長野県短期大学紀要　第48号　57-64頁　1993年12月
Journal of Nagano Prefectural College, No. 48, pp. 57-64, December 1993

2分木の符号化とその応用

清水　道夫*

Coding of Binary Trees and its Application

Michio SHIMIZU*

Abstract: We show one-one correspondence between all binary trees with n internal nodes and certain bit strings, and the number of the bit strings. We then show the algorithms of ranking and unranking for the bit strings. As one of application problem, algorithms for drawing of all binary trees with n nodes are discussed.

1　はじめに

近年，組合せ論的な木の符号化問題，とくに2分木の符号化問題がいくつか研究されている。2分木はデータ構造として最も基本的で重要なもので，その総数がカタラン数として容易に表現できることが知られている[1~3]。木の符号化とは，本来二次元的である木を一次元的な整数列で表現して，ジェネレーティング，ランキング，アンランキングの3つの基本アルゴリズムを考えることである。ジェネレーティングは同じ大きさのすべての2分木（実際には対応する整数列）を生成することで，ランキングは与えられた2分木（整数列）の順序（ランク）を決定することである。アンランキングはランキングの逆で，ランクが与えられたとき対応する木（整数列）を生成する。2分木の符号化問題は数学的に興味深いものであるが，応用面としては，木に関するアルゴリズムのふるまいを考えるときのランダムデータとしての使用を志向している。

いろいろ提案されている2分木の符号化アルゴリズムのうち，Zaksの2進数列（ビットストリング）への符号化がいちばん簡潔である。これは，2分木の内部節に1，葉に0とラベリングして，木を先行順（root-left-right）にたどったときのラベルをならべたものに対応している。このようなビットストリングをX-seqとよび，辞書式順序のもとでのジェネレーティング，ランキング，アンランキングを考えている。しかし，X-seqをランダムデータとして使おうとするとこのままでは非常に使いにくい。これは，木の構造についての情報が不足しているためと考えられる。

そこで，2分木を虫が這うようにたどったときのビットストリングを考える。虫は根から出発して左回りで根に戻って来るが，根から葉に向かう枝（up）を1，その逆（down）を0とする。これに対応するビットストリングをY-seqとよぶ。

*〒380　長野市三輪8-49-7　長野県短期大学
*Nagano Prefectural College, 49-7 Miwa 8-chome, Nagano 380, Japan.

Y-seq は符号という意味では多少冗長であるが，木の構造を反映しており，いろいろな応用に適している。たとえば，あとで述べるように，2分木を描画するときのランダムデータとして使ったり，2分木間の距離[8]（レーベンシュタイン距離の拡張）を計算するときのデータとして用いられる。したがって，符号操作に関する Y-seq 自身の性質を明かにしておくことは意味があると考えられる。

まず，X-seq と Y-seq の性質および相互の変換アルゴリズムを示す。つぎに，Y-seq に関するランキング・アンランキングアルゴリズムを考察する。最後に，符号化問題の一応用として，大きさ n のすべての2分木をグラフィック画面に描画するアルゴリズムを提案する。

2 定 義

2分木の辞書式順序，ランク，ビットストリングなどを定義する。2分木は，空集合であるか，または根と2個の2分木（左右の部分木）から成る。5個の内部節を持つ2分木の例を Fig. 1 に示す。○は内部節を表し，□は葉を表す。節を結ぶものが枝で，ちょうど2本の枝に接続している節が根である。

