2分探索木における挿入・探索アルゴリズムの解析†

中 村 良 三††

2分探索木で，見出しの探索頻度という重みを付けて一般化したモデルにおける挿入・探索アルゴリズムの系統的な解析は見当たらない．すなわち，任意の順序で登録される個々の見出しが，2分木のどの位置に，どのような確率で配置され，2分木がどのように構成されるかという挿入アルゴリズムの解析から，この2分木において個々の見出しの探索頻度を考慮した探索コストを評価する探索アルゴリズムの解析に至る系統的な解析は明らかでない．本稿では，この問題に対する挿入・探索アルゴリズムの系統的な解析を提示する．はじめに，挿入アルゴリズムの解析では，ある順序で登録される見出しが，ある指定された探索回数で挿入できる可能な場所の数を，新しい順列を生成する方法をもとに解析的に求め，それを表す母関数を導き出している．そして，ある順序で登録される見出しが，2分木のどの位置にどのような割合で配置されるかを表す確率を求めている．次に，探索アルゴリズムの解析では，この母関数を用いることによって，個々の見出しの探索頻度を考慮した探索コストの評価式を見通しよく系統的に導出できることを示している．

1. まえがき

計算機で用いるデータ構造の中で，木はよく使われる構造の一つである．木は有限個の要素すなわち節点の集合からなる．この木の中で，分節数すなわち一つの節点から出ている部分木の数が最大2の順序のある木は，2分木または2分探索木とよばれる．

2分探索木に関する研究は，1960年代から盛んになり，その後B木，Trei木などの新しい探索木も提案されている．それらの木構造を用いた見出し探索法に関する解説ならびに論文は，Knuthによって集大成されている[2]．

これまで2分探索木に関する研究は数多くなされているが，しかし，見出しの探索頻度という重みを付けて一般化したモデルにおいては，2分探索木の挿入・探索アルゴリズムの系統的な解析は見当たらない．すなわち，n個の相異なる見出しの$n!$通りの順列から2分木を生成するとき，任意の順序で登録される個々の見出しが2分木のどの位置にどのような確率で配置され，2分木がどのように構成されるかという挿入アルゴリズムの解析から，n個の見出しからなる$n!$個のすべての2分木における探索コストを評価する探索アルゴリズムの解析に至る系統的な解析は明らかでない[2)~4)]．

本稿では，見出しの探索頻度という重みを付けて一般化したモデルのもとで[5]，2分木における挿入・探索アルゴリズムの系統的な解析を提示する．はじめに，挿入アルゴリズムの解析では，ある順序で登録される見出しが，ある指定された探索回数で挿入できる可能な場所の数を，新しい順列を構成する方法をもとに解析的に求め，それを表す母関数を導き出している．そして，ある順序で登録される見出しが，2分木のどの位置にどのような割合で配置されるかを表す確率を求めている．次に，探索アルゴリズムの解析では，この母関数を用いることによって，個々の見出しの探索頻度を考慮した探索コストの評価式を見通しよく系統的に導出できることを示している．

以下，2章では，木および2分木の基礎概念と定義を概観する．次に，3章で，従来提示されている解析を検討したのち，4章においては，新しい解析法を提示し，見出しの探索頻度を考慮したもとでの探索コストの評価式を導出する．

なお，この解析では，見出しの総数をN，見出しが探索される確率を登録順序に従い，$\rho_i(i=1,\cdots,N)$とし，探索が成功する場合の平均探索コストをS_N，分散をV_N，不成功のときのそれらをそれぞれ$\bar{S}_N$，$\bar{V}_N$とする．

2. 2分探索木法

2.1 木および2分探索木の性質

木は根と呼ぶ節点が一個だけ指定され，根を除いた節点は共通部分をもたない集合に分割され，それらはそれぞれ再び木になるが，この木のことをその根の部分木という．部分木をもたない節点は，端点または葉と呼ばれる．

根は水準1と定義され，この根の直接下にある節点

---

† An Analysis of Insertion and Search Algorithms of a Binary Search Tree by RYOZO NAKAMURA (Faculty of Engineering, Kumamoto University).
†† 熊本大学工学部

は水準2となる．一般に水準 i の直接下にある節点の水準は $i+1$ となる．根から任意の節点に至る枝の数を路長といい，水準 i の節点の路長は $i-1$ となる．

