

# Some New Optimum Golomb Rectangles

James B. Shearer  
IBM Research Division  
T.J. Watson Research Center  
P.O. Box 218  
Yorktown Heights, N.Y. 10598  
jbs@watson.ibm.com

Submitted: April 3, 1995; Accepted: June 1, 1995

**Abstract:** We give some new optimum Golomb rectangles found by computer search.

**AMS Subject Classification.** 05B99

In [2] Robinson defined a Golomb rectangle as an  $N \times M$  array of ones and zeros such that the two-dimensional autocorrelation has three values: 0, 1 and  $K$ , where  $K$  is the number of ones in the array. This means that the positions of the ones in any nonzero integral translation of the rectangle will overlap with the positions of the ones in the original position of the rectangle in at most one place. Equivalently, the differences between the positions of every pair of ones in the rectangle, considered as vectors, are distinct. See also [1]. Let  $G(N, M)$  be the maximum number of ones that can be present in an  $N \times M$  Golomb rectangle. For example  $G(2, 2) = 3$ . Robinson defined an optimum Golomb rectangle to be one containing  $G(N, M)$  ones. We prefer to add the conditions  $G(N, M) > G(N - 1, M)$  and  $G(N, M) > G(N, M - 1)$ .

In table 1 we give a number of new optimum Golomb rectangles found by computer search. In most cases they are far from unique. Note there exists a  $2 \times 18$  rectangle with 9 ones. The rectangle in Robinson's table V is  $2 \times 20$  an apparent misprint.

A brief description of the computer program used to find these rectangles follows. Recall that a Golomb ruler is a set of integers  $a_1 < a_2 < \cdots < a_k$  for which the  $\binom{k}{2}$  differences  $\{a_j - a_i | 1 \leq i < j \leq k\}$  are distinct. We will use the following easy lemma.

**Lemma 1:**  $N \times M$  Golomb rectangles with  $K$  ones correspond 1-1 with  $K$  element Golomb rulers with elements chosen from the set  $\{i + (2N - 1)(j - 1) | 1 \leq i \leq N, 1 \leq j \leq M\}$ .

**Proof:** Let  $\{(b_i, c_i) | 1 \leq b_i \leq N, 1 \leq c_i \leq M, 1 \leq i \leq K\}$  be a set of  $K$  positions in a  $N \times M$  rectangle. We claim this set consists of the positions of the ones in a  $N \times M$  Golomb rectangle with  $K$  ones iff the set  $\{a_i = b_i + (2N - 1)(c_i - 1) | 1 \leq i \leq K\}$ , is a

Golomb ruler with elements chosen from the set  $\{i + (2N - 1)(j - 1) \mid 1 \leq i \leq N, 1 \leq j \leq M\}$ . For suppose  $(b_{i_1}, c_{i_1}) - (b_{i_2}, c_{i_2}) = (b_{i_3}, c_{i_3}) - (b_{i_4}, c_{i_4})$ . Then

$$\begin{aligned} a_{i_1} - a_{i_2} &= (b_{i_1} + (2N - 1)(c_{i_1} - 1)) - (b_{i_2} + (2N - 1)(c_{i_2} - 1)) \\ &= (b_{i_1} - b_{i_2}) + (2N - 1)(c_{i_1} - c_{i_2}) \\ &= (b_{i_3} - b_{i_4}) + (2N - 1)(c_{i_3} - c_{i_4}) \\ &= (b_{i_3} + (2N - 1)(c_{i_3} - 1)) - (b_{i_4} + (2N - 1)(c_{i_4} - 1)) \\ &= a_{i_3} - a_{i_4}. \end{aligned}$$

Conversely suppose  $(a_{i_1} - a_{i_2}) = (a_{i_3} - a_{i_4})$ . Then

$$\begin{aligned} (b_{i_1} + (2N - 1)(c_{i_1} - 1)) - (b_{i_2} + (2N - 1)(c_{i_2} - 1)) \\ = (b_{i_3} + (2N - 1)(c_{i_3} - 1)) - (b_{i_4} + (2N - 1)(c_{i_4} - 1)). \end{aligned}$$

It follows that

$$(b_{i_1} - b_{i_2}) - (b_{i_3} - b_{i_4}) + (2N - 1)((c_{i_1} - c_{i_2}) - (c_{i_3} - c_{i_4})) = 0.$$