2つの2分木間の辞書式順序（lexicographic order）をつぎのように再帰的に定義する。T_l, T_rはそれぞれ左部分木，右部分木を表す。

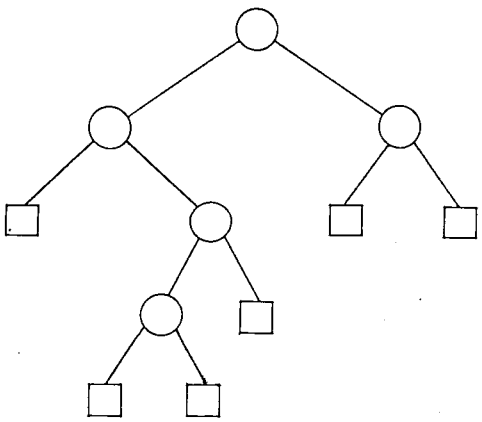

Fig. 1 An example of binary tree

[定義1][1] 2分木 T と T'に対して，次の条件を満たすとき，$T<T'$であるという。

(1) T が空（empty）で T'が空でない。または

(2) T が空でないとき

(a) $T_l<T_l'$，または

(b) $T_l=T_l'$かつ $T_r<T_r'$

ところで，n 個の内部節を持つ2分木の総数 b_nはカタラン数

$$b_n=\frac{\binom{2n}{n}}{n+1}$$

で表されるが[5]，b_n個の2分木を辞書式順序で並べた順番をランクと呼ぶ。

つぎに，2種類のビットストリング（bit string）を定義する。

[定義2][1]次の条件を満足する系列 x を X-seq とする。

(1) x は n 個の1と $n+1$ 個の0を含む。（$n\geq 1$）

(2) x を左から走査するとき，末尾のケタを除く任意のケタまでの0の個数は1の個数よりも多くない。

[定義3] Y-seq y を次のように再帰的に構成する。

$n=0$ ：$y=\phi$（空を表す）

$n=1$ ：$y=$ ”1 0 1 0”

$n>1$ ：$y=$ ”1”$+\alpha+$”0 1”$+\beta+$”0”

ここに，α と β は $|\alpha+\beta|=4$（$n-1$）をみたす Y-seq で，＋は文字列のつながり（concatenation）を示す。

さらに，ビットストリングに対する辞書式順序を次のように定義する。2つのビットストリングを

$u=u_1,\ u_2,\ \cdots,\ u_n,$

$v=v_1,\ v_2,\ \cdots,\ v_n,$

Table 1　RANK, X-seq, Y-seq

R	X-seq	Y-seq
1	101010100	1011011011010000
2	101011000	1011011101001000
3	101100100	1011101001101000
4	101101000	1011101101000100
5	101110000	1011110100100100
6	110010100	1101001101101000
7	110011000	1101001110100100
8	110100100	1101101000110100
9	110101000	1101101101000010
10	110110000	1101110100100010
11	111000100	1110100100110100
12	111001000	1110100110100010
13	111010000	1110110100010010
14	111100000	1111010010010010

とする。$u_j=v_j$ $(j=1,2,\cdots,l\text{-}1)$ で，$u_l<v_l$ なる l $(1\leqq l\leqq n)$ があるとき，$u<v$ であるという。Table 1 に $n=4$ に対する X-seq，Y-seq を示す。

3　ビットストリングの性質

ここでは，まず X-seq と Y-seq が 2 分木に 1 対 1 に対応していることを証明する。つぎに，X-seq と Y-seq の共通点を示し，X-seq について既に得られている結果が，Y-seq に対しても適用できることを示す。

X-seq が 2 分木に 1 対 1 に対応していることは，文献1) で証明されている。X-seq は 2 分木の内部節に 1 を，葉に 0 をラベリングして，そのラベルを先行順に並べたものに対応しており，その長さは $2n+1$ である。たとえば，Fig. 1 の木に対する X-seq は11011000100となる。

Y-seq が 2 分木に 1 対 1 に対応していることは，その構成法からも明かであるが，これが組合せ論における「括弧の問題」[6]に置き換えられることに注意すると，より理解しやすくなる。まず，Y-seq の " 0 1 " を " 0 " に置き換え，つぎに残りの " 1 " と " 0 " をそれぞれ " (" と ") " に変換する。たとえば，$n=2$ に対する11010010と10110100は，それぞれ ((0)0) と (0(0)) に対応する。そして，括弧のつけかたの数は，カタラン数に等しいことが証明されており，次の補題が成り立つ。

