Iterative Formulas for Enumerating Binary Trees

Xiang, Limin
Department of Computer Science and Communication Engineering,Kyushu University : Graduate Student

Ushijima, Kazuo
Department of Computer Science and Communication Engineering, Kyushu University

https://doi.org/10.15017/1522388

九州大学大学院
システム情報科学研究科報告
第2巻 第2号 平成9年9月

Research Reports on Information Science and
Electrical Engineering of Kyushu University
Vol.2, No.2, September 1997

Iterative Formulas for Enumerating Binary Trees

Limin XIANG* and Kazuo USHIJIMA**

(Received June 20, 1997)

Abstract: Enumeration is an important aspect for combinatorial properties of binary trees. Traditional solutions for enumerating binary trees are expressed by algorithms and most of them are recursive. In this paper, we give our solutions by iterative formulas for enumerating binary trees. Iterative algorithms can be obtained easily based on the iterative formulas for enumerating binary trees, and the iterative algorithms can be guaranteed to be correct.

Keywords: Binary tree, Enumeration, Lexicographic, Natural ordering

1. Introduction

Various methods have been presented for enumerating binary trees with n nodes[1)–13)]. In general, a one-to-one correspondence is established between the set of binary trees and the set of certain integer sequences, such as *0-1 pairs sequences*[2),5)], *bit-strings*[3),9),12)], *tree permutations*[4)], *weight sequences*[7)], *P-sequences*, *L-sequences*[8)], *codewords*[13)], and *ballot sequences*[10)]; then all the integer sequences are enumerated for the corresponding binary trees in *Natural Ordering*[4),8)], in *Lexicographic Ordering*[3),7),12),13)], or in some other orderings[2),8)]. Obviously, the series of binary trees in *Lexicographic Ordering* may be non-isomorphic from sequences to sequences, while those in *Natural Ordering* are isomorphic and independent of their representation, i.e. sequences.

All the solutions above are described by algorithms and most of them are recursive. It is clear that the problem of enumerating binary trees has been well-researched, but it is also indicated that to reveal properties of the problem, different algorithms reflect different aspects of the problem, in other words, different methods are needed to solve the problem. On the other hand, since enumerating binary trees belongs to combinatorial problems in mathematics, a solution expressed by a mathematical formula should be significant at least in the formalization for solutions and in the correctness for the algorithm based on the formula. In this paper, we establish a one-to-one correspondence between binary trees with n nodes and a kind of character sequences with n length, then we give and prove two iterative formulas for enumerating the character sequences with n length in *Natural Ordering* and in *Lexicographic Ordering* respectively. By a bijection, the iterative formula is shown for enumerating *0-1 pairs sequences* of binary trees in *Lexicographic Ordering.*

* Department of Computer Science and Communication Engineering, Graduate Student

** Department of Computer Science and Communication Engineering

2. Concepts

Nodes can be classified into 4 kinds in a binary trees, i.e., leaf nodes, nodes with a right subtree, nodes with both a left subtree and a right subtree, and nodes with a left subtree. Nodes are labeled by **a**, **b**,**c** and **d** corresponding to their classification in a binary tree, i.e., **a**:leaf nodes, **b**:nodes with a right subtree, **c**:nodes with both a left subtree and a right subtree, and **d**:nodes with a left subtree. Then, these labels are read in preorder to form a word which is called a *B-word.* The set is denoted as $\mathcal{B}_n$ of *B-words* for binary trees with n nodes. Obviously, non-empty $\mathcal{B}_n$ has the following properties.

Properties

1. Binary trees with n nodes and $\mathcal{B}_n$ are one-to-one correspondent.

2. For $\mathbf{W} = \mathbf{w}_1\mathbf{w}_2...\mathbf{w}_n \in \mathcal{B}_n$,

(1) $\mathbf{w}_n = \mathbf{a}$.

(2) The number of **a**'s in $\mathbf{W}$ = the number of **c**'s in $\mathbf{W}$ +1.

(3) If $\mathbf{w}_i = \mathbf{c}$ and $\mathbf{w}_j \neq \mathbf{c}$ for $1 \leq j < i$, then $\mathbf{w}_j \neq \mathbf{a}$ for $1 \leq j < i$.

To give formulas for enumerating *B-words* of $\mathcal{B}_n$ in the next section, some basic concepts must be defined here. The definition of *natural ordering* can be found in [4),8)] for binary trees.

Definition 1.

