notation, which however should be explained for a broader audience: \( A \sqcup B = C \) means that \( A \cup B = C \) and \( A \cap B = \emptyset \); \( \mathcal{P}(X) \) is the set of all subsets of \( X \); \( f[X] \) is the image of the set \( X \); \( \text{Argmax}_{x \in X} f(x) \) is the set of the elements of \( X \), at which \( f(x) \) takes the maximum value; a singleton is a 1-element set.

Let \( I \) be the set of all items, \( f : I \to \mathbb{R} \) be the frequency of usage function, \( \mathcal{C} = \{C_1 \ldots C_n\} \) be the set of unsplittable clusters constructed a priori on the basis of some similarity function and let

\[
\bigcup_{i=1}^{n} C_i = I.
\]

Let \( N \) be the number of tabs. If \( N \) is not known in advance, it can be taken large enough, e.g. \( N = M \); the optimization process will pull all the clusters leftward and leave the redundant right tabs empty. The optimization problem consists of four interconnected parts: 1) optimal distribution of the clusters from \( \mathcal{C} \) among \( N \) tabs; 2) optimization of the order of the tabs from left to right; 3) optimization of the clusters within each tab; 4) optimization of the items within each cluster. The problem statement in this section is simplified: the semantic consistency of the tabs is not considered.

Let \( t_i = \{C_{i1}, \ldots, C_{iK_i}\}, i \in \overline{1,N} \), be the set containing the clusters assigned to the \( i \)-th tab, and

\[
\bigcup_{i=1}^{N} t_i = \mathcal{C}.
\]

One of the approaches to minimization of access time is to balance the load of the tabs:

\[
\max_{i \in 1,N} \{F(t_i)\} \to \min_{t_1 \sqcup \ldots \sqcup t_N = \mathcal{C}} \min_{i \in 1,N} \{t_i\}
\]

(1)

where \( F(X_i) \) is the summary cost of the \( i \)-th tab, computed based on the best possible order of the weighted tab’s clusters:
\[ F(t_i) = \min_{\gamma: 1, K_i \to 1, K} \left\{ \sum_{j=1}^{K_i} W(C_{\gamma(j)}^i) \right\}. \] 

The weight \( W \) of each cluster is summed from the weights of its items, and the weight of each item is the product of its frequency and its distance from the top-left corner of the menu:

\[ W(C_j^i) = \sum_{l=1}^{\lvert C_j^i \rvert} \left[ f(c_l^{(i,j)})(i - 1) + \Delta(i, j) + l \right], \] 

where \( c_l^{(i,j)} \) is the \( l \)-th item of the \( j \)-th cluster in the \( i \)-th tab (in this notation, the elements of each cluster are assumed to be sorted by descending frequency \( \forall i \in \overline{1, N} \forall j \in \overline{1, K_i} \) \( k < l \Rightarrow (f(c_k^{(i,j)}) \geq f(c_l^{(i,j)})) \)); \( f(c) \) is the frequency of the item \( c \); \( (i - 1) \) is the items’ distance shift due to the horizontal position of the whole \( i \)-th tab; \( \Delta(i, j) \) is the distance shift due to the position of the cluster in the tab, which depends on the summary power of the “upper” clusters:

\[ \Delta(i, j) = \sum_{k=1}^{j-1} |C_{\gamma(k)}^i|. \] 

The distance components of (3) may be taken with different weights \( \omega_1, \omega_2, \omega_3 \), if such weights can be estimated from some cognitive model:

\[ W(C_j^i) = \sum_{l=1}^{\lvert C_j^i \rvert} \left[ f(c_l^{(i,j)})(\omega_1(i - 1) + \omega_2\Delta(i, j) + \omega_3l) \right]. \] 

Problem (1) is a complicated variant of the partition problem [22] (NP-hard). Problem (2) is a TSP-like problem with a possible dependence of the cost function on a task list [14] (NP-hard). If problem (1)-(4) appears too complex from the computational point of view, the distribution and ordering components can be separated from each other. In this case, for example, at the first step, the clusters can be distributed among the tabs with the help of some simple function \( \tilde{F}(t) \) (instead of the unknown optimal \( F(t) \) from (2)), and at the second step, where the clusters composing each tab are known, the problem (2)-(4) can be solved independently for each tab to find the optimum order of the clusters.

