

最適なメニュー階層構造を求めるアルゴリズム

高田 喜朗

辻野 嘉宏

都倉 信樹

y-takata@ics.es.osaka-u.ac.jp tsujino@ics.es.osaka-u.ac.jp tokura@ics.es.osaka-u.ac.jp

大阪大学基礎工学部情報工学科

計算機アプリケーションのユーザインタフェースとしてメニューがよく用いられているが、アプリケーションが複雑になるにつれ、より多数の項目を持つメニューが見られるようになっていく。多数の項目を持つメニューシステムにおいては、一般にメニュー階層構造を用いることが多い。

与えられた項目の集合を平均アクセス時間の小さいメニュー階層構造に配置することを考える。このとき、メニューの階層構造がその意味関係を反映したものでなければ、ユーザにとって扱いにくい。

本稿では、与えられた項目の意味的分類を保つ階層構造すべての中からメニューの平均アクセス時間が最小な階層構造を求める、効率のよいアルゴリズムを示す。

An Algorithm for Finding Optimal Menu Hierarchy

Yoshiaki Takata Yoshihiro Tsujino Nobuki Tokura

Department of Information and Computer Sciences
Faculty of Engineering Science, Osaka University

Toyonaka 560 Japan

As the computer application becomes more complex, it should have more complex menu structure.

The menu options should be arranged in a menu hierarchy which preserves the natural semantical relation, while the access time (menu option selection time) should be minimized so that a user navigates it easily and efficiently.

This paper presents an efficient algorithm to find the optimal menu hierarchy which minimize the expected access time among all hierarchy which keeps a given semantical grouping of menu options.

1 はじめに

近年、計算機アプリケーションの高度化に伴いその操作方法も複雑になる傾向にあり、そのためグラフィカルユーザインタフェース (GUI) を採用するアプリケーションが増加してきた。マウスやキーボードにより複数の項目 (オプション) から必要なものを選択するメニューがよく用いられているが、アプリケーションが複雑になるにつれ、より多数の項目を持つメニューが見られるようになっていく。その場合、一般に、まずオプションを適当に階層的に分類して、上位階層から順に下位へ選択していくという形のメニュー階層構造を使うことが多い。よりよいユーザインタフェースを実現するためには、良好なメニュー方式とメニュー階層構造を設計する必要がある。メニュー方式についてはすでにいろいろな研究が行なわれている [3],[4]。本稿では、メニュー階層構造の問題を扱う。

メニュー階層構造の良否の尺度として、平均アクセス時間が考えられる。これは、階層構造の根から最終的に選択すべきオプション (終端オプション) を選択するまでに必要な時間の平均である。これを小さくするようなメニュー階層構造が望ましい。

ここで、終端オプションの集合が与えられたとしてそれをメニュー階層構造に配置することを考える。極端な解として、全終端オプションをひとつのメニューに置いたような高さが 0 の階層構造、あるいは各メニューのオプション数を 2 とした高さの大きい階層構造が考えられる。しかし、前者ではその 1 個のメニューでの選択時間が大きくなり、後者では各メニューごとの選択時間は小さいが選択操作を階層の深さだけ繰り返すために、どちらも平均アクセス時間は小さくない。

また、メニューの階層構造がその意味関係を反映したものでなければ、その階層構造はユーザにとって扱いにくいものになってしまう。すなわち、オプションの選択は、その子孫のオプションを意味的に分類するものでなくてはならない。

Fisher らは、メニュー設計者から与えられたオプションの意味的分類がなされた階層構造から、その分類を崩さないようなある制限の下で平均アクセス時間を最小にする階層構造を求める方法を示した [2]。

しかし、Fisher らの方法では次のような点で問題がある。

- Fisher らの方法では、制限が強すぎて、メニュー設計者から与えられた階層構造から得られる意味的分類を保った階層構造すべてを調べ尽くしてはいない。
- 現実のメニューでは多く見られるような、中間オプション

