数理解析研究所講究録
第1148巻 2000年 88-93

発表番号 16

反復型アルゴリズムによるフィボナッチ列の生成について

三河 賢治 (Kenji MIKAWA)　　仙波 一郎 (Ichiro SEMBA)

茨城大学工学部情報工学科

E-mail: {mikawa,isemba}@cis.ibaraki.ac.jp

1 はじめに

本論文は, 組合せ生成問題の一つであるフィボナッチ列の生成について考察する. フィボナッチ数は最もよく知られた数のひとつで黄金比に深く関わる. とくに組合せ数学では, パスカルの三角形にフィボナッチ数が潜んでいたり, 1×2 の板で $2 \times n$ の盤を覆う問題の解の数がフィボナッチ数となっていたり, この数に関わる多くの組合せ問題が知られている.

本論文で扱うフィボナッチ列は次のとおり定義される: 集合 $\{0,1\}$ 上の n 個の要素からなるすべての部分集合を $\mathcal{B}$, フィボナッチ列の集合を $\mathcal{F}$ とする. このとき, 集合 $\mathcal{F}$ は, $\mathcal{F} \subseteq \mathcal{B}$ かつ要素 $\forall x \in \mathcal{F}$ は文字列 11 を含まない. また, 集合 $\mathcal{F}$ の要素の数がフィボナッチ数に一致する. Hsu [4] によって, 集合 $\mathcal{F}$ の要素に対応する頂点により誘導されるハイパーキュウブの部分グラフ (フィボナッチキュウブ) 上のハミルトン道の構成法は示された. 本論文では, フィボナッチ列を生成する 2 つの生成法を与える. いずれも最小交換則 (Minimal Change Property, MCP [2]) を満たす.

再帰的に定義された組合せリストは, その解析しやすさ, 構造のわかりやすさの点で評価でき, 組合せ生成を扱ういくつかの文献で論じられている. 再帰的な生成法を反復的な (すなわち再帰手続きを用いない) 生成法に変換する一般的な方法はなく, 生成問題個々に解決されているのが現状である. 現在までに, Gray code [1], 順列 [3], 組合せ [7], カッコ列 [10, 11, 6, 7], 重複順列 [5, 8], 重複組合せ [9] を反復的に生成する方法が知られている.

再帰的に定義された組合せリストは, 組合せリストの生成過程で再帰木 (recursion tree または computation tree) を構成し, 経路上の特定の節点の情報から次の組合せ列を生成する. 訪問する節点の順序はリストの定義式から一意に定まり, ある節点から $O(1)$ 時間で次の節点を見つける反復型アルゴリズムの構成法は文献 [6, 7] で詳しく述べられている. これらによって示された方法は, いずれも文献 [1] を基にし, 改良された方法である. また文献 [1, 6] で扱われた再帰木は, 根から各葉までの経路が等しい長さであった. 組合せ問題で扱う再帰木は, 根から葉までの経路の長さが異なる場合も多い. 文献 [7] で扱われた n 個の要素から k 個を選ぶ組合せ (以下, $C_{n,k}$ と表す) に対応する再帰木は, 根から葉までの経路の長さが異なり, $O(1)$ 時間で次の節点を見つける方法は高く評価できる. しかしながら, その方法は, その再帰木の特徴的な構造に依存するところも多い.

本論文で定義するフィボナッチ列の生成式から得られる 2 つの再帰木は, いずれも根から葉までの経路の長さが異なり, $C_{n,k}$ の再帰木と比べて再帰的な構造がより複雑である. このような場合についても, ある節点から $O(1)$ 時間で次の節点を見つける反復型アルゴリズムの構成法を与える. また, 異なる 2 つの再帰木を分析および比較することによって得られた反復型アルゴリズムの構成法は, より一般的な構成方法であると期待される.

2 表記法

組合せ生成のための表記法を以下のとおり定義する. リスト L, 記号 x に対して, $L \cdot x$ は L に含まれるそれぞれの文字列の最後に x を付加する. 例えば $L = (01, 10)$ とすると, $L \cdot 1 = (011, 101)$ である. L, L' をリストとするとき, $L \circ L'$ は二つのリストを連結する. 例えば $L_1 = (01, 10)$, $L_2 = (00, 11)$ とすると,

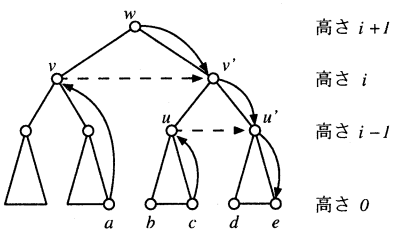

図 1 再帰木 $\mathcal{P}$ 上の節点の走査法

$L \circ L' = (01, 10, 00, 11)$ である.