Given $\mathbf{W} = \mathbf{w}_1\mathbf{w}_2...\mathbf{w}_n \in \mathcal{B}_n$, let

(1) $\mathbf{W}_N^N$ be the successor of $\mathbf{W}$ in *Natural Ordering*;

(2) $\mathbf{W}_L^N$ be the successor of $\mathbf{W}$ in *Lexicographic Ordering*;

(3) $\mathbf{w}_k^B = \begin{cases} \mathbf{b} & ,\mathbf{w}_k=\mathbf{d} \\ \mathbf{w}_k & ,\text{otherwise} \end{cases}$, for $1 \le k \le n$;

(4) $\mathbf{W}(i,j) = \begin{cases} \mathbf{w}_i\mathbf{w}_{i+1}...\mathbf{w}_j & ,1 \le i \le j \le n \\ \epsilon & ,\text{otherwise} \end{cases}$;

(5) $\mathbf{W}^B(i,j) = \begin{cases} \mathbf{w}_i^B\mathbf{w}_{i+1}^B...\mathbf{w}_j^B & ,1 \le i \le j \le n \\ \epsilon & ,\text{otherwise} \end{cases}$;

(6) $\mathbf{s}$, $\mathbf{n}_1$ and $\mathbf{n}_2$ be nonnegative integers, s.t. $\mathbf{W}(\mathbf{s},n)$ is the longest sequence of $\{\mathbf{a},\mathbf{d}\}^*$ ended at $\mathbf{w}_n$, and $\mathbf{W}(\mathbf{s},n)=\mathbf{d}^{\mathbf{n}_1}\mathbf{a}\mathbf{d}^{\mathbf{n}_2}\mathbf{a}\mathbf{W}(\mathbf{s}+\mathbf{n}_1+\mathbf{n}_2+2,n)$, or else, $\mathbf{n}_2 = 0$ and $\mathbf{W}(\mathbf{s},n)=\mathbf{d}^{\mathbf{n}_1}\mathbf{a}\mathbf{W}(\mathbf{s}+\mathbf{n}_1+1,n)$;

(7) $\mathbf{t}$, $\mathbf{n}_a$ and $\mathbf{n}_d$ be nonnegative integers, s.t. $\mathbf{W}(\mathbf{t},n)$ is the longest sequence of $\{\mathbf{d}\}^*\{\mathbf{a}\}^*$ ended at $\mathbf{w}_n$, and $\mathbf{W}(\mathbf{t},n)=\mathbf{d}^{\mathbf{n}_d}\mathbf{a}^{\mathbf{n}_a}$.

3. Two Iterative Formulas

For $\mathcal{B}_n$, the series of binary trees in *Natural Ordering* and that in *Lexicographic Ordering* are non-isomorphic when $n > 4$. For example, $\mathbf{W}$=**cdada** and $\mathbf{W}'$=**cbbaa**, and the corresponding trees are $\mathbf{T}$ and $\mathbf{T}'$ respectively. In *Natural Ordering* $\mathbf{T} < \mathbf{T}'$, while in *Lexicographic Ordering* $\mathbf{T} > \mathbf{T}'$. But there exist the following results.

Lemma

(1) $\mathbf{b}^{n-1}\mathbf{a}$ is the first *B-word* of $\mathcal{B}_n$ in *Natural Ordering* and in *Lexicographic Ordering*;

(2) $\mathbf{d}^{n-1}\mathbf{a}$ is the last *B-word* of $\mathcal{B}_n$ in *Natural Ordering* and in *Lexicographic Ordering.*

Proof.

(1) In *Natural Ordering*, the smallest tree of binary trees with n nodes is one in which every node has a smallest left subtree, i.e. empty left subtree. Therefore, the smallest tree has the corresponding *B-word* $\mathbf{b}^{n-1}\mathbf{a}$. While in *Lexicographic Ordering*, by **Properties 2.(2)** and **(3)**, any *B-word* with two or more **a**'s in $\mathcal{B}_n$ must have one **c** at least and no **a** can be arranged before the first **c**. Therefore, $\mathbf{b}^{n-1}\mathbf{a}$ is the smallest *B-word* of $\mathcal{B}_n$;

(2) In *Natural Ordering*, the biggest tree of binary trees with n nodes is one in which every node has a biggest left subtree, i.e. empty right subtree. Therefore, the biggest tree has the corresponding *B-word* $\mathbf{d}^{n-1}\mathbf{a}$. While in *Lexicographic Ordering*, $\mathbf{d}^{n-1}\mathbf{a}$ is obviously the biggest *B-word* of $\mathcal{B}_n$.

□

Theorem 1.

