STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLVI**, Number 1, 2001

CODING OBJECTS RELATED TO CATALAN NUMBERS

ANTAL BEGE AND ZOLTÁN KÁSA

ABSTRACT. A coding method using binary sequences is presented for different computation problems related to Catalan numbers. This method proves in a very easy way the equivalence of these problems.

1. Introduction

The Catalan numbers, named after the french mathematician E. C. Catalan, defined as

$$C_n = \frac{1}{n+1}\binom{2n}{n},$$

are as known as the Fibonacci numbers. These numbers arise in a lot of combinatorial problems as the number of some objects. The Catalan number C_n describe, among other things,

- the number of binary trees with n nodes,
- the number of ways in which parantheses can be placed in a sequence of $n+1$ numbers to be multiplied two at a time,
- the number of well-formed reverse Polish expressions with n operands and $n+1$ operators,
- the number of paths in a grid from $(0,0)$ to (n,n), increasing just one coordinate by one at each step, without crossing the main diagonal,
- the number of n-bit sequences that the number of 1s never exceeds the number of 0s in each position from left to right,
- the number of ways you can draw non-crossing segments between $2n$ points on a circle in the plane,
- the number of sequences $(x_1, x_2, \ldots, x_{2n})$, with $x_i \in \{-1, 1\}$ for all i between 1 and $2n$ and having the following properties for all partial sums: $x_1 \geq 0$, $x_1+x_2 \geq 0$, $\ldots$, $x_1+x_2+\ldots+x_{2n-1} \geq 0$, $x_1+x_2+\ldots+x_{2n} = 0$,
- the number of ways a polygon with $n+2$ sides can be cut into n triangles,
- the number of frieze pattern with $n+1$ rows,

2000 *Mathematics Subject Classification.* 11B75, 68P30, 68R05.

1998 *CR Categories and Descriptors.* G.2.1 [**Mathematics and Computing**]: Discrete Mathematics – *Combinatorics, Counting problems.*

Research supported by Sapientia Foundation: `http://www.sapientia.ro`.

- the number of mountain ranges you can draw using n upstrokes and n downstrokes,
- the number of ways n votes can come in for each of two candidates in an election, with the first never behind the second.

The Catalan numbers are the solution of the following recurrence equation:

$$C_{n+1} = C_0C_n + C_1C_{n-1} + \ldots + C_nC_0 \quad \text{for } n \geq 0, \text{ with } C_0 = 1.$$

Another recurrence equation for the Catalan numbers is:

$$(n+2)C_{n+1} = (4n+2)C_n \quad \text{for } n \geq 0, \text{ with } C_0 = 1.$$

The generating function of these numbers is

$$\sum_{n\geq 0} C_n z^n = \frac{1-\sqrt{1-4z}}{2z},$$

which can be obtained from the first recurrence equation given above using generating function techniques (see e. g. [5] for computing the number of n-node binary trees).

Let $C(z) = \sum_{n\geq 0} C_n z^n$ be the generating function corresponding to the Catalan numbers. By the recurrence equation this function satisfy the following equation:

$$zC^2(z) = C(z) - 1, \quad \text{with } C(0) = 1.$$

From this

$$C(z) = \frac{1-\sqrt{1-4z}}{2z}$$

results. By developping in series we will get the followings:

$$C(z) = \frac{1}{2z}\left(1-\sqrt{1-4z}\right) = \frac{1}{2z}\left(1 - \sum_{n\geq 0}\binom{\frac{1}{2}}{n}(-4z)^n\right) =$$

$$= \sum_{n\geq 0}\binom{\frac{1}{2}}{n+1}(-1)^n 2^{2n+1} z^n = \sum_{n\geq 0}\frac{1}{n+1}\binom{2n}{n},$$

and from this the formula for Catalan numbers results.

2. The Encoding

We shall present here an encoding method of objects whose number is a Catalan number. Each object will be codified by a binary sequence in which the number of 0s is equal to the number of 1s, and from the beginning to any position of the sequence, the number of 1s never exceeds the number of 0s. Let us call these sequences *Catalan sequences*.

