

# THE BALLOT PROBLEM

Michael L. GARGANO<sup>1</sup>, Louis V. QUINTAS<sup>2</sup>, and Eric M. WAHL<sup>2</sup>

<sup>1</sup>Computer Science Department, Pace University  
New York, NY 10038 U.S.A.  
mgargano@pace.edu

<sup>2</sup>The New York Institute for Bioengineering and Health Science  
30 Fifth Avenue #1E  
New York, NY 10011 U.S.A.  
qmail02@attglobal.net, ericwahl.md@verizon.net

## Abstract

If there are two candidates for an elective office and  $2n$  voters such that  $n$  vote for one candidate and  $n$  vote for the other candidate, then in how many ways can one sequentially count the votes so that a given candidate is always ahead or tied with the other candidate? This is called the *Ballot Problem*. Variations of this problem and algorithmic lexicographic orderings for these sequences will be discussed.

**Keywords:** Ballot Problem, lexicographic ordering, angiogenesis

## 1. Introduction

### 1.1. The Ballot Problem: $n(B) + n(C) = 2n$

The following is called the *Ballot Problem*. Suppose Bob and Carol are candidates for an elective office and there are  $2n$  voters such that  $n$  vote for Bob and  $n$  vote for Carol. In how many ways can the votes be sequentially counted so that Bob is always ahead of or tied with Carol? (see [1]).

The answer to this question is the same as the number of sequences of length  $2n$  consisting of  $n$   $B$ 's,  $n$   $C$ 's, and such that for every *initial subsequence* the number of  $B$ 's is greater than or equal to the number of  $C$ 's. It is known (see [1]) that the solution of the Ballot Problem is the *Catalan Number*

$$C_n = \frac{1}{n+1} \binom{2n}{n} = \frac{(2n)!}{(n+1)!n!}.$$

Call a sequence consisting solely of  $B$ 's and  $C$ 's a *BC-sequence* and let  $n(x)$  denote the *number of  $x$ 's* in a given sequence (not necessarily restricted to *BC*-sequences).

A restatement of the Ballot Problem and its solution is as follows.

If  $A(m)$  is the *number of BC-sequences of length  $m$  such that  $n(B) = n(C) = n$  and for each initial subsequence  $n(B) \geq n(C)$* , then  $m = 2n$  and  $A(2n) = C_n$ . A *BC*-sequence of this type is called an  *$A(m)$ -sequence*.

A recursive formula for  $A(m)$  is obtained as follows.

If  $m = 0$ , then the sequence is empty and we define  $A(0) = 1$ .

If  $m = 2$ , then  $A(2) = 1$ , realized by the  $A(2)$ -sequence  $BC$ .

For  $m \geq 4$  and  $i = 0, 2, \dots, m-2$ , every  $A(m)$ -sequence is of the form: a  $B$  followed by an  $A(i)$ -sequence (of length  $i$ ) followed by a  $C$  followed by a  $A(j)$ -sequence (of length  $j$ ), where  $i + j = m - 2$ .

Then, recursively we have

$$A(m) = A(0)A(m-2) + A(2)A(m-4) + \dots + A(m-2)A(0)$$

or equivalently

$$A(m+2) = A(0)A(m) + A(2)A(m-2) + A(4)A(m-4) + \dots + A(m)A(0)$$

Keeping in mind that  $m = 2n$ , let  $A(m) = b_{\frac{m}{2}} = b_n$ . This yields,

$$b_{n+1} = b_0b_n + b_1b_{n-1} + b_2b_{n-2} + \dots + b_nb_0$$

Applying Grimaldi's result on pp. 501-502 of [2], namely, the solution of this recursion is  $b_n = C_n$ , we obtain,  $A(m) = b_n = C_n$ . Therefore,  $A(2n) = C_n$ .

## 1.2. An algorithm for the listing of $A(m)$ -sequences

The following algorithm gives a lexicographic listing of all  $A(m)$ -sequences.

Using Rosen's pseudocode style (see Chapter 2, p. 21 and A-2, in [5]) we have:

### ALGORITHM 1.

$n(B) + n(C) = m = 2n$ ,  $n(B) = n(C) = n$ , and each initial subsequence satisfies  $n(B) \geq n(C)$ .

```

get( $n$ )
 $m = 2n$ 
initialize ( $b_p[1], \dots, b_p[n] = 1, 2, \dots, n$ )
process ( $b_p[1], \dots, b_p[n]$ )
while ( $b_p[1], \dots, b_p[n] \neq n + 1, n + 2, \dots, 2n$ )
    next- $a(b_p[1], \dots, b_p[n])$ 
    process ( $b_p[1], \dots, b_p[n]$ )

```

```

procedure next- $a(b_p[1], \dots, b_p[n])$ 
     $i = n$ 
    while  $b_p[i] = n + i$ 
         $i = i - 1$ 
     $b_p[i] = b_p[i] + 1$ 
    for  $j = (i + 1)$  to  $n$ 
         $b_p[j] = b_p[j] + j - i$ 
    return ( $b_p[1], \dots, b_p[n]$ )

