Eight Pieces Cannot Cover a Chess Board*

A. D. ROBISON[1], B. J. HAFNER[1] AND S. S. SKIENA[2]

[1] *Department of Computer Science, University of Illinois, Urbana, IL 61801, USA*

[2] *Department of Computer Science, State University of New York, Stony Brook, Stony Brook, NY 11794, USA*

The problem of maximising the number of squares on a chess board which can be attacked by a configuration of the eight main pieces was first posed in 1849. We report on a computer search which proves that at most 63 squares can be simultaneously attacked, and we give results for other variations of the problem. Our search technique, which pruned the space of 2.27×10^{12} positions to 1.03×10^{8}, is of independent interest.

Received July 1988

1. INTRODUCTION

Chess is a game that has fascinated mankind for thousands of years. In addition, it has inspired a number of combinatorial problems of independent interest. The combinatorial explosion was first recognised in the legend that the inventor of chess demanded as payment one grain of rice for the first square of the board, and twice the amount of the ith square for the $(i+1)$st square, for a total of $\Sigma_{i=1}^{64} 2^i = 2^{65} - 1$ grains. In beheading him, the wise king first established pruning as a technique for dealing with this type of problem. The n-queens[1,2] and re-entrant knight's tour[3] problems have histories dating back to the nineteenth century and today are famous exercises in computer science illustrating backtracking and Hamiltonian tours. Refs 4 and 5 pose the question of whether all 64 squares on the board can be simultaneously threatened by an arrangement of the eight main pieces on the chess board – the king, queen, two knights, two rooks, and two oppositely coloured bishops. Configurations which simultaneously threaten 63 squares have been known for a long time, but whether this is the best possible was an open problem. The problem was first proposed by Kling in 1849 and configurations exist where each square can be uniquely unthreatened.[5] Some configurations which cover 63 squares are given in Fig. 1.

This paper describes the results of an exhaustive computer search which proves that no configuration exists which covers all squares. Also, we provide some new results for related problems. We have recently learned that ref. 6 asserts without comment that 63 is the maximum number achievable. However, due to its efficiency the pruning technique we developed to reduce the search space is of independent interest.

2. SEARCH TECHNIQUE

We generalise the problem somewhat by waiving the usual restriction that no two pieces can be positioned on the same square. We reduce the problem somewhat by taking advantage of some symmetry. First, in any solution the board may be rotated 180° or reflected about a diagonal to bring the queen into the lower left quadrant. Second, the on-white bishop (the bishop restricted to white squares) may be brought into the lower triangle by reflection about the diagonal, while keeping the queen in the lower left quadrant. Therefore there are only 16 distinct positions for the queen and 16 spots for the on-white bishop, as shown in Fig. 2. There are 32 places for the on-black bishop and 64 places for the king. Finally, there are 2,080 distinct ways to position a pair of rooks or knights. Thus to perform an exhaustive search, we must test 2,268,279,603,200 distinct positions.

We call an arrangement of chess pieces on a chess board a *board*. A board b may have any number of pieces and more than one piece on a square. The union of a set of boards $\{b_1, b_2, \ldots, b_n\}$ is the superposition of the boards, and is denoted $\bigcup_i b_i$.

For a given board, we distinguish two kinds of attack on a square: strong and weak. The notion of strong attack corresponds to the usual notion of attack in chess. A square is *weakly attacked* if the square is strongly attacked by any subset of the board, that is, weak attack ignores blocking effects of intervening pieces. Fig. 3 shows that all 64 squares can be weakly attacked with eight pieces, though in general fewer squares will be strongly attacked. (In fact Fig. 3 is an extreme case: 64 squares are weakly attacked, but only 43 squares are strongly attacked.) We call a square *safe* if it is not