また, n 個の文字列 $l_1, l_2, \ldots, l_n$ からなるリスト $L = (l_1, l_2, \ldots, l_n)$ に対して, 先頭の文字列 l_1 を示す $\mathrm{first}(L)$, 末尾の文字列 l_n を示す $\mathrm{last}(L)$ をそれぞれ定義する. また, リスト L の文字列を逆順に並べたリスト $\overline{L}$, すなわち $\overline{L} = (l_n, \ldots, l_2, l_1)$ を定義する. 明らかに $\mathrm{first}(\overline{L}) = \mathrm{last}(L)$ と $\mathrm{last}(\overline{L}) = \mathrm{first}(L)$ が成立する.

3 反復型アルゴリズムの構成法

文献 [6, 7] による再帰木の走査法について述べる. 走査する再帰木を $\mathcal{P}$ として, 高さ i の節点を記憶する配列 s と, 高さ i の節点から直接訪問できる節点 v を記憶する配列 d を用意する. 高さ i に節点 v があるとき, $s[i]$ と $d[i]$ にはそれぞれ v が記憶されている. これら 2 つの配列を制御して, 次に訪問する節点を $O(1)$ 時間で見つける. 初期値設定として, 再帰木の最左端の経路上の節点を s, d それぞれに記憶しておく.

$\mathcal{P}$ 上の隣接する 2 つの経路 a, b を考える (図 1 参照). このとき, $s[i] = d[i] = v$, $s[i+1] = d[i+1] = w$ である. この例では, 葉 e から節点 w へ直接訪問できることを示す. 葉 a から v へ直接訪問できるとしよう. v を訪問したとき, v を根とする部分木のすべての節点は訪問済みである. 以下, v を訪問した後の処理を列挙する.

(処理 1) v の右兄弟 v' について, ある節点の最右端の子ならば, $d[i]$ を $d[i+1]$ $(= w)$ に, $d[i+1]$ を w の右隣りの節点 w' に, それぞれ更新する. 最右端の子でなければ (処理 3).

(処理 2) $s[i]$ を, v' の右隣りの節点 v'' にあらかじめ更新しておく.

(処理 3) v' から葉 b を直接訪問する.

この処理を c について繰り返すと, 葉 d を訪問したとき, (処理 1) によって, $d[i-1]$ には w が記憶されている. d から e までの経路についても, 同様の処理を繰り返すと, 葉 e を訪問するとき, $d[0]$ には w が記憶されていることになる. したがって, 葉 e から直接訪問できる節点は w となる. このとき, 葉 e と w の間にある経路上の各節点は, (処理 2) によって, e の右隣りの経路上の対応する節点に更新されている. 以上の処理を, 再帰木の最左端の葉から再帰木の根を訪問するまで繰り返す. 再帰木の根を訪問したとき, 節点の走査を終了する.

次にこの走査法の問題点を述べる. 再帰木の根から葉までの経路の長さが等しい場合, (処理 2) について, v' を v'' に更新することが可能である. つまり, v' に対応する節点 v'' が, e の右に隣接する経路上に存在する. 一方経路の長さが異なる場合, v' に対応する v'' が, e の右に隣接する経路上に存在しない場合も起こりうる. 文献 [7] では, ${}_nC_r$ に対応する再帰木の対称性に注目して v' から v'' への予測を可能とした. フィボナッチ列のリストに対応する再帰木は内部の部分木について非対称である. 以下に続く節で, フィボナッチ列のリストに対応した再帰木の節点を走査する反復型アルゴリズムを実際に構築していく.