```

```

procedure process ( $b_p[1], \dots, b_p[n]$ )
     $i = 1$ 
    while ( $b_p[i] \leq 2i - 1$  and  $i \leq n$ )
         $i = i + 1$ 
    if  $i = n + 1$ , then initialize ( $a[1], a[2], \dots, a[m] = C, C, \dots, C$ )
        for  $j = 1$  to  $n$ 
             $a[b_p[j]] = B$ 
        print ( $a[1], a[2], \dots, a[m]$ )

```

## 2. A Variation on the Ballot Problem: $n(B) + n(C) = m$

Let  $A^*(m)$  denote the number of  $BC$ -sequences of length  $m$  such that  $n(B) + n(C) = m$  and for each initial subsequence  $n(B) \geq n(C)$ . Call a  $BC$ -sequence of this type an  $A^*(m)$ -sequence.

$$\textbf{Theorem 2.1. } A^*(m) = \begin{cases} 1 & \text{if } m = 0 \\ 2A^*(m-1) - C_{\frac{m-1}{2}} & \text{if } m \text{ is odd, } m \geq 1 \\ 2A^*(m-1) & \text{if } m \text{ is even, } m \geq 2 \end{cases}$$

**Proof.**  $A^*(0) = 1$ , by definition.

If  $m$  is odd, then an  $A^*(m-1)$ -sequence has even length. If such a sequence is extended with either a  $B$  or a  $C$ , we obtain  $2A^*(m-1)$  sequences of length  $m$ . However, for those  $A^*(m-1)$ -sequences that have  $n(B) = n(C)$  the addition of a  $C$  does not produce an  $A^*(m)$ -sequence (since this would create too many  $C$ 's). Thus, we must subtract the number of  $BC$ -sequences of length  $m-1$  such that  $n(B) = n(C) = \frac{m-1}{2}$  and each initial subsequence satisfies  $n(B) \geq n(C)$ . This number is  $A^*(m-1) = C_{\frac{m-1}{2}}$  (see Section 1.1). Therefore,  $A^*(m) = 2A^*(m-1) - C_{\frac{m-1}{2}}$ .

If  $m$  is even, then every  $A^*(m-1)$ -sequence has  $n(B) > n(C)$ . Thus, every  $A^*(m-1)$ -sequence can be extended by adding either a  $B$  or a  $C$  in the  $m$ -th position. Thus,  $A^*(m) = 2A^*(m-1)$ , when  $m$  is even. ■

$$\textbf{Corollary 2.2. } A^*(m) = \begin{cases} 2^m - \sum_{i=0}^{\frac{m-2}{2}} 2^{m-1-2i} C_i & \text{when } m \text{ is even} \\ 2^m - \sum_{i=0}^{\frac{m-1}{2}} 2^{m-1-2i} C_i & \text{when } m \text{ is odd} \end{cases}$$

**Proof.** Recursively substitute  $A^*(m-1)$  using the result of Theorem 2.1. ■

**Problem 1.** Obtain a closed formula for  $A^*(m)$ .

### 2.1. An algorithm for the listing of $A^*(m)$ -sequences

The following algorithm gives a lexicographic listing of all  $A^*(m)$ -sequences.

**ALGORITHM 2.**

$n(B) + n(C) = m$  and each initial subsequence satisfies  $n(B) \geq n(C)$ .

```

get( $m$ )
initialize ( $a[1], \dots, a[m] = B, \dots, B$ )
print ( $a[1], \dots, a[m]$ )
while ( $a[1], \dots, a[m] \neq B, C, B, C \dots$ )
    next-a ( $a[1], \dots, a[m]$ )

procedure next-a ( $a[1], \dots, a[m]$ )
     $i = m$ 
    while ( $a[i] = C$ )
         $a[i] = B$ 
         $i = i - 1$ 
     $a[i] = C$ 
     $n(C) = 0$ 
     $n(B) = 0$ 
     $j = 1$ 
    while ( $j \leq m$  and  $n(B) \geq n(C)$ )
        if  $a[j] = B$  then  $n(B) = n(B) + 1$ 
        else  $n(C) = n(C) + 1$ 
         $j = j + 1$ 
    if ( $j = m + 1$  and  $n(B) \geq n(C)$ ) then print ( $a[1], \dots, a[m]$ )
    return ( $a[1], \dots, a[m]$ )

```

**3. A further variation on the Ballot Problem:**

$$n(B) + n(S) + n(C) = m$$

Assume there is a third choice for voting in the Bob and Carol election. This could be a third candidate but with interest still focused on the Bob and Carol votes or the third choice could simply be a "no preference" vote denoted with an  $S$ .

Here the sequence formulation of this problem is as follows.

