from ICCA Journal, Vol. 9, No. 3 (September 1986)

RETROGRADE ANALYSIS OF CERTAIN ENDGAMES

Ken Thompson

AT&T Bell Laboratories
Murray Hill, New Jersey, U.S.A.

Introduction

Computers have been able to exhaustively solve certain simple chess endgames. The general method is to work backwards from mates (or known winning positions). Wins-in-one are marked by generating predecessor positions to mates. Wins-in-two are then marked by generating predecessor positions to mates and wins in one, etc. When no more winning positions are generated, the unmarked positions are either illegal, draws or losses.

The major complexity of this problem is the number of pieces on the board. A minor complexity is the existence of a pawn, which complicates the symmetry. The first work in this field[1] consisted of analysis of one and two-piece (not counting Kings) endgames. Since that time there has been much material published on endgames with less than three pieces. This paper describes work on three-piece endgames. There has been some prior work on three-piece endgames.[2, 3] There is also a good recent bibliography to the field.[4]

The Gödel Function

The basis of a retrograde analyzer is a subprogram that converts a chess position into a unique Gödel number (G) that is used to index a database of positions. The inverse transformation (from G to position) is also important. Different transformations are used for positions with and without pawns. The existence of a pawn destroys one of the symmetries enjoyed with pure pieces. Let me first describe the pure-piece transformations.

The canonical position has two Kings and three pure pieces. The White King is confined to the "octant" defined by the squares a8-d8-d5-a8. This is not a restriction since any position can be rotated and reflected to place the White King in this area. The Black King encoding is based on the position of the White King. In particular, the Black King cannot be next to the White King and if the White King is on the a8-d5 diagonal, the Black King can be rotated to be above that diagonal. The King encodings are implemented with a table lookup. This lookup also selects what rotation/reflection to apply to the other three pieces. There are 462 legal positions of two Kings where the described symmetries are removed. The other three pieces are encoded by their position on the board after the rotation/reflections indicated by the King positions. If the piece is missing (captured) then it is assigned the position of the White King, which would otherwise be illegal. Thus the encoded G has the following parts:

0-461 Position of two Kings
0-63 Position of Piece 1.
0-63 Position of Piece 2.
0-63 Position of Piece 3.

giving G a range of $462 \times 64 \times 64 \times 64$ or 121,110,528.

There are potentially other symmetries that could be exploited depending on what pieces are on the board. For example, if there are two White Rooks, they could exchange places without altering the position. More notably, if there are two White Bishops, and it is assumed that they are of opposite color, each could be assigned a 32 square subset of the board rather than 64 squares. None of these potential symmetries were exploited.

The Method

The retrograde analysis is accomplished by 4 programs: P_1 thru P_4. These programs operate on files of sets of chess positions. Each file is a bit map of 121,110,528 bits. If a bit is on in a file then the corresponding G encoded chess position is considered in the set of chess positions represented by the file. There are 5 files that are successively manipulated by the programs. File W is a list of all currently known White-to-move and win positions. B is all currently known Black-to-move and lose positions. W_i is the latest newly found White-to-move and win in i moves. This file is added (logically *or*ed) to W to bring W up to date. B_i is the latest Black-to-move and lose in i moves. The remaining file, J_i is a temporary file that is a superset of B_i as described below.

Program P_1 is for initialization. It is run exactly once at the beginning and creates the file B_0, those positions where Black-to-move and Black is mated. P_1 loops through all G positions, converts each to chess-board representation and examines each for a legal mate. If it is mate, then the corresponding bit is set in file B_0. B is initialized to B_0. W is initialized to all zeros and i is set to zero.

Program P_2 examines each position in file B_i and for each position, generates all possible legal predecessor positions. These positions have White-to-move and with at least one White move can obtain a B_i position. They are therefore the new W_{i+1} positions if they have not been found before — that is if they are not in W. After P_2 finishes reading B_i, consulting W and creating W_{i+1}, W_{i+1} is added to W.

Predecessor positions are formed by an un-move generator. This is the same as a move generator but a) it is illegal to start in check, but legal to un-move into check and b) it is illegal to capture, but legal to un-capture by leaving an enemy piece behind. In the games that we are discussing, un-castle and un-*enpassant* are not encountered. Un-promotion is described later under pawn endgames.

Program P_3 is exactly the same as P_2 except with Black-to-move. P_3 reads W_{i+1}, generates Black predecessor positions, consults B, and creates J_{i+1}. The positions in J_{i+1} are Black-to-move and lose if Black wanted to mate himself. Of these positions, only those that can be forced to a winning White position are losses.