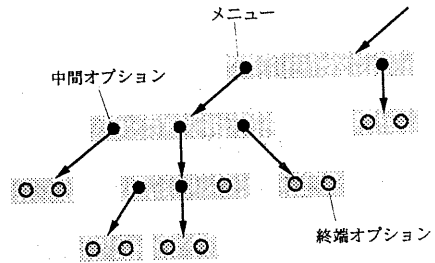


図 1: メニュー階層構造

と終端オプションが混在するメニューを許さない。

本稿では、与えられた項目の意味的分類を保つ階層構造すべての中からメニューの平均アクセス時間が最小な階層構造を求める、効率のよいアルゴリズムを示す。

2 モデル

2.1 メニュー階層構造

[定義 1] メニュー階層構造は、メニューを節点とする木である (図 1)。

メニューは、1 個以上の項目 (オプション) の集合である。

メニュー階層構造の有向辺の始点となるメニューには、その有向辺の終点となるメニューを表すオプションが一つの有向辺に対して一つずつ存在する。そのオプションを中間オプションと呼ぶ。それ以外のオプションを終端オプションと呼ぶ。

根メニューから順にメニューオプションの選択を行なうとき、各メニューで目的のオプションに対する選択すべき親オプションを決定できなければならない。すなわちこれは、1 個のメニュー内のオプションが、そこから辿れる終端オプションの集合を分割していることに等しい。その分割は、一般的にはオプションが表す意味によって行なわれるのが自然である。意味によってグループ化されたオプションの集合をカテゴリと呼ぶことにする。ここで対象とすべきメニュー階層構造は、親オプションの選択が目的のオプションの属するカテゴリの選択と等しいような階層構造である。

[定義 2] 次のような性質を満たすメニュー階層構造を *semantically well formed* (以下 SWF と略す) を満たす階層構造という。

各メニュー M について、 M のオプション o から到達できるすべての終端オプションは、 o で表されるカテゴリに属す。

o から到達できない終端オプションは o で表されるカテゴリに属さない。 □

SWF を満たすメニュー階層構造は、終端オプションのカテゴリの集合を表現していると考えられる。つまり、各中間オプションがカテゴリの名前を表し、そこから辿れる終端オプションの集合がそのカテゴリに一致する。メニュー階層構造が SWF を満たしていれば、ユーザにとって使いやすい階層構造であると考えられる。

ここでは、メニュー設計者が予め考えられるカテゴリをすべて与えとし、それを基に、各カテゴリが含む終端オプションを変えないような階層構造の中から適するものを選ぶことを考える。

[定義 3] メニュー設計者が終端オプションを分類したカテゴリの階層をメニュー階層構造として表したものを、seed 階層構造という (図 2)。 □

ここでは議論を簡単にするため、seed 階層構造が木であると仮定する。一般の場合 (seed 階層構造が DAG の場合) については、後に議論する。

[定義 4] 次のような操作を縮約操作という。

階層構造 H 中の任意のメニュー M の任意の中間オプションを、それが指す子メニュー M' の全オプションに置き換える操作。

こうして得られるメニューと階層構造をそれぞれ $[M \leftarrow M']$, $H[M \leftarrow M']$ と表す。また、階層構造 H 中のメニュー M に縮約されている子メニュー M' の縮約を解消して得られる階層構造を $H[M \rightarrow M']$ と表す。また、メニュー M に縮約されているメニューの集合を $reduced[M]$ と表す。 □

図 3 の構造はどちらも図 2 の構造に 1 回の縮約操作を行なったものである。すなわち、(i) は図形、(ii) は作図という中間オプションを縮約した。

[定義 5] seed 階層構造に縮約操作を 0 回以上繰り返して得られる階層構造を縮約階層構造と呼ぶ。 □

縮約操作と SWF について以下の補題が成り立つ。

[補題 1] SWF を満たす階層構造中の任意のメニュー M に任意の子メニュー M' を縮約したとき、縮約後の階層構造も SWF を満たす。 □

[補題 2] 与えられた seed 階層構造に対し、SWF を満たす階層構造はすべて seed 階層構造の縮約階層構造である。 □

補題 1, 2 より縮約階層構造のみを考えれば必要十分であることがわかる。

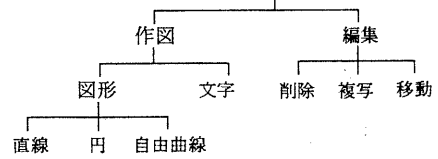


図 2: seed 階層構造

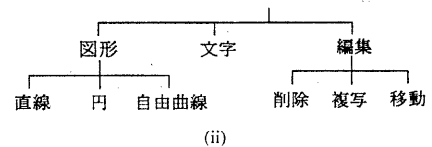
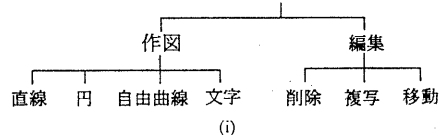