**Algorithms**

As it was noted before, a good menu should solve at least two (often conflicting) problems: 1) group similar items together; 2) provide fast access to the most frequently used items. The priority here should be given to the grouping, since a user could put up with the increasing access time, but would be definitely annoyed at the destruction of the groups he got used to. The optimization process considered in the previous section preserves conventional item groups (for example, the elements of the group \{“Cut”, “Copy” and “Paste”\} are not spread over different tabs). In this section, I consider a more careful grouping, which allows assembling reasonable “homogeneous” tabs that could be given reasonable names and provide navigation hints.
The primary grouping problem is addressed by a specific 2-level UPGMA-based (Unweighted Pair Group Method with Arithmetic Mean) [18] clustering method. At the first level of the proposed method, a known association function on unordered pairs of items is used to create unsplittable clusters of highly associated items. At the second, these clusters are merged into tabs. The frequency-connected optimization is conducted during the grouping process whenever possible, as far as it does not contradict the main clustering.

The first-level clustering method (Algorithm 1) is a variant of the classical UPGMA, which is used to construct unsplittable groups of closely related menu items. The merging process continues until there are no clusters, the similarity between which exceeds some preselected threshold. It should be noted that UPGMA algorithms implemented with heaps have polynomial time complexity $O(n^2 \log n)$ [18].

The second-level tab formation (Algorithm 2) is a UPGMA-inspired approach. The merging elements here are the unsplittable clusters generated at the first level. The second-level merging process continues until the largest cluster becomes large enough to be taken as one single tab. After that, the constructed tab is withdrawn, the remaining second-level cluster structure is destroyed and a new second-level merging process is started over the rest of the first-level unsplittable clusters. The resulting tabs are ordered by descending summary frequency, the first-level clusters within each tab are ordered by descending average frequency and the items within each first-level cluster are ordered by decreasing frequency. The number of tabs in this approach is not predefined. Different numbers of tabs are tried to construct the resulting menus and the one with the best value of frequency-motivated criterion is selected.

**Algorithm 1** (First-level UPGMA clustering: find unsplittable groups).

// Data
1. $I$, a set of items; use it to construct the set of unordered pairs of the items $P := \{B \in \mathcal{P}(I) : |B| = 2\}$;
2. $A_1 : P \rightarrow [0, 1]$, an association function defined on $P$ (the larger the closer);
3. $\tau$, the stop threshold, which takes one of the following values: mean, mean ± standard deviation or some percentile of $A_1[P]$;

// Results
4. $\mathcal{C} = \{C_1, \ldots, C_n\}$, the set of clusters such that $\cup_{i=1}^n C_i = I$;
5. $A_2 : \{B \in \mathcal{P}(\mathcal{C}) : |B| = 2\} \rightarrow [0, 1]$, an association function, defined on the unordered pairs of the clusters, where $A_2(\{C_i, C_j\}) < \tau$ if $i \neq j$;

// Initialize
7. let $\mathcal{C}$ (current set of clusters) initially be the set of all possible singletons $\{x\}, x \in I$;
8. $A_2$ is the association function defined on the singleton clusters as follows: $A_2(\{\{x\}, \{y\}\}) := A_1(\{x, y\})$;
9. $\mathcal{E}$ is the set of all unordered pairs of different clusters from $\mathcal{C}$, the members of which are close enough to be merged: $\mathcal{E} := \{\{X, Y\} \in \mathcal{P}(\mathcal{C}) : X \neq Y \land A_2(\{X, Y\}) \geq \tau\}$;
// Main cycle
while \( \mathcal{C} \neq \emptyset \) do:
10. select an arbitrary element \( z_0 = \{C_i, C_j\} \) from \( \text{Argmax}_{z \in \mathcal{C}} A_2(z) \);
11. replace \( C_i \) and \( C_j \) with their union in the current set of clusters \( \mathcal{C} \):
\[
\mathcal{C} := (\mathcal{C} \setminus \{C_i, C_j\}) \cup \{C_i \cup C_j\};
\]
12. extend the association function \( A_2 \) considering the new item:
\[
\forall C \in \mathcal{C} \setminus \{C_i \cup C_j\}, A_2\left(\{C, C_i \cup C_j\}\right) := w_1 A_2\left(\{C, C_i\}\right) + w_2 A_2\left(\{C, C_j\}\right),
\]
where \( w_1 = \frac{|C_i|}{(|C_i| + |C_j|)} \) and \( w_2 = \frac{|C_j|}{(|C_i| + |C_j|)} \);
13. refresh the set of pairs to join:
\[
\mathcal{E} := \{\{X, Y\} \in \mathcal{P}(\mathcal{C}) : X \neq Y \land A_2(\{\{X, Y\}\}) \geq \tau\};
\]
end
return \( \mathcal{C}, A_2 \).