This brings us to P_4 that reads J_{i+1}, generates Black successor positions and examines each in W. If all such successors are in W, then that position is added to B_{i+1}. When P_4 finishes, the new B_{i+1} positions are added to B. The process is iterated by incrementing i and repeating programs P_2, P_3, and P_4 in turn until no more positions are generated.

When the process finishes, W contains all positions in which White can force a win, with White-to-move. These positions are partitioned into the files W_i; White-to-win in i, with best play. The easiest way to save the results in a database is to make a file with 121,110,528 bytes. Byte j contains i if bit j is set in file W_i. In other words, byte j in the result file is the number of moves for White to win, and zero if the position is not a win for White. This result file is incrementally updated after the generation of each W_i file. The total space required is then the result file, W, B, and only two active files from B_i, W_i and J_i. This totals $12 \times 121{,}110{,}528$ bits of secondary storage, or about 175 megabytes.

Results

These programs were implemented on a Sequent Balance 8000 computer. This computer consists of 12 National 32032 microprocessors on a 16 megabyte shared memory. The programs were designed to divide the work by assigning each processor every *N*th position. The disk traffic was drastically reduced by allocating a megabyte of real memory per processor as a cache on the files that were randomly accessed. The programs were run in the background with an average of four processors working simultaneously. A typical pure-piece endgame would be solved in two to three weeks of real time.

Some of the results tabulated below are labeled "Max to Mate." These games were solved in one pass as described above. Max refers to the number of moves to mate with best play.

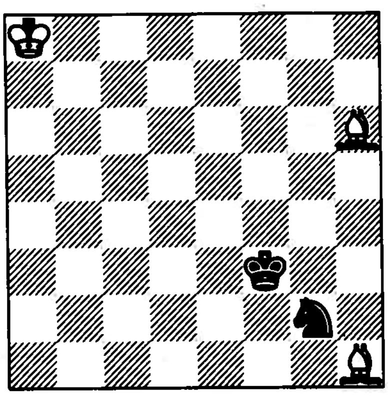
The other positions were solved in two passes. These are labeled "Max to Win." In these endgames, the first pass solved all subgames with fewer than the full complement of pieces to mate. The second pass then used these positions as B_0 along with all mates to obtain results with the complete complement of pieces. Thus the two-pass, "Max to Win," method solves positions with an objective function of either mate or capture into a won sub-game. This is precisely the conditions for the 50-move rule.

The "Percent Wins" column of Table 1 represents the number of positions that are wins divided by the total number of legal positions with White to move. These numbers will vary depending on the Gödel function since some positions are counted more than once. Also note that it is White to move in what is essentially a random position. This is a large advantage and the win percentage favors White. It is hard to characterize, but a win percentage of about 40 is indicative of a drawn endgame and a percentage of about 90 is a won endgame.

White	Black	Max to Mate	Max to Win	Percent Wins
♗♗	♘		66	91.8
♕	♘♘		63	89.7
♕	♘♗		42	93.1
♕	♗♗		71	92.1
♖♘	♖		33	35.9
♖♗	♖		59	40.1
♖♖	♖	31		94.3
♖♕	♖	35		95.9
♕♘	♕	41		48.4
♕♗	♕	33		53.4
♕♖	♕	67		92.1
♕♕	♕	30		94.0

Table 1

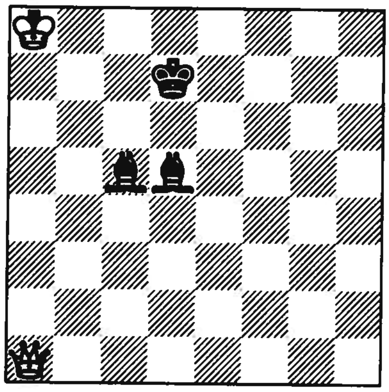
The first group of endgames in Table 1 were solved for chess interest. All four of the endgames in this group are considered to be drawn. This work shows that they are really wins. The next two groups were done as terminal promotion positions for pawn endgames. Some extreme cases are illustrated by the following "best play" examples.