4 生成アルゴリズムその 1

長さ n のフィボナッチ列のリストを次のように再帰的に定義する.

$$F_n = \begin{cases} (0,1) & n = 1 \text{ のとき} \\ (00, 10, 01) & n = 2 \text{ のとき} \\ F_{n-1} \cdot 0 \circ \overline{F_{n-2}} \cdot 01 & n \geq 3 \text{ のとき.} \end{cases}$$

リスト F_n によって与えられる再帰木を $\mathcal{F}_n$ とする. $\mathcal{F}_n$ の根には, 2 つの部分木 $\mathcal{F}_{n-1}$ と $\overline{\mathcal{F}_{n-2}}$ が接続する. リスト F_6 に対応する再帰木 $\mathcal{F}_6$ を図 2 に示す. $\mathcal{F}_n$ の各葉から根までの経路をたどることによって得られるラベル列が, フィボナッチ列に対応する.

補題 4.1 リスト F_n は以下の性質を満たす.

$\mathrm{first}(F_n) = 0^n$
$\mathrm{last}(F_n) = 0^{n-1}1$

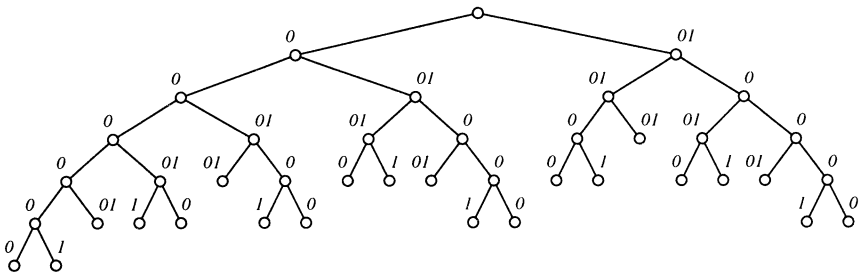

図 2 再帰木 $\mathcal{F}_6$

また補題 4.1 から以下の定理を得る.

定理 4.1 リスト F_n 上の連続する文字列は高々 3 ビットだけ相異なる.

証明. n に関する帰納法で証明する. 明らかに $n = 2$ 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. 帰納法の仮定より, 各部分リストの内部では定理は成立する. したがって, 部分リストの境界について定理を示せばよい. 部分リスト $F_m \cdot 0$ の末尾の文字列は, 補題 4.1 より,

$$\mathrm{last}(F_m \cdot 0) = 0^{m-1}10$$

となる. 一方部分リスト $\overline{F_{m-1}} \cdot 01$ の先頭の文字列は

$$\begin{aligned}\mathrm{first}(\overline{F_{m-1}} \cdot 01) &= \mathrm{last}(F_{m-1} \cdot 01) \\ &= 0^{m-2}101\end{aligned}$$

となる. したがって, $m-1$, m, $m+1$ 番目の要素だけが異なる. 以上, $n = m + 1$ の場合も定理は成立する. ゆえに定理は証明された. □

次に, リスト F_n 上のフィボナッチ列について, 隣り合うフィボナッチ列の相異なるビット数の平均はある定数以下になることを示す. リスト F_n 上の隣り合うフィボナッチ列の相異なるビット数の総和を g_n とする. g_n は, 定理 4.1 の証明中で示されたとおり, $n \geqq 3$ について,

$$g_n = g_{n-1} + g_{n-2} + 3$$

を満たす. ただし $g_1 = 1$, $g_2 = 3$ である. この漸化式を解いて,

$$g_n = \frac{1}{\sqrt{5}}\left\{\left(\frac{1+\sqrt{5}}{2}\right)^{n+3} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+3}\right\} - 1$$

を得る. したがって交換されるビット数の平均は, 文字列ひとつあたり

$$g_n/f_n \approx (1+\sqrt{5})/2 \approx 1.61803$$

となる. リスト F_n 上のフィボナッチ列の総数 f_n は

$$f_n = \frac{1}{\sqrt{5}}\left\{\left(\frac{1+\sqrt{5}}{2}\right)^{n+2} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+2}\right\}$$

で与えられる. g_n/f_n により計算される値は, 黄金比と呼ばれる. 以上, リスト F_n 上の隣り合うフィボナッチ列は, 平均 1.62 ビット, 最悪 3 ビットだけ相異なる.

4.1 反復型アルゴリズムの実装

はじめに $\mathcal{F}_n$ 上の節点 v に対して次のとおりラベル $s(v)$ を定義する.

定義 4.1 再帰木 $\mathcal{F}_n$ の節点 v について, $s(v)$ は v を根とする部分木によって生成されるフィボナッチ列の長さを示す. ただし v に与えられたラベルの長さは含まない.