Algorithm 2 (Second-level “careful” hierarchical clustering: build tabs).

// Data
1. \( \mathcal{C} \), a set of clusters;
2. \( A_2 \), an association function, defined on the unordered pairs of the clusters;
3. \( f : I \to \mathbb{R} \), the frequency of usage of the items;

// Result
4. \( \mathbb{T} = (T_1, \ldots, T_N) \), a sequence of disjoint tabs, where each tab \( T_i \) is a sequence of “ordered clusters” \( s(C_i^1) \ldots s(C_{K_i}^i) \) and each “ordered cluster” \( s(C_j^i) \) is a sequence of the elements of the cluster \( C_j^i \in \mathcal{C} \) such that \( \sqcup_{i=1}^{N} \sqcup_{j=1}^{K_i} C_j^i = \mathcal{C} \);

// Initialize
5. find the maximum size \( z(N) \) of a tab for the given number of tabs \( N \) and the known number of items \( |I| \) (see Fig.2): \( (z(N) - 0) + (z(N) - 1) + \ldots + (z(N) - (N - 1)) = |I| \), from where \( z(N) = \text{IntegerPartOf} \left( \frac{2|I| + N^2 - N}{2N} + 0.5 \right) \);
6. build a frequency function on the clusters: \( \forall C \in \mathcal{C}, F(C) := \left( \sum_{x \in C} f(x) \right) / |C| \) (alternatively, one could use the median);
7. for each \( C \in \mathcal{C} \) let \( s(C) \) be the sequence of the items \( x \in C \) sorted by descending \( f(x) \);
8. \( Q^* \) is an upper bound for the criterion function \( Q(M) \) from [6], e.g. \( Q^* = 2|I|^4 \max_{x \in I} f(x) \).
Figure 2. Second-level “careful” hierarchical clustering: build tabs (initialization part). Different sizes of “ideal” tabs depending on their distance from the upper-left corner (due to Fitt’s law [1,7]). Here we assume that the horizontal unit steps (from one tab to the next) and the vertical unit steps (between neighbor items in a tab) have equal weight

```
// Main cycle
for N ∈ 2, |C| do: // go through all reasonable numbers of tabs
9. ˜C := C; ˜A₂ := A₂;
10. M := min{z(N), N − 2}; // limit the number of tabs to reduce the number of empty ones
11. for i ∈ 0, M do: // construct the i-th of M tabs
   a. continue an UPGMA-like clustering process (“second-level”) similar to that one described in Algorithm 1, iteratively merging the clusters from ˜C into tabs (instead of merging the items into clusters in Algorithm 1) and updating the similarity function ˜A₂ (instead of A₁) until the overall number of the items in the largest second-level cluster is greater than z(N) − i;
   b. let T’ᵢ₊₁ be any one of the largest second-level clusters resulted from the previous step;
   c. exclude the clusters of T’ᵢ₊₁ from the initial cluster set, forget about the remaining constructed tab structure and forget about the extensions made to ˜A₂:
      ˜C := C \ \bigcup_{j=1}^{i+1} T’ᵢ,
      ˜A₂ := A₂|˜C;
end
12. let M be i + 2; put all the remaining first-level clusters (if any) into the last second-level cluster: T’ₘ := ˜C; T’ₘ is a “freak tab” consisting of the first-level clusters that could hardly be associated with any other first-level clusters (have small values of A₂) and thus were not included into the previously formed second-level clusters at step 10;
13. rearrange the indexes 1...M such that T’₁,...,T’ₘ become sorted by the summary frequency of the contained items (∑ₕ∈C רוב \sumᵢ∈C f(i)) in descending order;
```
14. for each \( i \in \overline{1,M} \) construct the tab \( T_i \) from the corresponding second-level cluster \( T'_i \) by sorting the first-level clusters of \( T'_i \) by the average frequency \( F(C) \) in descending order;