Initial Position

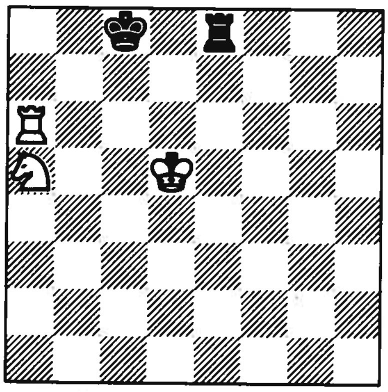
1 ♗f8 ♔g3 2 ♗d6† ♔f3 3 ♔a7 ♔f2 4 ♗h2 ♔f1 5 ♔b7 ♔f2 6 ♔c6 ♔f1 7 ♔d6 ♔f2 8 ♔e6 ♔f3 9 ♔f5 ♔f2 10 ♔g4 ♘e3† 11 ♔h3 ♘c4 12 ♗g2 ♘a3 13 ♗f4 ♘c4 14 ♗e4 ♔e2 15 ♔h4 ♘d2 16 ♗g6 ♘f1 17 ♔g5 ♔f3 18 ♗d6 ♔e3 19 ♗c5† ♔f3 20 ♔f5 ♘e3† 21 ♔e5 ♘g2 22 ♗h5† ♔g3 23 ♗b6 ♘f4 24 ♗d1 ♘g2 25 ♔d4 ♘e1 26 ♔e3

♘g2† 27 ♔d2 ♔f4 28 ♔e2 ♔f5 29 ♗c2† ♔e6 30 ♗b3† ♔d6 31 ♗f2 ♘f4† 32 ♔e3 ♘e6 33 ♗g3† ♔e7 34 ♗e5 ♘c5 35 ♗d5 ♘e6 36 ♔e4 ♘c5† 37 ♔f5 ♘d7 38 ♗f4 ♘b6 39 ♗f3 ♘c4 40 ♔e4 ♘d6† 41 ♔d5 ♘f7 42 ♗d1 ♔f6 43 ♗c2 ♘g5 44 ♗e5† ♔e7 45 ♗g3 ♘e6 46 ♔e5 ♘d8 47 ♗e1 ♘f7† 48 ♔d5 ♘h8 49 ♗h4† ♔f7 50 ♔d6 ♘g6 51 ♗b3† ♔f8 52 ♗f6 ♔e8 53 ♗c3 ♘f4 54 ♗d4 ♘g6 55 ♗d1 ♔f8 56 ♗c3 ♘e7 57 ♗h5 ♘f5† 58 ♔e5 ♘h6 59 ♔e6 ♘g8 60 ♗d4 ♘h6 61 ♔f6 ♘g8† 62 ♔g6 ♘e7† 63 ♔h7 ♘d5 64 ♗c5† ♘e7 65 ♔h6 ♔g8 66♗×e7



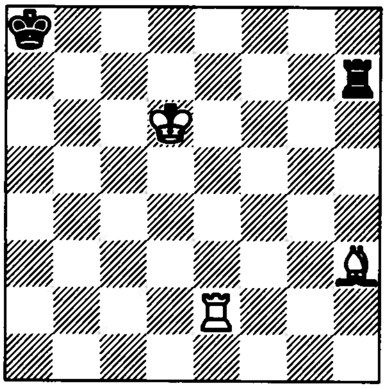
Initial Position

1 ♔b8 ♗d6† 2 ♔a7 ♗c5† 3 ♔a6 ♗c4† 4 ♔a5 ♔d6 5 ♕c1 ♔d5 6 ♕c3 ♗d4 7 ♕f3† ♔e5 8 ♕g3† ♔e4 9 ♕h4† ♔e3 10 ♕e7† ♔d3 11 ♔b4 ♗d5 12 ♕e1 ♗f3 13 ♕h4 ♔e3 14 ♔c4 ♗e2† 15 ♔d5 ♗f3† 16 ♔e6 ♗e4 17 ♕g3† ♔d2 18 ♕f4† ♔d3 19 ♔d6 ♗c3 20 ♕g3† ♔d2 21 ♕g1 ♗d3 22 ♔c5 ♔c2 23 ♕e3 ♗b2 24 ♔b4 ♗c3† 25 ♔a3 ♗b2† 26 ♔a4 ♗c3 27 ♕c5 ♗e4 28 ♕c4 ♗f3 29 ♔b5 ♗d1 30 ♕a2† ♔d3 31 ♕g2 ♗e2 32 ♔c5 ♗d1 33 ♕h3† ♔d2 34 ♔d5 ♗b3† 35 ♔e4 ♗c2† 36 ♔f4 ♗d4 37 ♕a3 ♗d3 38 ♕b4† ♗c3 39 ♕a4 ♗b2 40 ♔f3 ♗c3 41 ♕a7 ♗b2 42 ♕c5 ♗e2† 43 ♔f2 ♗d3 44 ♕g5† ♔c2 45 ♕d5 ♔d2 46 ♕a2 ♔c3 47 ♔e1 ♗c4 48 ♕a5† ♔b3 49 ♕b6† ♔c3 50 ♕b7 ♔c2 51 ♕h7† ♔b3 52 ♕b1 ♔c3 53 ♕e4 ♗a3 54 ♕e3† ♔b2 55 ♔d1 ♗b3† 56 ♔d2 ♗b4† 57 ♔d3 ♗a2 58 ♕e2† ♔b3 59 ♕c2† ♔a3 60 ♔d4 ♗b3 61 ♕c1† ♔a2 62 ♔d3 ♗a3 63 ♕c7 ♗b2 64 ♕a7† ♔b1 65 ♔d2 ♗d5 66 ♕b6 ♗e4 67 ♕b5 ♗g2 68 ♕e2 ♗b7 69 ♕f1† ♔a2 70 ♕f7† ♔a3 71 ♕×b7