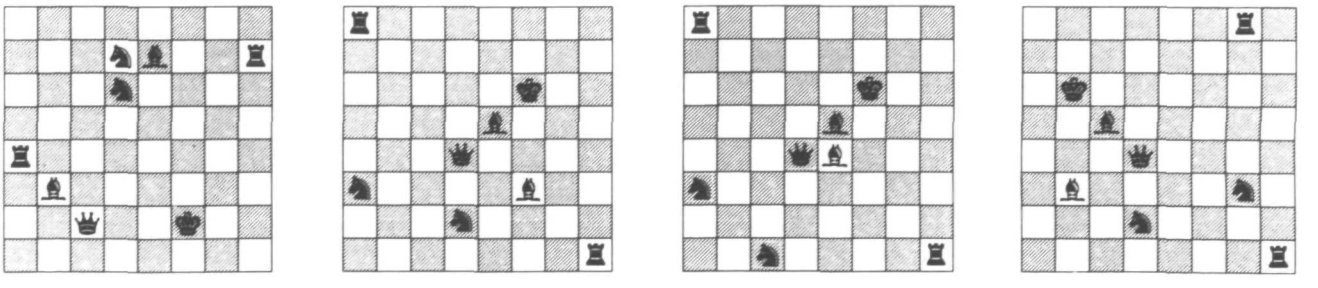

Figure 1. Some configurations which threaten 63 squares.

* The research of the first author was supported by a Shell Fellowship in Computer Science.

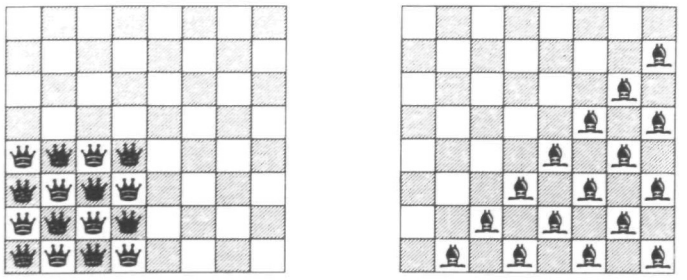

Figure 2. The 16 unique positions for the queen and on-white bishop.

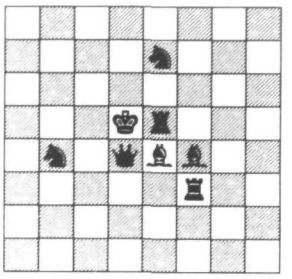

Figure 3. All squares can be weakly attacked by eight pieces.

attacked. The *attack set* $A(b)$ of board b is the set of squares attacked.

Our algorithm consists of two passes. The first pass lists all boards such that every square is weakly attacked. The second pass filters the list by considering blocking and reports any boards with n or fewer safe squares. As the first pass emits only 8,715 boards, high speed is not critical for the second pass. The advantage of separating weak and strong attack computations is that weak attack has three nice algebraic properties. First, weak attack is distributive over unions:

$$A(\bigcup_i b_i) = \bigcup_i A(b_i).$$

Second, weak attack is monotonic:

$$A(b_1) \subseteq A(b_1 \cup b_2).$$

Finally, the strong attack set is always a subset of the weak attack set.

We view our search space as eight-dimensional, with one axis for each piece. The coordinates along each axis are the possible boards containing only that piece. The king axis consists of the 64 boards with just a king. The queen and on-white bishop axes each consist of the 16 boards with just a queen or bishop as discussed above. The on-black bishop axis is 32 in length.

We save some searching by noting that the two rooks and two knights are identical. Consider the two rook axes of the search space as shown in Fig. 4. Swapping the rooks is equivalent to reflecting the search space about the diagonal shown. Therefore we need only report solutions within the lower triangular region. Using this observation to save time is a bit tricky, however, as our search algorithm does not search individual points, but rather rectangular regions. Regions completely outside the lower triangular region will be rejected. (These rejected regions are dashed in Fig. 3.) For those familiar with computer graphics, the problem is that of clipping a picture (the search region) against a triangular window. Regions which partially overlap the lower triangle, if not rejected for other reasons, will be subdivided (unfolded) and the subdivisions outside the lower triangle rejected.

Each point of our space is a possible board with coordinates $(b_1, b_2, \ldots, b_8)$. Therefore the attack set of a point is the union of the attack sets of its coordinates, that is

$$A(b_1, b_2, \ldots, b_8) = \bigcup_{i=1}^{8} A(b_i).$$

All together our search space contains approximately 2.27×10^{12} points. To search this space in a day would require us to examine approximately one board every 40 nanoseconds, which is unmanageable without substantial pruning.