(1) When $\mathbf{s} = 1$, $\mathbf{W}$ is the last *B-word* of $\mathcal{B}_n$ in *Natural Ordering* ;

(2) when $\mathbf{s} > 1$,

$$\mathbf{W}_N^N = \begin{cases} \mathbf{W}(1,\mathbf{s}-2)\mathbf{d}\mathbf{W}^B(\mathbf{s},n) & ,case_1 \\ \mathbf{W}(1,\mathbf{s}-2)\mathbf{ca}\mathbf{W}^B(\mathbf{s}+1,n) & ,case_2 \\ \mathbf{W}(1,\mathbf{s}-2)\mathbf{d}\alpha & ,case_3 \\ \mathbf{W}(1,\mathbf{s}-1)\beta & ,case_4 \end{cases}$$

here,

$\alpha = \mathbf{W}^B(\mathbf{s},\mathbf{s}+\mathbf{n}_1-1)\mathbf{b}\mathbf{W}^B(\mathbf{s}+\mathbf{n}_1+1,n)$,

$\beta = \mathbf{W}^B(\mathbf{s},\mathbf{s}+\mathbf{n}_1-1)\mathbf{ba}\mathbf{W}^B(\mathbf{s}+\mathbf{n}_1+2,n)$,

$case_1$: $\mathbf{w}_{s-1}$=**b** and $\mathbf{n}_1 = 0$,

$case_2$: $\mathbf{w}_{s-1}$=**b** and $\mathbf{n}_1 > 0$,

$case_3$: $\mathbf{w}_{s-1}$=**c** and $\mathbf{n}_2 = 0$, and

$case_4$: $\mathbf{w}_{s-1}$=**c** and $\mathbf{n}_2 > 0$.

Proof.

(1) Since s=1, $\mathbf{W}=\mathbf{W}(1,n) \in \{\mathbf{a},\mathbf{d}\}^*$, by **Properties 2.(1)** and **(2)**, $\mathbf{W}=\mathbf{d}^{n-1}\mathbf{a}$, by **Lemma**, $\mathbf{W}$ is the last *B-word* of $\mathcal{B}_n$ in *Natural Ordering*;

(2) Let $\overset{(i)}{\mathbf{W}} \in \{\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}\}^*$ and $|\overset{(i)}{\mathbf{W}}|=i$, $\mathbf{FW}(i)=\mathbf{b}^{i-1}\mathbf{a}$ and $\mathbf{LW}(i)=\mathbf{d}^{i-1}\mathbf{a}$, i.e. $\mathbf{FW}(i)$ is the first *B-word* and $\mathbf{LW}(i)$ is the last *B-word* of $\mathcal{B}_i$ in *Natural Ordering.*
For $i = 1$, $\mathcal{B}_i = \{\mathbf{a}\}$.
For $i > 1$, when $\overset{(i)}{\mathbf{W}} \in \mathcal{B}_i$ and $\overset{(i)}{\mathbf{W}} \neq \mathbf{LW}(i)$, by the definition of *Natural Ordering*, $\overset{(i)}{\mathbf{W}}{}_N^N$ can be expressed in the following recursive form.

$$\overset{(i)}{\mathbf{W}}{}_N^N = \left\{ \begin{array}{lll} \mathbf{d}\,\mathbf{FW}(i-1) & (1'), c_{11} & \multirow{2}{*}{C_1} \\ \mathbf{b}\overset{(i-1)}{\mathbf{W}}{}_N^N & (2'), c_{12} & C_1 \\ \mathbf{d}\,\mathbf{FW}(i-1) & (3'), c_{21} & \\ \mathbf{c}\,\mathbf{FW}(j+1)\,\mathbf{FW}(k-1) & (4'), c_{22} & \\ \mathbf{c}\overset{(j)}{\mathbf{W}}{}_N^N\mathbf{FW}(k) & (5'), c_{23} & C_2 \\ \mathbf{c}\overset{(j)}{\mathbf{W}}\overset{(k)}{\mathbf{W}}{}_N^N & (6'), c_{24} & \\ \mathbf{d}\overset{(i-1)}{\mathbf{W}}{}_N^N & (7'), & C_3 \end{array} \right.$$

here,

C_1: $\overset{(i)}{\mathbf{W}} = \mathbf{b}\overset{(i-1)}{\mathbf{W}}$,

c_{11}: $\overset{(i-1)}{\mathbf{W}} = \mathbf{LW}(i-1)$,

c_{12}: $\overset{(i-1)}{\mathbf{W}} \neq \mathbf{LW}(i-1)$,

C_2: $\overset{(i)}{\mathbf{W}} = \mathbf{c}\overset{(j)}{\mathbf{W}}\overset{(k)}{\mathbf{W}}$ $(j \ge 1, k \ge 1, j+k = i-1)$,

