数理解析研究所講究録 1325 巻 2003 年 63-68

自然数の組成の $O(1)$ 時間生成について

三河 賢治 (Kenji MIKAWA)
茨城大学総合情報処理センター
mikawa@mx.ibaraki.ac.jp

仙波 一郎 (Ichiro SEMBA)
茨城大学工学部情報工学科
isemba@cis.ibaraki.ac.jp

あらまし

自然数 n を r 個の和因子に分割する問題は，次式 $a_1 > \cdots > a_r$ と $a_1 + \cdots + a_r = n$ を満たす文字列 $a_1 a_2 \ldots a_r$ を求める問題である．本論文で扱う問題は，条件 $a_1 > \cdots > a_r$ を省いて，順序を考慮した自然数の分割について考える．順序を考慮した自然数の分割は，自然数の組成と呼ばれる．Grimaldi と Meadows により，最大の和因子が k であるような自然数の組成の個数は k 段フィボナッチ数列に一致することが証明された．本論文では，自然数の組成を一つ当たり $O(1)$ 時間で生成するアルゴリズムを提案する．

キーワード： 自然数の組成，k 段フィボナッチ数列，グレイコード，逐次生成，組合せアルゴリズム

1 まえがき

フィボナッチ数列を拡張した k 段フィボナッチ数列 $f(n,k)$ は，境界条件 $f(0,k) = 1$ として，

$$f(n,k) = \begin{cases} 0 & k = 0 \text{ のとき} \\ 1 & k = 1 \text{ のとき} \\ \sum_{i=1}^{k} f(n-i,k) & k \leqq n \text{ のとき} \\ f(n,n) & k > n \text{ のとき} \end{cases}$$

のように再帰的に定義される．フィボナッチ数列は 2 段フィボナッチ数列と考えることができる．

自然数の分割とは，自然数 n を適当な自然数の和に表現する問題である．例えば，5 の分割は $1+1+1+1+1$，$2+1+1+1$，$2+2+1$，$3+1+1$，$3+2$，$4+1$，5 の 7 通り考えられる．自然数の分割の個数を分割数と呼ぶ．この例では，5 の分割数は 7 になる．和を構成する数字を和因子と呼び，最大の和因子を指定して，自然数の分割を考えることもできる．n 例えば，最大の和因子を 3 とすると，5 の分割は $1+1+1+1+1$，$2+1+1+1$，$2+2+1$，$3+1+1$，$3+2$ の 5 通り考えられる．従って，この場合の分割数は 5 となる．自然数を幾つかの和因子に分解する問題では，$3+2$ と $2+3$ は等しい分割とする見方と異なる分割とする見方の二つの考え方があるが，前者を自然数の分割，後者を自然数の組成と呼んで明確に区別される．

本論文では，最大の和因子が k であるような自然数 n のすべての組成を生成するアルゴリズムを提案する．文字列を一定の順序で並べたものをリストと呼ぶことがあるが，以下，本論文でも組成を並べたものをリストと呼ぶことにする．Savage [5] は，自然数の分割におけるグレイコードを次のように定義し，これを満たす自然数 n の分割を生成する手法を提案した．

定義 1.1 自然数 n の分割のリスト A について，i 番目の自然数の分割が $i-1$ 番目の自然数の分割から次式

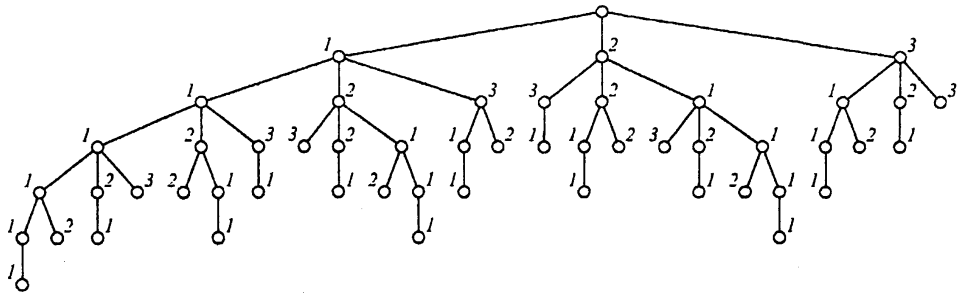

図 1 列挙木 $T(6,3)$

(1) 和因子の値を 1 増やし，1 より大きい和因子の値を 1 減らす，
(2) 和因子の値を 1 増やし，和因子 '1' を取り除く，
(3) 1 より大きい和因子の値を 1 減らし，和因子 '1' を新しく加える，