Initial Position

1 ♖a8† ♔d7 2 ♖a7† ♔c8 3 ♔d6 ♖d8† 4 ♔c6 ♔b8 5 ♖b7† ♔a8 6 ♖h7 ♖c8† 7 ♔b6 ♖b8† 8 ♔c5 ♖g8 9 ♖h4 ♖b8 10 ♘c6 ♖b2 11 ♖h7 ♖c2† 12 ♔d6 ♖d2† 13 ♔c7 ♖h2 14 ♖d7 ♖d2 15 ♘d4 ♖b2 16 ♔c6 ♖b7 17 ♖d5 ♖b4 18 ♘b5 ♖c4† 19 ♔b6 ♔b8 20 ♖e5 ♖c1 21 ♖e8† ♖c8 22 ♖e1 ♖c2 23 ♘d4 ♖b2† 24 ♔c6 ♔a8 25 ♖f1 ♖b4 26 ♘b5 ♖c4† 27 ♔b6 ♔b8 28 ♘d6 ♖b4† 29 ♔c6 ♔a8 30 ♖f8† ♖b8 31 ♘c8 ♖b3 32 ♘b6† ♔a7 33 ♖a8#



Initial Position

1 ♗f5 ♖h4 2 ♗d3 ♖f4 3 ♗e4† ♔a7 4 ♗c6 ♖g4 5 ♔c7 ♖g7† 6 ♗d7 ♖g6 7 ♗e6 ♖g7† 8 ♔c6 ♖g1 9 ♖a2† ♔b8 10 ♖b2† ♔a8 11 ♔b6 ♖c1 12 ♗f5 ♖c3 13 ♖b1 ♔b8 14 ♖b4 ♖a3 15 ♗d7 ♖a2 16 ♖h4 ♖b2† 17 ♗b5 ♖c2 18 ♗c4 ♖b2† 19 ♔c6 ♖f2 20 ♖h8† ♔a7 21 ♖h7† ♔b8 22 ♖b7† ♔a8 23 ♖b4 ♖g2 24 ♗d3 ♖g3 25 ♖d4 ♖f3 26 ♗c4 ♖h3 27 ♖d8† ♔a7 28 ♗d5 ♖h2 29 ♖d7† ♔b8 30 ♖b7† ♔a8 31 ♖b1 ♖c2† 32 ♔b6† ♔b8 33 ♗e6 ♖d2 34 ♔c6† ♔a7 35 ♖a1† ♔b8 36 ♗d5 ♖h2 37 ♖b1† ♔a7 38 ♗e4 ♖h6† 39 ♔c5 ♖b6 40 ♖h1 ♖a6 41 ♖h8 ♖a5† 42 ♔c6 ♖g5 43 ♖h7† ♔a6 44 ♗d5 ♔a5 45 ♔c5 ♖g6 46 ♖h2 ♖g4 47 ♖b2 ♖h4 48 ♖b7 ♖h6 49 ♗f7 ♖f6 50 ♗c4 ♖f5† 51 ♗d5 ♖f6 52 ♖b5† ♔a6 53 ♖b2 ♔a7 54 ♖b7† ♔a6 55 ♖e7 ♔a5 56 ♗e6 ♔a6 57 ♗c8† ♔a5 58 ♖a7† ♖a6 59 ♖×a6#