[補題 1] Y-seq は 2 分木と 1 対 1 に対応する。

実際，Y-seq は 2 分木の根を出発し，左回りにすべての枝を行きと帰りの 2 度たどるときの系列に対応している。n 個の内部節を持つ 2 分木の枝の数は $2n$ であるから，行きを 1 とし，帰りを 0 とすると，Y-seq は長さが $4n$ の 2 進数列になる。たとえば，Fig. 1 の木に対する Y-seq は11011101001000110100となる。

つぎに，X-seq と Y-seq が辞書式順序の意味で一致していることを示す。

[補題 2] n 個の内部節をもつ 2 つの 2 分木をそれぞれ T，T' とする。T に対する X-seq，Y-seq をそれぞれ x，y とし，T' に対する X-seq，Y-seq をそれぞれ x'，y' とする。このとき，$x<x'$ $(T<T')$ であれば $y<y'$ である。

(証明) $T<T'$ ならば $x<x'$ であることは文献1) に示されている。まず，2 分木の左端路上の左枝の数と内部節の数が等しいから，この部分に対する X-seq と Y-seq の連続する 1 の数は同じである。また，$T<T'$ であるから，T と T' を先行順にたどったとき，左端の路長が異なるような部分木 S と S' が存在するはずである。部分木 S に至るまでの x と x'，および部分木 S' に至るまでの y と y' の部分列はそれぞれ共通であるから，辞書式順序は部分木 S と S' の左端の路長によって決まる。したがって，$x<x'$ であれば $y<y'$ であるといえる。　(証明終)

とくに，2 分木 T の X-seq を x，Y-seq を y とすると，x と y の左端から連続する 1 の個数は等しい。このことは Table 1 を見ても明かである。じつは，Table 1 の R はランクを示してお

り，先に2分木とビットストリングとの1対1対応が証明されたから，2分木のランクをビットストリングのランクと考えてもよい。

5節では，2分木およびビットストリングのランキングを考えるが，そのために同じ特徴を持つ木またはビットストリングごとにグループ分けして，その個数をあらかじめ数え上げておく必要がある。その特徴とは，2分木では左端路上の内部節の数または左端路上の枝の数であり，ビットストリングでは左端から連続する1の個数（これをLとする）である。たとえば，Table 1では$R=1\sim5$が$L=1$，$R=6\sim10$が$L=2$，$R=11\sim13$が$L=3$，$R=14$が$L=4$である。

ところで，左端路上の内部節の数と枝の数が等しいから，X-seqについてのZaksの結果がY-seqにもそのままあてはまる。それによると，内部節の数がnで，連続する1の個数（左端路上の内部節の数）が$j+1$以上である2分木の総数$a(n,j)$は，

$$a(n,j)=a(n,j+1)+a(n-1,j-1)$$
$$=\frac{j+2}{2n+j}\binom{2n-j}{n-j-1}\quad(0\leqq j\leqq n-1)$$

で表される。このnについての再帰的な関係が，5節で述べるアルゴリズムのキーポイントになっている。$a(n,j)$はjの少ないほうに累積した2分木の総数であることに注意する。たとえば，

Table 2 $a(n,j)$

n \ j	0	1	2	3	4	5	6	7	8
1	1								
2	2	1							
3	5	3	1						
4	14	9	4	1					
5	42	28	14	5	1				
6	132	90	48	20	6	1			
7	429	297	165	75	27	7	1		
8	1430	1001	572	275	110	35	8	1	
9	4862	3432	2002	1001	429	154	44	9	1

$n=4$のとき$a(4,0)=14$，$a(4,1)=9$，$a(4,2)=4$，$a(4,3)=1$となるが，これはTable 1で確かめられる。Table 2に$n=1\sim9$に対する$a(n,j)$を示す。

4 変換アルゴリズム

X-seqとY-seqの相互の変換アルゴリズムを示す。つぎの補題は，Y-seqからX-seqに容易に変換できることを示している。

[補題3] Y-seqは，つぎのアルゴリズムによってX-seqに変換される。

アルゴリズムYX：Y-seqを左から右に走査し，1のつぎの01を0におきかえる。かつ，0のつぎの01および0のつぎの0を除く。

（証明）nについての帰納法により証明する。

[I] $n=1$のとき：1010 → 100

$n=2$のとき：11010010 → 11000

10110100 → 10100

だから成り立つ。

[II] $n<n'$のとき成り立つと仮定して，$n=n'$のときを示す。

左部分木に対するY-seqとX-seqを，それぞれ，A，A'とし，右部分木に対するY-seqとX-seqをそれぞれB，B'とすると，Y-seqは1A01B0で表され，X-seqは1$A'B'$で表される。