図 3: 縮約操作

2.2 平均アクセス時間

メニュー階層構造の平均アクセス時間は、終端オプションのアクセス時間の期待値である。これを $E[T]$ と書く。終端オプションのアクセス時間は、途中アクセスされるメニューそれぞれでかかる選択時間の和である。ここではメニュー階層構造と、各終端オプションへのアクセス確率が与えられるとする。

まず各メニューでの選択時間について考える。オプション数 m のメニューでの平均選択時間 $t(m)$ は次のような式で表されると仮定する。

$$t(m) = c(m+1)/2 + s + r,$$

c : オプション 1 個あたりの選択判定時間,

s : 選択操作 (打鍵, クリック) 時間,

r : 計算機の反応時間。

c, s, r は定数である。これは、メニュー内の各オプションを順に 1 個ずつ目的のものかどうか判定し目的のオプションに到達するまでの平均時間が $c(m+1)/2$ 、その後ユーザが選択操作を行なうのに s 、計算機がその操作を受け取り中間オプションの場合には次のメニューを表示するのに r だけ時間がかかるというモデルである。これは選択時間の第 1 次近似モデルとして妥当であると考えられる。

終端オプション o のアクセス確率を p_o とし、メニュー M のアクセス確率を $P(M)$ 、中間または終端オプション o_i のア

クセス確率を $P(o_i)$ とすると,

$$P(M) = \sum_{o_i \in M} P(o_i),$$

$$P(o_i) = \begin{cases} p_{o_i}, & o_i \text{が終端オプション} \\ P(M'), & o_i \text{が中間オプション} \\ & (M' \text{は } o_i \text{が表す子メニュー}) \end{cases}$$

が成り立つ.

メニュー M を根とする部分階層構造の平均アクセス時間を $E[T|M]$ で表す. $E[T] = E[T|\text{根}]$ である. これを用いてメニュー階層構造の平均アクセス時間は次のように再帰的に表せる.

$$E[T|M] = t(|M|) + \sum_{M' \in \text{child}[M]} E[T|M']P(M'|M) \quad (1)$$

ただし, $|M|$ はメニュー M のオプション数, $\text{child}[M]$ は M 中の中間オプションが表す子メニューの集合である. $P(M'|M)$ はメニュー M がアクセスされたときにその子メニュー M' がアクセスされる遷移確率で, $P(M'|M) = P(M')/P(M)$ である.

3 最適なメニュー階層構造を求めるアルゴリズム

3.1 問題の定義

[定義 6] 最適なメニュー階層構造を求める問題は次のように定義される.

入力 seed 階層構造と終端オプションのアクセス確率

出力 seed 階層構造から得られる縮約階層構造のうち最も平均アクセス時間の小さいもの $\square$

この出力のような階層構造を最適なメニュー階層構造と呼ぶ.

seed 階層構造のメニュー数を n とすると縮約階層構造は, 根以外の各メニューについて縮約操作をするかどうか選べるので 2^{n-1} とおりある. これらの中から効率良く最適な階層構造を見つけたい.

3.2 メニュー階層構造の性質

任意の縮約階層構造に 1 回縮約操作を行なって新しい縮約階層構造を作った際, 階層構造の平均アクセス時間の変化について次の補題が成り立つ. これにより 1 個のメニューについて, 各子メニュー独立に平均アクセス時間を小さくするよう縮約するかしないか決めれば, 平均アクセス時間を最小にする縮約操作の組合せが得られる.

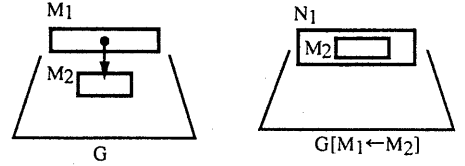


図 4: 縮約による階層構造の変化

[補題 3] メニュー階層構造 H の任意のメニュー M_1 について, その任意の子メニュー M_2 を縮約したときの H の平均アクセス時間の変化は, M_1 の M_2 以外の子孫メニューを縮約するかどうかによらずに定まる.

(証明) M_1 を根とする部分階層構造を G とする.

M_1 への縮約が変化したときの H の平均アクセス時間の変化は定義より, G の平均アクセス時間の変化に M_1 のアクセス確率 $P(M_1)$ を掛けたものである. $P(M_1)$ は M_1 への縮約の変化に関わらず一定なので, G の平均アクセス時間の変化について調べればよい.