And with a Pawn

When one of the pieces is a pawn, everything gets harder. The Gödel function loses symmetries; the pawns can promote into sub-games that must be solved independently; each pawn position must be treated as a sub-game and solved independently; and there are numerous smaller annoyances. The overall structure remains. First all promoted sub-games are solved. These are all combined to create the file W_{8*}, White-to-move, pawn on the 8th, and win in any number of moves. Second, a new program P_5 makes pawn un-moves to create the file B_{70}. These positions are combined with mates on the 7th and also wins without the pawn. Programs P_2, P_3 and P_4 are then iterated to create files W_{7*}, B_{7*} and J_{7*}. Again, this is done in two passes to first create wins without the full complement and second to generate all wins. And then P_5 is run to back up W_{7*} into B_{60}. All goes well until rank 2 when, because of the initial pawn move, files W_{3*} and W_{4*} must be combined before creating B_{20}.

The basic size of the G function is 83,886,080 to accommodate the pawn on one of the four files and one rank at a time. The total storage required is about 120 megabytes per rank. After the six result files of 83 megabytes each are complete, they are inverted into four rank-oriented files of 117 megabytes each. It takes about six weeks real time to create a complete pawn data base, not counting the time to create the promotion sub-games.

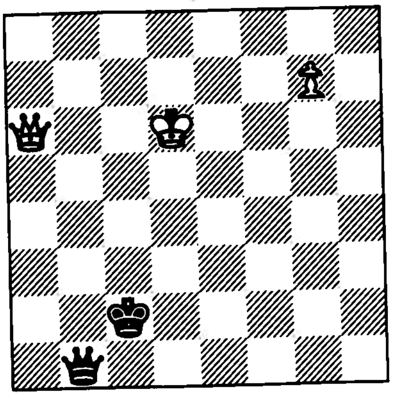
♕♙ versus ♕

The first pawn endgame attempted was Queen and Pawn vs. Queen. Table 2 below summarizes the results "max to win" and "percent wins" for each of the 24 initial pawn positions. Note that "max to win" means moves until capture, mate or pawn move; precisely the 50-move definition. Unfortunately, a clerical error surfaced. The under-promotions were accidentally discarded and so the results are for Queen promotion only. The chess literature contains some Rook under-promotions but every one that was examined could be won in a few more moves with Queen promotion. So the max figures for the 7th rank may have small errors, but the other figures in table 2 are probably correct.

7	70 84.0	55 84.7	43 85.4	42 85.6
6	71 71.0	61 76.5	46 79.3	58 77.6
5	33 57.2	38 63.8	43 72.7	45 70.1
4	29 51.8	30 55.6	48 67.2	64 65.6
3	20 48.5	51 52.7	53 61.5	54 58.3
2	17 48.6	31 52.7	47 62.7	41 59.0
	a	b	c	d

Table 2

There is one QPvQ example in Komissarchik and Futer. That analysis is presented below annotated with the current data base. The numbers in parentheses are the discrepancies in number of moves to mate, capture or promotion.



Initial Position

1 ... ♕b4† 2 ♔e6 ♕g4† 3 ♔f6 ♕f4† 4 ♔g6 ♕e4† 5 ♔g5 ♕e3† 6 ♔h5 ♕f3† 7 ♔h6 ♕h1† 8 ♔g5 ♕d5† 9 ♔f6 ♕d4† 10 ♔f7 ♕d7† 11 ♔g6 ♕g4† 12 ♔h7 ♕h3† 13 ♔g8 ♕f5 14 ♕a2† ♔c1 15 ♕h2 ♕d5† 16 ♔h8 ♕d4 17 ♕c7† ♔b1 18 ♔h7 ♕e4† 19 ♔h6 ♕e3† 20 ♔g6 ♕e6† 21 ♔g5 ♕d5† 22 ♔f6 ♕f3† 23 ♔e7 ♕e4† 24 ♔d8 ♕a8† 25 ♔d7 ♕d5† 26 ♔c8 ♕e6† 27 ♔b8 ♕e8† 28 ♔a7 ♕a4† 29 ♔b6 ♕b3† 30 ♔a6 ♕a2† 31 ♕a5 ♕g8 32 ♕b4† ♔a2 33 ♕d4 ♕e6†

33 ... ♕c8† (1)

34 ♔b5 ♕e8† 35 ♔b4 ♕b8†

35 ... ♕e1† (5)