ところで，AとBの末尾のケタはともに0だから，Y-seqから01と0を除くと1ABになる。帰納法の仮定により，AとBはそれぞれA'とB'に変換されるから，Y-seqはX-seqに変換される。（証明終）

アルゴリズムYXは，Y-seqを一回だけ走査する。したがって，計算時間はY-seqの長さに比例するから，$O(n)$である。

つぎに，X-seqをY-seqに変換するアルゴリズムXYを示す。ステップ1のiはX-seqのケタ位置を表し，jはスタックポインタである。ステップ2ではX-seqの連続する3ケタを調べ，

条件に合っていればその部分に対応するY-seqをスタック$B(j)$に保存する。"102"，"120"はそれぞれ右部分木，左部分木として結合することを示している。ここで注意することは，スタックにあとから保存されるものから結合されること(LIFO)である。これによって結合の順番が保証されている。ステップ4では部分的な変換が終了したことを示すために，連続する3けたを"2"（0と1以外であればなんでもよい）に圧縮する。これによって，X-seqの長さLが2ケタだけ短くなる。Lが1になったときに終了し，最終的なY-seqは$B(1)$に入る。

アルゴリズムXY

ステップ1：$i=3$，$j=0$。

ステップ2：$x=x(1)x(2)\cdots x(L)$のうち，$x(i-2)x(i-1)x(i)$を調べ，それが

"100"であれば，$j=j+1$，$B(j)=$"1010"，ステップ4へ

"102"であれば，$B(j)=$"101"$+B(j)+$"0"，ステップ4へ

"120"であれば，$B(j)=$"1"$+B(j)+$"010"，ステップ4へ

"122"であれば，$j=j-1$，$B(j)=$"1"$+B(j)+$"01"$+B(j+1)+$"0"，ステップ4へ。

ステップ3：$i=i+1$，ステップ2へもどる。

ステップ4：$x=x(1)\cdots x(i-3)+$"2"$+x(i+1)\cdots x(L)$とする。

ステップ5：$i=i-2$，$L=L-2$。

ステップ6：$L=1$ならば終了。

ステップ7：$i=2$ならば$i=3$とする。

ステップ8：ステップ2へジャンプ。　□

このアルゴリズムの能率を評価するために，スタックの深さとステップ2での比較回数を考える。X-seqの連続する3ビットのうち，"100"はすべて"1010"に変換されてスタックに保存されるから，最大個の"100"を含むようなX-seqを考えれば，スタックの深さがわかる。これは2分木における部分木の最大数に等しいから$(n+1)/2$である。たとえば，$n=9$のとき，$x=110011001100100$を考えると，スタックの深さは$(9+1)/2=5$となる。つぎに，比較回数であるが，これはステップ4での圧縮回数に関係している。圧縮回数は内部節の数nに等しいから，比較回数はたかだか，ケタ数$-2+$圧縮回数$=2n+1-2+n=3n-1$となる。したがって，アルゴリズムXYの計算時間は$O(n)$である。また，作業領域は，スタックの深さとY-seqの長さの積に比例するから$O(n^2)$である。

5　ランキングアルゴリズム

Y-seqに対するランキング・アンランキングアルゴリズムを示す。ランキングアルゴリズムは，内部節の数nとY-seqが与えられたとき，ランクRを求めるもので，アンランキングアルゴリズムは，nとRが与えられたとき，対応するY-seqを求めるものである。アルゴリズムは，$n=2$の2分木を初期木として，左端路上の適当な位置に$n=1$の2分木を逐次挿入することによって，任意の木が生成できることに基づいている。挿入とはFig. 2のように，部分木αが右部分木となることであり，丸で囲った部分が$n=1$の木の挿入を示している。Y-seqでこの部分を考えると，

