数理解析研究所講究録
1489巻 2006年 102-105

ビット列で表現された完全二分木の判定アルゴリズム

漆山 龍哉* Tatsuya URUSHIYAMA
三河 賢治† Kenji MIKAWA
長谷川 誠‡ Makoto HASEGAWA

概要

与えられた二分木が完全二分木であるかを線形時間で判定するアルゴリズムを提案する. 本アルゴリズムでは, 論理演算と反復手法を応用し, 長さ n のビット列で表現された二分木を判定するために必要な計算量は $O(n)$ 時間である. また, 入力のビット列を記憶する領域を除くと, 判定に必要な記憶領域は $O(1)$ 領域である. 現在のところ, ビット列で表現されたすべての二色木 (赤黒木) を効率よく列挙するアルゴリズムは知られておらず, 本アルゴリズムが二色木の列挙問題を解決するための一助となることを期待する.

1 まえがき

組合せ的な集合のすべての要素を列挙する問題を扱う場合, 集合の要素をどうのように分類し, 系統的に列挙するアルゴリズムを構築できるか, が問題となる. 集合の要素が $a_1a_2...a_n$ の文字列で表現される集合であれば, 一定の条件を満たすように前半部分または後半部分の文字列を固定して, 要素を分類できる集合も多い. 実際, このような分類を利用した高速な列挙アルゴリズムは数多く発表されており, 文献 [3] に詳しく紹介されている.

データ構造で利用される平衡木の一つである二色木は, 赤と黒の二色の節点を持つことから赤黒木と呼ばれており, (1) 赤節点の子は必ず黒節点である, (2) 葉は必ず黒節点である, (3) 根から葉までの経路に含まれる黒節点の個数 (黒高さ) はどの経路も等しい, という性質を満たす.

二色木の内部節点のうち, 赤節点に 2 のラベル, 黒節点に 1 のラベル, 外部節点 (葉) に 0 のラベルを与えて, 各節点に与えられたラベルを行きがけ順に並べる. このように得られた二色木のラベル列を列挙する問題は, 未解決の問題の一つである. Ball らにより, 2-3-4 木から二色木を生成するアルゴリズムが提案されたが, ラベル列で表現された二色木を系統的に分類し, 列挙するアルゴリズムは知られていない [2].

ラベル列で表現された二色木を列挙する上で, 最も大きな問題は, 与えられたラベル列が二色木であるための必要十分条件が示されていない点にある. 赤節点を含まない二色木は完全二分木であるから, 完全二分木のラベル列の性質から二色木の必要十分条件を導出できるのではないかと考えている. 本論文では, 二色木の列挙問題を解決するための助走的な研究として, 完全二分木を表現するラベル列の性質を明らかにし, 与えられたラベル列が完全二分木であるかを判定するアルゴリズムを提案する.

2 準備

完全二分木の各節点に対する二つのラベル付けを次のように定義する. 完全二分木の内部節点に 1 のラベル, 葉に 0 のラベルを与えたものをラベル付け A(図 1 参照) として, 内部節点の左の子に 0 のラベル, 右の子に 1 のラベルを与えたものをラベル付け B(図 2 参照) とする. ラベル付け B は, 2 進数の辞書式順序 (昇順) に対応するラベル付けであり, 便宜

* 新潟大学大学院自然科学研究科 (Graduate School of Science and Technology, Niigata University)
† 新潟大学総合情報処理センター (Integrated Information Processing Center, Niigata University)
‡ 近畿大学工学部 (School of Engineering, Kinki University)

として, 根に 0 のラベルを与える.

ラベル付け A により与えられたラベルについて, 行きがけ順に並べたラベル列を $\boldsymbol{a} = a_1a_2...a_n$ とする. 完全二分木上で隣り合う葉を v_i, v_j とおき, v_i, v_j に与えられたラベルをそれぞれ a_i, a_j とする. このように定義すると, a_i と a_j はそれぞれ 0 となり, ラベル列 $\boldsymbol{a}$ 上で a_i の次に 0 のラベルとなるものが a_j である.

a_i と a_j の間に含まれる 1 の個数は, v_i と v_j の最小共通先祖 (least common ancestor) の高さに深く関わっている. 行きがけ順に v_i の次にたどられる節点は, v_i と v_j の最小共通先祖の右の子であることに注目すると, a_i と a_j の間に含まれる 1 の個数は, v_i と v_j の最小共通先祖の右の子から v_j までの経路上の節点に与えられた 1 の個数である. v_i と v_j の最小共通先祖の高さを h とおけば, a_i と a_j の間には $h-1$ 個の 1 が連続して含まれる. a_i がレベル列 $\boldsymbol{a}$ の先頭から第 s 番目の 0 であるとき, 次の補題から h を求めることができる.