$[M_1 \leftarrow M_2]$ を N_1 とする (図 4). このとき $|N_1| = |M_1| + |M_2| - 1$ である.

また, メニューのアクセス確率は縮約によって変化しないので, $P(M_1) = P(N_1)$ である.

定義より G の平均アクセス時間は

$$E[T|M_1] = t(|M_1|) + \sum_{M' \in \text{child}[M_1]} E[T|M']P(M'|M_1),$$

$G[M_1 \leftarrow M_2]$ の平均アクセス時間は

$$E[T|N_1] = t(|N_1|) + \sum_{M' \in \text{child}[N_1]} E[T|M']P(M'|N_1)$$

である. ここで,

$$\text{child}[N_1] = \text{child}[M_1] \cup \text{child}[M_2] - \{M_2\}$$

であるから, 縮約による平均アクセス時間の変化 g は

$$\begin{aligned} g &= E[T|N_1] - E[T|M_1] \\ &= t(|N_1|) - t(|M_1|) + \sum_{M' \in \text{child}[M_2]} E[T|M']P(M'|M_1) \\ &\quad - E[T|M_2]P(M_2|M_1) \\ &= t(|M_1| + |M_2| - 1) - t(|M_1|) \\ &\quad + \sum_{M' \in \text{child}[M_2]} E[T|M']P(M'|M_1) \\ &\quad - \left(t(|M_2|) + \sum_{M' \in \text{child}[M_2]} E[T|M']P(M'|M_2) \right) \\ &\quad \times P(M_2|M_1) \end{aligned}$$

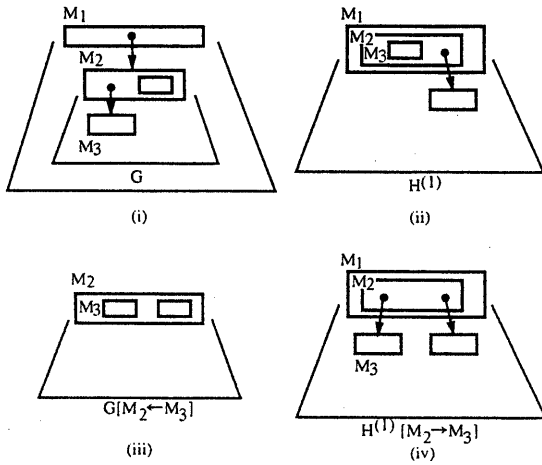


図 5: 縮約とその解消

$$\begin{aligned}
 &= t(|M_1| + |M_2| - 1) - t(|M_1|) - t(|M_2|)P(M_2|M_1) \\
 &= \frac{c}{2}(|M_2| - 1)(1 - P(M_2|M_1)) - (c + s + r)P(M_2|M_1) \\
 &\quad \dots\dots (2)
 \end{aligned}$$

となり、 g は M_2 のオプション数と M_1 からの遷移アクセス確率によって決まる。遷移アクセス確率 $P(M_2|M_1)$ は M_1 や M_2 に子孫メニューが縮約されても一定であるから、 g は M_1 の M_2 以外の子孫メニューの縮約に関係しない。□

また、次の補題が成り立つ。

[補題 4] メニュー階層構造 H の任意のメニュー M_1 とその子メニュー M_2 について、 M_2 を根とする部分階層構造がその部分のみを見たとき最適であるとする。 M_2 への子孫メニューの縮約を変化させたのち M_1 に M_2 を縮約することを考える。 H 中で M_2 に縮約されていなかったメニュー M_3 を M_2 に縮約して得られる階層構造を $H^{(1)}$ とする。すなわち $H^{(1)} = H[M_2 \rightarrow M_k]_{M_k \in S[M_2 \leftarrow M_3]}[M_1 \leftarrow M_2]$ 、ただし $S \subseteq \text{reduced}[M_2]$ 。このとき、 $H^{(1)}$ が、 M_3 を縮約しない階層構造すべて、すなわち $H[M_2 \rightarrow M_k]_{M_k \in S'}[M_1 \leftarrow M_2]$ より平均アクセス時間が短くなることはない。ただし $S' \subseteq \text{reduced}[M_2]$ 。