$\alpha \rightarrow$ "101"$+\alpha+$"0"

になる。ただし，αが葉のときは$\alpha=$""(空)になる。この考え方はアンランキングそのもので

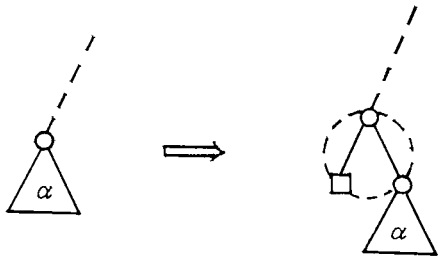

Fig. 2　Insertion

あり，ランキングはまったくこの逆である。

アルゴリズム RANK

入力 n, y；出力 R

ステップ1：$i=n$。

ステップ2：yを$y=\gamma+$"101"$+\delta$のように3つの部分列に分ける。ここに，γはすべて1からなる部分列で，その長さをsとする。$x(i)=s+1$。

ステップ3：部分列δをさらに$\delta=\alpha+$"0"$+\beta$のように3つの部分に分ける。ここに，δの左端のケタが0ならば$\alpha=$""とし，そうでなければ，αは（$|\alpha| \bmod 4=0$）かつ（αの1の個数と0の個数が等しい）を満たす最小の部分列とする。

ステップ4：$y=\gamma+\alpha+\beta$。

ステップ5：$i=3$ならばステップ6へ，そうでなければ$i=i-1$としてステップ2へ。

ステップ6：$y=$"11010010"ならば$Z=1$とし，$y=$"10110100"ならば$Z=2$とする。

ステップ7：$i=3$。

ステップ8：$j=x(i)$，$Z=Z+a(i,j)$。

ステップ9：$i=n$ならばステップ10へ，そうでなければ$i=i+1$としてステップ8へ。

ステップ10：$R=a(n,0)-Z+1$。 □

アルゴリズム RANK の2つの特徴を説明しておく。1つは部分列αを取り出すところである。これは部分木に対応しているはずだから，αのケタの長さが4の倍数になっていること，および1の個数と0の個数が等しいことの2つの条件で判別している。もう1つは，Zをカウントしているところである。$a(n,j)$は，jの少ないほうに累積した数であるから，まず，対象外の個数Zを求め，最後に全体の数（カタラン数）からそれを引いてランクRを求めている。

（例）$n=8$，$y=$"11011101110100100010001110100100"のとき，ランクRを求める。

まず，$n=8$に対して，$j+1=3$以上のものは対象外であるから，$a(8,2)=572$を除く。Y-seqから"101"と"0"を除くと，$n=7$に対して$y=$"11101110100100010011101001 00"となる。ここで，$j+1=4$以上のものは対象外であるから，$a(7,3)=75$を除く。同様にして対象外のものを求めると，

$Z=a(8,2)+a(7,3)+a(6,4)+a(5,3)+a(4,2)+a(3,1)+1$

$=572+75+6+5+4+3+1=668$

となる。最後の1は，yが最終的に$y=$"11010010"となるからである。よってランクは

$R=1430-666+1=765$

となる。参考までに，yの変化を書いておく。丸で囲まれたところが削除される。

$n=8$：$y=$"1(10)1101110100100010(0)1110100100"

$n=7$：$y=$"11(01)1010010(0)1001110100100"

$n=6$：$y=$"111(01)001001001110100100"

$n=5$：$y=$"11(101)001001110100100"

$n=4$：$y=$"1(101)01110100100"

$n=3$：$y=$"(10)110100100"

$n=2$：$y=$"11010010"