木を構成している節点を内部節点と呼び，任意の節点で，部分木が欠けているところに特別な節点を付け加えたものを外部節点と呼ぶ．このとき，根からすべての内部節点へ至る路長の総和を内部路長，また根からすべての外部節点へ至る路長の総和を外部路長と定義する．

木の中で分節数が2の順序のある木を2分木または2分探索木という．この2分木は，空または根のみの節点からなるか，あるいは根とそれ自身も2分木である左部分木と右部分木からなる．2分木の各節点には，見出しの値がひとつずつ入っており，各節点の見出しの値は，その節点の左部分木の節点の見出しの値より大きく，右部分木の節点の見出しの値より小さい順序木である．このような2分木を構成して見出しの探索を行うことを2分探索木法と言う．

この探索法は，主記憶装置上でのデータ探索法として代表的なもののひとつであり，見出しの集合があらかじめ分かっていない場合，表を用いる2分探索法に比べより有効である．また，この探索法は，2分木を通りがけ順になぞることによって，見出しの値を小さいものから大きいものへ順次探すことも可能で実用上有効である．ここで，2分木と拡張2分木を**図1**に示す．

図1に，7個の丸形の節点すなわち内部節点よりなる2分木と，この2分木に箱形の節点すなわち外部節点を付加した拡張2分木を示す．この図1に示されるように，外部節点の数は内部節点の数よりひとつだけ大きい．この関係は，任意の2分木に対して一般的に成立する．また，内部路長は根から丸形の節点へ至る路の長さの総和として求まり，その値は 11 となる．また，外部路長は根から箱形の節点へ至る路の長さの総和とし，その値は 25 となる．

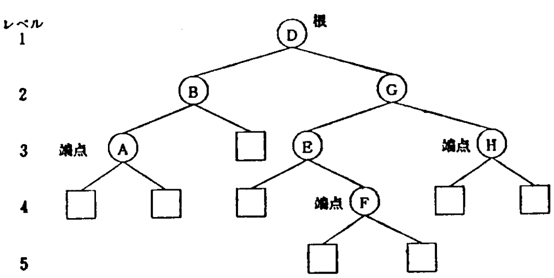


図1 2分木と拡張2分木
Fig. 1 Binary tree and extended binary tree.

2.2 挿入・探索アルゴリズム

2分木の任意の節点において，その節点に置かれる見出しの値は，その節点の左部分木中にあるすべての見出しの値より大きく，また，右部分木中にあるすべての見出しの値より小さいという性質がある．この性質から，与えられた見出しXを登録するアルゴリズムは次のようになる．

[A.1] 2分木が空ならば，新しい節点を作り，そこに見出しを置き登録は終了する．空でなければ，根の見出しの値と登録する見出しXの値とを比較する．

[A.2] 登録する見出しXの値が根の見出しの値より小さいならば，左部分木を再帰的に2分木と見做して，[A.1] へ帰る．見出しXの値が根の見出しの値より大きいならば，右部分木を再帰的に2分木と見做して，[A.1] へ帰る．

たとえば，図1の2分木に見出しCを登録するには，まず，[A.1] により根の見出しDとCとを比較する．次に，[A.2] により見出しCはDより小さいから，左部分木について再帰的に [A.1] を行う．また，[A.2] によりCはBより大きいから右部分木について [A.1] を行う．これは空であるから，そこに新しい節点を作り，Cを登録する．

次に，上述の挿入アルゴリズムによって構成された2分木を用いて，見出しXを探す手順は，次のようになる．

探索する見出しXの値と根にある見出しの値と比較する．一致すれば探索は成功，すなわち，成功探索となり探索は終了する．一致しないならば，見出しXの値が根の見出しの値より小さければ左部分木，大きければ右部分木へ移り，水準をひとつ上げて，この節点を根として同じ比較を繰り返す．最後に，端点に至っても一致する見出しがなければ，探索は失敗，すなわち不成功探索となる．したがって，成功探索では，探す見出しが内部節点のいずれかに一致し，不成功探索では，探す見出しが外部節点のいずれかに到達する．