c_{21}: $\overset{(j)}{\mathbf{W}} = \mathbf{LW}(j), \overset{(k)}{\mathbf{W}} = \mathbf{a}$,

c_{22}: $\overset{(j)}{\mathbf{W}}= \mathbf{LW}(j), \overset{(k)}{\mathbf{W}}= \mathbf{LW}(k)$,

c_{23}: $\overset{(j)}{\mathbf{W}}\neq \mathbf{LW}(j), \overset{(k)}{\mathbf{W}}= \mathbf{LW}(k)$,

c_{24}: $\overset{(k)}{\mathbf{W}}\neq \mathbf{LW}(k)$, and

C_3: $\overset{(i)}{\mathbf{W}}= \mathbf{d}\ \overset{(i-1)}{\mathbf{W}}$.

Formulas (1′), (3′) and (4′) are non-recursive, formulas (2′), (6′) and (7′) are tail recursive, and formula (5′) is recursive with the change from $\mathbf{LW}(k)$ to $\mathbf{FW}(k)$. So, $\overset{(n)}{\mathbf{W}}$ can be expressed as $\overset{(x)}{\mathbf{W}}\overset{(y)}{\mathbf{W}}\overset{(z)}{\mathbf{W}}$, here, $\overset{(z)}{\mathbf{W}}\in \{\mathbf{a},\mathbf{d}\}^*$, $\overset{(y)}{\mathbf{W}}\in \mathcal{B}_y$ and $\overset{(y)}{\mathbf{W}}{}^N_N$ can be gotten by (1′), (3′) or (4′), and $\overset{(n)}{\mathbf{W}}{}^N_N= \overset{(x)}{\mathbf{W}}\overset{(y)}{\mathbf{W}}{}^N_N\overset{(z)}{\mathbf{W}}{}^B$. Therefore, parameters $\mathbf{s}$, $\mathbf{n}_1$ and $\mathbf{n}_2$ of **Definition 1.(6)** are needed. Such being the case, $\mathbf{w}_{\mathbf{s}-1}(\mathbf{s} > 1)$ can only be $\mathbf{b}$ or $\mathbf{c}$.

(i) When $\mathbf{w}_{\mathbf{s}-1} = \mathbf{b}$,

if $\mathbf{n}_1 = 0$, i.e. $\mathbf{W}= \mathbf{W}(1,\mathbf{s}-2)\mathbf{baW}(\mathbf{s}+1,n)$,

$\overset{(y)}{\mathbf{W}}= \mathbf{ba}$ and $\overset{(z)}{\mathbf{W}}=\mathbf{W}(\mathbf{s}+1,n)$,

$$\mathbf{W}^N_N= \mathbf{W}(1,\mathbf{s}-2)\mathbf{daW}^B(\mathbf{s}+1,n) =\mathbf{W}(1,\mathbf{s}-2)\mathbf{dW}^B(\mathbf{s},n);$$

if $\mathbf{n}_1 > 0$, i.e.

$\mathbf{W}= \mathbf{W}(1,\mathbf{s}-2)\mathbf{bd}^{\mathbf{n}_1}\mathbf{aW}(\mathbf{s}+\mathbf{n}_1+1,n)$,

$\overset{(y)}{\mathbf{W}}=\mathbf{bd}^{\mathbf{n}_1}\mathbf{a}$ and $\overset{(z)}{\mathbf{W}}=\mathbf{W}(\mathbf{s}+\mathbf{n}_1+1,n)$,

$$\mathbf{W}^N_N= \mathbf{W}(1,\mathbf{s}-2)\mathbf{cab}^{\mathbf{n}_1-1}\mathbf{aW}^B(\mathbf{s}+\mathbf{n}_1+1,n) =\mathbf{W}(1,\mathbf{s}-2)\mathbf{caW}^B(\mathbf{s}+1,n);$$

(ii) When $\mathbf{w}_{\mathbf{s}-1} = \mathbf{c}$,

if $\mathbf{n}_2 = 0$, i.e.