36 ♔c3 ♕g3† 37 ♔d2

37 ♔c2 (10) 37 ♔c4 (8) 37 ♔b4 (3)

37 ... ♕g2† 38 ♔e1 ♕h1† 39 ♔f2 ♕h2† 40 ♔f3 ♕h3† 41 ♔f4 ♕h2† 42 ♔g5 ♕g3† 43 ♔f6 ♕f3† 44 ♔e6 ♕c6† 45 ♔e5 ♕e8† 46 ♔f4 ♕f7† 47 ♔g3 ♕g6† 48 ♔h3 ♕h7† 49 ♔g2 ♕g6† 50 ♔f1 ♕b1† 51 ♔e2 ♕b5† 52 ♔d2 ♕b3 53 ♕a7†

53 ♕d3 (1)

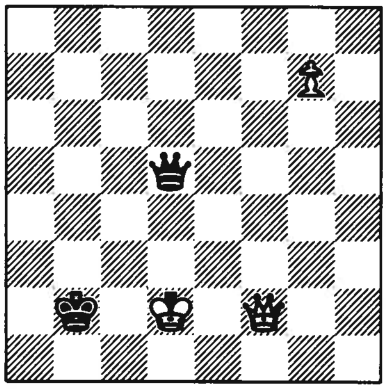
53 ... ♔b2 54 ♕f2 ♕g8

54 ... ♕d5† (1)

55 ♕b6† ♔a3 56 ♕b7 ♔a4 57 ♔c3 ♔a5 58 ♕b4† ♔a6 59 ♕c4†

The one-move discrepancies near the end can be explained by under-promotions. For example:

a



Position after 54 ... ♕d5†

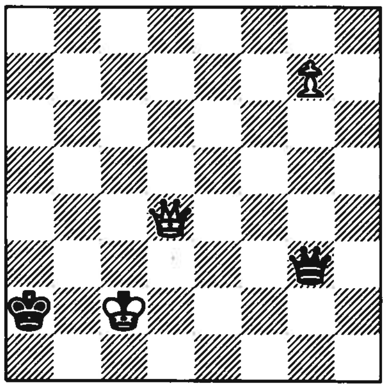
The data base play is:

55 ♔e3† ♔c3 56 ♕f3 ♕g8 57 ♔f2† ♔d4 58 ♕f8 ♕a2† 59 ♔g3 ♕b3† 60 ♔h4 etc.

But this can be improved by:

55 ♔e3† ♔c3 56 ♕f6† ♔c4 57 ♕f4† ♔c3 58 ♕c7† ♔b2 59 ♕b8† ♔a1 60 g8♖!

The discrepancies at moves 33 and 35 are simply to avoid the position at move 37. Thus the only real dispute is the position at move 37:



Position after 37 ♔c2

Komissarchik and Futer imply that this position is a win in at least 23 moves. The current work claims that it is a win in 13. The following analysis is given in support of the current work.

37 ... ♕g2†? 38 ♕d2 ♕c6† 39 ♔d3†

and promote next.

37 ... ♕c7†? 38 ♕c3 ♕h2† 39 ♕d2 ♕h7† 40 ♔c3†

and mate.

37 ... ♕g6†! 38 ♔d1 ♕e6? 39 ♕a7† ♔b2 40 ♕b8† ♔a1 41 ♕a8† ♔b1 42 g8♕

or more simply

41 g8♖!

38 ... ♕h5†? 39 ♔c1 ♕h1† 40 ♕d1

38 ... ♕f7! 39 ♔d2 ♔b3 40 ♔e1 ♕e6† 41 ♔f2 ♕f7† 42 ♔g3 ♕c7† 43 ♔g2 ♕b7† 44 ♔g1 ♕f7 45 ♕g4 ♕g8 46 ♕f3†

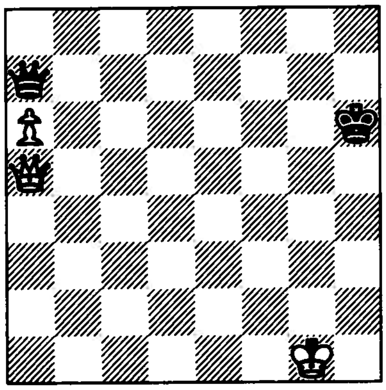
39 ... ♔a3 40 ♕g4 ♕g8? 41 ♕f3† ♔a4 42 ♕b7

and Black will have to leave the pawn to prevent mate.