の何れかから得られるとき，A はグレイコードである．

近年，Grimaldi と Meadows [3] は，自然数の組成の分割数が k 段フィボナッチ数列に一致することを証明した．本論文では，上記で定義されたグレイコードの性質を満たすような組成のリストを生成するアルゴリズムを提案する．

2 生成アルゴリズム

始めに，リストの表記法を定義しよう．n 個の文字列 $l_1, l_2, \ldots, l_n$ からなるリスト L を $L = \langle l_1, l_2, \ldots, l_n \rangle$ と表す．リスト L，記号 x に対して，$x \cdot L$ は L に含まれる各文字列の先頭に x を付加する．例えば $L = \langle 21, 31 \rangle$ のとき，$4 \cdot L = \langle 421, 431 \rangle$ である．リスト L，L' に対して，$L \circ L'$ は二つのリストを連結する．例えば $L = \langle 21, 31 \rangle$，$L' = \langle 43, 53 \rangle$ のとき，$L \circ L' = \langle 21, 31, 43, 53 \rangle$ である．n 個の文字列 $l_1, l_2, \ldots, l_n$ からなるリスト $L = \langle l_1, l_2, \ldots, l_n \rangle$ に対して，先頭の文字列 l_1 を示す $\mathrm{first}(L)$ と末尾の文字列 l_n を示す $\mathrm{last}(L)$ を定義する．また，リスト L の文字列を逆順に並べたリスト $\overline{L} = \langle l_n, \ldots, l_2, l_1 \rangle$ を定義し，$\overline{L}$ を L の反転と呼ぶ．明らかに $\mathrm{first}(\overline{L}) = \mathrm{last}(L)$，$\mathrm{last}(\overline{L}) = \mathrm{first}(L)$ が成立する．次に，組成の表記法を定義する．和因子 a が b 個連続するとき，a^b と表記する．例えば組成 22211 ならば $2^3 1^2$ である．

リスト $L(n,k)$ は，$L(n,0) = \lambda$，$L(n,1) = \langle 1^n \rangle$ を基底として，$1 < k \leqq n$ について，

$$\begin{aligned} L(n,k) = {} & 1 \cdot L(n-1,k) \circ 2 \cdot \overline{L(n-2,k)} \\ & \circ 3 \cdot L(n-3,k) \circ 4 \cdot \overline{L(n-4,k)} \\ & \circ 5 \cdot L(n-5,k) \circ \cdots \end{aligned}$$

のように k 個の部分リストの連結として定義する．このとき，$L(n,k)$ の先頭の部分リストが反転しないように k 個の部分リストを交互に反転する．すなわち偶数番目の部分リストは反転し，奇数番目の部分リストは反転しない．一方，$k > n$ について，

$$L(n,k) = L(n,n)$$

のように定義する．例として，リスト $L(6,3)$ の構造を表すグラフ $T(n,k)$ を図 1 に示す．このようなグラフは列挙木と呼ばれ，$T(n,k)$ の根から葉までの経路が $L(n,k)$ の組成に対応する．

組成 $a_1 a_2 \ldots a_r$ について，a_1 の選び方は $1, 2, \ldots, k$ の k 通りある．ここで $a_1 = i$ に固定すると，文字列 $a_2 \ldots a_r$ は部分リスト $L(n-i,k)$ の要素と考えることができる．$a_1 = i$ を満たす組成を要素とする $L(n,k)$ の部分リスト $i \cdot L(n-i,k)$ について，a_2 以下の文字についても順に値を特定していく．末尾の文字 a_r の値を特定したとき，即ち $n = 0$ のとき，$L(n,k)$ の要素が一つ決定する．従って，リスト $L(n,k)$ は，最大の和因子が k であるような自然数 n の組成を系統的に並べている．

補題 2.1 リスト $L(n,k)$ の先頭の文字列について，

$$\mathrm{first}(L(n,k)) = 1^n$$

が成立し，末尾の文字列について，

$$\mathrm{last}(L(n,k)) = \begin{cases} k^{\lfloor n/k \rfloor} s & k \text{ が奇数のとき} \\ k\, 1^{n-k} & k \text{ が偶数のとき} \end{cases}$$

が成立する．だだし，$s = n \bmod k$ である．

定理 2.1 リスト $L(n,k)$ はグレイコードである．

（証明） n に関する帰納法で証明する．明らかに $n = 1$ について定理は成立する．$n \leqq m$ について定理は成立すると仮定し，$n = m+1$ について考える．奇数 i について，$L(m+1,k)$ を

$$\begin{aligned} L(m+1,k) = {} & 1 \cdot L(m,k) \circ \cdots \\ & \circ\, i-1 \cdot \overline{L(m-i+2,k)} \\ & \circ\, i \cdot L(m-i+1,k) \\ & \circ\, i+1 \cdot \overline{L(m-i,k)} \circ \cdots \end{aligned}$$