(証明) M_1 を根とする部分階層構造について平均アクセス時間の大小を考えれば十分である。

M_2 を根とする最適な階層構造を G とする。 G と $G[M_2 \leftarrow M_3]$ の平均アクセス時間の差を考えると、式(2)と同様に

$$\begin{aligned}
 &(E[T] \text{ of } G[M_2 \leftarrow M_3]) - (E[T] \text{ of } G) \\
 &= \frac{c}{2}(|M_3| - 1) - t(|M_3|)P(M_3|M_2)
 \end{aligned}$$

である(図 5 (i),(iii))。 G は最適な階層構造であるので左辺は 0 以上。

$H^{(1)}$ と $H^{(1)}[M_1 \rightarrow M_3]$ の平均アクセス時間の差は

$$\begin{aligned}
 &(E[T] \text{ of } H^{(2)}) - (E[T] \text{ of } H^{(1)}) \\
 &= -\left\{\frac{c}{2}(|M_3| - 1) - t(|M_3|)P(M_3|M_1)\right\}
 \end{aligned}$$

である(図 5 (ii),(iv))。 $0 \leq P(M_3|M_1) \leq P(M_3|M_2)$ 、 $0 \leq t(|M_3|)$ であるから

$$\begin{aligned}
 &(E[T] \text{ of } H^{(1)}[M_1 \rightarrow M_3]) - (E[T] \text{ of } H^{(1)}) \\
 &= -\frac{c}{2}(|M_3| - 1) + t(|M_3|)P(M_3|M_1) \\
 &\leq -\frac{c}{2}(|M_3| - 1) + t(|M_3|)P(M_3|M_2) \\
 &= (E[T] \text{ of } G) - (E[T] \text{ of } G[M_2 \leftarrow M_3]) \\
 &\leq 0
 \end{aligned}$$

$H^{(1)}[M_1 \rightarrow M_3]$ は、 M_2 への縮約を変化させる際に M_3 を M_2 に縮約しなかった階層構造と同じものである。すなわち M_3 を縮約せずに $H^{(1)}$ より平均アクセス時間が大きくない階層構造が存在する。□

最適な部分階層構造の根すべてをその親メニューに縮約したものが最適であるとはいえない。しかし、この補題により、最適な部分階層構造の根について、それに縮約されている子孫メニューからいくつかの縮約を解消し、親メニューに縮約する場合のみ考えればよいことがわかる。

[補題 5] 最適な縮約階層構造の任意の部分階層構造は、同じメニューを根とする seed 階層構造の部分階層構造に対して最適な縮約階層構造となる。

(証明) 背理法により示す。

最適な縮約階層構造 H の部分階層構造 H' が、同じメニューを根とする seed 階層構造の部分階層構造 H'_{seed} に対して最適でないと仮定する。seed 階層構造 H'_{seed} に対して最適な階層構造を H'_{opt} とすると、 H'_{opt} の平均アクセス時間は H' より小さい。よって H 内の H' を H'_{opt} に換えた階層構造を考えると、式(1)よりこの階層構造の平均アクセス時間は H より小さく、 H が最適であることに反する。□

よって縮約を解消する場合、そこから下位の部分階層構造について再び最適な部分階層構造に戻せばよいことがわかる。

3.3 アルゴリズム

以上からアルゴリズムの動作の概略は次のようになる。

seed 階層構造のメニュー M について次のことを順に行なう。

1. M の子メニューを根とする部分階層構造を再帰的に最適化する。
2. M の子メニューを M に縮約すべきかを調べる。
 M の各子メニュー M' について、 M' を根とする最適部分階層構造から 0 個以上の縮約を解消した構造を考え、その根を M に縮約した場合と、 M' を M に縮約しない場合のうち、最適なものを選ぶ。
3. 縮約を解消した部分を再び最適な階層構造に戻す。

上の 2. で、縮約を解消するメニューを探すときは解消した部分を最適な階層構造に戻した場合の平均アクセス時間の変化も考慮しなければならない。3. も含め、これは再帰的に最適な部分階層構造を求めた際にその形状と平均アクセス時間についての情報を記憶しておくことで再計算の手間を減らせる。

アルゴリズムの詳細を図 6 に示す。ここで、階層構造 H の根を $root[H]$ と書く。ただし、メニューのアクセス確率は終端オプションのアクセス確率から計算されるが、これらはメニューの数に比例する時間で全部求まるので、簡単のためここでは予め計算されているものとする。