$s(v)$ をラベルとする再帰木を $\mathcal{P}_n$ と表す (図 3 参照). ラベル木 $\mathcal{P}_n$ について, 根から葉までの経路に対応するラベル列を $\langle p_n, p_{n-1}, \ldots, p_k\rangle$ と表し, $\mathcal{P}_n$ の最左経路と最右経路から得られるラベル列をそれぞれ $\mathrm{fseq}(\mathcal{P}_n)$, $\mathrm{lseq}(\mathcal{P}_n)$ と表す. ここで k は葉の高さである. 明らかに $\mathrm{fseq}(\overline{\mathcal{P}_n}) = \mathrm{lseq}(\mathcal{P}_n)$ と $\mathrm{lseq}(\overline{\mathcal{P}_n}) = \mathrm{fseq}(\mathcal{P}_n)$ が成立する. また, ラベル列 x に対して, x の要素数を $|x|$ と表す.

補題 4.2 $\mathcal{P}_n$ は以下の性質を満たす.

$$\mathrm{fseq}(\mathcal{P}_n) = \langle n, \ldots, 1, 0\rangle$$

$$\mathrm{lseq}(\mathcal{P}_n) = \begin{cases}\mathrm{fseq}(\mathcal{P}_n) & n \leqq 1 \text{ のとき} \\ \langle n, n-2, \ldots, 1, 0\rangle & n \geqq 2 \text{ のとき.}\end{cases}$$

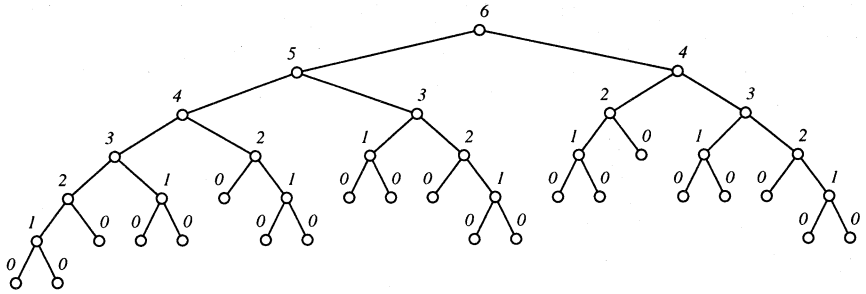

図 3 再帰木 $\mathcal{P}_6$

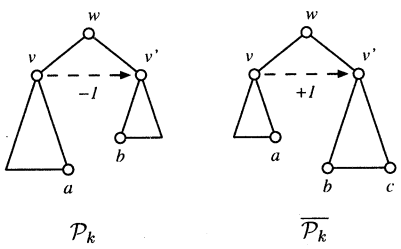

図 4 再帰木 $\mathcal{P}_n$ の部分木

$\mathcal{P}_n$ に含まれる部分木は, 部分リスト F_k, $\overline{F_k}$ に対応する 2 通り考えられる (図 4 参照). 第 3 節で述べたように, 配列 s の内容について, 隣接する経路上の節点のラベルをどのように更新するかが問題の本質であるから, $k \leqq 2$ の場合を考慮する必要はない.

はじめに部分木 $\mathcal{P}_k$ について考える. 定義から $s(v) = k-1$, $s(v') = k-2$ である. したがって経路 a, b に対応するラベル列はそれぞれ $\mathrm{lseq}(\mathcal{P}_{k-1})$, $\mathrm{fseq}(\overline{\mathcal{P}_{k-2}})$ である. 補題 4.2 より,

$$|\mathrm{lseq}(\mathcal{P}_{k-1})| \geqq |\mathrm{fseq}(\overline{\mathcal{P}_{k-2}})|$$

が成立する. したがって経路 b 上の高さ i の節点 v'_i に対応する経路 a 上の節点を v_i とすると,

$$s(v'_i) = \begin{cases} 0 & k = 3 \text{ のとき} \\ s(v_i) - 1 & k \geqq 4 \text{ のとき} \end{cases} \tag{1}$$

が成立する. ただし $j-k+2 \leqq i < j-1$. ここで j は節点 w の高さである. (処理 2) において, 節点のラベルを (1) 式によって更新する. 次に葉 b を訪問したとき, b 上の各節点は正しくラベル付けされている.

