

Ordering method to accelerate the solution of mate-in-two chess problems by computer

W. Proskurowski

Royal Institute of Technology, Department for Computer Sciences, S-100 44 Stockholm 70, Sweden

The problem of solving a mate-in-two chess problem might be described as searching a move tree with the depth limit of four or five, depending on the algorithm used. Here we describe methods to increase the probability that the order of investigation of the moves will in some way approach the best order. The gain in the computing time due to these ordering methods was substantial (at least a factor of two).

(Received August 1972)

The problem of solving a mate-in-two chess problem might be described as searching a move tree with the depth limit of four or five, depending on the algorithm used (Bell, 1970; Manning, 1971). The total size of the tree is approximately b^L , where b is the branching factor (almost constant at each move) and L is the depth of branches. Nevertheless, as for any AND/OR tree, it is not necessary to traverse the whole tree; some branches can be pruned without hazard. Namely, it is enough to find one white (black) continuation which does (not) end forcingly in mate to refute the black (white) move. Thus, all the other white (black) responses need not be investigated. Such a pruning should not, however, be done at random. To achieve the maximum saving in computation one must investigate the moves in the best order. That means selecting at each time that move which (most probably) would refute the opponent's move (Nilsson, 1971). It is estimated that with an optimal order, the branching factor can be reduced nearly to the square root of its original value (Samuel, 1967).

Here we describe methods to increase the probability that the order of investigation of the moves will in some way approach the best order.

The selection rules are based on the information:

1. contained in the initial description
2. available as the search proceeds.

We shall examine them separately.

1. For any position a list of moves is generated according to both an ordering of the chessmen and to a prescribed order of moves for each piece (i.e. clockwise with the beginning in West-North-West square for a Knight). Initially the chessmen are listed (as an input) according to their position on a chessboard (i.e. at random) or in prescribed order corresponding to the decreasing power of pieces, with the white King listed last and the black King listed first. Note that the latter enables an early escape of the black King.

2. Our search strategy uses the information discovered at one move to improve the evaluations at other moves.

The order of the chessmen may be changed during the computation in accordance with the information received from the search. The black piece which refutes the current opening move, being the currently best piece, is then moved to the beginning of the list of blackmen. Other pieces are shifted correspondingly (Appendix 1).

Each piece (excluding pawns) can make about six moves on the average (up to 27 moves for the Queen on an empty chessboard). This indicates that the move, rather than the piece, which refutes the opponent's move, should be placed at the beginning of the list. But the order of moves for a piece is strictly prescribed. Consequently this best move is added (with the elimination of repetitions) to the beginning of the list of moves (Appendix 2). Then the program checks whether this move is present on the current list of moves; if so, this double

move is removed from the list (Appendix 3): This procedure is applied only to the black first move.

An example

Most modern two-movers consist of 19 pieces, white and black together (Petrović, 1968). This gives an average of about 80 legal (according to Bell's definition) moves in the initial position or in other words a branching factor of order 40. The following example taken from Petrović (1968) consists of $8 + 11 = 19$ pieces with $42 + 31 = 73$ legal moves.

The initial order of chessmen is as in Fig. 1, the only exception being that the white King is at the end of the list of whitemen. The 15 Queen opening moves are refuted by only two black moves (Kf5 and Kd3) which, one after the other, are placed by the program at the beginning of the current list of moves. The next opening move $1. Ra7$ is refuted by $1... Rd5$. And this move succeeds against the next 13 opening moves but fails against $1. Ktd4$ which is refuted by $1... Rg3$. This new best move succeeds against all the other opening moves except the key move $1. Ktb5$ which leads to a mate (as we want to check the uniqueness of the solution all white's opening moves must be tried). Here the branching factor for the first black move is reduced from about eight for the random search to about three. This considerably reduces the size of the search tree and correspondingly the time of a run.

Several test runs were made on an IBM 360/75 computer. The gain in the computing time due to the described ordering method was substantial (at least of factor of two).

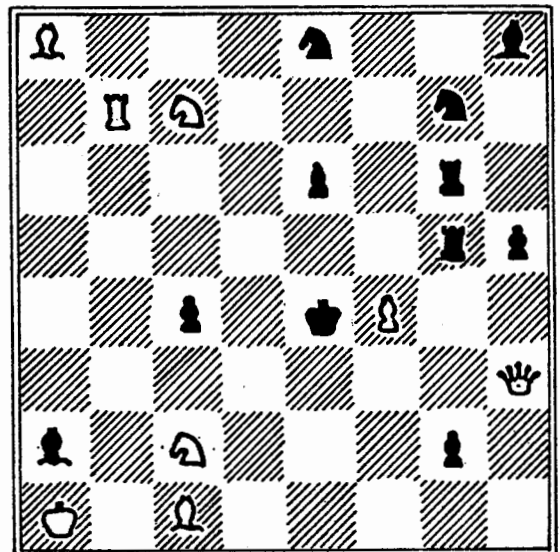


Fig. 1 White: Ka1, Qh3, Rb7, Ba8, Bc1, Ktc2, Ktc7, pf4 (8 pieces)
Black: Ke4, Rg5, Rg6, Ba2, Bh8, Kte8, Ktg7, pc4, e6, g2, h5 (11 pieces).

Algorithm

Our method was tested with the help of Algorithm 50 in *The Computer Journal* (Bell, 1970) and we describe here only the additions to it.