$\mathbf{W}= \mathbf{W}(1,\mathbf{s}-2)\mathbf{cd}^{\mathbf{n}_1}\mathbf{aaW}(\mathbf{s}+\mathbf{n}_1+2,n)$,

$\overset{(y)}{\mathbf{W}}=\mathbf{cd}^{\mathbf{n}_1}\mathbf{aa}$ and $\overset{(z)}{\mathbf{W}}=\mathbf{W}(\mathbf{s}+\mathbf{n}_1+2,n)$,

$$\mathbf{W}^N_N= \mathbf{W}(1,\mathbf{s}-2)\mathbf{db}^{\mathbf{n}_1}\mathbf{baW}^B(\mathbf{s}+\mathbf{n}_1+2,n) =\mathbf{W}(1,\mathbf{s}-2)\mathbf{dW}^B(\mathbf{s},\mathbf{s}+\mathbf{n}_1-1)\mathbf{b}\ \mathbf{W}^B(\mathbf{s}+\mathbf{n}_1+1,n);$$

if $\mathbf{n}_2 > 0$, i.e.

$\mathbf{W}= \mathbf{W}(1,\mathbf{s}-2)\mathbf{cd}^{\mathbf{n}_1}\mathbf{ad}^{\mathbf{n}_2}\mathbf{aW}(\mathbf{s}+\mathbf{n}_1+\mathbf{n}_2+2,n)$,

$\overset{(y)}{\mathbf{W}}=\mathbf{cd}^{\mathbf{n}_1}\mathbf{ad}^{\mathbf{n}_2}\mathbf{a}$ and $\overset{(z)}{\mathbf{W}}=\mathbf{W}(\mathbf{s}+\mathbf{n}_1+\mathbf{n}_2+2,n)$,

$$\mathbf{W}^N_N= \mathbf{W}(1,\mathbf{s}-2)\mathbf{cb}^{\mathbf{n}_1}\mathbf{bab}^{\mathbf{n}_2-1}\mathbf{a}\ \mathbf{W}^B(\mathbf{s}+\mathbf{n}_1+\mathbf{n}_2+2,n) =\mathbf{W}(1,\mathbf{s}-1)\mathbf{W}^B(\mathbf{s},\mathbf{s}+\mathbf{n}_1-1)\mathbf{ba}\ \mathbf{W}^B(\mathbf{s}+\mathbf{n}_1+2,n).$$

□

Theorem 2.

(1) When $\mathbf{t} = 1$, $\mathbf{W}$ is the last *B-word* of $\mathcal{B}_n$ in *Lexicographic Ordering* ;

(2) when $\mathbf{t} > 1$,

$$\mathbf{W}^N_L= \begin{cases} \mathbf{W}(1,\mathbf{t}-2)\mathbf{ba}^{\mathbf{n}_a}\mathbf{b}^{\mathbf{n}_d-1}\mathbf{a} & ,case_1 \\ \mathbf{W}(1,\mathbf{t}-2)\mathbf{da}^{\mathbf{n}_a} & ,case_2 \\ \mathbf{W}(1,\mathbf{t}-2)\mathbf{ca}^{\mathbf{n}_a}\mathbf{b}^{\mathbf{n}_d-1}\mathbf{a} & ,case_3 \\ \mathbf{W}(1,\mathbf{t}-2)\mathbf{da}^{\mathbf{n}_a-2}\mathbf{b}^{\mathbf{n}_d+1}\mathbf{a} & ,case_4 \end{cases}$$

here,

case$_1$: $\mathbf{w}_{t-1}=\mathbf{a}$,

case$_2$: $\mathbf{w}_{t-1}=\mathbf{b}$ and $\mathbf{n}_d = 0$,

case$_3$: $\mathbf{w}_{t-1}=\mathbf{b}$ and $\mathbf{n}_d > 0$, and

case$_4$: $\mathbf{w}_{t-1}=\mathbf{c}$.

Proof.

(1) Since $\mathbf{t}$=1, $\mathbf{W}=\mathbf{W}(1,n) \in \{\mathbf{d}\}^*\{\mathbf{a}\}^*$, by **Properties 2.(1)** and **(2)**, $\mathbf{W}=\mathbf{d}^{n-1}\mathbf{a}$, by **Lemma**, $\mathbf{W}$ is the last *B-word* of $\mathcal{B}_n$ in *Lexicographic Ordering*;

(2) In *Lexicographic Ordering*, the rule for the successive increase of *B-words* is that: checking $\mathbf{w}_i$ for i from n down to 1, when $\mathbf{w}_i$ can be changed into the next possible character, after this change $\mathbf{w}_j$ is changed into the smallest possible character for j from $i+1$ to n. Or else, $\mathbf{w}_{i-1}$ is to be checked continously. For $\mathcal{B}_n$, any *B-word* must end at $\mathbf{a}$, and this $\mathbf{a}$ can not be changed into any other character. For a *B-word* ended with several $\mathbf{a}$'s, by **Properties 2.(2)**, any $\mathbf{a}$ of them can not be changed into any other character either. If $\mathbf{w}_i$ checked is $\mathbf{d}$, $\mathbf{w}_{i-1}$ is to be checked continously. Therefore, parameters $\mathbf{t}$, $\mathbf{n}_d$ and $\mathbf{n}_a$ of **Definition 1.(7)** are needed. Such being the case, $\mathbf{w}_{\mathbf{t}-1}(\mathbf{t} > 1)$ can only be $\mathbf{a}$, $\mathbf{b}$ or $\mathbf{c}$.