3. 従来の解析法

Knuth の解析では[2]，N個の節点から構成される2分木の探索コストを，内部路長 I_N と外部路長 E_N を用いて，成功時の探索コスト S_N および不成功時の探索コスト $\bar{S}_N$ を次のように定式化している．ただし，個々の見出しの探索頻度は一様とする．

$$S_N=1+\frac{I_N}{N} \tag{1}$$

$$\bar{S}_N=\frac{E_N}{N+1} \qquad (2)$$

また，根からの路長 k の外部節点を内部節点にすれば，内部路長は k 増加し，外部路長は実質的に $k+2$ 増加する．この関係から，N個の節点からなる2分木では，内部路長 I_N と外部路長 E_N の間には，

$$E_N=I_N+2N \qquad (3)$$

なる関係が成立する．したがって，成功時と不成功時の探索コスト間には次の関係が成り立つ．

$$S_N=\left(1+\frac{1}{N}\right)\bar{S}_N-1 \qquad (4)$$

ところで，n 個の節点（見出し）からなる2分木に $n+1$ 番目の見出しを登録するときの平均比較回数は，$E_n/(n+1)$ となる．これは，n 個の節点からなる2分木の不成功時の平均探索コストの評価式（2）に等しい．

一方，ある見出しを探すために要する比較回数は，その見出しを登録するときに要した他の見出しとの比較回数に1を加えたものに等しい．したがって，N個の見出しからなる2分木のすべての見出しを探すために要する比較回数は NS_N となり，それは次のように表される．

$$NS_N=N+\sum_{i=0}^{N-1}\frac{E_i}{i+1}$$
$$=N+\bar{S}_0+\bar{S}_1+\cdots+\bar{S}_{N-1} \qquad (5)$$

このことから，成功探索コスト S_N と不成功探索コスト $\bar{S}_N$ との間には

$$S_N=1+\frac{\bar{S}_0+\bar{S}_1+\cdots+\bar{S}_{N-1}}{N} \qquad (6)$$

が成立する．

したがって，（4）と（6）から，次の式を得る．

$$(N+1)\bar{S}_N=2N+\bar{S}_0+\bar{S}_1+\cdots+\bar{S}_{N-1}$$

これから，帰納的に次の関係が成立する．

$$N\bar{S}_{N-1}=2(N-1)+\bar{S}_0+\bar{S}_1+\cdots+\bar{S}_{N-2}$$

よって，

$$(N+1)\bar{S}_N-N\bar{S}_{N-1}=2+\bar{S}_{N-1}$$

$$\bar{S}_N=\bar{S}_{N-1}+\frac{2}{N+1} \qquad (7)$$

となる再帰関係式を得る．

ここで，$\bar{S}_0=0$ なる関係から，（7）を非再帰的な閉じた形で表すと次のようになる．

$$\bar{S}_N=2H_{N+1}-2 \qquad (8)$$

また，（4）と（8）より，成功探索コストを次のように導出している．ただし H_N は $1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{N}$ を表す．

$$S_N=2\left(1+\frac{1}{N}\right)H_N-3 \qquad (9)$$

さらに，個々の見出しの探索頻度を考慮すれば，成功時の平均探索コストは，（6）に準じて，次のようになる．

$$S_N=1+\rho_1\bar{S}_0+\rho_2\bar{S}_1+\cdots+\rho_N\bar{S}_{N-1}$$
$$=1+\sum_{i=1}^{N}\rho_i\bar{S}_{i-1} \qquad (10)$$

(10)に（8）を代入すると成功時の平均探索コストは次のようになる．

$$S_N=1+\sum_{i=1}^{N}\rho_i(2H_i-2)$$
$$=2\sum_{i=1}^{N}\rho_iH_i-1 \qquad (11)$$

上述したように，従来の解析においては，探索コストの評価式の導出過程は簡潔であるが，しかし，その挿入アルゴリズムの解析では，ある順序で登録される見出しを2分木に挿入するために要する他の見出しとの比較回数の平均値を解析の基本としているので，任意の順序で登録される見出しが2分木のどの位置にどのような確率で配置されるかは明らかでない．

4. 提示する解析

4.1 挿入アルゴリズムの解析

任意の順序で登録される個々の見出しが，2分木のどの位置にどのような確率で配置されるかを解析する．

はじめに，順列に関して次のことを考察する．

n 個の相異なる対象を1列に並べた列を n 個の対象の順列と言う．この n 個の対象の順列から $n+1$ 個の対象の新しい順列を得る主な方法として次の二つがある．ただし，順列の要素 $\{a_1\ a_2\cdots a_n\}$ は $\{1,2,\cdots,n\}$ なる自然数とする．

方法1　n 個の要素の順列 $a_1\ a_2\cdots a_n$ に自然数 $n+1$ を可能なすべての場所に挿入することによって，新しい $n+1$ 個の順列を構成する．