Appendix 1

Added just before the line 'goto w1continue;'
for $i := \text{locate}[2]$ step 1 until $\text{blackpiece}[0]$ do
 $\text{blackpiece}[i] := \text{blackpiece}[i + 1]$;
 $\text{blackpiece}[\text{blackpiece}[0]] := \text{backto}[2]$;

Appendix 2

(excludes the usage of Appendix 1)

Declarations at the beginning of program:

integer blhelp , blopt ;

integer array $\text{blfirst}[0:49]$;

In the procedure makemove, after the line starting with 'a6:' is added:

if $\text{depth} = 1$ then

begin

for $j := 0$ step 5 until $\text{blopt} - 1$ do

for $i := 0$ step 1 until 4 do

$\text{moves}[i + \text{blhelp} + j] := \text{blfirst}[i + \text{blopt} - 5 - j]$;

$\text{to} := \text{blhelp}$;

$\text{numberofmoves}[1] := \text{blhelp} + \text{blopt}$;

end else $\text{to} := \text{numberofmoves}[\text{depth}]$;

'to :=' is removed from the line next to the one starting with 'ep3:'.

Near the end of a program, just before the line

'goto w1continue;' is added:

for $i := 1$ step 5 until blopt do

if $q[2] = \text{blfirst}[i]$ and $\text{moves}[\text{bl} + 1] = \text{blfirst}[i + 2]$ then

begin

if $i + 4 = \text{blopt}$ then goto by;

for $j := i - 1$ step 1 until $\text{blopt} - 6$

do $\text{blfirst}[j] := \text{blfirst}[j + 5]$;

$\text{blopt} := \text{blopt} - 5$; goto out;

end;

out:

$\text{blfirst}[\text{blopt}] := \text{locate}[2]$;

$\text{blfirst}[\text{blopt} + 1] := q[2]$;

$\text{blfirst}[\text{blopt} + 2] := \text{backto}[2]$;

$\text{blfirst}[\text{blopt} + 3] := \text{moves}[\text{bl} + 1]$;

$\text{blfirst}[\text{blopt} + 4] := -\text{backto}[2]$;

$\text{blopt} := \text{blopt} + 5$;

by:

In the lines:

for $w1 := 3$ step 1 until $\text{numberofmoves}[1]$ do

for $\text{bl} := \text{numberofmoves}[1] + 2$ step 1 until
 $\text{numberofmoves}[2]$ do
' $\text{numberofmoves}[1]$ ' is replaced by 'blhelp'.

Appendix 3

In procedure Makemove is added, just after the line 'start:'

if $\text{moves}[\text{pointer} + 1] = 0$ then

begin from := $\text{moves}[\text{pointer}]$; $\text{pointer} := \text{pointer} + 1$;

goto next end;

if $\text{depth} = 2$ and $\text{pointer} < \text{numberofmoves}[1]$

and $\text{moves}[\text{pointer} + 2] < 0$ then

begin

for $i := \text{numberofmoves}[2] - 1$ step -1

until $\text{numberofmoves}[1] + 4$ do

begin

if $\text{moves}[i] \geq 0$ then goto nexti;

if $\text{moves}[i] = \text{moves}[\text{pointer} + 2]$ then

begin

for $k := i - 1, k - 1$ while $\text{moves}[k] > 0$ do

if $\text{moves}[k] = \text{moves}[\text{pointer} + 1]$ then

begin $\text{moves}[k] := 0$; goto fin end;

goto om;

end;

nexti: end;

om: $\text{pointer} := \text{pointer} + 5$;

$\text{locate}[2] := \text{moves}[\text{pointer} - 2]$;

$q[2] := \text{moves}[\text{pointer} - 1]$;

$\text{backto}[2] := \text{moves}[\text{pointer}]$;

goto start;

fin: end check of list;

'next:' is added just before the line

'to := $\text{moves}[\text{pointer} + 1]$;'

Appendix 4

A minor improvement based on the fact that the Queen cannot be a masking piece in a battery.

Added after the line ' $n := \text{numberofmoves}[4]$:'

if $q[3] = 5$ then

begin

integer array $\text{acc}[0:1]$;

$\text{acc}[0] := 1$; $\text{acc}[1] := \text{whitepiece}[\text{locate}[3]]$;

$\text{listmoves}(\text{acc}, \text{whitemen}, \text{blackmen}, n, \text{notstalemate})$;

end else

Appendix 5

The line after the declaration of procedure Reversemove:

$c[1] := 0$;

is replaced by:

for $i := 1$ step 1 until 5 do $c[i] := 0$;

References

- BELL, A. G. (1970). Algorithm 50: How to program a computer to play legal chess, *The Computer Journal*, Vol. 13, pp. 208-219.
MANNING, J. R. (1971). Algorithm 68: White to move and mate in n moves, *The Computer Journal*, Vol. 14, pp. 209-213.
NILSSON, N. J. (1971). *Problem-Solving Methods in Artificial Intelligence*, New York: McGraw-Hill.
PETROVIĆ, N. ed. (1968). *III FIDE Album 1962-1964*, Zagreb.
SAMUEL, A. L. (1967). Some Studies in Machine Learning, Part II, *IBM Journal*, Vol. 11, pp. 601-627.

Erratum

There is an error in the paper 'An Information Measure for Hierarchic Classification' by D. M. Boulton and C. S. Wallace (this *Journal*, Vol. 16, No. 3, pp. 254-261). The error is on page 261, in the second paragraph after Figure 1. The second sentence 'The two classes (4, 6) and (1, 2, 3, 5, 7)—' should read 'The two classes (2, 7) and (1, 3, 4, 5, 6)—'.

Journal of the Computer Society of India

The *Journal of the Computer Society of India* is published twice annually and contains articles on the research, development and applications of computer science and technology. Contributions are welcomed and authors will be notified of the referees' suggestions. Subscriptions in the U.K. are £1.50 annually, including postage. A communication to the Editor, CSI Journal, Computer Group, Tata Institute of Fundamental Research, Colaba, Bombay 400 005, India.