次に部分木 $\overline{\mathcal{P}_k}$ について考える. 定義から $s(v) = k-2$, $s(v') = k-1$ である. したがって経路 a, b に対応するラベル列はそれぞれ $\mathrm{lseq}(\mathcal{P}_{k-2})$, $\mathrm{fseq}(\overline{\mathcal{P}_{k-1}})$ である. 補題 4.2 より,

$$|\mathrm{lseq}(\mathcal{P}_{k-2})| \leqq |\mathrm{fseq}(\overline{\mathcal{P}_{k-1}})|$$

が成立する. したがって経路 b 上の高さ i の節点 v'_i に対応する経路 a 上の節点を v_i とすると,

$$s(v'_i) = \begin{cases} 0 & k = 3 \text{ のとき} \\ s(v_i) + 1 & k \geqq 4 \text{ のとき} \end{cases} \tag{2}$$

が成立する. ただし $j-k+2 \leqq i < j-1$. ここで j は節点 w の高さである. (処理 2) において, 節点のラベルを (2) 式によって更新する.

(処理 3) によって葉 b を訪問したとき, $\overline{\mathcal{P}_{k-1}}$ 上の節点のうち, 高さ $j-k+1$ と $j-k$ の節点のラベルが更新されていない場合がある. この問題はそれぞれ $k \geqq 4$, $k \geqq 3$ の場合に起こりうる. しかし, これらの節点のラベルは $\mathrm{fseq}(\overline{\mathcal{P}_{k-1}})$ と $\mathrm{lseq}(\overline{\mathcal{P}_{k-1}})$ から求めることができる. そこで (処理 3) の前に, 高さ $j-k+1$ と $j-k$ の節点のラベルを更新する.

以上の変更によって, $\mathcal{P}_n$ 上の葉から $O(1)$ 時間で特定の節点を訪問することができる. 葉から特定の節点 v を訪問したときに得られるラベル $s(v)$ の値が, 定理 4.1 の証明中で示された相異なるビットの位置を示している. したがって, v を訪問したときに, 新しいフィボナッチ列を生成する.

5 生成アルゴリズムその 2

長さ n のフィボナッチ列のリストを次のように再帰的に定義する.

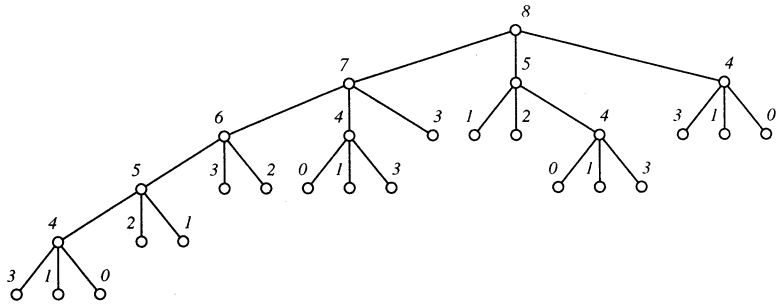

図 5 再帰木 $\mathcal{P}_8$

$$F_n = \begin{cases} (0,1) & n = 1 \text{ のとき} \\ (00,10,01) & n = 2 \text{ のとき} \\ F_{n-1} \cdot 0 \circ F_{n-2} \cdot 01 & n = 3 \text{ のとき} \\ F_{n-1} \cdot 0 & \\ \quad \circ \overline{F_{n-3}} \cdot 001 & \\ \quad \circ F_{n-4} \cdot 0101 & n \geqq 4 \text{ のとき.} \end{cases}$$

リスト F_n によって与えられる再帰木を $\mathcal{F}_n$ とする. 定義から, $\mathcal{F}_n$ は 2 分木と 3 分木を部分木としてもつ. すなわち, $n \leqq 3$ のとき, $\mathcal{F}_n$ は 2 分木となる. 再帰木 $\mathcal{F}_n$ 上の節点 v に対して, $s(v)$ をラベルとする再帰木を $\mathcal{P}_n$ とする. 再帰木の構造を簡単にするため, $s(v) \leq 3$ となる節点 v をすべて葉とする. 当然 $\mathcal{P}_n$ は 3 分木である (図 5 参照).

定理 5.1 リスト F_n 上の連続する文字列は 1 ビットまたは連続する 2 ビットだけ相異なる.

定理 5.1 から, 部分リスト $F_{n-1} \cdot 0$ と $\overline{F_{n-3}} \cdot 001$ の境界では, n 番目と $n-1$ 番目の要素だけが入れ替わる. また, 部分リスト $\overline{F_{n-2}} \cdot 001$ と $F_{n-3} \cdot 0101$ の境界では, $n-2$ 番目の要素だけが相異なる.