15. the resulting ordered sequence \( (T_1, \ldots, T_M) \), where each tab \( T_k \) is an ordered sequence \( (s(C^k_1), \ldots, s(C^k_{|T'_k|})) \) of “ordered clusters” and where each “ordered cluster” \( C^k_i \) is an ordered sequence of items \( (x^{(k,i)}_1, \ldots, x^{(k,i)}_{|C^k_i|}) \), is referred to as a “candidate menu”; compute the “quality” (expected access efficiency) of this menu:

\[
Q(M) := \sum_{k=1}^{M} \sum_{i=1}^{|T'_k|} \sum_{j=1}^{|C^k_i|} \left( f(x^{(k,i)}_j) \left[ (k - 1) + \left( \sum_{q=1}^{i-1} C^k_q \right) + j \right] \right) ;
\]

(6)

16. remember the new “candidate menu” if it is better than all the previous:

if \( Q(M) < Q^* \), then \( Q^* := Q(M) \) and \( T := (T_1, \ldots, T_M) \).

\end

\textbf{return} \, T

If there is a considerable amount of equal values of the association functions \( A_1 \) and/or \( A_2 \), the clustering process may become random since an arbitrary best pair is selected for merging. In this case, multiple runs of both algorithms could result in multiple feasible solutions with different structures, which can be compared on the basis of pure “quality” \( Q^* \) as well as by human designers.

The proposed heuristic is parametrized by the only parameter – stop threshold \( \tau \). In the general case, where no additional information is available, from practical experience it can be recommended to divide the percentile interval \([1, 99]\) regularly into \( L \) almost equal parts and to construct the menus for the values of \( \tau \) equal to all the obtained endpoints. For example, if \( L = 5 \), the endpoint percentiles could be \( \{1, 20, 40, 60, 80, 99\} \). After that, the designer (human) could select a couple of neighbor percentile endpoints corresponding to the menus that have the smallest values of \( Q^* \) (or just those he prefers from the structural point of view) and use the bisection method between them, iteratively generating new percentiles and the corresponding menus that may converge finally to some local “optimum” (see Figure 3).

\textbf{Results}

The examples of application of Algorithms 1,2 are given in the following tables. Tables 1–3 show the results of construction of menus using the 51 items of the Firefox browser original menu system (version 36). This problem is referred to below as “Firefox problem”. The values of frequency (\( f \) in Algorithm 2) and association (\( A_1 \) in Algorithm 1) functions (Tables 1,2) were taken initially from the Mozilla Lab Test Pilot project (not supported anymore).

Table 4 shows the results of application of Algorithms 1,2 to a demonstrative toy problem, where a group of 44 organisms is organized into a virtual “menu system” (this demonstrative case is referred to below as “taxonomy problem”). “Taxonomic” association between two organisms here is the number of shared taxonomic ranks, for example, the blue whale (Eukaryota/Animalia/Chordata/Mammalia/Artiodactyla/Balaenopteridae/Balaenoptera) and the telephone-pole beetle (Eukaryota/Animalia/Arthropoda/Insecta/Coleoptera/Micromalthidae/Micromalthus)
Figure 3. Iterative application of the proposed heuristic by the bisection method

have only two shared taxons: Eukaryota/Animalia, so the association between them (function $A_1$ in Algorithm 1) will be equal to $2/7$ (where 7 is the maximum number of taxon ranks considered). The “frequency” of an organism is the number of the results Google Search returned for the name of the species on July 6, 2017, for example, the frequency for Giraffe is $130 \cdot 10^6$ search results. The purpose of this toy problem is to show that in an ideal situation the application of the proposed method allows one to construct the tabs that could have been given sensible names.