Now  $1 \leq b_{i_1}, b_{i_2}, b_{i_3}, b_{i_4} \leq N$ . It follows that

$$-(2N - 1) < (b_{i_1} - b_{i_2}) + (b_{i_3} - b_{i_4}) < (2N - 1).$$

Therefore we must have  $(b_{i_1} - b_{i_2}) - (b_{i_3} - b_{i_4}) = 0$  and  $(c_{i_1} - c_{i_2}) - (c_{i_3} - c_{i_4}) = 0$ . Hence  $(b_{i_1}, c_{i_1}) - (b_{i_2}, c_{i_2}) = (b_{i_3}, c_{i_3}) - (b_{i_4}, c_{i_4})$ . This suffices to prove the claim and the lemma.  $\square$

Lemma 1 means that searches for Golomb rectangles can be performed with a modified version of the author's Golomb ruler search program (see [3]). This program performs a straightforward depth first backtrack search. It builds up a Golomb ruler by picking  $a_1, a_2, \dots, a_k$  in order ( $a_j$  being picked at level  $j$  of the search tree). At each node of the search tree the program keeps track of which differences have been used (i.e. are formed by pairs of the elements which have already been picked) and of which integers can be adjoined to the current ruler without violating the distinct difference condition. The sons of a node at level  $j$  are formed by setting  $a_{j+1}$  to each of the elements in the level  $j$  eligibility list in turn. The search tree is pruned (i.e. the program backtracks) when too few integers remain eligible to allow completion of the ruler, when not enough small differences remain unused to allow completion of the ruler or when allowed by symmetry conditions. (Symmetry conditions allow search trees to be pruned because we need only generate one member of each symmetry class of solutions. Clever use of symmetry can produce dramatic improvements in running times.) The following symmetry conditions were used in the Golomb rectangle program. Assume at least half the ones are in the left half of the rectangle (flip left and right if necessary). Assume the top half of the first column

contains a one (clearly in an optimum rectangle the first column must contain a one, flip top and bottom if necessary). These conditions are probably not the best (in particular the second condition could assume the top half of the first column contains at least half the ones in the first column) but they were simple to implement. One can choose to search for  $N \times M$  rectangles or for  $M \times N$  rectangles. The main runs were done with  $N \leq M$  although that is not always the best choice.

The entire program amounts to about 100 lines of VS Fortran code. As is often the case for this sort of program running times (on a 3090 IBM mainframe) increased rapidly with the problem size. For example showing  $G(7, 11) < 14$  required 640 seconds of cpu time but showing  $G(9, 12) < 16$  required 44000 seconds of cpu time. On average each node visited in the search tree required about 6 microseconds of cpu time.

A skeptical reader might ask why he should believe the program is correct. Checking that the rectangles found are in fact Golomb rectangles is fairly simple. An independent program checked this for the rectangles in table 1. It is possible (although tedious) to do these checks by hand. The validity of the assertion that the rectangles in table 1 are optimum (i.e. that Golomb rectangles with better parameters do not exist) is more problematic. However, the following factors give the author confidence that his search program has not missed any superior rectangles. The program successfully reproduced (with the exception noted above) the results Robinson obtained with a completely different program. The program is reasonably small (100 lines of code). Additionally, the basic algorithm and much of the code is the same as for the author's Golomb ruler program. The results obtained by the Golomb ruler program are in agreement with those obtained by other researchers using independent programs. Of course, additional checking is always possible. For example, it would be nice to do searches on  $N \times M$  rectangles and on  $M \times N$  rectangles as the search trees will be completely different (when  $N \neq M$ ). However, as noted above this was not done for the large cases.

Table 1

$G(2,18)=9$	110000010000000010 100101000000010001
$G(2,29)=11$	1100000010000000000010100010 10010000000010000100000000001
$G(2,35)=12$	10100000000000110000000010000000100 100100010000000000000000100001000001
$G(2,43)=13$	1100000000000010100000000000000000100010010

1000001000010000000000010000000010000000001

G(2,52)=14 110000000100000000000001000000100000000000001000101  
1001000000000100000000000000010000000000010000100000

G(2,59)=15 11000000001000000100000000000000000000001000001000000010001  
10010000000000000000000101000000000010000000000000100001000