リスト F_n 上のフィボナッチ列について, 隣り合うフィボナッチ列の相異なるビット数の平均を計算すると, $1 + 1/\sqrt{5} \approx 1.45$ ビットとなる. すなわち, リスト F_n 上の隣り合うフィボナッチ列は, 平均 $1 + 1/\sqrt{5} \approx 1.45$ ビット, 最悪 2 ビットだけ相異なる.

5.1 反復型アルゴリズムの実装

$\mathrm{fseq}(\mathcal{P}_n)$ と $\mathrm{lseq}(\mathcal{P}_n)$ に関する補題を与える.

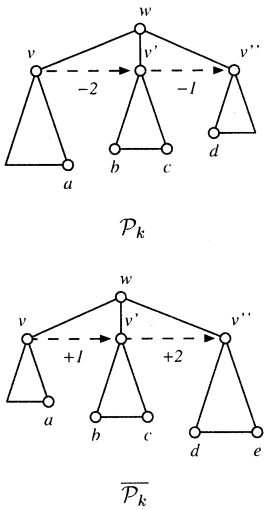

図 6 再帰木 $\mathcal{P}_n$ の部分木

補題 5.1 $\mathcal{P}_n$ は以下の性質を満たす.

$$\mathrm{fseq}(\mathcal{P}_n) = \langle n, \ldots, 1, 0 \rangle$$
$$\mathrm{lseq}(\mathcal{P}_n) = \langle n, n-4, n-8, \ldots, n \bmod 4 \rangle$$

$\mathcal{P}_n$ に含まれる部分木は, 部分リスト F_k, $\overline{F_k}$ に対応する 2 通り考えられる (図 6 参照). 配列 s の内容について, 隣接する経路上の節点のラベルをどのように更新するかが問題の本質であるから, $k \leqq 4$ の場合を考慮する必要はない.

はじめに, 部分木 $\mathcal{P}_k$ について考える. 定義から, $s(v) = k-1$, $s(v') = k-3$, $s(v'') = k-4$ である. したがって, 経路 a, b, c, d に対応するラベル列はそれぞれ $\mathrm{lseq}(\mathcal{P}_{k-1})$, $\mathrm{fseq}(\overline{\mathcal{P}_{k-3}})$, $\mathrm{lseq}(\overline{\mathcal{P}_{k-3}})$, $\mathrm{fseq}(\mathcal{P}_{k-4})$ で

ある. 補題 5.1 より,

$$|\mathrm{lseq}(\mathcal{P}_{k-1})| \geqq |\mathrm{fseq}(\overline{\mathcal{P}_{k-3}})|$$
$$|\mathrm{lseq}(\overline{\mathcal{P}_{k-3}})| \geqq |\mathrm{fseq}(\mathcal{P}_{k-4})|$$

が成立する. したがって, 経路 b 上の高さ i の節点 v'_i に対応する経路 a 上の節点を v_i とすると,

$$s(v'_i) = s(v_i) - 2 \tag{3}$$

が成立する. ただし $j - \lfloor \frac{k-3}{4} \rfloor \leqq i < j$. ここで, j は $\mathcal{P}_k$ の根 w の高さである. 一方, 経路 d 上の高さ i の節点 v''_i に対応する経路 c 上の節点を v'_i とすると,

$$s(v''_i) = s(v'_i) - 1 \tag{4}$$

が成立する. ただし $j - k + 3 \leqq i < j$. (処理 2) において, 節点のラベルを (3), (4) 式によって更新する. (処理 3) で b (d) の葉を訪問したとき, b (d) 上のすべての節点は正しくラベル付けされている.