The mathematical definition of the Catalan sequence is given below. Let us denote by $n_1(u)$ the number of 1s and by $n_0(u)$ the number of 0s in the sequence

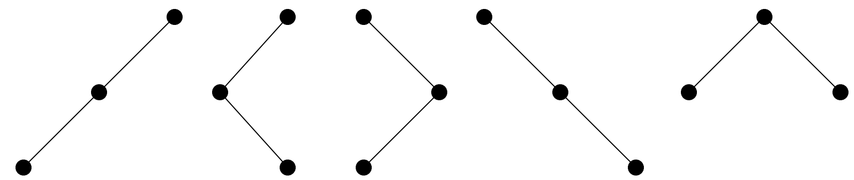
u. A sequence $u = u_1u_2\ldots u_{2n}$, with $u_i \in \{0,1\}$ for $i = 1,2,\ldots,2n$, is a *Catalan sequence* if

$$\begin{aligned} n_1(u_1u_2\ldots u_i) &\le n_0(u_1u_2\ldots u_i) \quad \text{for} \quad i = 1,2,\ldots,2n \\ n_1(u) &= n_0(u) \end{aligned}$$

Our coding method is different from the one given in [8] for binary trees.

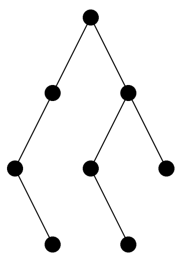
There are a lot of papers which deal with the Catalan numbers, in references we give only a few of them.

2.1. **Encoding of binary trees.** The encoding of a binary tree is the following: when a vertex has only one descendant, we put the sequence 01 for a single left edge, 10 for a single right edge, and 00 for the left edge resp. 11 for the right edge when there are two descendants. We complete the resulting sequence with 0 at the beginning and 1 at the end. The encoding is made using a preorder traversal of the binary tree. In the case of the binary trees with 3 nodes we shall have the encoding in Fig. 1.



0 01 01 1   0 01 10 1   0 10 01 1   0 10 10 1   0 00 11 1

Fig. 1. Encoding of binary trees



0 00 01 10 11 00 10 11 1

Fig. 2. A more complex example

A more complex example, when the preorder traversal can be easily seen, is given in Fig. 2.

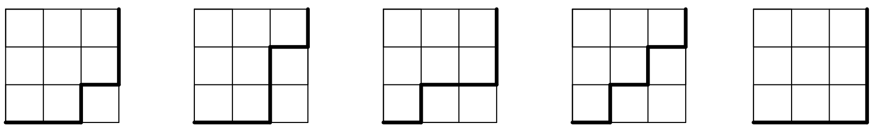
The encoding algorithm for a binary tree B is given as follow in a pseudocode-form. Let us denote by $\emptyset$ the empty binary tree (with no vertices). The **put** statement puts its argument in the resulting output sequence.

Algorithm for encoding a binary tree
put 0
procedure encoding (B):
  Let B_L *be the left and* B_R *the right subtree of* B
  if $B_L \neq \emptyset$ and $B_R = \emptyset$ **then**
    put 01
    call encoding (B_L)
  if $B_L = \emptyset$ and $B_R \neq \emptyset$ **then**
    put 10
    call encoding (B_R)
  if $B_L \neq \emptyset$ and $B_R \neq \emptyset$ **then**
    put 00
    call encoding (B_L)
    put 11
    call encoding (B_R)
end procedure
put 1

For an empty binary tree the procedure has no effect. The proof that the resulting sequence is a Catalan sequence is immediately by the above algorithm.

2.2. **Encoding of paths in grid.** We shall put 0 for a horizontal unit of the path and 1 for a vertical one. The resulting sequence is a Catalan one because the path never cross the main diagonal of the grid.

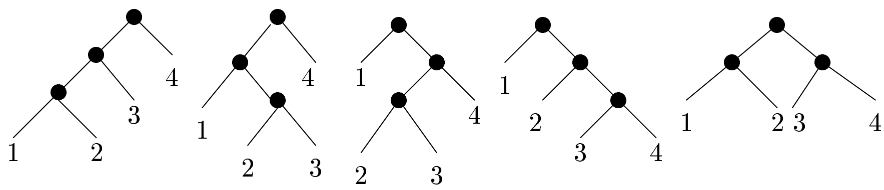
In the case of a grid 3×3 the following paths and codes results (see Fig. 3).



001011 001101 010011 010101 000111

Fig. 3. Encoding of paths in grid

2.3. **Encoding of expressions with multiplications.** To encode expressions we first attach to each expression for multiplication a binary tree by a very simple method. If we multiple a by b, this yields a binary tree with a root and two descendant nodes a and b. A multiplication of two expressions yields a binary tree with two subtrees which are the binary trees corresponding to the two expressions. In the resulting binary tree each internal nodes has exactly two descendants. Such trees are called *extended binary trees.* To encode an extended binary tree we shall omit all leaves (with of course the corresponding edges) in the tree corresponding to the multiplication expression and use the encoding method presented before for the resulting binary tree. For $n = 4$ we shall have the expressions and the corresponding extended binary trees in Fig. 4. If we omit all leaves with the adjacent edges in these extended trees the binary trees and the corresponding encoding result.