40 ... ♕f2† 41 ♔d3 ♕f1† 42 ♔e3 ♕e1† 43 ♔f3 ♕d1† 44 ♔g3 ♕d6† 45 ♔g2 ♕d2† 46 ♔g1 ♕c1† 47 ♔h2 ♕c2† 48 ♕g2 ♕h7† 49 ♕h3†

The author would welcome any input which helps resolve this discrepancy.

The following is the maximal QPvQ play:



Initial Position

1 ♔g2 ♕g7† 2 ♔h1 ♕f6 3 ♕d2† ♔g7 4 ♕d7† ♔g6 5 ♕d3† ♔g7 6 ♕g3† ♔f8 7 ♕b8† ♔f7 8 ♕c7† ♔g6 9 ♕c4 ♕f3† 10 ♔h2 ♕e3 11 ♕f1 ♕e5† 12 ♔h1 ♕h8† 13 ♔g2 ♕a8† 14 ♔g1 ♕a7† 15 ♔h1 ♕d7 16 ♕f2 ♔h5 17 ♕e2† ♔g6 18 ♕e4† ♔h5 19 ♕c4 ♕a7 20 ♕e2† ♔g5 21 ♔g2 ♕d4 22 ♕f2 ♕e4† 23 ♕f3 ♕d4 24 ♕e2 ♔g6 25 ♕b5 ♕e3 26 ♔f1 ♕e4 27 ♔f2 ♕d4† 28 ♔f3 ♔h6 29 ♕c6† ♔g7 30 ♕c7† ♔g6 31 ♕g3† ♔h5 32 ♕h3† ♔g5 33 ♕e6 ♕d1† 34 ♕e2 ♕d5† 35 ♔f2 ♕f5† 36 ♔g1 ♕b1† 37 ♔g2 ♕g6 38 ♕c4 ♔h5† 39 ♔f2 ♕f6† 40 ♔e3 ♕e5† 41 ♔d3 ♔g5 42 ♔c2 ♕e3 43 ♔b2 ♔g6 44 ♕b5 ♔h6 45 ♔c2 ♕e6 46 ♔c3 ♕d6 47 ♔c4 ♔g7 48 ♕g5† ♔h8 49 ♕h5† ♔g7 50 ♕g4† ♔f7 51 ♕f5† ♔g7 52 ♕c8 ♕f4† 53 ♔b5 ♕f6 54 ♕c6 ♕b2† 55 ♔c5 ♕f2† 56 ♔d6 ♔g6 57 ♔d5† ♔h7 58 ♕c7† ♔h8 59 ♕c3† ♔h7 60 ♔c6 ♕f5 61 ♔b6 ♕e6† 62 ♔b7 ♕e4† 63 ♔b8 ♕b1† 64 ♔c7 ♕a2 65 ♕c6 ♔h8 66 ♔b8 ♕b3† 67 ♕b7 ♕g3† 68 ♔a8 ♕h3 69 ♕c6 ♕g4 70 ♕c3† ♔h7 71 a7

♖♙ versus ♖

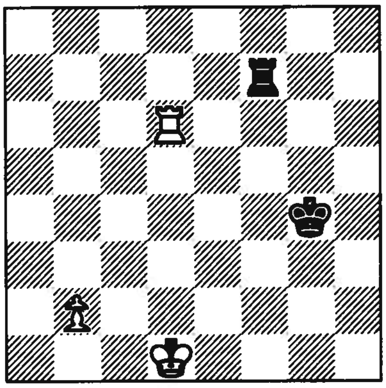
Table 3 is for Rook and Pawn vs Rook.

7	20 88.1	24 88.6	20 88.8	23 89.1
6	28 73.0	19 75.1	20 75.1	20 75.8
5	20 61.7	19 66.6	18 65.9	20 65.9
4	27 54.2	18 62.1	19 60.7	20 59.0
3	25 45.4	30 56.4	24 55.0	32 50.4
2	25 44.0	35 56.1	33 53.8	33 48.7
	a	b	c	d

Table 3

There is one RPvR example in Arlazarov and Futer. It is not possible to compare that to the current work because of differences in objective function. The goal in Arlazarov and Futer is to promote, while in the current work, the goal is to push the pawn to the next square.