のように k 個の部分リストの連結で表す．帰納法の仮定により部分リストの内部で定理は成立しているので，各部分リストの境界について定理を示せばよい．補題 2.1 で示しているように，k のパリティによって部分リストの末尾の文字列の振る舞いが異なる．そこで，k が奇数である場合と偶数である場合の二つの場合に分けて考えていく．

(1) k が奇数である場合．始めに，部分リスト $i-1 \cdot \overline{L(m-i+2,k)}$ の末尾の文字列と部分リスト $i \cdot L(m-i+1,k)$ の先頭の文字列との境界で定理は成立することを示す．補題 2.1 より，部分リスト $i-1 \cdot \overline{L(m-i+2,k)}$ の末尾の文字列は，

$$\begin{aligned} & \mathrm{last}(i-1 \cdot \overline{L(m-i+2,k)}) \\ & \quad = \mathrm{first}(i-1 \cdot L(m-i+2,k)) \\ & \quad = i-1\, 1^{m-i+2} \end{aligned}$$

である．一方，部分リスト $i \cdot L(m-i+1,k)$ の先頭の文字列は，

$$\mathrm{first}(i \cdot L(m-i+1,k)) = i\, 1^{m-i+1}$$

である．二つの文字列を比較すると，部分リスト $i-1 \cdot \overline{L(m-i+2,k)}$ の末尾の文字列の先頭の文字の値を 1 増やし，$m-i+3$ 番目の和因子 '1' を取り除くことによって，部分リスト $i \cdot L(m-i+1,k)$ の先頭の文字列を得る．従って，定義 1.1 の (2) を満たしている．

次に，部分リスト $i \cdot L(m-i+1,k)$ の末尾の文字列と部分リスト $i+1 \cdot \overline{L(m-i,k)}$ の先頭の文字列との境界で定理は成立することを示す．補題 2.1 より，部分リスト $i \cdot L(m-i+1,k)$ の末尾の文字列は，

$$\mathrm{last}(i \cdot L(m-i+1,k)) = i\, k^{\lfloor m-i+1/k \rfloor} s$$

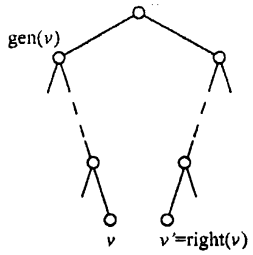

図 2 節点 ν と定義された節点との関係

である．一方，部分リスト $i+1\cdot\overline{L(m-i,k)}$ の先頭の文字列は，

$$\begin{aligned}\mathrm{first}(i+1\cdot\overline{L(m-i,k)}) &= \mathrm{last}(i+1\cdot L(m-i,k))\\ &= i+1\,k^{\lfloor m-i/k\rfloor}s'\end{aligned}$$

である．s の取りうる値は $\lambda,1,2,\ldots,k-1$ なので，$s=\lambda$，$s=1$，$s=2,\ldots,k-1$ の三つの場合に分けて両者を比較しよう．和因子 'k' の個数はそれぞれ $t=\lfloor m-i+1/k\rfloor$，$t'=\lfloor m-i/k\rfloor$ である．

(1A) $s=\lambda$ の場合．$t'=t-1$，$s'=k-1$ が成立するので，両者の文字列長は等しく，先頭と末尾の文字だけが異なっている．従って，定義 1.1 の (1) を満たす．

(1B) $s=1$ の場合．$t'=t$，$s'=\lambda$ が成立する．従って，定義 1.1 の (2) を満たす．

(1C) $s=2,\ldots,k-1$ の場合．$t'=t$，$s'=s-1$ が成立する．従って，定義 1.1 の (1) を満たす．

以上，k が奇数である場合，リスト $L(n,k)$ はグレイコードであることが確かめられた．

同様に，k が偶数である場合も，リスト $L(n,k)$ がグレイコードであることを確かめることができるので，$n=m+1$ の場合も定理は成立する．故に，定理は証明された． □

3 $O(1)$ 時間アルゴリズム

列挙木は組合せ的な集合の要素の構造を表し，計算機上で実現するデータ構造に適している．また，列挙木上の各要素を効率良く生成するアルゴリズムは文献 [6, 4] に詳しく述べられている．本節では，これらの文献に習い $T(n,k)$ 上の組成を $O(1)$ 時間で生成するアルゴリズムを提案する．