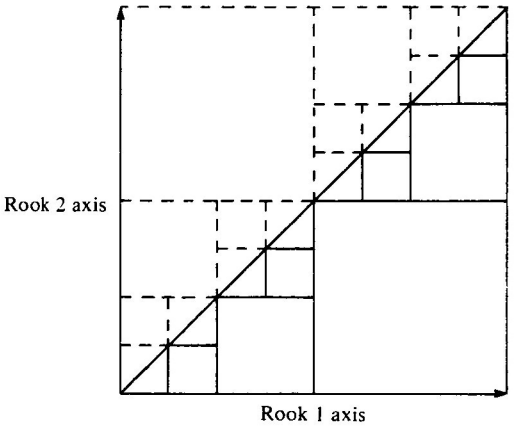

Figure 4. Clipping transpositions in rook plane.

To speed up the search, we employ a folding operation on the search space. To fold the search space, we split the space in half along an axis and then take the element-wise union of the two halves. We can view the folding operation as doubling up a piece, that is making each coordinate along the folded axis stand for two possible locations of the piece. The key observation is that if a square is (weakly) safe in the folded space, then it must be safe in both halves of the unfolded space. Most of the time we will be able to fold some sub-space many times and still be able to show some square is safe.

We begin by folding the search space down to a single point, that is we repeatedly fold until each axis is of unit length. This is equivalent to placing all eight pieces on each of the 64 squares. Now we unfold the space along some axis and check the two sub-space halves. Unfolding stops when one of three conditions hold:

(1) Some square is weakly safe, implying that the sub-space contains no solutions.

(2) The knights or rooks are out of order, meaning that the sub-space is entirely outside the clipping window.

(3) The sub-space is completely unfolded, meaning the position contains no squares which are weakly safe.

Positions satisfying the third condition are output by the first pass for further analysis.

When unfolding a subspace, we must choose which axis to unfold, and how the axis should be unfolded. Our first heuristic is to unfold the most-folded axis first, with some weight factors thrown in so that preference is given to certain axes. The weight factors were determined empirically by measuring the time required to search a fraction of the total space. A second heuristic is to prefer unfolding the axis that maximises the difference between the attack sets of the two halves of the axis. The idea is that if we split along an axis and the two sub-axes have the same attack sets, we haven't gained any possibilities

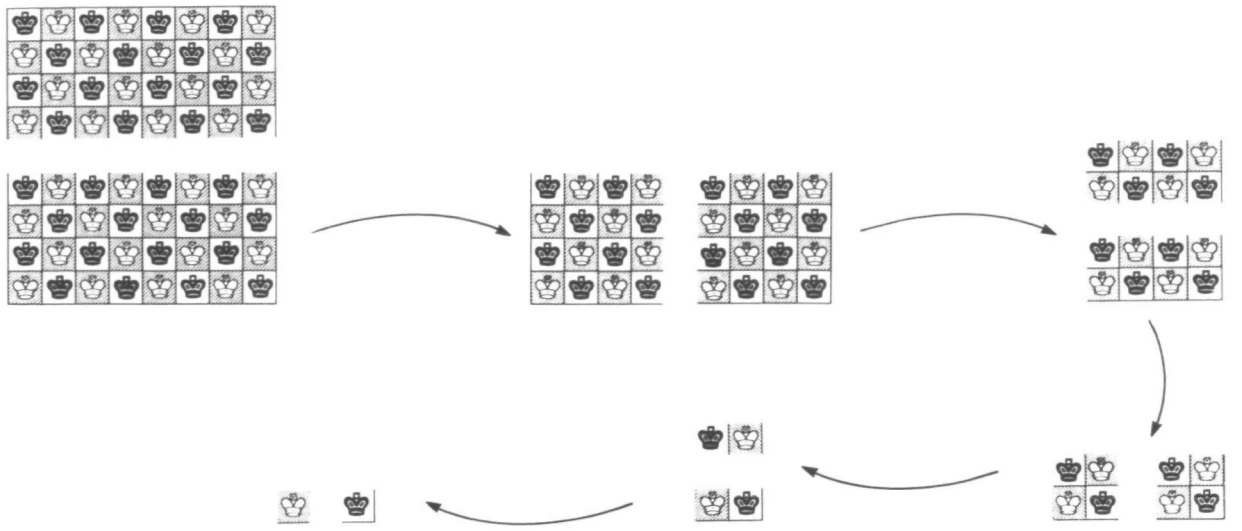

Figure 5. Unfolding on the king axis.

yet for pruning. A third heuristic is to try unfolding along each of the eight axes and pick the first unfolding we find which rejects half of the search space. This amounts to a one-level look ahead.