3.4 アルゴリズムの正当性

[定理 6] OPT は、与えられた *seed* 階層構造 H から得られる縮約階層構造のうち最も平均アクセス時間が小さい階層構造を返す。

(証明) H の高さについての帰納法を用いる。

H の高さが 0 のとき、 H の縮約階層構造は H 自身のみであるから最適な縮約階層構造は H 自身である。このとき OPT は H を返すので題意は成り立つ。

H の高さが 1 以上のとき、 H より高さの低い *seed* 階層構造について題意が成り立つと仮定する。

補題 4, 5 より、最適な H の縮約階層構造において、

- 根に縮約されるメニューは、いずれも H の根の子を根とする最適部分階層構造の根またはそれに縮約されているメニューのいずれか、
- 根の子を根とする部分階層構造はその部分のみを見たとき最適な階層構造

となる。この条件にあう H の縮約階層構造すべてを調べ、平均アクセス時間が最小のものを選べばよい。

$OPT(H: \text{hierarchy})$

/* seed 階層構造 H に縮約操作を行ない、最適化する */

```

1  opt-nests[H] ← ∅
   /* 最適化したときに根に縮約されている子孫メニューの集合 */
2  if  $H$  の高さが 0
3  then return
4  for each  $H' \in \{root[H]$  の子を根とする部分階層構造}
5  do  $OPT(H')$ 
6   $g \leftarrow \frac{c}{2}(|root[H']|-1) - t(|root[H']|)P(root[H']|root[H])$ 
   /*  $H'$  の根を縮約したときの  $E[T]$  の変化量 */
7   $diff\text{-}by\text{-}nest[root[H']] \leftarrow -(c+s+r)$ 
8   $FCC(H, H', root[H'], cutpoints, \Delta g)$ 
9  if  $g + \Delta g < 0$ 
10 then for each  $M \in cutpoints$ 
11     do  $H' \leftarrow H'[root[H'] \leftarrow M]$ 
12         $H'' \leftarrow M$  を根とする
           seed 階層構造の部分階層構造
            $REOPT(H'')$ 
13         $H \leftarrow H[root[H]] \leftarrow root[H']$ 
14         $opt\text{-}nests[H] \leftarrow opt\text{-}nests[H]$ 
            $\cup reduced[root[H]]$ 
15         $diff\text{-}by\text{-}nest[root[H']] \leftarrow g + \Delta g$ 
   /*  $H'$  の根を縮約したことによる  $E[T]$  の変化量 */

```

$FCC(H, H': \text{hierarchy}; M: \text{menu};$

$C: \text{set of menu}; \Delta g: \text{time})$

/* H の根に H' の根を縮約するとして、 H' の根に縮約されたメニュー M および M に縮約された子孫メニューから、分離すると $E[T]$ of H が最小になるようなメニューの集合 C とそのときの $E[T]$ of H の変化量 Δg を求める */

```

1   $C_1 \leftarrow \emptyset$ 
2   $\Delta g_1 \leftarrow 0$ 
3  for each  $M' \in reduced[M] \cup \{\text{seed 階層構造での } M \text{ の子}\}$ 
4  do  $FCC(H, H', M', C', \Delta g')$ 
5   $C_1 \leftarrow C_1 \cup C'$ 
6   $\Delta g_1 \leftarrow \Delta g_1 + \Delta g'$ 
   /*  $M$  の内側での最適な分離位置と  $E[T]$  の変化量 */
7   $\Delta g_2 \leftarrow -diff\text{-}by\text{-}nest[M]P(root[H']|root[H])$ 
    $-\frac{c}{2}(|M|-1)(1-P(root[H']|root[H]))$ 
   /*  $M$  を分離したときの  $E[T]$  of  $H$  の変化量 */
8  if  $\Delta g_1 < \Delta g_2$ 
9  then  $C \leftarrow C_1$ 
10      $\Delta g \leftarrow \Delta g_1$ 
11  else  $C \leftarrow \{M\}$ 
12      $\Delta g \leftarrow \Delta g_2$ 
13 if  $\Delta g \geq 0$ 
14 then  $C \leftarrow \emptyset$ 
15      $\Delta g \leftarrow 0$ 

```

$REOPT(H: \text{hierarchy})$

/* seed 階層構造 H を再び最適化する */

```

1  for each  $M \in opt\text{-}nests[H]$ 
2  do  $H \leftarrow H[root[H]] \leftarrow M$ 
3  for each  $H' \in \{root[H]$  の子を根とする部分階層構造}
4  do  $REOPT(H')$ 

```

図 6: アルゴリズム

OPT の 5 行目で, H の根の子を根とする部分階層構造に OPT を再帰的に適用して得られる階層構造は, 帰納仮定より最適な階層構造である. この最適な部分階層構造を H'_{opt} とする.