(((1 2) 3) 4) ((1 (2 3)) 4) (1 ((2 3) 4)) (1 (2 (3 4))) ((1 2) (3 4))

Fig. 4. Encoding of multiplications

2.4. **Encoding of sequences.** We encode sequences $(x_1, x_2, \ldots, x_{2n})$, with $x_i \in \{-1, 1\}$ for all i between 1 and $2n$ and having the following properties for all partial sums: $x_1 \geq 0,\ x_1 + x_2 \geq 0,\ \ldots,\ x_1 + x_2 + \ldots + x_{2n-1} \geq 0,\ x_1 + x_2 + \ldots + x_{2n} = 0$. We shall code -1 in the sequence by 1 and 1 by 0. It is easy to see that in any positions the number of 1s never exceeds the number of 0s, and they are equals in the sequence (because the sum of all $2n$ elements is equal to 0), so the resulting sequence is a Catalan one. For example:

1,	1	1,	-1,	-1,	-1	coded by:	000111
1,	1	-1,	1,	-1,	-1	coded by:	001011
1,	1	-1,	-1,	1,	-1	coded by:	001101
1,	-1	1,	1,	-1,	-1	coded by:	010011
1,	-1	1,	-1,	1,	-1	coded by:	010101

2.5. **Encoding of segments.** If we have $2n$ points on a circle in the plane and n non-crossing segments between them, the encoding is the following: Let us mark the points clockwise on the circle with numbers from 1 to $2n$. For a segment between i and j $(i < j)$ put 0 in the i^{th} position and 1 in the j^{th} position in the code sequence. For $n = 3$ see Fig. 5. It is easy to see that the resulting sequence is a Catalan one.

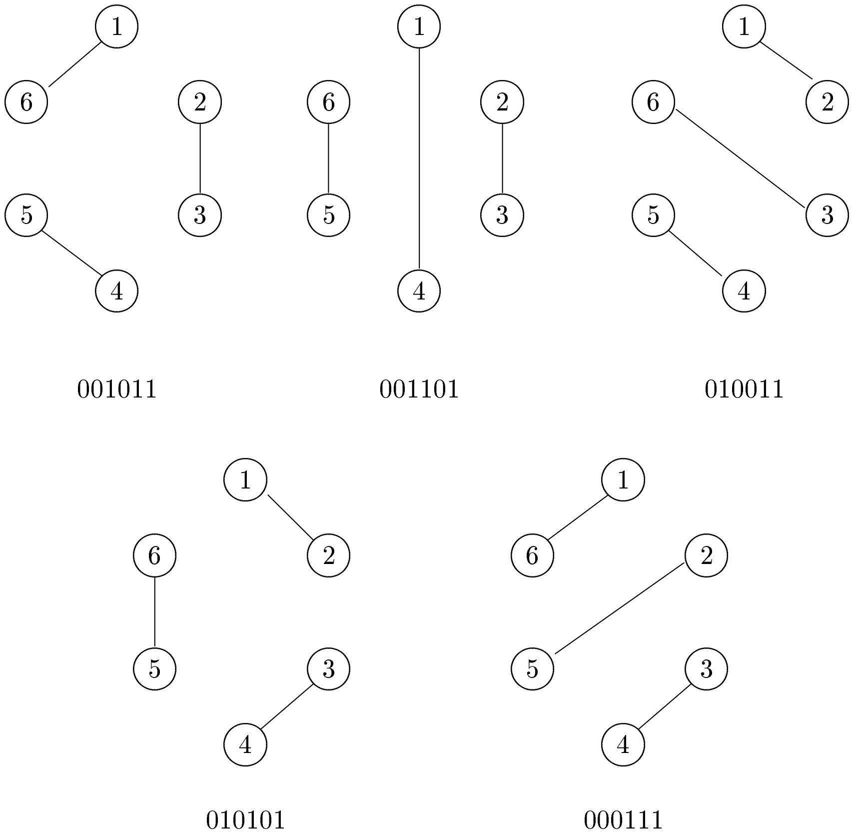


Fig. 5. Encoding of segments

2.6. **Encoding of reverse Polish expressions.** We shall code each operand by 0 and each operator by 1, and add at the end of the resulting sequence an 1. For example, if we have the reverse Polish expression $aaa \times a \times \times$ — which corresponds to the expression $(a \times ((a \times a) \times a))$ — the resulting code is 00010111.