(i) When $\mathbf{w}_{\mathbf{t}-1} = \mathbf{a}$, i.e.,

$\mathbf{W}= \mathbf{W}(1,\mathbf{t}-2)\mathbf{ad}^{\mathbf{n}_d}\mathbf{a}^{\mathbf{n}_a}$ and $\mathbf{n}_d> 0$,

$\mathbf{W}^N_L= \mathbf{W}(1,\mathbf{t}-2)\mathbf{ba}^{\mathbf{x}}\mathbf{b}^{\mathbf{n}_d+\mathbf{n}_a-\mathbf{x}-1}\mathbf{a}$,

by **Properties 2.(2)**, $\mathbf{x}$ can be evaluated as $\mathbf{n}_a$, i.e., $\mathbf{W}^N_L= \mathbf{W}(1,\mathbf{t}-2)\mathbf{ba}^{\mathbf{n}_a}\mathbf{b}^{\mathbf{n}_d-1}\mathbf{a}$;

(ii) When $\mathbf{w}_{\mathbf{t}-1} = \mathbf{b}$,

if $\mathbf{n}_d = 0$, i.e. $\mathbf{W}= \mathbf{W}(1,\mathbf{t}-2)\mathbf{ba}^{\mathbf{n}_a}$,

by **Properties 2.(2)**, $\mathbf{b}$ can not be changed into $\mathbf{c}$, but into $\mathbf{d}$. Therefore,

$\mathbf{W}^N_L= \mathbf{W}(1,\mathbf{t}-2)\mathbf{da}^{\mathbf{n}_a}$;

if $\mathbf{n}_d > 0$, i.e., $\mathbf{W}= \mathbf{W}(1,\mathbf{t}-2)\mathbf{bd}^{\mathbf{n}_d}\mathbf{a}^{\mathbf{n}_a}$,

$\mathbf{W}^N_L= \mathbf{W}(1,\mathbf{t}-2)\mathbf{ca}^{\mathbf{x}}\mathbf{b}^{\mathbf{n}_d+\mathbf{n}_a-\mathbf{x}-1}\ \mathbf{a}$,

by **Properties 2.(2)**, $\mathbf{x}$ can be evaluated as $\mathbf{n}_a$, i.e., $\mathbf{W}^N_L= \mathbf{W}(1,\mathbf{t}-2)\mathbf{ca}^{\mathbf{n}_a}\mathbf{b}^{\mathbf{n}_d-1}\mathbf{a}$; (iii) When $\mathbf{w}_{\mathbf{t}-1} = \mathbf{c}$, i.e. $\mathbf{W}= \mathbf{W}(1,\mathbf{t}-2)\mathbf{cd}^{\mathbf{n}_d}\mathbf{a}^{\mathbf{n}_a}$,

$\mathbf{W}^N_L= \mathbf{W}(1,\mathbf{t}-2)\mathbf{da}^{\mathbf{x}}\mathbf{b}^{\mathbf{n}_d+\mathbf{n}_a-\mathbf{x}-1}\mathbf{a}$,

by **Properties 2.(2)**, $\mathbf{x}$ can be evaluated as $\mathbf{n}_a-2$, i.e.

$\mathbf{W}_L^N = \mathbf{W}(1, \mathbf{t}-2)\mathbf{d}\mathbf{a}^{\mathbf{n}_a - 2}\mathbf{b}^{\mathbf{n}_d + 1}\mathbf{a}.$

□

It is easy to obtain an iterative algorithm from iterative formulas above. For example, an iterative algorithm can be given as follows for enumerating *B-words* of binary trees in *Natural Ordering.*

Algorithm EBsN

```
(1) $\mathbf{W} \longleftarrow$ the first B-word of $\mathcal{B}_n$;
    ( by Lemma (1) )
(2) Output $\mathbf{W}$ ;
(3) Evaluate $\mathbf{s}, \mathbf{n}_1$ and $\mathbf{n}_2$; ( by Definition 1.(6) )
(4) if $\mathbf{s} = 1$ , then go to (6); ( byTheorem 1.(1) )
(5) $\mathbf{W} \longleftarrow \mathbf{W}_N^N$, go to (2); ( byTheorem 1.(2) )
(6) End .
```