始めに，文献 [4] による $O(1)$ 時間アルゴリズムの戦略について述べる．列挙木 $T(n,k)$ の根から葉までの経路で深さ i に位置する節点を $a(i)$ として，根 $a(0)$ を除いた節点の系列 $a(1)\ldots a(r)$，$r\leq n$，が組成である．根 $a(0)$ に接続する k 個の子 $1,2,\ldots,k$ の中から任意に $1\leq j<k$ を選び，部分木 $T(n-j,k)$ と $T(n-j-1,k)$ の境界に面する二つの組成について考える．部分木 $T(n-j,k)$ の最右端の経路に対応する組成を $\boldsymbol{a}=a(1)\ldots a(r)$，部分木 $T(n-j-1,k)$ の最左端の節点の経路に対応する組成を $\boldsymbol{a}'=a'(1)\ldots a'(r')$ とする．組成 $\boldsymbol{a}$ の次列 $\boldsymbol{a}'$ は，定理 2.1 の証明の中で示された通り，j の奇偶によって求めることができる．即ち，j が偶数である場合，補題 2.1 より，

$$a(i)=\begin{cases}j & i=1\text{ のとき}\\ 1 & i=2,\ldots,r\text{ のとき}\end{cases}$$

と

$$a'(i)=\begin{cases}j+1 & i=1\text{ のとき}\\ 1 & i=2,\ldots,r-1\text{ のとき}\end{cases}$$

```
A1   output a(1)...a(n);
A2   i := d(n); r := n;
A3   while i > 0 do begin
A4     2 文字 a(i) とその他を変換して，次列を生成する;
A5     output a(1)...a(r);
A6     a(i) := right(a(i));
A7     r := a(i) を 根とする部分木の最左端の葉の深さ;
A8     if a(i) = 末弟の節点 then begin
A9       a(i) := right(a(i));
A10      d(i) := d(i − 1); d(i − 1) := i − 1
A11    end;
A12    i := d(r); d(r) := r
A13  end;
```

図 3 生成アルゴリズム

が成立するので，$a'(1) = a(1)+1$, $r' = r-1$ によって組成 $\boldsymbol{a}$ を得る．一方，j が奇数である場合，$a'(1) = a(1)+1$ として，$a(r) = 1$ ならば $r' = r-1$，$a(r) > 1$ ならば $r' = r$ かつ $a'(r') = a(r)-1$ によって組成 $\boldsymbol{a}$ を得る．列挙木 $T(n,k)$ は再帰的に定義されるので，上記の交換規則は $T(n,k)$ の内部についても帰納的に言える．組成の文字列長の差による変換も 1 文字として考えると，どの組成も 2 文字だけ相異なるように並んでいる．ここで，ある組成から次列を生成するときに変換される文字の中で最も先頭に近い文字に注目する．文献 [4] では，このような文字を列挙木の節点になぞらえて生成節点と呼び，次のように定義した．$T(n,k)$ の節点を行きがけ順にたどるときの節点の系列を考える．このとき任意の節点 v について，v より後にたどられる節点の中で v に最も近くかつ $T(n,k)$ の根から等しい深さの節点を $\mathrm{right}(v)$ と表す．また任意の 2 節点 v，v' について，v と v' から最も近い共通の先祖を w とする．このとき，v と w を結ぶ経路上に位置する w の子を $\mathrm{gen}(v,v')$ と表し，これを生成節点と呼ぶ．特に $\mathrm{gen}(v,\mathrm{right}(v))$ を $\mathrm{gen}(v)$ と表す．節点 v と，節点 v に関係する $\mathrm{gen}(v)$，$\mathrm{right}(v)$ との関係を図 2 に示す．また $T(n,k)$ の深さ i の節点 $a(i)$ について，生成節点 $\mathrm{gen}(a(i))$ の深さを記憶する配列 $d(i)$ を用意する．便宜的であるが $d(0) = 0$ として，$T(n,k)$ の最右端の節点の系列 $a(1)\ldots a(r)$ に対して $\mathrm{gen}(a(i)) = a(0)$ とする．