H'_{opt} の根およびそれに縮約されているメニューの一部または全部を H の根に縮約する場合を考える. これはすなわち H'_{opt} の根に縮約されている子孫メニューのうち 0 個以上の縮約を解消したのち, H の根に縮約することである. ただし, このとき縮約を解消したメニューを根とする部分階層構造は, その部分について最適な階層構造に戻すとする.

補題 3 より, 互いに先祖/子孫関係のないメニュー同士について, H の平均アクセス時間が最も減少するように縮約を解消するかどうかを独立に選べば, すべての組合せのうち最も H の平均アクセス時間を小さくするものとなる. 互いに先祖/子孫関係のあるメニュー同士の場合は, 先祖を分離すると子孫も分離されるので, 子孫メニューの分離の組合せと上位のメニューを分離した場合とを再帰的に調べ, より H の平均アクセス時間を小さくするものを選べばよい.

OPT の 8 行目で FCC が呼ばれると, FCC の 3 行目から 6 行目までのループと再帰呼出しにより, 上記のような, H'_{opt} の根を縮約するとき, H'_{opt} の根に縮約されている各子孫メニューの分離の組合せのうち最も H の平均アクセス時間を小さくするものが得られる. ここで得られた組合せに従って H'_{opt} の根から子孫メニューを分離し, 分離したメニューを根とする部分階層構造をその部分について最適な階層構造に戻したものを H' とする.

補題 3 より, 各 H' について, H の平均アクセス時間が最も減少するように H' の根を縮約するかどうかを独立に選べば, すべての組合せのうち最も H の平均アクセス時間を小さくするものとなる.

OPT の 4 行目から 16 行目のループで, 各 H' について独立に縮約するかどうか選ぶことですべての組合せのうち最も H の平均アクセス時間を小さくするものが得られる.

OPT の 9 行目で H' の根を縮約することを選んだとき, OPT は計算結果に従ってメニューの分離と部分階層構造の再構築, および H' の根の縮約を行なう.

よって OPT は, 上記の条件にあう H の縮約階層構造すべてのうち H の平均アクセス時間を最小にするものを返す.

帰納法により, すべての seed 階層構造 H について題意は成り立つ. $\square$

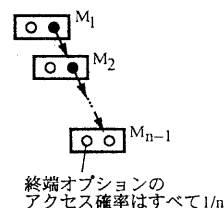


図 7: 最悪時間計算量を実現する場合

3.5 アルゴリズムの評価

seed 階層構造 H 中のメニューの数を n とする.

データ構造として, 各メニューは seed 階層構造での子メニューとその子メニューを縮約したかどうかのフラグのリストを持ち, そのフラグを変更することで縮約およびその解消を行なうとする. これにより, 1 個のメニューの縮約およびその解消は単位時間で実行できる.

最悪時間計算量

- OPT は, H 中の各メニューにつきちょうど 1 回ずつ呼ばれる.
- FCC が OPT から 1 回呼ばれたとき, FCC は再帰的に, H' の根に縮約されたメニューについてちょうど 1 回ずつ呼ばれる.
OPT 4 行目からの for ループで, 各 H' について根にネストされたメニューは重複していない. よって, OPT 1 回につき FCC は $O(n)$ 回呼ばれる.
- REOPT が OPT から 1 回呼ばれたときの計算量は, 入力された部分階層構造内のメニュー数に比例する. また, OPT 12 行目で, M を根とする seed 階層構造の部分階層構造を得る計算の計算量も, 得られる部分階層構造内のメニューの数に比例する.
OPT 10–13 行目の for ループで, M を根とする部分階層構造内のメニューは重複していない. よって, OPT 1 回につき REOPT の計算量は $O(n)$ である.

以上より, OPT の最悪時間計算量は $O(n^2)$ 時間である.

また, 図 7 のような階層構造を seed 階層構造とした場合を考える. $s+r$ が c より十分大きい ($s+r > nc/2$) とし, 最適な縮約階層構造は根メニューに全メニューを縮約した高さが 0 の階層構造となる. このときの OPT の時間計算量は $\Omega(n^2)$ である.

