LEFTIST 木の総数にたいする一評価

清　水　道　夫

1. まえがき

情報処理の基本的な技法として，分類（sort）や探索（search）がよく知られているが，これらは，数学的には組合せアルゴリズム（combinatorial algorithm）と呼ばれる。このアルゴリズムを議論するとき，必ずデータ構造（data structure）という概念を伴っており，両者は車の両輪の関係になっている。アルゴリズムの表現はフロー（flow）であるのにたいし，データ構造は階層的表現である木（tree）の場合が多い。

本論文では，組合せアルゴリズムにおけるオープン・プロブレムの1つである「leftist 木の数え上げ問題」を考察する（文献(1), p. 159, 問題34)。leftist 木は，順位付きキュー（priority queue）を表現するものとして，1971年に Clark A. Crane によって提案されたデータ構造である（文献（(1), p. 150)。順位付きキューとは，有効なデータ構造を実現するモデルの1つであり，最小値または最大値を持つデータの見出し（key)が常に先頭に保存され，かつデータの挿入と削除の処理が容易に行なわれるものをいう（文献(1), p. 150)。

データ構造を写像する木にたいする数え上げは，アルゴリズムの性能評価やデータの符号化の問題に関連してしばしば考察されている。ここでは，leftist 木の総数にたいする一評価として，その漸近的な下限を表現する。表現の具体的な特徴は，指数 $e=2.718\cdots$ で表されることである。

leftist 木の例を図1に示す。□は葉を，○は節を表わす。いちばん上の節は根と呼ばれるが，ふつうのイメージでいくと木が逆さになっている。節には葉からの距離を表す値 DIST が付けられており，これについて leftist 木は次の条件を満たしている。葉の DIST はすべて0である。節 p の左子の DIST を X，右子の DIST を Y とすると，$X \geqq Y$ である。節 p の DIST を Z とすると，$Z=Y+1$ である。

2. 漸化式

leftist 木の総数にたいする漸化式を考えてみる。葉の数が n で根の DIST が d であるような leftist 木の総数を $f(n, d)$ とする。根の DIST が d のとき，葉の数が最小な木は 2^d 個の葉を持つ完全2分木である。完全2分木とは，葉がすべて同じレベルにある2分木をいう。根の DIST が

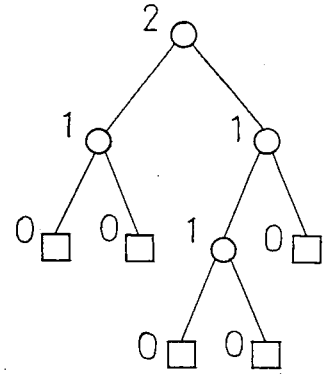

図1　leftist 木

最小になるのは，右部分木が1個の葉である場合だから，その木の葉数にかかわらず常に $d=1$ である。したがって

$$\left.\begin{array}{l} d=0:\ n=1 \\ 1\leqq d\leqq\lfloor\log_2 n\rfloor\dagger:\ n\geqq 2 \end{array}\right\} \quad (1)$$

である。n 個の葉を持つ leftist 木の総数を $f(n)$ とすると，それは $f(n,d)$ をすべての d について加えたものだから，

$$f(n)=\sum_{d=1}^{\lfloor\log_2 n\rfloor} f(n,d):\ n\geqq 2 \quad (2)$$

である。

〔補題1〕 $f(n,d)$ は次式で表わされる。

$$\left.\begin{array}{l} d=0\ \text{のとき} \\ \quad f(n,d)=\begin{cases} 1: n=1 \\ 0: n\geqq 2 \end{cases} \\ d\geqq 1\ \text{のとき} \\ \quad f(n,d)=0: 1\leqq n<2^d \\ \quad f(n,d)=\displaystyle\sum_{s=d-1}^{\lfloor\log_2(n-2^{d-1})\rfloor}\ \sum_{j=2^s}^{n-2^{d-1}} f(j,s) \\ \qquad\qquad \cdot f(n-j,d-1) \\ \qquad\qquad :n\geqq 2^d \end{array}\right\} \quad (3)$$