The correctness of this algorithm is guaranteed by **Definition 1**, **Lemma (1)** and **Theorem 1**. Based on **Definition 1**,**Lemma (2)** and **Theorem 2**, the iterative algorithm is similar to **algorithm EBsN** for enumerating $\mathcal{B}_n$ of binary trees in *Lexicographic Ordering*,

4. An Iterative Formula For 0-1 Pairs Sequences

V.Bapiraju and V.V.B.Rao [2] gave an algorithm to enumerate *0-1 pairs sequences* of binary trees neither in *Natural Ordering* nor in *Lexicographic Ordering.* By a bijection, an iterative formula is easily obtained for enumerating *0-1 pairs sequences* of binary trees in *Natural Ordering* or in *Lexicographic Ordering.* Here, we will discuss the method for enumerating *0-1 pairs sequences* of binary trees in *Lexicographic Ordering* only, and the method is similar for enumerating *0-1 pairs sequences* of binary trees in *Natural Ordering.*

We denote **00, 01, 10** and **11** by **0, 1, 2** and **3** respectively.

Definition 2.

Given $\mathbf{W} = \mathbf{w}_1\mathbf{w}_2 \ldots \mathbf{w}_n \in \mathcal{B}_n$, let

(1) $\mathbf{g}(\mathbf{w}_i) = \mathbf{g}_i = \begin{cases} \mathbf{0}, \ \mathbf{w}_i = \mathbf{a} \\ \mathbf{1}, \ \mathbf{w}_i = \mathbf{b} \\ \mathbf{2}, \ \mathbf{w}_i = \mathbf{d} \\ \mathbf{3}, \ \mathbf{w}_i = \mathbf{c} \end{cases}, \ 1 \le i \le n$;

(2) $\mathbf{G}(i,j) = \mathbf{g}(\mathbf{W}(i,j))$

$= \begin{cases} \mathbf{g}(\mathbf{w}_i)\mathbf{g}(\mathbf{w}_{i+1}) \ldots \mathbf{g}(\mathbf{w}_j) & 1 \le i \le j \le n \\ \epsilon & \text{otherwise} \end{cases}$;

(3) $\mathbf{g}(\mathcal{B}_n) = \{\mathbf{g}(\mathbf{W}) \mid \mathbf{W} \in \mathcal{B}_n\}$;

(4) $\mathbf{G}^N$ be the successor of $\mathbf{G}$ for $\mathbf{g}(\mathcal{B}_n)$ in *Lexicographic Ordering* ;

(5) $\mathbf{u}$ and $\mathbf{v}$ be nonnegative integers, s.t. $\mathbf{G}(\mathbf{u}, n)$ is the longest sequence of $\{\mathbf{3}\}^*\{\mathbf{2}\}\{\mathbf{0}\}^* \cup \{\mathbf{3}\}^*\{\mathbf{0}\}^*$ ended at $\mathbf{g}(\mathbf{w}_n)$, and $\mathbf{v}$ = the number of $\mathbf{0}$'s in $\mathbf{G}(\mathbf{u}, n)$ – the number of $\mathbf{3}$'s in $\mathbf{G}(\mathbf{u}, n)$.

Theorem 3.

(1) $\mathbf{1}^{n-1}\,\mathbf{0}$ is the first word of $\mathbf{g}(\mathcal{B}_n)$ in *Lexicographic Ordering* ;

(2) when $\mathbf{u} = 1$, $\mathbf{G}$ is the last word of $\mathbf{g}(\mathcal{B}_n)$ in *Lexicographic Ordering*;

(3) when $\mathbf{u} > 1$,

$$\mathbf{G}^N = \begin{cases} \mathbf{G}(1,\mathbf{u}-2)\mathbf{1}\mathbf{0}^{\mathbf{v}}\mathbf{1}^{n-\mathbf{u}-\mathbf{v}}\mathbf{0} & ,\mathbf{g}_{u-1} = \mathbf{0} \\ \mathbf{G}(1,\mathbf{u}-2)\mathbf{2}\mathbf{0}^{\mathbf{v}-1}\mathbf{1}^{n-\mathbf{u}-\mathbf{v}+1}\mathbf{0} & ,\mathbf{g}_{u-1} = \mathbf{1} \\ \mathbf{G}(1,\mathbf{u}-2)\mathbf{3}\mathbf{0}^{\mathbf{v}}\mathbf{1}^{n-\mathbf{u}-\mathbf{v}}\mathbf{0} & ,\mathbf{g}_{u-1} = \mathbf{2} \end{cases}$$

Proof.