すなわち，

$n+1\ a_1\ a_2\cdots a_n$,

$a_1\ n+1\ a_2\cdots a_n, \cdots$,

$a_1\ a_2\cdots a_n\ n+1$

によって，$n+1$ 個の対象の $n+1$ 個の順列が得られ，どの順列もただ1回だけ構成される．

方法2　n 個の要素の順列 $a_1\ a_2\cdots a_n$ から次のよ

うにして $n+1$ 個の新しい順列を構成する．

まず，

$a_1\ a_2\cdots a_n\ k_1$,

$a_1\ a_2\cdots a_n\ k_2, \cdots$,

$a_1\ a_2\cdots a_n\ k_{n+1}$

(ただし，$k_i=i-1/2$, $(i=1,2,\cdots,n+1)$)

を作り，ついで各列ごとにその列の要素を小さい順に $1, 2, \cdots, n+1$ で置き換える．

たとえば，順列 $a_1\ a_2\ a_3$ を 2 3 1 とすると，まず，

2 3 1 1/2, 2 3 1 3/2, 2 3 1 5/2, 2 3 1 7/2

が得られ，ついでそれぞれの列において各要素を小さい方から順に 1 2 3 4 で置き換えて，

3 4 2 1, 3 4 1 2, 2 4 1 3, 2 3 1 4

が得られる．

この方法では，k_i $(i=1, 2, \cdots, n+1)$ を順列の並びの右端に付け加える場合を示したが，k_i は左端あるいはそれ以外のある任意の決められた位置に付け加えても同じである．

この提示する解析では，方法2を用いて k_i $(i=1, 2, \cdots, n+1)$ を順列の右端の直前に付け加える方法をとる．すなわち，任意の順列 $a_1\ a_2\cdots a_n$ から

$a_1\ a_2\cdots a_{n-1}\ 1/2\ a_n$,

$a_1\ a_2\cdots a_{n-1}\ 3/2\ a_n$,

$\cdots$,

$a_1\ a_2\cdots a_{n-1}\ n+1/2\ a_n$

なる $n+1$ 個の順列を作り，ついで各列ごとに，その列の要素を小さい順に $1, 2, \cdots, n+1$ で置き換える．このようにして得られた任意の列を

$b_1\ b_2\cdots b_{n-1}\ m\ b_n$

とする．このとき，b_j は a_j または a_j+1 $(j=1, 2, \cdots, n)$ となる．

ここで，元の順列 $a_1\ a_2\cdots a_n$ によって生成される2分木と新しく作られた $n+1$ 個の順列によって生成される2分木の生成過程を考察する．

まず，b_j が a_j または a_j+1 となる上述の関係から，$b_1\ b_2\cdots b_{n-1}$ と $a_1\ a_2\cdots a_{n-1}$ とからなる部分列による2分木の生成過程は等しい．

次に，m が a_n ならば，$b_1\ b_2\cdots b_{n-1}\ m$ までの2分木の生成過程は元の順列 $a_1\ a_2\cdots a_n$ の生成過程に等しく，このとき，b_n は a_n+1 となる．したがって，b_n が2分木に配置される水準は，a_n が配置された水準よりひとつだけ大きくなる．

一方，m が a_n+1 ならば，b_n は a_n となり，b_n が配置される水準は，前と同様に a_n が配置された水準よりひとつだけ大きくなる．

上の考察から，順列 $a_1\ a_2\cdots a_n$ で n 番目に登録された見出し a_n が2分木の水準 k の位置に配置されているとき，この順列から新しく作られた $n+1$ 個の順列のうち，m の値が a_n または a_n+1 の2個の順列では，$n+1$ 番目に登録される見出し b_n は2分木の水準 $k+1$ の位置に配置され，m の値がそれ以外の $n-1$ 個の順列では，b_n は a_n が配置された水準 k と同じ位置に配置される．