$T(n,k)$ 上の組成を生成するアルゴリズムを図 3 に示す．アルゴリズムの基本動作は次の通りである．$T(n,k)$ の根 v に接続する k 個の節点 $v_1,\ldots,v_k$ に対して，v_i を根とする部分木 $T(n-i,k)$ の先頭の葉を x_i，末尾の葉を y_i とする（図 4 参照）．生成アルゴリズムでは，$T(n,k)$ の先頭の葉 x_1 から順々に葉を移動し，末尾の葉 y_k に到達するとき，y_k から v に移動してアルゴリズムを終了する．この動作を基本として，v に接続する先頭の部分木 $T(n-1,k)$ について，先頭の葉 x_1 から始めて末尾の葉 y_1 に到達するとき，部分木の根 v_1 に移動する．これで $T(n-1,k)$ の節点の探索は終了したので，次に隣接する部分木 $T(n-2,k)$ の根 v_2 に移動し，先頭の葉 x_2 から同様の移動を繰り返す．これを末尾の部分木 $T(n-k,k)$ まで繰り返すのだが，$k-1$ 番目の部分木 $T(n-k+1,k)$ の節点の探索を終了したとき，即ち y_{k-1} から v_{k-1} に移動するとき，$d(1)$ の値を $d(0)$ の値に変更し，$d(0)$ の値を 0 に戻す．そして v_{k-1} から v_k に移動し，先頭の葉 x_k から $T(n-k,k)$ の節点の移動を始める．

$T(n-k,k)$ の内部についても，上で述べた節点の移動を行うので，$T(n-k,k)$ の根 v_k と末尾の葉 y_k を結ぶ経路上の深さ j の節点に到達するとき，$d(j)$ の値を $d(j-1)$ の値に変更し，$d(j-1)$ の値を $j-1$ に戻す．$T(n-k,k)$ の末尾の葉 y_k は深さ r の節点であると仮定すると，$d(r)$ の値は連鎖的に $d(0)$ の値に等しくなるので，y_k に到達するとき $d(r)$ を参照すると v の深さ 0 を得る．以上，列挙木 $T(n-k,k)$ の節点の移動と，$O(1)$ 時間の計算量で末尾の葉 y_k から根 v に移動する原理を述べた．$T(n,k)$ の内部の部分木についても帰納的に同様の議論が成立する．従って，

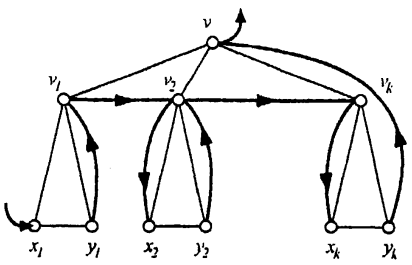

図 4 $T(n,k)$ の巡回方法

生成アルゴリズムは $O(1)$ 時間の計算量で各生成節点を計算している.

次に，生成アルゴリズムの A7 で計算される組成の文字列長について考えていく．$T(n,k)$ 上の組成を $\boldsymbol{a} = a(1)\ldots a(r)$，その次列を $\boldsymbol{a}\ = a'(1)\ldots a'(r')$，生成節点を $a(i)$ とする．組成の文字列長は，$a(i-1)$ を根とする部分木が反転している場合と，反転していない場合に分けて考える必要がある．$a(i-1)$ を根とする部分木が反転していない場合，即ち $a(i) < a'(i)$ の場合は，定理 2.1 の証明の中で示された通り，

$$r' = \begin{cases} r-1 & a(r) = 1 \text{ のとき} \\ r & a(r) > 1 \text{ のとき} \end{cases}$$

が成立する．一方，$a(i-1)$ を根とする部分木が反転している場合，即ち $a(i) > a'(i)$ の場合は，

$$r' = \begin{cases} r & a(r) = 1 \text{ のとき} \\ r+1 & a(i) \text{ が奇数のとき，または } a(r) = k \text{ のとき} \end{cases}$$

が成立する.

配列 d の初期値は，生成節点の定義から，$d(i) = i,\ \ i = 0,\ldots,n$ である．また $L(n,k)$ の先頭の文字列は補題 2.1 で与えられる．最後に次の定理を与える.

定理 3.1 リスト $L(n,k)$ の各組成を $O(1)$ 時間で生成することができる.

参考文献

[1] M. Develin, "A complete categorization of when generalized Tribonacci sequences can be avoided by additive partitions," Electron. J. Combin., vol.7, #R53, 2000.

[2] I. Dumitriu, "On generalized Tribonacci sequences and additive partitions," Discrete Math., vol.219, pp.65–83, 2000.

[3] R. P. Grimaldi and D. S. Meadows, "Compositions of integers," Congre. Numer., vol.76, pp.119–125, 1990.

[4] 三河賢治, 仙波一郎, "フィボナッチ列の $O(1)$ 時間生成について, " 信学論, vol.J85-D-I, no.2, pp.116–121, 2002.

[5] C. D. Savage, "Gray code sequences of partitions," J. Algorithms, vol.10, pp.577–595, 1989.

[6] T. Takaoka, "$O(1)$ time algorithms for combinatorial generation by tree traversal," Comput. J., vol.42, pp.400–408, 1999.