Acknowledgements

This research was supported by the Russian Science Foundation, grant no. 14-11-00109. The author wants to thank Prof. Antti Oulasvirta, MSc. Morteza Shiripour and Dr. Niraj Dayama from Aalto University, Finland for the fruitful discussions and valuable comments.
Table 1. The correspondence between menu items, their labels and frequency for the “Firefox problem”

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Menu item</th>
<th>Frequency</th>
<th>Abbreviation</th>
<th>Menu item</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Recently Closed Tab</td>
<td>0.1798</td>
<td>b</td>
<td>New Window</td>
<td>0.0862</td>
</tr>
<tr>
<td>B</td>
<td>Recently Closed Windows</td>
<td>10^-4</td>
<td>c</td>
<td>Open Location</td>
<td>0.0124</td>
</tr>
<tr>
<td>C</td>
<td>User History Item</td>
<td>0.5146</td>
<td>d</td>
<td>Open File</td>
<td>0.0496</td>
</tr>
<tr>
<td>D</td>
<td>Show All Bookmarks</td>
<td>10^-4</td>
<td>e</td>
<td>Close Tab</td>
<td>10^-4</td>
</tr>
<tr>
<td>E</td>
<td>Bookmark This Page</td>
<td>0.9857</td>
<td>f</td>
<td>Close Window</td>
<td>0.011717</td>
</tr>
<tr>
<td>F</td>
<td>Subscribe To This Page</td>
<td>10^-4</td>
<td>g</td>
<td>Save Page As</td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>Bookmark All Pages</td>
<td>0.0744</td>
<td>h</td>
<td>Email Link</td>
<td>10^-4</td>
</tr>
<tr>
<td>H</td>
<td>Unsorted Bookmarks</td>
<td>10^-4</td>
<td>i</td>
<td>Page Setup</td>
<td>10^-4</td>
</tr>
<tr>
<td>I</td>
<td>Minimize</td>
<td>10^-4</td>
<td>j</td>
<td>Print</td>
<td>0.1302</td>
</tr>
<tr>
<td>J</td>
<td>Zoom</td>
<td>10^-4</td>
<td>k</td>
<td>Work Offline</td>
<td>0.1674</td>
</tr>
<tr>
<td>K</td>
<td>Start Page</td>
<td>10^-4</td>
<td>l</td>
<td>Undo</td>
<td>10^-4</td>
</tr>
<tr>
<td>L</td>
<td>About Mozilla Firefox</td>
<td>0.124</td>
<td>m</td>
<td>Redo</td>
<td>10^-4</td>
</tr>
<tr>
<td>M</td>
<td>Add-ons</td>
<td>0.16987</td>
<td>n</td>
<td>Cut</td>
<td>10^-4</td>
</tr>
<tr>
<td>N</td>
<td>Check for Updates</td>
<td>0.4836</td>
<td>o</td>
<td>Copy</td>
<td>0.0558</td>
</tr>
<tr>
<td>O</td>
<td>Clear Recent History</td>
<td>0.16987</td>
<td>p</td>
<td>Paste</td>
<td>0.031</td>
</tr>
<tr>
<td>P</td>
<td>Customize</td>
<td>0.7254</td>
<td>q</td>
<td>Delete</td>
<td>0.0186</td>
</tr>
<tr>
<td>Q</td>
<td>Downloads</td>
<td>0.2852</td>
<td>r</td>
<td>Select All</td>
<td>0.0124</td>
</tr>
<tr>
<td>R</td>
<td>Private Browsing</td>
<td>0.0806</td>
<td>s</td>
<td>Find</td>
<td>0.0558</td>
</tr>
<tr>
<td>S</td>
<td>Reload</td>
<td>0.0124</td>
<td>t</td>
<td>Find Again</td>
<td>10^-4</td>
</tr>
<tr>
<td>T</td>
<td>Stop</td>
<td>10^-4</td>
<td>u</td>
<td>Start Dictation</td>
<td>10^-4</td>
</tr>
<tr>
<td>U</td>
<td>Reset</td>
<td>0.093</td>
<td>v</td>
<td>Back</td>
<td>0.0062</td>
</tr>
<tr>
<td>V</td>
<td>Zoom In</td>
<td>0.0124</td>
<td>w</td>
<td>Forward</td>
<td>10^-4</td>
</tr>
<tr>
<td>W</td>
<td>Zoom Out</td>
<td>10^-4</td>
<td>x</td>
<td>Home</td>
<td>0.0124</td>
</tr>
<tr>
<td>X</td>
<td>Web Search</td>
<td>10^-4</td>
<td>y</td>
<td>Show All History</td>
<td>10^-4</td>
</tr>
<tr>
<td>Y</td>
<td>Page Info</td>
<td>0.031</td>
<td>z</td>
<td>Restore Previous Session</td>
<td>10^-4</td>
</tr>
<tr>
<td>a</td>
<td>New</td>
<td>0.12151</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Association function $A_1$ for the “Firefox problem” (all absent pairs have zero association)