The following is the longest RPvR variation:



Initial Position

1 ♖d5 ♖f2 2 ♔c1 ♔f4 3 ♖d2 ♖f1† 4 ♖d1 ♖f2 5 ♖e1 ♖g2 6 ♖e8 ♖h2 7 ♔b1 ♖h7 8 ♔c2 ♖c7† 9 ♔d3 ♖d7† 10 ♔c3 ♖c7† 11 ♔d4 ♖b7 12 ♖f8† ♔g4 13 ♔c3 ♖c7† 14 ♔d3 ♖b7 15 ♔c2 ♖c7† 16 ♔b1 ♖h7 17 ♖f6 ♖b7 18 ♖f2 ♖b8 19 ♔c1 ♔g5 20 ♔c2 ♖c8† 21 ♔d2 ♖b8 22 ♔c3 ♖c8† 23 ♔b4 ♖b8† 24 ♔a5 ♖a8† 25 ♔b6 ♖b8† 26 ♔a7 ♖b5 27 ♔a6 ♖b8 28 ♖c2 ♔f6 29 ♖c6† ♔e5 30 ♖b6 ♖a8† 31 ♔b5 ♔d4 32 ♖d6† ♔e5 33 ♖c6 ♖b8† 34 ♖b6 ♖a8 35 b4

References

1. T. Ströhlein, "Untersuchungen über kombinatorische Spiele," *Dissertation, Fakultät für Allgemeine Wissenschaften der Technischen Hochschule München* (1970).
2. Z. A. Komissarchik and A. L. Futer, "Ob Analize Ferzevogo Endshpilia pri Pomoshchi EVM," *Problemy Kybernet*, Moscow **29**, pp. 211-220 (1974).
3. V. L. Arlazarov and A. L. Futer, "Computer Analysis of a Rook Endgame," pp. 361-371 in *Machine Intelligence 9*, ed. J. E. Hayes, D. Michie, and L. J. Mikulich, Ellis Horwood Limited, Chichester, England (1979).
4. H. J. van den Herik and I. S. Herschberg, "A Data Base on Data Bases," *ICCA Journal* 9(1), pp. 29-34 (March 1986).

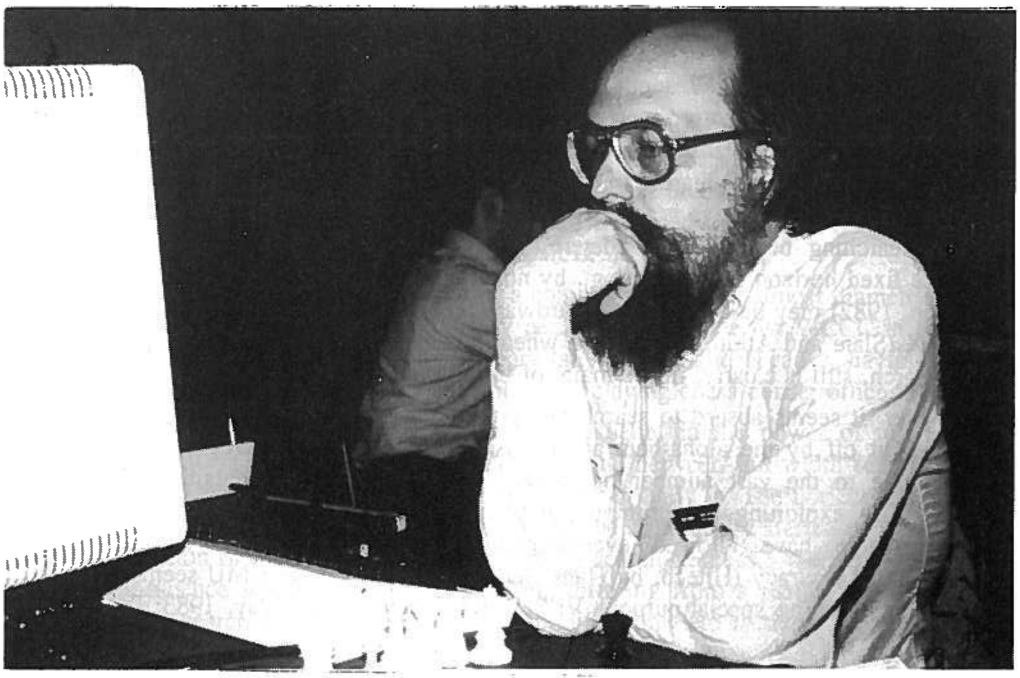


Photo by M.T. Fürstenberg

KEN ON THE SCAN
Five digits for digitally contemplating five-piece endgames.