We also must choose how to unfold each axis. Our unfolding is geometrically based, with the intent to minimise the squares attacked by a set of pieces. The king axis folding is the simplest. The completely folded king axis corresponds to putting 64 kings on the board, one on each square. The first unfolding corresponds to putting the kings on each half of the board, subsequent unfoldings subdivide these halves as shown in Fig. 5. This grouping of kings tends to reduce the number of squares attacked by the group. The queen, bishops, and rooks are treated similarly, except for the aforementioned symmetry and colour constraints. Knights are partitioned in a slightly different manner. The first knight unfolding puts all knights on black squares in one half and all the knights on on-white squares in the other half. Since knights change colour when moving, each of these two arrangements of 32 simultaneous knights attacks only 32 squares.

3. IMPLEMENTATION

Since the program is computationally intensive, we make every attempt to provide efficient data structures. Weak attack sets are represented by 64-bit vectors, each bit corresponds to a square. The bits 0 and 1 represent safe and attacked squares respectively. The union of two (weak) attack sets is computed as the bitwise OR of their vectors. Some square is safe if the word is not all 1's.

For each of the eight pieces, we precompute a tree of axis foldings. Each internal node has two children, which are the two halves of the unfolded axis. For example, the root of the king tree represents 64 simultaneously placed kings, and its two subtrees represent the two 32-king sets when the king axis is unfolded. Eventually we reach leaf nodes, which represent boards with a single piece. Each node also precomputes the attack set of its piece set.

A sub-space is represented as eight pointers, one for each axis. Each pointer points to some node of its respective axis tree. The two halves of the unfolded sub-space are simply computed by replacing a pointer by each of its child pointers in turn. The attack set of the sub-space is computed as the union of the eight precomputed attack sets of its coordinates.

Table 1. The effectiveness of our pruning strategy

Foldings	Clipped	Safe square
41	0	1
39	0	3
36	0	12
32	0	96
30	0	576
28	432	0
27	2106	0
26	3235	0
25	10381	1372
24	18981	1447
23	58894	280
22	71128	457
21	84498	13785
20	124978	10012
19	279656	4652
18	495630	2282
17	1171970	913
16	1204445	797
15	1280363	788
14	1300040	1519
13	1363224	1528
12	1226053	687
11	1030734	377
10	814020	387
9	637183	498
8	461268	303
7	324552	184
6	214378	121
5	119433	109
4	64665	119
3	42533	30
2	23080	112
1	12854	1
0	15206	41

4. RESULTS

As reported above, our program did not locate a single position covering all 64 squares with the bishops on

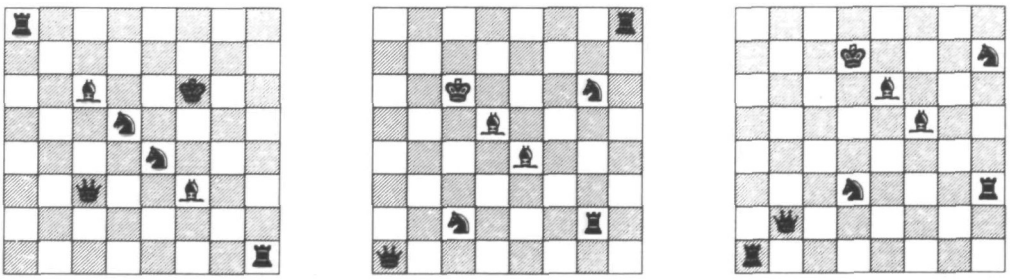

Figure 6. The three ways 8 pieces with similar bishops can cover the board.