[補題 2.1] ラベル付け A の完全二分木を行きがけ順にたどって得られるラベル列 $a_1a_2...a_n$ に対して, a_i と a_j はラベル列の先頭から第 s 番目と第 $s+1$ 番目の 0 とする. このとき, a_i と a_j の間に連続して含まれる 1 の個数 t について,

$$t = \log_2(s \times (s \oplus (s-1)))$$

が成り立つ. ここで, $\times$ と $\oplus$ は, それぞれ論理積と排他的論理和を表す二項演算子である.

証明. 完全二分木上で隣り合う葉を v_i, v_j として, v_i と v_j の最小共通先祖の高さを h とする. 最小共通先祖の右の子の高さが

$$\log_2(s \times (s \oplus (s-1))) = h-1$$

であることを示せばよい.

ラベル付け B の完全二分木について, 根から v_i までのラベル列を $\boldsymbol{b} = b_m...b_1b_0$, 根から v_j までのラベル列を $\boldsymbol{b'} = b'_m...b'_1b'_0$ とおく. ラベル列 B の定義から, b_k について,

$$b_k = \begin{cases} 1 & k = 0,1,...,h-2 \text{ のとき} \\ 0 & k = h-1 \text{ のとき} \\ b'_k & k = h,h+1,...,m \text{ のとき} \end{cases}$$

が成り立ち, b'_k について,

$$b'_k = \begin{cases} 0 & k = 0,1,...,h-2 \text{ のとき} \\ 1 & k = h-1 \text{ のとき} \\ b_k & k = h,h+1,...,m \text{ のとき} \end{cases}$$

が成り立つ. したがって, $k = 0,1,...,m$ に対して,

$$b'_k \times (b'_k \oplus b_k) = \begin{cases} 1 & k = h-1 \text{ のとき} \\ 0 & k \neq h-1 \text{ のとき} \end{cases}$$

が成り立つ. ここで, $s-1$ の 2 進数表現は $\boldsymbol{b}$ に一致し, s の 2 進数表現は $\boldsymbol{b'}$ に一致するので,

$$s \times (s \oplus (s-1)) = 2^{h-1}$$

を得る. ゆえに, $\log_2(s \times (s \oplus (s-1))) = h-1$ が成り立つ. 以上, 補題は証明された. □

また, 補題 2.1 から, 次の系を得る.

[系 2.1] ラベル付け A の完全二分木を行きがけ順にたどって得られるラベル列 $\boldsymbol{a} = a_1a_2...a_n$ に対して, $\boldsymbol{a}$ 上の 0 の個数を s' とすると,

$$\log_2(s' \times (s' \oplus (s'-1))) = h$$

が成り立つ. ここで, h は完全二分木の高さである.

3 完全二分木の判定アルゴリズム

与えられた文字列 $\boldsymbol{a} = a_1a_2...a_n, a_i \in \{0,1\}$, が高さ h の完全二分木を表す文字列のとき, a_1 から第 1 番目の 0 までの間に連続する 1 の個数がこの完全二分木の高さ h に等しい. 前章の補題 2.1 で述べたように, $\boldsymbol{a}$ 上の第 s 番目の 0 から第 $s+1$ 番目の 0 までの間に $\log_2(s \times (s \oplus (s-1)))$ 個の 1 が連続する. 系 2.1 から, a_n を第 s' 番目の 0 として, $\log_2(s' \times (s' \oplus (s'-1))) = h$ が成立する.