（証明） $f(1,0)$ は1個の葉を表している。いま，葉の数が n で根の DIST が d であるような木を T とする。$f(j,s)$ と $f(n-j,d-1)$ はそれぞれ木 T の左部分木と右部分木に対する総数を表現している。木の定義より右部分木の根の DIST は $d-1$ で，左部分木の根の DIST s は $s\geqq d-1$ である。完全2分木を考えると，左部分木の葉の数 j は $j\geqq 2^s$ であり，右部分木の葉の数 $n-j$ は $n-j\geqq 2^{d-1}$ であることがわかる。したがって，$2^s\leqq j\leqq n-2^{d-1}$ である。つまり，左部分木の葉の数がたかだか $n-2^{d-1}$ であるということは，$s\leqq\lfloor\log_2(n-2^{d-1})\rfloor$ であるといえる。よって，$d-1\leqq s\leqq\lfloor\log_2(n-2^{d-1})\rfloor$ となる。（証明終）

〔補題2〕 特に，$d=1$ のとき，次式が成り立つ。

$$\left.\begin{array}{l} f(n,1)=\displaystyle\sum_{d=1}^{\lfloor\log_2(n-1)\rfloor} f(n-1,d)=f(n-1), \\ \qquad\qquad :n\geqq 3 \end{array}\right\} \quad (4)$$

（証明） 式(3)を $d=1$ とすると

$$f(n,1)=\sum_{s=0}^{\lfloor\log_2(n-1)\rfloor}\ \sum_{j=2^s}^{n-1} f(j,s)f(n-j,0)$$

となる。ところで，

$$f(n-j,0)=\begin{cases} 1: n-j=1 \\ 0: n-j\geqq 2 \end{cases}$$

だから，$j=n-1$ のときだけ $f(n-j,0)=1$ である。よって，

$$f(n,1)=\sum_{s=0}^{\lfloor\log_2(n-1)\rfloor} f(n-1,s)$$

となり，$s=0$ のとき $f(n-1,s)=0$ だから $s\geqq 1$ である。s を d とおくと式(4)の最初の等式が成り立ち，さらに式(2)より2番目の等式が成り立つ。

（証明終）

3. 漸近的下限

本論文の成果を次の定理1で示す。

〔定理1〕 $n>>2^d$ のとき，$f(n,d)$ の漸近的下限は次式で表現される。

$$f(n,d)>C_d\frac{e^n}{n}O\left(\frac{1}{\log n}\right):\ d\geqq 1 \quad (5)$$

ここに，C_d は d によって決まる適当な定数である。記号 O については，文献(2)，p.103 参照。e は指数で $e=2.718\cdots$ である。

この定理を証明するのに，次の補題3が必要である。

〔補題3〕（文献(2)，p.93，問題6）

総和 $\sum_{j=1}^{n-1} 1/[j(n-j)]$ は次式で表される。

$$\sum_{j=1}^{n-1}\frac{1}{j(n-j)}=\frac{2H_{n-1}}{n} \quad (6)$$

ここに，H_n は調和数(harmonic number，文献(2)，p.73）とよばれるもので，

$$H_n=\log_e n+\gamma+\frac{1}{2n}+\cdots \quad (7)$$

である。γ はオイラー定数と呼ばれ，$\gamma=0.57721\cdots$ である。

†: $\lfloor x\rfloor$ は x を超えない最大の整数を表す。

（定理１の証明） 小さい $n(n<80)$ については，$C_d e^n/n\log n$ とおいて（このようにおいてさしつかえないのは $e^n>>n>>\log n$ だからである）適当に C_d を選べば，漸化式(3)で求めた具体的な値と比較して，式(5)が成り立つことが計算機によって確かめられる。つまり，$(f(n,d)\cdot n\log n)/e^n$ が一定の値になることが確かめられる。