<table>
<thead>
<tr>
<th>Label 1</th>
<th>Label 2</th>
<th>Value</th>
<th>Label 1</th>
<th>Label 2</th>
<th>Value</th>
<th>Label 1</th>
<th>Label 2</th>
<th>Value</th>
<th>Label 1</th>
<th>Label 2</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>0.99</td>
<td>M</td>
<td>k</td>
<td>0.4</td>
<td>a</td>
<td>b</td>
<td>0.99</td>
<td>d</td>
<td>e</td>
<td>0.2</td>
</tr>
<tr>
<td>A</td>
<td>e</td>
<td>0.75</td>
<td>N</td>
<td>O</td>
<td>0.4</td>
<td>a</td>
<td>c</td>
<td>0.4</td>
<td>d</td>
<td>f</td>
<td>0.2</td>
</tr>
<tr>
<td>A</td>
<td>f</td>
<td>0.6</td>
<td>N</td>
<td>X</td>
<td>0.4</td>
<td>a</td>
<td>d</td>
<td>0.6</td>
<td>e</td>
<td>f</td>
<td>0.99</td>
</tr>
<tr>
<td>B</td>
<td>e</td>
<td>0.6</td>
<td>N</td>
<td>h</td>
<td>0.75</td>
<td>a</td>
<td>e</td>
<td>0.2</td>
<td>e</td>
<td>g</td>
<td>0.4</td>
</tr>
<tr>
<td>B</td>
<td>f</td>
<td>0.75</td>
<td>O</td>
<td>y</td>
<td>0.9</td>
<td>a</td>
<td>f</td>
<td>0.2</td>
<td>g</td>
<td>i</td>
<td>0.75</td>
</tr>
<tr>
<td>C</td>
<td>O</td>
<td>0.75</td>
<td>O</td>
<td>z</td>
<td>0.4</td>
<td>a</td>
<td>g</td>
<td>0.5</td>
<td>g</td>
<td>j</td>
<td>0.6</td>
</tr>
<tr>
<td>C</td>
<td>y</td>
<td>0.99</td>
<td>P</td>
<td>V</td>
<td>0.4</td>
<td>a</td>
<td>i</td>
<td>0.5</td>
<td>i</td>
<td>j</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>E</td>
<td>0.75</td>
<td>P</td>
<td>W</td>
<td>0.4</td>
<td>a</td>
<td>j</td>
<td>0.6</td>
<td>l</td>
<td>m</td>
<td>0.99</td>
</tr>
<tr>
<td>D</td>
<td>G</td>
<td>0.75</td>
<td>Q</td>
<td>k</td>
<td>0.4</td>
<td>a</td>
<td>l</td>
<td>0.2</td>
<td>l</td>
<td>n</td>
<td>0.6</td>
</tr>
<tr>
<td>D</td>
<td>H</td>
<td>0.75</td>
<td>Q</td>
<td>y</td>
<td>0.4</td>
<td>a</td>
<td>m</td>
<td>0.2</td>
<td>l</td>
<td>o</td>
<td>0.6</td>
</tr>
<tr>
<td>E</td>
<td>G</td>
<td>0.99</td>
<td>Q</td>
<td>z</td>
<td>0.4</td>
<td>a</td>
<td>n</td>
<td>0.2</td>
<td>l</td>
<td>p</td>
<td>0.6</td>
</tr>
<tr>
<td>E</td>
<td>H</td>
<td>0.6</td>
<td>S</td>
<td>T</td>
<td>0.6</td>
<td>a</td>
<td>o</td>
<td>0.2</td>
<td>l</td>
<td>q</td>
<td>0.4</td>
</tr>
<tr>
<td>F</td>
<td>Q</td>
<td>0.6</td>
<td>S</td>
<td>U</td>
<td>0.75</td>
<td>a</td>
<td>p</td>
<td>0.2</td>
<td>l</td>
<td>v</td>
<td>0.2</td>
</tr>
<tr>
<td>G</td>
<td>H</td>
<td>0.6</td>
<td>S</td>
<td>Y</td>
<td>0.4</td>
<td>a</td>
<td>q</td>
<td>0.2</td>
<td>l</td>
<td>w</td>
<td>0.2</td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>0.75</td>