たとえば，順列 3 1 4 2 から，まず

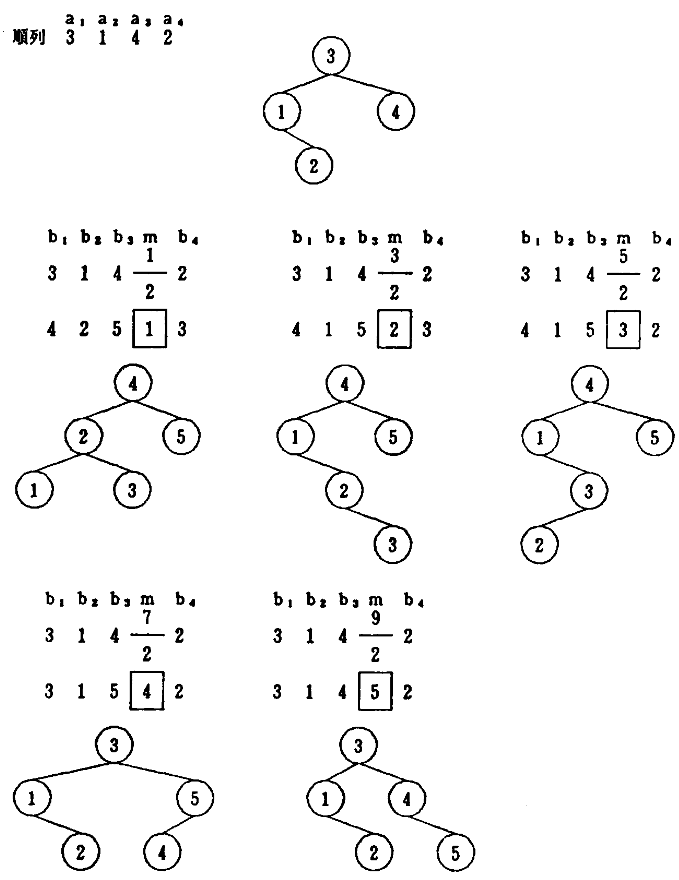


図 2 順列 3 1 4 2 から新しく作られた順列によって生成された2分木

Fig. 2 The binary tree by the new permutations got from the permutation 3 1 4 2.

$$3\ 1\ 4\ \frac{1}{2}\ 2,\quad 3\ 1\ 4\ \frac{3}{2}\ 2,\quad 3\ 1\ 4\ \frac{5}{2}\ 2,$$

$$3\ 1\ 4\ \frac{7}{2}\ 2,\quad 3\ 1\ 4\ \frac{9}{2}\ 2$$

なる5個の順列が作られ，ついでそれぞれの列において各要素を小さい順に 1, 2, 3, 4, 5 で置き換えて，

4 2 5 1 3, 4 1 5 2 3, 4 1 5 3 2, 3 1 5 4 2
3 1 4 5 2

が作られる．これらの順列から構成される2分木を図2に示す．

図2から，元の順列 3 1 4 2 の4番目の要素2は，2分木の水準3の位置に配置されている．新しく作られた5個の順列のうち，m の値が2と3の場合の順列，すなわち 4 1 5 2 3 と 4 1 5 3 2 での5番目の要素3および2は，2分木の水準4の位置に配置されるが，残りの 4 2 5 1 3, 3 1 5 4 2, 3 1 4 5 2 の順列の5番目の要素3および2は，いずれも水準3の位置に配置されていることが容易に確かめられる．

上述から，n 個の見出しで生成された個々の順列から新しく $n+1$ 個の順列を作るとき，この $n+1$ 個の順列のうちの2個の順列では，$n+1$ 番目に登録される見出しが2分木に配置される水準は，元の順列で n 番目に登録された見出しのその水準よりひとつだけ大きいことが分かる．

前述の考察から，n 個の見出しによって生成された2分木に $n+1$ 番目の見出しを登録するとき，k 回の比較を要するような順列の数を C_{nk} とすれば，それは次のような再帰関係式で表すことができる．

$$C_{nk}=2C_{n-1,k-1}+(n-1)C_{n-1,k} \tag{12}$$

ただし，$C_{00}=1$ とする．

ここで，再帰関係式(12)において，順列の数 C_{nk} に伴う各順列とそれらの順列によって構成される2分木を**図3**に示す．ただし，見出しの集合 $\{a_1, a_2, \cdots, a_n\}$ の各要素には，$a_1<a_2<\cdots<a_n$ なる関係がある．また，C_{31}, C_{32}, C_{33} については，それらの2分木は省略している．

この順列の数 C_{nk} からなる母関数を $G_n(z)$ とすると，それは次のような再帰関係式で表される．

$$\begin{aligned}G_n(z)&=\sum_{k=0}^{n}C_{nk}z^k\\&=\sum_{k=0}^{n}\{2C_{n-1,k-1}+(n-1)C_{n-1,k}\}z^k\\&=(2z+n-1)\sum_{k=0}^{n-1}C_{n-1,k}z^k\\&=(2z+n-1)G_{n-1}(z)\end{aligned} \tag{13}$$