Under the function **g**, the proof for **Theorem 3** is similar to that for **Theorem 2**.

□

Based on **Definition 2** and **Theorem 3**, the iterative algorithm for enumerating *0-1 pairs sequences* of binary trees in *Lexicographic Ordering* is also similar to **algorithm EBsN** wholly.

As a comparison with [2], words of $\mathbf{g}(\mathcal{B}_n)$ for $n = 5$ in *Lexicographic Ordering* are given as follows.

*01.***11110** *02.***11120** *03.***11210** *04.***11220** *05.***11300**
*06.***12110** *07.***12120** *08.***12210** *09.***12220** *10.***12300**
*11.***13010** *12.***13020** *13.***13100** *14.***13200** *15.***21110**
*16.***21120** *17.***21210** *18.***21220** *19.***21300** *20.***22110**
*21.***22120** *22.***22210** *23.***22220** *24.***22300** *25.***23010**
*26.***23020** *27.***23100** *28.***23200** *29.***30110** *30.***30120**
*31.***30210** *32.***30220** *33.***30300** *34.***31010** *35.***31020**
*36.***31100** *37.***31200** *38.***32010** *39.***32020** *40.***32100**
*41.***32200** *42.***33000**

5. Conclusion

If the correspondence is changed between { **a, b, c, d** } and 4 kinds of nodes in a binary tree, some non-isomorphic series of binary trees can be obtained easily by enumerating the corresponding *B-words* in *Lexicographic ordering.*

If the one-to-one correspondence is considered between the set of binary trees and the set of full binary trees, such as in [3),7)−9),12)], there are only 2 kinds of nodes in a full binary tree, i.e., leaf nodes and nodes with both a left and a right subtrees. *B-words* for full binary trees can be formed from $\{\mathbf{a}, \mathbf{c}\}^*$. When the length is n of a *B-word* of binary trees, the length is $2n+1$ of the corresponding *B-word* of full binary trees. The iterative formulas for enumerating full binary trees can be obtained by the discussion similar to that above.

Traditional methods for enumerating binary trees are

Step 1. integer sequences are used to code binary trees, and

Step 2. (recursive) algorithms are given for enumerating all the integer sequences.

Our methods for enumerating binary trees are

Step 1. character sequences are used to code binary trees, and

Step 2. iterative formulas are given for enumerating all the character sequences.

Two steps in our method are both formalized while those in traditional methods are not.

References

1) M.D.Atkinson and J.R.Sack, "Generating binary trees at random", *Inform.Process.Lett.* 41 (1992) 21-23.
2) V.Bapiraju and V.V.B.Rao, "Enumeration of binary trees", *Inform.Process.Lett.* 51 (1994) 125-127.
3) M.C.Er, "Enumerating Ordered Trees Lexicographically", *Comput.J.* 28 (5)(1985) 538-542.
4) G.D.Knott, "A Numbering System for Binary Trees", *Comm.ACM* 20 (2)(1977) 113-115.
5) J.F.Korsh, "Counting and randomly generating binary trees", *Inform.Process.Lett.* 45 (1993) 291-294.
6) J.M.Lucas, D.Roelants van Baronaigien, and F.Ruskey, "On Rotations and the Generation of Binary Trees", *J.Algorithms* 15 (1993) 343-366.
7) J.M.Pallo, "Enumerating, Ranking and Unranking Binary Trees", *Comput.J.* 29 (2)(1986) 171-175.
8) J.M.Pallo and R.Racca, "A Note on Generating Binary Trees in A-order and B-order", *Intern.J.Computer Math.* 18 (1985) 27-39.
9) A.Proskurowski, "On the Generation for Binary Trees", *J.ACM* 27 (1)(1980) 1-2.
10) D.Rotem and Y.L.Varol, "Generation of Binary Trees from Ballot Sequences", *J.ACM* 25 (3)(1978) 396-404.
11) M.Solomon and R.A.Finkel, "A Note on Enumerating Binary Trees", *J.ACM* 27 (1)(1980) 3-5.
12) S.Zaks, "Lexicographic Generating of Ordered Trees", *Theoretical Comput. Sci.* 10 (1980) 63-82.
13) D.Zerling, "Generating Binary Trees Using Rotations", *J.ACM* 32 (3)(1985) 694-701.