2.7. **Encoding of polygons.** The polygon is divided into triangles. We consider one node in each triangle, and one outside of each side of the polygon. Join by an edge two nodes if the corresponding triangles (or a triangle and the outside of the polygon) have a side in common. We shall get a tree, on which the encoding will be made. If we mark one side of the polygon and the corresponding edge of the tree, and eliminate all edges from the tree that have an endpoint as a leave, we shall get a binary tree (the root will be the node which is adjacent with the marked edge). The exemplification will be made for $n = 3$ (pentagon). The marked side is AB. (See Fig. 6)

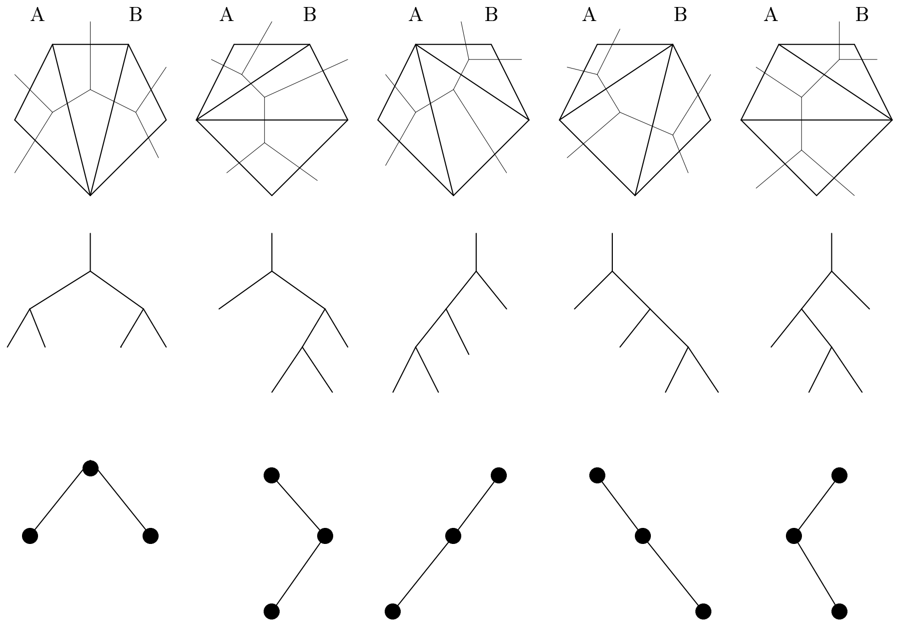


Fig. 6. Encoding of polygons

3. The Decoding

If we have a Catalan sequence, from this the corresponding object can be easily obtained. Let us consider for exemplification the sequence 00010111.

If we want to obtain the corresponding binary tree, we shall omit the first 0 and the last 1. The subsequence 00 is for a left edge (in a stack we shall keep its position), the following subsequence is 10 corresponding to a single right edge, the remaining subsequence 11 is a right edge (corresponding to the edge kept in the stack). The binary tree obtained is in Fig. 7.a.

For the segments we search the first subsequence 01, trace the corresponding segment, omit it from the sequence and continue with the remaining sequence (keeping the original positions) (Fig. 7.b).

For the multiplication we first draw the corresponding binary tree (Fig. 7.a), complete it to having two descendants for each node. The resulting extended binary tree give us the order of multiplications (Fig. 7.c).

The path in the grid is obtained immediately: we draw a horizontal unit segment for each 0 and a vertical one for each 1 (Fig. 7.d).

From these examples general algorithms to obtain related objects from the Catalan sequences can be easily given. We shall describe only the algorithm to obtain a binary tree from a Catalan sequence.