アルゴリズム UNRANK

入力 n, R；出力 y

ステップ1：$Z=a(n,0)-R+1$。

ステップ2：$a(n,i)<Z\leqq a(n,i-1)$なるiにたいし，$X(n)=i$とする。$Z=Z-a(n,i)$。

ステップ3：$n=3$ならばステップ4へ，そうでなければ$n=n-1$としてステップ2へ。

ステップ4：$Z=1$ならば$y=$"11010010"，$s=2$とし，$Z=2$ならば$y=$"10110100"，$s=1$とする。

ステップ5：$i=3$。

ステップ6：yを$y=\gamma+\delta$のように2つの部分列に分ける。ここに，$|\gamma|=x(i)-1$である。$s>x(i)-1$ならばステップ7へ，そうでなければ$\alpha=$""（空），$\beta=\delta$としてステップ8へ。

ステップ7：$\delta=\alpha+\beta$とする。ここに，αは

（$|\alpha| \bmod 4 = 0$）かつ（αの1の個数と0の個数が等しい）なる最小の部分列である。

ステップ8：$y=\gamma+$"101"$+\alpha+$"0"$+\beta$, $s=x(i)$。ステップ9：$i=n$ならば終了，そうでなければ$i=i+1$としてステップ6へ。 □

RANKおよびUNRANKの計算時間は，2からnまでの各ループにおけるY-seqの長さに比例するからO（n^2）である。なお，ランクRのY-seqは，UNRANKのかわりに，X-seqに対するアンランキングと変換アルゴリズムXYの組合せによっても求めることができる。Zaksによると，X-seqに対するアンランキングの計算時間はO（n）で，アルゴリズムXYがO（n）だから，ランクRのY-seqをO（n）の計算時間で求めることができる。ただし，アルゴリズムXYの作業領域がO（n^2）であるのに対し，UNRANKの作業領域はO（n）である。

6　2分木の描画

Y-seqによる符号化の一応用として，大きさnのすべての2分木をグラフィック画面に描画する方法を提案する。このアルゴリズムは2段階から成っている。

(1) 与えられたnに対するすべてのY-seqを生成する。

(2) 生成された個々のY-seqをもとに，2分木を画面に描く。

大きさnに対するすべてのY-seqを生成するには，2通りの方法が考えられる。一つは，大きさnに対するカタラン数b_nをもとめ，ランク1～b_nに対して前節のアンランキングアルゴリズムUNRANKを適用する方法である。もう一つは，つぎに示すZaksのジェネレーティングを用いるものである。このアルゴリズムは，大きさnに対するすべてのX-seqを容易に生成する。そして，4節のアリゴリズムXYによって個々のX-seqからY-seqに変換すればよい。

Zaksのジェネレーティングアルゴリズムの記述を簡潔にするために，X-seqのビット1のケタの位置を表す整数列Z-seqを用いる。たとえば，X-seq 10110100に対するZ-seqは1346である。つぎのアルゴリズムGENEは，Z-seqを$\{Z\}=z(1), z(2), \cdots, z(n)$で表し，$\{Z\}$から$\{Z'\}$を求める。Z-seqからX-seqへの変換は明かであろう。

アルゴリズムGENE

ステップ1：初期設定として$\{Z\}=1, 2, \cdots, n$を与える。

ステップ2：$\{Z\}$を右から左に走査し，$z(j)<2j-1$をみたす最右のけたjをみつける。そのようなケタが存在しないときは終了。

ステップ3：$z'(i)=z(i)$, $i<j$,

$z'(j)=z(j)+1$,

$z'(i)=z'(i-1)+1$,

$i=j+1, \cdots, n$。

ステップ4：$\{Z'\}$をあらたに$\{Z\}$とし，ステップ2へジャンプ。 □

つぎに，個々のY-seqから一定の条件で2分木を描くアルゴリズムDRAWを示す。ここに，一定の条件とは，親が左子と右子の中間にあること，同じレベルの子は一直線上にあること，レベル間の距離は等しいことなどである。枝がなるべく交差しないように，根からはなれるほど同じレベルの節の間隔が狭くなるようにしている。変数iはY-seqのケタの位置を表し，Tは木の深さを表す。