次に，(13)を非再帰的な閉じた形で表すと，次のようになる．

$$\begin{aligned}G_n(z)&=(2z+n-1)(2z+n-2)\cdots2z\\&=\sum_{k=0}^{n}\begin{bmatrix}n\\k\end{bmatrix}2^kz^k\end{aligned} \tag{14}$$

ただし，$\begin{bmatrix}n\\k\end{bmatrix}$ は第1種の Stirling 数を表す[1]．

順列の数

$C_{00}=1$

a_1

$C_{11}=2C_{00}$
$=2$

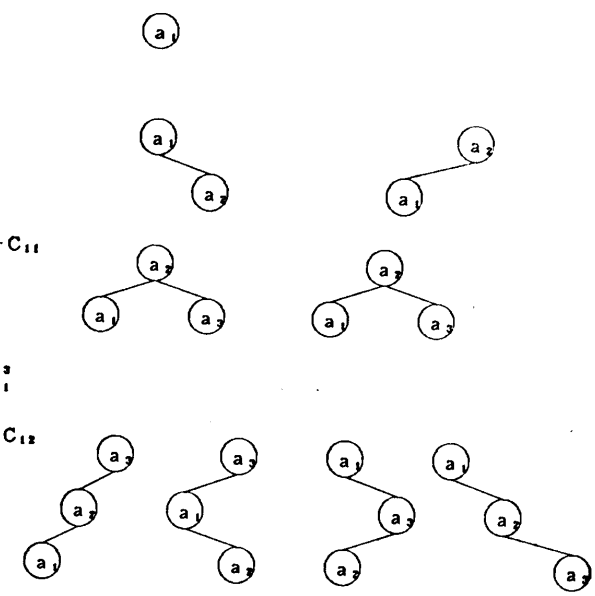
$a_1\ a_2$
$a_2\ a_1$

$C_{21}=2C_{10}+C_{11}$
$=C_{11}$
$=2$

$a_2\ a_1\ a_3$
$a_2\ a_3\ a_1$

$C_{22}=2C_{11}+C_{12}$
$=2C_{11}$
$=4$

$a_3\ a_2\ a_1$
$a_3\ a_1\ a_2$
$a_1\ a_3\ a_2$
$a_1\ a_2\ a_3$



$C_{31}=2C_{20}+2C_{21}$
$=2C_{21}=4$

$a_3\ a_2\ a_1\ a_4$
$a_3\ a_1\ a_2\ a_4$
$a_2\ a_4\ a_3\ a_1$
$a_2\ a_3\ a_4\ a_1$

$C_{32}=2C_{21}+2C_{22}$
$=4+8=12$

$a_2\ a_1\ a_3\ a_4$
$a_2\ a_1\ a_4\ a_3$
$a_3\ a_4\ a_1\ a_2$
$a_3\ a_4\ a_2\ a_1$

$a_3\ a_2\ a_4\ a_1$
$a_4\ a_2\ a_3\ a_1$
$a_3\ a_1\ a_4\ a_2$
$a_4\ a_2\ a_1\ a_3$
$a_1\ a_3\ a_4\ a_2$
$a_2\ a_4\ a_1\ a_3$
$a_2\ a_3\ a_1\ a_4$
$a_1\ a_3\ a_2\ a_4$

$C_{33}=2C_{22}+2C_{23}$
$=2C_{22}=8$

$a_1\ a_2\ a_4\ a_3$
$a_1\ a_2\ a_3\ a_4$
$a_1\ a_4\ a_3\ a_2$
$a_1\ a_4\ a_2\ a_3$
$a_4\ a_1\ a_3\ a_2$
$a_4\ a_1\ a_2\ a_3$
$a_4\ a_3\ a_2\ a_1$
$a_4\ a_3\ a_1\ a_2$

図3 順列の数 C_{nk} に伴う各順列とそれらの順列によって構成される2分木
Fig. 3 The permutations got from C_{nk} and those binary trees.