Using notation analogous to that used for  $A^*(m)$ -sequences we shall seek the number  $A^{**}(m)$  of  $BSC$ -sequences of length  $m = n(B) + n(S) + n(C)$  such that for each initial subsequence  $n(B) \geq n(C)$ . Call such a sequence an  $A^{**}(m)$ -sequence. The solution is given in the following theorem.

**Theorem 3.1.**  $A^{**}(m) = \sum_{k=0}^m \binom{m}{k} A^*(m-k)$

**Proof.** Let  $k$  be the number of  $S$ 's in a given  $A^{**}(m)$ -sequence.

For a given  $k$ , the  $S$ 's can be put in  $\binom{m}{k}$  different positions in an  $A^{**}(m)$ -sequence. Corresponding to each such distribution of  $k$ 's an  $A^*(m-k)$ -sequence can be placed in order in the remaining  $m-k$  positions. Since there are  $A^*(m-k)$  of the latter sequences, we have the number of  $A^{**}(m)$ -sequences having  $k$   $S$ 's is  $\binom{m}{k} A^*(m-k)$ . Therefore, the total number of  $A^{**}(m)$ -sequences is  $A^{**}(m) = \sum_{k=0}^m \binom{m}{k} A^*(m-k)$ . ■

**Problem 2.** Obtain a closed formula for  $A^{**}(m)$ .

### 3.1. An algorithm for the listing of $A^{**}(m)$ -sequences

The following algorithm gives a lexicographic listing of all  $A^{**}(m)$ -sequences.

#### ALGORITHM 3.

$n(B) + n(S) + n(C) = m$  and each initial subsequence satisfies  $n(B) \geq n(C)$ .

```

get( $m$ )
initialize ( $a[1], \dots, a[m] = S, \dots, S$ )
print ( $a[1], \dots, a[m]$ )
while ( $a[1], \dots, a[m] \neq BCBC \dots$ )
    next- $a$  ( $a[1], \dots, a[m]$ )

```

```

procedure next- $a$  ( $a[1], \dots, a[m]$ )
     $i = m$ 
    while ( $a[i] = C$ )
         $a[i] = S$ 
         $i = i - 1$ 
    if  $a[i] = S$ , then  $a[i] = B$ 
    if  $a[i] = B$ , then  $a[i] = C$ 
     $n(C) = 0$ 
     $n(B) = 0$ 
     $j = 1$ 
    while ( $j \leq m$  and  $n(B) \geq n(C)$ )
        if  $a[j] = B$  then  $n(B) = n(B) + 1$ 
        if  $a[j] = C$  then  $n(C) = n(C) + 1$ 

```

```
         $j = j + 1$ 
    if ( $j = m + 1$  and  $n(B) \geq n(C)$ ) then print ( $a[1], \dots, a[m]$ )
    return ( $a[1], \dots, a[m]$ )
```

**Problem 3.** Can Algorithms 1, 2, and 3 be made more efficient?

## 4. An application in biology

In [3](also see [4]) a model for angiogenesis in the renal glomerulus was defined. It is assumed that the vascular network of a renal glomerulus starts with a single vessel and then by three mechanisms called budding, splitting, and connecting, the single vessel develops into an adult vascular network. Denote these mechanisms by  $B$ ,  $S$ , and  $C$ , respectively.

Thus, a given adult vascular network is assumed to have developed through a  $BSC$ -sequence corresponding to the mechanisms noted above. The splitting and budding mechanisms can occur at any step of the development. However, it is a property of these mechanisms that a connecting mechanism cannot occur without an available budding mechanism to have preceded it. Thus, we have the  $n(B) \geq n(C)$  condition for each initial subsequence of the  $BSC$ -sequence that leads to an adult vascular network.

Therefore, the number of sequences of length  $m$  corresponding to  $B$ ,  $S$ , or  $C$  mechanisms that result in a renal glomerulus vascular network is  $A^{**}(m)$ . This number is given in Theorem 3.1.

### Acknowledgments

MLG acknowledges the partial support through research grants from The School of Computer Science and Information Systems, Pace University during the preparation of this work.

### References

- [1] E. Weisstein, *Eric Weisstein's World of Mathematics*, see WEB page, <http://mathworld.wolfram.com/BallotProblem.html>.
- [2] R.P. Grimaldi, *Discrete and Combinatorial Mathematics*, Addison-Wesley, 3rd Edition (1994).

- [3] L.V. Quintas and E.M. Wahl, Random graph process models for angiogenesis, CSIS Pace University Technical Report Series, Report No. 183 (2002).
- [4] M.L. Gargano, L.L. Lurie, L.V. Quintas, and E.M. Wahl, A graph theory analysis of renal glomerular networks *Microvascular Research* (in press).
- [5] K.H. Rosen, *Discrete Mathematics and its Applications, Fifth Edition*, McGraw Hill, New York (2003).