以上より OPT の最悪時間計算量は $\Theta(n^2)$ 時間である.

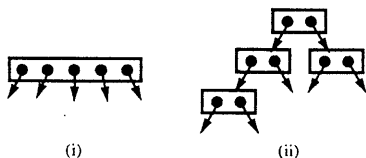


図 8: 階層構造の変形の例

平均時間計算量

ここでの平均は、同じメニュー数の seed 階層構造の形がすべて等しい確からしさで現れるとしてその平均を考える。アクセス確率については最悪時で評価する。

まず、簡単のため各メニューに含まれる中間オプションの数を高々2に制限した seed 階層構造について考える。

メニュー数 n のときの OPT の平均時間計算量を $T(n)$ とする。階層構造の左部分木のメニュー数が $k = 0, 1, \dots, n-1$ である場合それぞれについて平均を取ることから

$$T(n) \leq \frac{1}{n} \sum_{k=0}^{n-1} \{T(k) + T(n-1-k)\} + O(n)$$

が成り立つ。 $T(0), T(1) = O(1)$ より $T(n) = O(n \log n)$ となる [1]。

次に一般の seed 階層構造について考える。

3 個以上の中間オプションのあるメニューに対し子孫メニューを追加して各メニューに含まれる中間オプション数が高々2になるように変形した階層構造を考える。たとえば図 8 (i) のメニューに対して (ii) のようなメニューを考える。ただし、このとき追加した子孫メニューが作る部分階層構造の葉の深さなるべく均等になるようにする。

追加したメニュー数は元のメニューの中間オプション数すなわち子メニュー数を k として $k-2$ であるから、変形した階層構造のメニュー数は高々 $2n$ である。したがって前述の結果より変形した階層構造に対する OPT の平均時間計算量は $O(n \log n)$ 時間である。元の階層構造に対する時間計算量は、メニュー数が少ないので明らかにこれより大きくない。

よって、一般の seed 階層構造に対して OPT の平均時間計算量は $O(n \log n)$ 時間である。

4 あとがき

本稿では、与えられたメニュー階層構造について項目の意味的分類を保つメニュー階層構造すべての中からメニューの平均アクセス時間が最小な階層構造を求める、効率のよいアルゴリズムを示した。このアルゴリズムは、与えられたメニュー

階層構造のメニュー数を n としたとき最悪時に $O(n^2)$ 時間、平均 $O(n \log n)$ 時間で解を求める。

本稿ではカテゴリの交差のない階層構造について考えた。しかし、ある終端オプションが互いに包含関係のない複数のカテゴリに属するような場合も考えられる。そのときは seed 階層構造も木構造ではなく DAG となる。

seed 階層構造に DAG を許す場合、複数のメニューを親にもつメニューを根とする部分階層構造の扱いに 2 通りが考えられる。ひとつは、その部分階層構造の根を親メニューに縮約せず、部分階層構造がどの親メニューから見ても同じであるようにするものである。この場合、そのような部分階層構造に対して最適な階層構造を求めた後その部分は固定されたものとするこで、本稿のアルゴリズムが適用できる。このとき、時間計算量は、DAG のメニュー数に対し本稿のアルゴリズムと同じである。もうひとつは、部分階層構造が各親メニューごとに存在するとするものである。この場合は、部分階層構造を複製し DAG を木構造に変形した後本稿のアルゴリズムを適用する方法が考えられる。しかし、木構造のメニュー数は DAG のメニュー数より多くなる。

今後の課題として、この問題の計算量の下界を求めること、アルゴリズムを実際のメニューに応用してみるなどがある。

参考文献

- [1] Cormen T. H., Leiserson C. E. and Rivest R. L.: "Introduction to Algorithms", McGraw-Hill, pp.165-167 (1990).
- [2] Fisher D. L., Yungkurth E. J. and Moss S. M.: "Optimal Menu Hierarchy Design: Syntax and Semantics", Human Factors, Vol.32, No.6, pp.665-683 (1990).
- [3] Walker N. and Smelcer J. B.: "A Comparison of Selection Times from Walking and Pull-Down Menus", Proc. of the CHI'90 Conference on Human Factors in Computing Systems, pp.221-225 (April 1990).
- [4] 山本, 魚井, 辻野, 都倉: "リモートプルダウンメニュー方式の提案とその評価", 電子情報通信学会論文誌 (D-I), Vol.J76-D-I, No.7, pp.364-370 (1993-07).