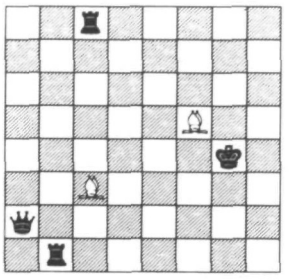

Figure 7. Superposition solution: knight/bishop pairs are shown as white bishops.

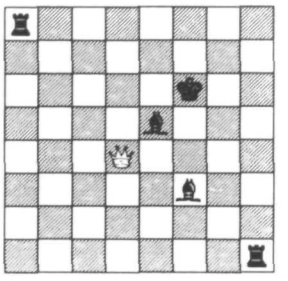

Figure 8. Seven pieces suffice with superposition, superqueen shown in white.

opposite coloured squares. Our program is efficient enough to complete the search on a Sun 3/360 in under 75 minutes. Table 1 provides statistics on the effectiveness of our pruning strategy. The table lists the frequency of search tree leaves, broken down by leaf type and sub-space size. The Foldings column is the $\log_2$ of the sub-space size, the two other columns list the number of such sub-spaces which were classified as containing a safe square or clipped. There were also 8,715 sub-spaces which were completely unfolded, for which neither of the other two classifications applied. To check our maths, we verified that the sum of the frequencies times the size of the sub-spaces is equal to the size of the unclipped search space (2^{43}).

All together there were approximately 1.03×10^8 search tree nodes (boards), of which 1.25×10^7 were leaves. This is a substantial reduction from the 2.27×10^{12} possible boards. The average safe leaf sub-space size was approximately $2^{17.9}$ boards; the median safe leaf sub-space size was 2^{13}. The latter is equivalent to placing at least 26 pieces (by folding 3 axes once and 5 axes twice to create 3 double pieces and 5 quadruple pieces) on the board simultaneously with possible superposition of pieces. It is remarkable that for the half of the leaves this many pieces left at least one square weakly safe! Furthermore, this upper 50 per cent of the leaves swallows 99.9 per cent of the volume of the unclipped search space.

Our program was applied to related problems, with interesting results. It has long been known that all 64 squares can be covered if the bishops are on similarly coloured squares. Fig. 6 gives the complete set of three such positions, independent of rotation and reflection. It is interesting to note that for all of these, the queen is on a different colour than the bishops.

Also, it is possible to cover the board with the eight pieces if we permit superposition, for example a knight and bishop occupy the same square (Fig. 7) or a queen and a knight combine to form a *superqueen*. Fig. 8 shows a very special position – the queen and *two* knights become a superqueen, so only seven pieces with superposition are necessary to cover the chess board.

5. CONCLUSIONS

An exercise such as this is useful to impress upon us what can be done in combinatorial computing with the proper search technique. The folding of the search space is a non-obvious algorithm, but can be very useful when the search space is sparse, since so much of the space can be compressed.

Ref. 6 gives a survey of many other variations on this problem, including identifying configurations which *minimise* the number of attacked squares instead of maximise them. This is a more difficult problem than maximising the attacks, since our folding technique does not apply. However, it should yield nicely to a branch-and-bound search.

REFERENCES

1. B. Abramson and M. M. Yung, Construction through decomposition: a divide-and-conquer algorithm for the N-queens problem. In *Proceedings Fall Joint Computer Conference*, edited H. Stone and S. Winkler, pp. 620–628. IEEE Computer Society Press, Washington (1986).
2. M. Gardner, The eight queens and other chessboard diversions. In *The Unexpected Hanging and Other Mathematical Diversions*, pp. 186–197. Simon and Schuster, New York (1969).
3. M. Gardner, Graph Theory. In *Martin Gardner's Sixth Book of Mathematical Games from Scientific American*, pp. 91–103. W. H. Freeman, San Francisco (1971).
4. C. S. Ogilvy, *Tomorrow's Math*. Oxford University Press, New York (1972).
5. M. Gardner, Eight Problems. In *The Unexpected Hanging and Other Mathematical Diversions*, pp. 76–89. Simon and Schuster, New York (1969).
6. M. Gardner, Chess Tasks. In *Wheels, Life, and Other Mathematical Amusements*, pp. 183–193. W. H. Freeman, New York (1983).