<td>T</td>
<td>U</td>
<td>0.75</td>
<td>a</td>
<td>r</td>
<td>0.2</td>
<td>l</td>
<td>x</td>
<td>0.2</td>
</tr>
<tr>
<td>J</td>
<td>V</td>
<td>0.9</td>
<td>T</td>
<td>l</td>
<td>0.6</td>
<td>a</td>
<td>s</td>
<td>0.2</td>
<td>n</td>
<td>o</td>
<td>0.99</td>
</tr>
<tr>
<td>J</td>
<td>W</td>
<td>0.9</td>
<td>T</td>
<td>m</td>
<td>0.6</td>
<td>a</td>
<td>t</td>
<td>0.2</td>
<td>n</td>
<td>p</td>
<td>0.99</td>
</tr>
<tr>
<td>K</td>
<td>y</td>
<td>0.6</td>
<td>T</td>
<td>v</td>
<td>0.4</td>
<td>a</td>
<td>v</td>
<td>0.2</td>
<td>n</td>
<td>q</td>
<td>0.8</td>
</tr>
<tr>
<td>K</td>
<td>g</td>
<td>0.6</td>
<td>T</td>
<td>w</td>
<td>0.4</td>
<td>a</td>
<td>w</td>
<td>0.2</td>
<td>o</td>
<td>p</td>
<td>0.99</td>
</tr>
<tr>
<td>K</td>
<td>i</td>
<td>0.6</td>
<td>U</td>
<td>v</td>
<td>0.4</td>
<td>a</td>
<td>x</td>
<td>0.2</td>
<td>s</td>
<td>t</td>
<td>0.99</td>
</tr>
<tr>
<td>L</td>
<td>N</td>
<td>0.75</td>
<td>U</td>
<td>w</td>
<td>0.4</td>
<td>c</td>
<td>d</td>
<td>0.9</td>
<td>v</td>
<td>w</td>
<td>0.99</td>
</tr>
<tr>
<td>L</td>
<td>Q</td>
<td>0.75</td>
<td>U</td>
<td>x</td>
<td>0.4</td>
<td>c</td>
<td>e</td>
<td>0.4</td>
<td>v</td>
<td>x</td>
<td>0.99</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>0.75</td>
<td>V</td>
<td>W</td>
<td>0.99</td>
<td>c</td>
<td>f</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 4. Two examples of solution of the “Firefox problem” (see Tables 1, 2). The upper menu corresponds to $\tau$ equal to the 30th percentile and the lower one corresponds to $\tau$ equal to $\text{mean} - \text{stdev}$ (see Algorithm 1 for the details). The gray lines contain the cluster centers of the corresponding tabs (with the association function $A_1$ as the distance; the number in parentheses to the right of the cluster center is the sum of the distances from this central element to all other elements of the tab, it reflects how well this center “represents” the cluster). The numbers in parentheses to the right of the menu items are the frequencies of the corresponding items. The numbers at the bottom of the tabs are the sums of the frequencies of the tab items. All the values are accurate to at most 2 significant figures.
<table>
<thead>
<tr>
<th>$T_1$ (Artiodactyla)</th>
<th>$T_2$ (Primates)</th>
<th>$T_3$ (Rodents)</th>
<th>$T_4$ (Snakes and lizards)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Llama (211·10⁶)</td>