また，母関数 $G_n(z)$ には，次の関係が成立する．

$$G_n(1)=(n+1)!$$

$$G_n'(z)=\sum_{k=0}^{n}k\begin{bmatrix}n\\k\end{bmatrix}2^kz^{k-1}\text{ から，}$$

$$G_n'(1)=2(n+1)!(H_{n+1}-1)$$

$$G_n''(z)=\sum_{k=0}^{n}k(k-1)\begin{bmatrix}n\\k\end{bmatrix}2^kz^{k-2}\text{ から，}$$

$$G_n''(1)=4(n+1)!(H_{n+1}^2-2H_{n+1}-H_{n+1}^{(2)}+2)$$

ただし，$H_N^{(S)}$ は $1+\frac{1}{2^S}+\frac{1}{3^S}+\cdots+\frac{1}{N^S}$ を表す．

ところで，(14)は n 個の見出しによって生成されている2分木に，$n+1$ 番目の見出しが k 回の比較で挿入される可能な場所が $\begin{bmatrix}n\\k\end{bmatrix}2^k$ 個あると解釈できる．また，任意の比較回数で挿入される場所，すなわち $n+1$ 番目の見出しの挿入可能場所は $(n+1)!$ 個ある．したがって，$n+1$ 番目の見出しが，k 回の比較で，すなわち2分木の水準 $k+1$ の位置に登録される確率を P_{nk} とすれば，P_{nk} は次のようになる．

$$P_{nk}=\frac{\begin{bmatrix}n\\k\end{bmatrix}2^k}{(n+1)!},\quad(k=0,1,2,\cdots,n)\tag{15}$$

4.2 探索アルゴリズムの解析

ある見出しを探索するときの比較回数は，その見出しを挿入するときの比較回数よりひとつだけ大きい．したがって，N個の見出しからなる2分木を探索するとき，ちょうど j 回の比較で見つかる見出しは，登録順番が j 番目以降の見出しのうち，ちょうど $j-1$ 回の比較で挿入された見出しである．

その結果，j 回の探索で見つかる確率は，各見出しの探索頻度を加味し，次のように表される．

$$\rho_jP_{j-1,j-1}+\rho_{j+1}P_{j,j-1}+\cdots+\rho_NP_{N-1,j-1},\quad(j=1,2,\cdots,N)$$

このことから，探索コストの評価式は次のように導出できる．

I）成功探索

このとき，平均探索コストおよび分散は次のように表現される．

$$S_N=\sum_{k=0}^{N-1}(k+1)\sum_{j=0}^{k}\rho_{j+1}P_{jk}$$

$$=\sum_{k=0}^{N-1}\rho_{k+1}\sum_{j=0}^{k}(j+1)P_{jk}\tag{16}$$

$$V_N=\sum_{k=0}^{N-1}\rho_{k+1}\sum_{j=0}^{k}(j+1)^2P_{jk}-S_N^2\tag{17}$$

ただし，$P_{k0}=\begin{cases}0 & (k>0)\\1 & (k=0)\end{cases}$

上の式の P_{jk} を(15)で置き換えると(16)，(17)は次のようになる．

$$S_N=\sum_{k=0}^{N-1}\rho_{k+1}\sum_{j=0}^{k}(j+1)\begin{bmatrix}k\\j\end{bmatrix}2^j/(k+1)!$$

$$=\sum_{k=0}^{N-1}\frac{\rho_{k+1}}{(k+1)!}\left\{\sum_{j=0}^{k}j\begin{bmatrix}k\\j\end{bmatrix}2^j+\sum_{j=0}^{k}\begin{bmatrix}k\\j\end{bmatrix}2^j\right\}$$

$$=\sum_{k=0}^{N-1}\frac{\rho_{k+1}}{(k+1)!}\{G_k'(1)+G_k(1)\}$$

$$=\sum_{k=0}^{N-1}\rho_{k+1}(2H_{k+1}-1)$$

$$=2\sum_{k=1}^{N}\rho_kH_k-1\tag{18}$$

$$V_N=\sum_{k=0}^{N-1}\rho_{k+1}\sum_{j=0}^{k}(j+1)^2\begin{bmatrix}k\\j\end{bmatrix}2^j/(k+1)!-S_N^2$$

$$=\sum_{k=0}^{N-1}\frac{\rho_{k+1}}{(k+1)!}\left\{\sum_{j=0}^{k}j^2\begin{bmatrix}k\\j\end{bmatrix}2^j+2\sum_{j=0}^{k}j\begin{bmatrix}k\\j\end{bmatrix}2^j+\sum_{j=0}^{k}\begin{bmatrix}k\\j\end{bmatrix}2^j\right\}-S_N^2$$

$$=\sum_{k=0}^{N-1}\frac{\rho_{k+1}}{(k+1)!}\{G_k''(1)+3G_k'(1)+G_k(1)\}-S_N^2$$

$$=\sum_{k=0}^{N-1}\rho_{k+1}(4H_{k+1}^2-2H_{k+1}-4H_{k+1}^{(2)}+3)-\left(2\sum_{k=0}^{N-1}\rho_{k+1}H_{k+1}-1\right)^2$$

$$=4\sum_{k=1}^{N}\rho_kH_k^2+2\sum_{k=1}^{N}\rho_kH_k-4\sum_{k=1}^{N}\rho_kH_k^{(2)}-\left(2\sum_{k=1}^{N}\rho_kH_k\right)^2+2\tag{19}$$