G(3,18)=11 100100000000010100  
100000000100000001  
010000011000100000

G(3,21)=12 100001000001000000011  
100000000000000010100  
001000100100000000010

G(3,26)=13 10001000000010000000001101  
0100000010000000000000001  
10000010000000010000100000

G(3,31)=14 1000010001000000010000001000001  
1000000000000000000010010000000  
011000000000100000000000000101

G(3,37)=15 100001000000001000000000000000000101  
10010000000000000000000010001000000  
00100000001000000100000100000000000110

G(4,13)=11 1001000000011  
100000000000  
0100001010000  
1000100000100

G(4,16)=12 1001000000001010  
100000000000001  
0000000100010000  
1000011000000100

G(4,19)=13 1010000100000000100  
1000000000100000001  
0000000001000001000

010011000000000100

G(4,23)=14 11000001000100001000000  
 1000000000000000010010  
 00000010000000000001000  
 1010000000000100000001

G(4,27)=15 100000000100010100000001000  
 1001000000000000000000100  
 000000011000000000010000001  
 10000100000000000000000010

G(5,11)=11 11000000001  
 10000000000  
 00000010010  
 10100001000  
 00001000001

G(5,12)=12 110001000001  
 100000000000  
 000000010010  
 100000001000  
 001010000001

G(5,15)=13 100001000001001  
 100000000000010  
 000000000010001  
 001010000000000  
 100000011000000

G(5,18)=14 100000001000100001  
 100000010010000000  
 000010000000000100  
 100000000000000000  
 010100000000000011

G(5,20)=15 10000000010000000100  
 00001000100000000011  
 10100000000000000000  
 00000000000001001000  
 01000010000001000001

G(6,8)=11      11000100  
                  00000001  
                  00000001  
                  01010000  
                  10000010  
                  01001000

G(6,10)=12     1100000001  
                  1000000100  
                  0001000001  
                  0000100000  
                  0000000010  
                  1010010000

G(6,12)=13     100010000110  
                  000010000001  
                  100000000000  
                  000101000000  
                  001000000000  
                  100000001001

G(6,14)=14     10010000000101  
                  00000110000010  
                  00100000000000  
                  10000000000010  
                  10000000000000  
                  00001000100001

G(6,17)=15     11000000000000101  
                  10000001000000000  
                  00000000010000010  
                  00010000000001000  
                  10000000000000000  
                  01000100001001000

G(7,7)=11        1000100  
                  0100001  
                  1000001  
                  0000000  
                  0001000

```
0000001
0110100

G(7,9)=12  110000101
            100000000
            000100000
            100000000
            000000001
            010010000
            001000100

G(7,10)=13 1100000010
            0000010000
            0000100100
            1000000000
            0000000001
            1000000001
            0101000100

G(7,12)=14 001010000001
            000000001000
            100000000001
            110000000000
            000000000010
            000100100000
            010001000001

G(7,15)=15 100000001000011
            100000010001000
            000000000000000
            101000000000000
            000010000000001
            000001000000000
            010000000010010

G(8,8)=12  11001010
            10000000
            00000001
            10000000
            00010000
            00000100
```

00100000  
10000001

G(8,11)=14 10000001001  
10001000010  
00000000000  
00001000000  
00000000010  
11000000000  
00000001000  
00100000101

G(8,13)=15 1000010001001  
1000000000000  
0000000001000  
0100000000000  
1000000000010  
0000000000001  
0000011000000  
0010000010100

G(9,9)=13 110000001  
100000000  
000010000  
100000000  
000000100  
010000010  
000000000  
100000000  
000101001

G(9,10)=14 1000001000  
1010000001  
0000000000  
1001000000  
0000000100  
0000000000  
0000000001  
1000000010  
0100011000

G(9,12)=15    110000010001  
                  100000000000  
                  000000001010  
                  100000000000  
                  000010000000  
                  000001000000  
                  000000000000  
                  100000000100  
                  000100100001

G(10,10)=15    1000010010  
                  0100000000  
                  0000000001  
                  1000100000  
                  0010000000  
                  0000000000  
                  0000000001  
                  0000000001  
                  0101000000  
                  1000001100

## References

- [1] Golomb, S.W. and Taylor, M., *Two-dimensional synchronization patterns for minimal ambiguity*, IEEE Transactions Information Theory, It-28, pp. 600-604, 1982.
- [2] Robinson, J.P., *Golomb Rectangles*, IEEE Transactions on Information Theory, It-31, pp. 781-787, 1985.
- [3] Shearer, J.B., *Some New Optimum Golomb Rulers*, IEEE Transactions on Information Theory, It-36, pp. 183-184, 1990.