<td>Gorilla (103·10⁶)</td>
<td>Beaver (136·10⁶)</td>
<td>Orangutan (769·10⁴)</td>
</tr>
<tr>
<td>Giraffe (130·10⁶)</td>
<td>Chimpanzee (17.3·10⁶)</td>
<td>Black rat (5.9·10⁶)</td>
<td>Marine iguana (122·10⁴)</td>
</tr>
<tr>
<td>Boar (34.1·10⁶)</td>
<td>Gibbon (14·10⁶)</td>
<td>Capybara (3.95·10⁶)</td>
<td>Ornate monitor (50·10⁴)</td>
</tr>
<tr>
<td>Blue whale (12.1·10⁶)</td>
<td>Orangutan (13·10⁶)</td>
<td>Ural field mouse (4.72·10⁴)</td>
<td>Tokay gecko (34.7·10⁴)</td>
</tr>
<tr>
<td>Sperm whale (302·10⁴)</td>
<td>Aye-aye (491·10⁴)</td>
<td>thermometer (8.03·10⁴)</td>
<td>Scincus scincus (8.03·10⁴)</td>
</tr>
<tr>
<td>Bactrian camel (120·10⁴)</td>
<td>Pygmy marmoset (63.4·10⁴)</td>
<td>145897200</td>
<td>Echis carinatus (5.69·10⁴)</td>
</tr>
<tr>
<td>Bottlenose dolphin. (114·10⁴)</td>
<td></td>
<td></td>
<td>Typhlops vern. (2.54·10⁴)</td>
</tr>
<tr>
<td>392560000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T_5$ (Crocodilia and turtles)</th>
<th>$T_6$ (Various insects)</th>
<th>$T_7$ (Beetles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>American alligator (327·10⁴)</td>
<td>European mantis (5.78·10⁴)</td>
<td>Orthogonius baconi (14·10⁴)</td>
</tr>
<tr>
<td>Nile crocodile (252·10⁴)</td>
<td>Deropteryx desiccata (218·10²)</td>
<td>Telephone-pole beetle (11.8·10⁴)</td>
</tr>
<tr>
<td>Gharial (52.5·10⁴)</td>
<td></td>
<td>Cucujus (494·10²)</td>
</tr>
<tr>
<td>Mary River turtle (120·10⁴)</td>
<td>Lepisma saccharina (11.3·10⁴)</td>
<td>Dorcus rectus (1.31·10⁴)</td>
</tr>
<tr>
<td>Pig-nosed turt. (107·10⁴)</td>
<td>Thermobia domestica (3.34·10⁴)</td>
<td>Mormolyce phyllodes (1.16·10⁴)</td>
</tr>
<tr>
<td>Loggerhead sea turt. (54.8·10⁴)</td>
<td></td>
<td>Orthogonius baconi (413)</td>
</tr>
<tr>
<td>Leatherback sea turt. (53.6·10⁴)</td>
<td>Velia caprai (2.18·10⁴)</td>
<td>Paracupes ascius (134)</td>
</tr>
<tr>
<td>Aldabra giant tort. (22.9·10⁴)</td>
<td>Ranatra fusca (2.09·10⁴)</td>
<td>192647</td>
</tr>
<tr>
<td>98980000</td>
<td>788900</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5. Seven-tab “menu” constructed of 44 animals by Algorithm 1,2 ($\tau$ is equal to the mean of $A_1[P]$) basing on “taxonomic” association function (see the details in the text); the names in parentheses in the first row near $T_i$ are possible names of the tabs, given by the author; for the details on other notation see the caption of Fig. 4

**References**