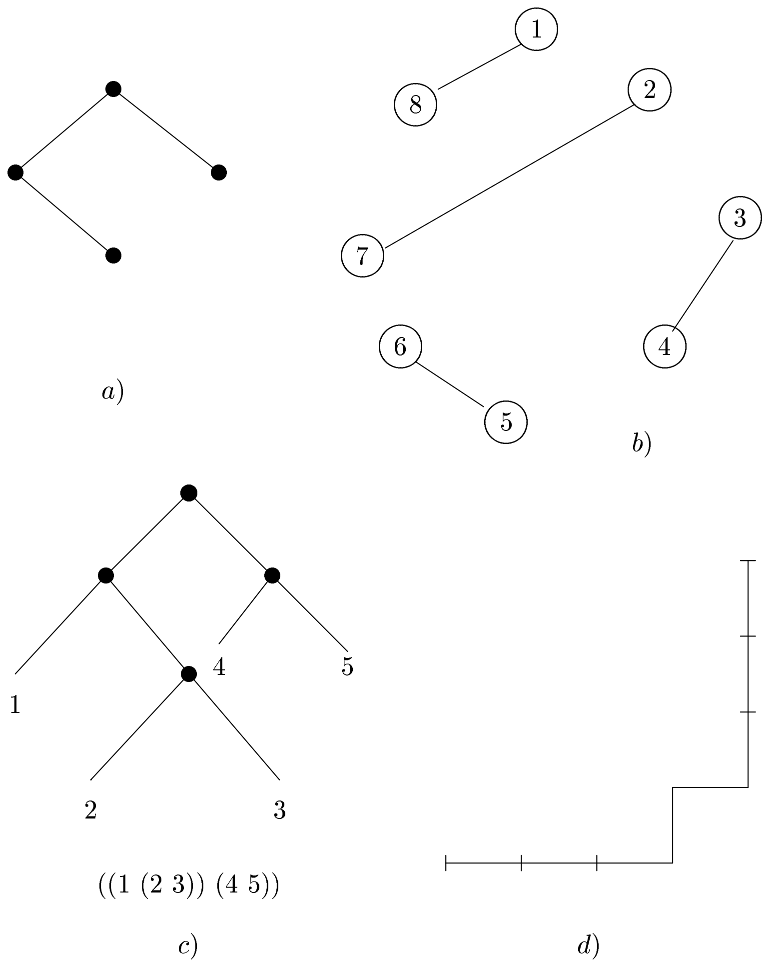


Fig. 7. Decoding

The following recursive algorithm is to decode a Catalan sequence in a binary tree. This algorithm is valid only for correct Catalan sequences. The **get** statement gets the next two digits from the sequence. We shall use the notion of current vertex to denote a vertex from which an edge is drawn. After drawing an edge from the current vertex the adjacent new vertex will be the current one.

Algorithm to decode a Catalan sequence into a binary tree

Input a Catalan sequence
Output a binary tree

delete 0 from the beginning and 1 from the end of the input sequence,
and draw a vertex (the root of the tree) as current vertex
procedure decoding (c):
get ab
delete ab from c
if $ab = 01$ **then**
draw a left edge from the current vertex
call decoding (c)
if $ab = 10$ **then**
draw a right edge from the current vertex
call decoding (c)
if $ab = 00$ **then**
put in the queue the position of the current vertex
draw a left edge from the current vertex
call decoding (c)
if $ab = 11$ **then**
get from the queue the position of a vertex,
this will be the current vertex
draw a right edge from the current vertex
call decoding (c)
end procedure

4. Conclusions

Our presentation give a uniform method to encode objects whose number is a Catalan number. The resulting code is a so-called Catalan sequence formed of equal number of binary digits 0 and 1, in which the number of 1s never exceeds the number of 0s from left to right. This method is important, beside the easy handling, because coding an object in a Catalan sequence and after decoding it in another kind of object, the equivalence of these problems can be easily seen. To prove that the number of objects in a class is a Catalan number it is enough to use the encoding method to obtain a Catalan sequence.

References

[1] Cofman, J., Catalan numbers for the classroom?, *Elemente der Mathematik*, **52** (1997), 108–117.
[2] Dickau, R. M., Catalan numbers,
`http://forum.swarthmore.edu/advanced/robertd/catalan.html`
[3] Dershowitz, N., Zaks, S., Enumeration of ordered trees, *Discrete Mathematics*, **31** (1980), 9–28.

[4] Egleton, R. B., Guy, R. K., Catalan strikes again! How likely is a function to be convex?, *Mathematics Magazine*, **67**, 4 (1988), 211–219.
[5] Knuth, D. E., The art of computer programming, Volume 1 (Fundamental algorithms), Addison-Wesley, 1981 (Second edition).
[6] Singmaster, D., An elementary evaluation of the Catalan numbers, *Amer. Math. Monthly*, **85**, 5 (1978), 366–368.
[7] Vilenkin, N. I., Combinatorica (in Russian), Nauka Moscow, 1969.
[8] Weisstein, E. W., Catalan numbers, `http://mathworld.wolfram.com/CatalanNumbers.html`
[9] ***, Information on binary trees, `http://www.theory.csc.uvic.ca/~cos/inf/tree/BinaryTrees.html`

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, RO 3400 Cluj-Napoca, Str. Kogălniceanu 1, Romania
E-mail address: `bege@math.ubbcluj.ro, kasa@cs.ubbcluj.ro`