アルゴリズムDRAW

ステップ1：$i=1$，$T=0$：初期設定。

ステップ2：$T=T+1$：アップ。

ステップ3：SUB 1：左枝をかく。

ステップ4：$i=i+1$：つぎのケタ。

ステップ5：$y(i)=1$ならばステップ2へジャンプ。

ステップ6：SUB 2：左枝をもどって右枝をか

く。
ステップ7：$i=i+2$：つぎのつぎのケタ。
ステップ8：$y(i)=1$ならばステップ2へジャンプ。
ステップ9：SUB 3：右枝をもどる。
ステップ10：$T=T-1$：ダウン。
ステップ11：$T=0$ならば終了。
ステップ12：$i=i+1$：つぎのケタ。
ステップ13：$y(i)=0$かつ$y(i+1)=1$ならばステップ6へジャンプ，そうでなければステップ9へジャンプ。
サブルーチンの変数：$X1$，$X2$：X座標，$Y1$，$Y2$：Y座標，W：X軸方向変位，D：Y軸方向変位
SUB 1：$X2=X1-W/T$：$Y2=Y1+D$：LINE($X1$, $Y1$)-($X2$, $Y2$)：$X1=X2$：$Y1=Y2$
SUB 2：$X1=X1+W/T$：$Y1=Y1-D$：$X2=X1+W/T$：$Y2=Y1+D$：LINE($X1$, $Y1$)-($X2$, $Y2$)：$X1=X2$：$Y1=Y2$
SUB 3：$X1=X1-W/T$：$Y1=Y1-D$ □

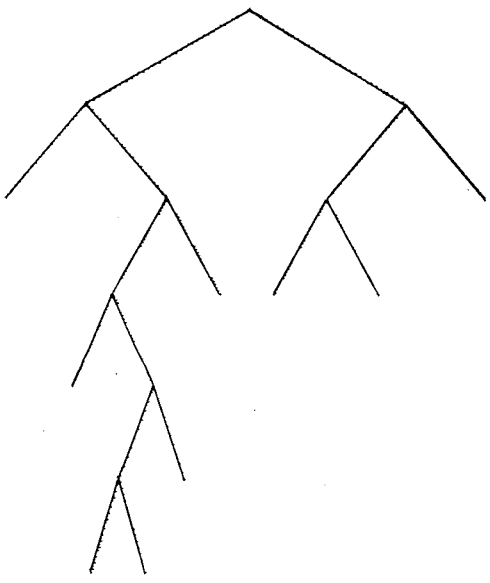

Fig. 3 An example ($n=8$, $R=765$)

ここに，LINEは2点間に直線を引くことを意味している。例として$n=8$，$R=765$に対する画面表示のハードコピーをFig. 3に示す。アルゴリズムDRAWの計算時間はY-seqの長さに比例するから，$O(n)$である。

7 おわりに

符号操作に関するY-seqの性質として，X-seqとの変換やランキング・アンランキングが可能であることを述べた。ジェネレーティングについてはいまのところ効率のよい方法がみつかっていないが，部分木のローテーション（rotation）による方法が可能と考えられる。これは分類（sort）や探索（search）における2分木の更新（update）を表現しており，ランダムデータとしてY-seqを用いる意味からも興味深い。

文 献

1) Zaks S.: Lexicographic Generation of Ordered Trees, Theoretical Computer Science, Vol. 10, pp. 63-82 (1980).
2) Zerling D.: Generating Binary Trees Using Rotations, J. ACM, Vol. 32, pp. 694-701 (1985).
3) Rotem D. and Varol Y. L.: Generation of Binary Trees from Ballot Sequences, J. ACM, Vol. 25, pp. 369-404 (1978).
4) Knuth D. E.: The Art of Computer Programming 1, 3, Addison-Wesley Reading, MA. (1971), (1973).
5) マーチン・ガードナー，数学ゲーム，サイエンス，Vol. 6, 8, pp. 120-126 (1976).
6) 数学セミナー増刊，数学100の問題，日本評論社，pp. 63-66 (1984).
7) Supowit K. J. and Reingold E. M.: The Complexity of Drawing Trees Nicely, Acta Informatica, Vol. 18, pp. 377-392 (1983).
8) 中林，鎌田：2分木間の距離とその計算法，信学論（D），J66-D, No. 4, pp. 445-451 (1983).