次に部分木 $\overline{\mathcal{P}_k}$ について考える. 定義から, $s(v) = k-4$, $s(v') = k-3$, $s(v'') = k-1$ である. したがって, 経路 a, b, c, d に対応するラベル列はそれぞれ $\mathrm{lseq}(\overline{\mathcal{P}_{k-4}})$, $\mathrm{fseq}(\mathcal{P}_{k-3})$, $\mathrm{lseq}(\mathcal{P}_{k-3})$, $\mathrm{lseq}(\overline{\mathcal{P}_{k-1}})$ である. 補題 5.1 より,

$$|\mathrm{lseq}(\overline{\mathcal{P}_{k-4}})| \leqq |\mathrm{fseq}(\mathcal{P}_{k-3})|$$
$$|\mathrm{lseq}(\mathcal{P}_{k-3})| \leqq |\mathrm{lseq}(\overline{\mathcal{P}_{k-1}})|$$

が成立する. したがって, 経路 b 上の高さ i の節点 v'_i に対応する経路 a 上の節点を v_i とすると

$$s(v'_i) = s(v_i) + 1 \tag{5}$$

が成立する. ただし $j - \lfloor \frac{k-3}{4} \rfloor \leqq i < j$. ここで, j は $\mathcal{P}_k$ の根 w の高さである. 一方, 経路 d 上の高さ i の節点 v''_i に対応する経路 c 上の節点を v'_i とすると,

$$s(v''_i) = s(v'_i) + 2 \tag{6}$$

が成立する. ただし $j - k + 3 \leqq i < j$. (処理 2) において, 節点のラベルを (5), (6) 式によって更新する.

(処理 3) によって b の葉を訪問したとき, $\mathcal{P}_{k-3}$ 上の節点のうち, 高さ $j-k+3$ の節点 (b の葉付近の節点である) のラベルが更新されていない場合がある. この問題は $k \geqq 7$ のときに起こる. しかし, この節点のラベルは $\mathrm{fseq}(\mathcal{P}_{k-3})$ から得ることができる. そこで (処理 3) を行う前に, 高さ $j-k+3$ の節点のラベルを更新する.

また, (処理 3) によって 葉 d を訪問したとき, $\overline{\mathcal{P}_{k-1}}$ 上の節点のうち, 高さ $j-k+1$ と $j-k+2$ の節点 (e の葉付近の節点である) のラベルが更新されていない場合がある. この問題はそれぞれ $k \geqq 5$, $k \geqq 6$ のときに起こる. しかし, これらの節点のラベルは $\overline{\mathcal{P}_{k-1}}$ から得ることができる. そこで (処理 3) を行う前に, 高さ $j-k+1$ と $j-k+2$ の節点のラベルを更新する.

一方, (処理 3) によって d の葉を訪問しようとしたとき, d の葉のラベルが更新されていない場合がある. この問題は $k \bmod 4 \leqq 1$ のときに起こる. しかし, この葉のラベルもまた $\overline{\mathcal{P}_{k-1}}$ から得ることができる. そこで (処理 3) を行う前に, 高さ $j - \lfloor \frac{k-3}{4} \rfloor - 1$ の節点のラベルを更新する.

以上の変更を加えることで, $\mathcal{P}_n$ 上の葉から $O(1)$ 時間で特定の節点を訪問することができる.

参考文献

[1] J. R. Bitner, G. Ehrlich, and E. M. Reingold, Efficient generation of the binary reflected Gray code and its applications, Comm. ACM 19 (1976) 517–521.

[2] P. Eades and B. McKay, An algorithm for generating subsets of fixed size with a strong minimal change property, Inform. Proc. Lett. 19 (1984) 131–133.

[3] G. Ehrlich, Loopless algorithms for generating permutations, combinations and other combinatorial configurations, J. ACM 20 (1973) 500-513.

[4] W.-J. Hsu, Fibonacci cubes – a new interconnection topology, IEEE Trans. Parallel and Distributed Systems 4 (1993) 2–12.

[5] J. Korsh and S. Lipschutz, Generating multiset permutations in a constant time, J. Algorithms 25 (1997) 321–335.

[6] K. Mikawa and T. Takaoka, Generation of parenthesis strings by transpositions, in: Proc. CATS97, Sydney, Australia, Feb. 1997, pp. 51–58.

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

[8] T. Takaoka, An $O(1)$ time algorithm for generating multiset permutations, ISAAC 99, Madras, India, Lecture Notes in Computer Science.

[9] T. Takaoka, Generation of multi-set permutations and multi-set combinations in $O(1)$ time, Tech. Report, Univ. Canterbury, Christchurch, New Zealand.

[10] V. Vajnovszki, On the loopless generation of binary tree sequences, Inform. Proc. Lett. 68 (1998) 113–117.

[11] T. R. Walsh, Generation of well-formed parenthesis strings in constant worst-case time, J. Algorithms 29 (1998) 165–173.