式(5)が $n-1$ 以下で成り立つとして n のとき成り立つことを示す。帰納法の仮定により，n 個の葉を持つ木 T の左部分木については，

$$f(j,s)>C_d\frac{e^j}{j}O\left(\frac{1}{\log j}\right)$$

が成り立ち，右部分木については

$$f(n-j,d-1)>C_{d-1}\frac{e^{n-j}}{n-j}O\left(\frac{1}{\log(n-j)}\right)$$

が成り立つから，これらを漸化式(3)に代入して

$$f(n,d)>\sum_{s=d-1}^{\lfloor\log_2(n-2^{d-1})\rfloor}\sum_{j=2^s}^{n-2^{d-1}}\left[C_s\frac{e^j}{j}O\left(\frac{1}{\log j}\right)\right]\cdot\left[C_{d-1}\frac{e^{n-j}}{n-j}O\left(\frac{1}{\log(n-j)}\right)\right],\quad d\geqq 2$$

ここで，

$$O\left(\frac{1}{\log j}\right)O\left(\frac{1}{\log(n-j)}\right)>O\left(\frac{1}{(\log n)^2}\right)$$

だから

$$>\sum_{s=d-1}^{\lfloor\log_2(n-2^{d-1})\rfloor}C_sC_{d-1}e^nO\left(\frac{1}{(\log n)^2}\right)\cdot\sum_{j=2^s}^{n-2^{d-1}}\frac{1}{j(n-j)}$$

ところで，$n>>2^s>2^{d-1}$ では式(6)より

$$\sum_{j=2^s}^{n-2^{d-1}}\frac{1}{j(n-j)}=\frac{O(\log n)}{n}$$

だから，

$$=\sum_{s=d-1}^{\lfloor\log_2(n-2^{d-1})\rfloor}C_sC_{d-1}e^nO\left(\frac{1}{(\log n)^2}\right)\cdot\frac{O(\log n)}{n}$$

$$=\frac{e^n}{n}O\left(\frac{1}{\log n}\right)\sum_{s=d-1}^{\lfloor\log_2(n-2^{d-1})\rfloor}C_sC_{d-1}$$

ここで，

$$C_d=\sum_{s=d-1}^{\lfloor\log_2(n-2^{d-1})\rfloor}C_sC_{d-1}$$

とおけば，式(5)が成り立つ。（証明終）

$f(n+1,1)$ および $f(n)$ についても，$C=\sum_{d=1}^{\lfloor\log_2 n\rfloor}C_d$ とおくと，式(2)より同様の下限が得られる。

［命題１］

$$f(n+1,1)=f(n)>C\frac{e^n}{n}O\left(\frac{1}{\log n}\right)\qquad(8)$$

4. むすび

leftist 木の総数にたいするひとつの下限を示したが，上限の評価は残された問題である。本論文の成果である定理１の証明は間接的な証明法である。つまり，漸近的表現を予測し，その表現を補題１の漸化式の右辺に代入することによって，左辺も同じ表現になることを示したものである。この証明ではなぜ指数 e が現れるかの説明にはなっていない。総数の具体的な表現がみつかれば，e の生じる理由や係数 C_d の性質が明らかになると考えられる。

２分木の部分集合としてその総数評価が興味深いものには，leftist 木のほかに，AVL 木（文献(1), p. 453）や完全２進符号木（文献(3), p. 71）がある。これらもまだ部分的にしか解明されておらず，今後の解析が待たれる。

文　献

(1) KNUTH D. E.,: "*The Art of Computer Programming* 3: *Sorting and Searching*", Addison-Wesley, Reading, MA (1973).

(2) KNUTH D. E.,: "*The Art of Computer Programming* 1: *Fundamental Algorithms*", Addison-Wesley, Reading, MA (1971).

(3) 喜安善市, : "符号論序説", 共立出版 (1984).