上記の性質を利用すると, 文字列 $\boldsymbol{a}$ が完全二分木であるかの判定は,

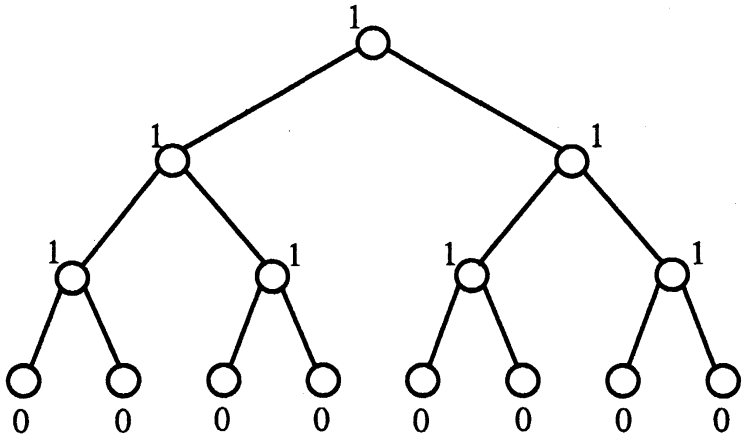

図 1　ラベル付け A の完全二分木

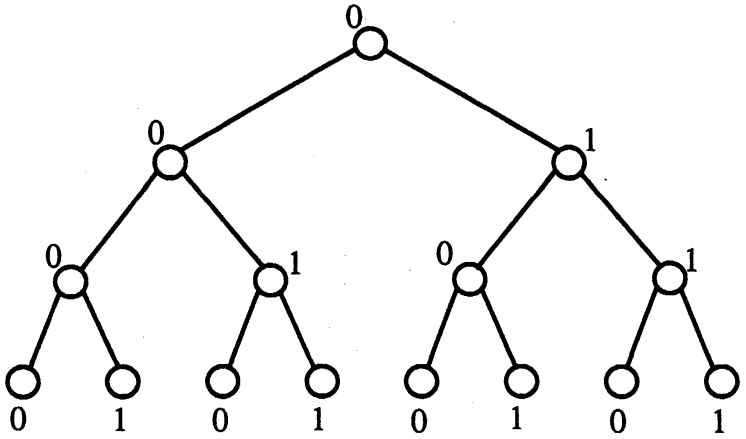

図 2　ラベル付け B の完全二分木

(1) a_1 から連続する 1 の個数 h を完全二分木である場合の木の高さとして記憶する．$h = 0$ のとき，$\boldsymbol{a}$ は完全二分木ではないので判定処理を終了する．

(2) 第 s 番目の 0 に対して，次に続く 1 の個数 t を計数し，$t = \log_2(s \times (s \oplus (s-1)))$ であるかを調べる．$t \neq \log_2(s \times (s \oplus (s-1)))$ のとき，$\boldsymbol{a}$ は完全二分木ではないので判定処理を終了する．この操作を a_{n-1} まで繰り返す．

(3) a_n について，$\log_2(s \times (s \oplus (s-1))) = h$ かつ $a_n = 0$ であるかを調べる．上式が成立しないとき，$\boldsymbol{a}$ は完全二分木ではないので判定処理を終了し，上式が成立するとき，$\boldsymbol{a}$ は完全二分木であると判定する．

のような流れになる．

4　むすび

本論分は完全二分木を判定するアルゴリズムを提案した．未解決の組合せ問題の一つとして，二色木の生成問題が存在する．Ball らにより，2-3-4 木か

ら二色木を生成するアルゴリズムが提案されたが, ビット列で表現された二色木を直接に生成するアルゴリズムは知られていない [2].

ビット列で表現された二色木を生成する上で, 最も大きな問題は, 与えられたビット列表現が二色木であるための必要十分条件が示されていない点にある. 著者らは, 完全二分木のビット列表現から二色木の必要十分条件を導出できるのではないかと考えている. 今後, 本アルゴリズムから二色木の生成問題を検討したい.

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. Assoc. Comput. Mach., Vol.19, No.9, pp.517-521, 1976.

[2] T. Ball, D. Hoffma, F. Ruskey, R. Webber, and L. White, "State Generation and Automated Class Testing," Softw. Test. Verif. Reliab., Vol.10, No.3, pp149-170, 2000.

[3] C.D. Savage, "A Survey of Combinatorial Gray Codes," SIAM Review Vol.39, No.4, pp.605-629, 1997.