II）不成功探索

不成功時の平均探索コスト $\bar{S}_N$ は，N個の見出しで生成された2分木を探索するとき，見出しが見つからない場合の探索回数の平均である．これは，$N+1$ 番目の見出しを登録するときの比較回数の平均と同値になる．よって，$\bar{S}_N$，$\bar{V}_N$ は次のように表現される．

$$\bar{S}_N=\sum_{k=0}^{N}kP_{Nk}$$

$$=\frac{1}{(N+1)!}\sum_{k=0}^{N}k\begin{bmatrix}N\\k\end{bmatrix}2^k$$

$$=\frac{1}{(N+1)!}G_N'(1)$$

$$=2(H_{N+1}-1)\tag{20}$$

$$\bar{V}_N=\sum_{k=1}^{N}k^2P_{Nk}-\bar{S}_N^2$$

$$
\begin{aligned}
&=\frac{1}{(N+1)!}\sum_{k=1}^{N} k^2\begin{bmatrix}N\\k\end{bmatrix}2^k-\{2(H_{N+1}-1)\}^2\\
&=\frac{1}{(N+1)!}\{G_N''(1)+G_N'(1)\}-\{2(H_{N+1}-1)\}^2\\
&=4H_{N+1}^2-6H_{N+1}-4H_{N+1}^{(2)}+6-4(H_{N+1}-1)^2\\
&=2H_{N+1}-4H_{N+1}^{(2)}+2 \qquad (21)
\end{aligned}
$$

上述したように，提示する解析では，２分探索木における挿入・探索アルゴリズムを見通しよく系統的に示した．まず，挿入アルゴリズムの解析では，ある順序で登録される見出しが，２分木のどの位置にどのような確率で配置されるかを明らかにした．次に，探索アルゴリズムの解析における探索コストの導出過程では，確率変数に比較回数すなわち探索回数を取り，その確率分布を，挿入アルゴリズムの解析で導出した確率を用いて表し，探索コストの評価式を見通しよく，かつ系統的に導出している．

5. む す び

ランダムな入力データに伴う２分木の生成過程を解析し，ある順序で登録される見出しが，ある比較回数で，すなわち２分木のある水準に挿入される可能な場所の数を与える母関数を導き出した．そして，この母関数を用いることによって，見出しの探索頻度を考慮に入れて一般化したモデルにおける探索コストを見通しよく系統的に導き出した．この導出された探索コストの評価式は，従来の評価式を包含する．

なお，挿入アルゴリズムの解析で提示した方法は，AVL バランス木[3] の挿入アルゴリズムの解析において，バランス木の高さの期待値，あるいは再バランス化を要する確率などの未解決な問題に対する有効な解決策を与える可能性があると考えている．

謝辞 本論文をまとめるにあたり，熊本大学 松山公一学長ならびに九州大学工学部 牛島和夫教授に貴重なご教示をいただいた．ここに記して謝意を表す．

参 考 文 献

1) Knuth, D. E.: *The Art of Computer Programming*, Vol. 1, *Fundamental Algorithms*, pp. 44-66, Addison-Wesley, Reading, Mass. (1973).

2) Knuth, D. E.: *The Art of Computer Programming*, Vol. 3, *Sorting and Searching*, pp. 406-432, Addison-Wesley, Reading, Mass. (1973).

3) Wirth, N.: *Algorithms + Data structures = Programs*, pp. 189-242, Prentice-Hall, Englewood Cliffs, N. J. (1976).

4) 渋谷，山本：データ管理算法，岩波講座 情報科学 11, pp. 29-52, 岩波書店，東京 (1983).

5) 中村，松山：見出しの探索頻度を考慮した探索路長の考察，情報処理学会論文誌，Vol. 24, No. 4, pp. 505-512 (1983).

(昭和 60 年 4 月 10 日受付)
(昭和 60 年 5 月 9 日採録)