The Game of Lasker Morris

Peter Stahlhacke
Lehrstuhl Mathematische Optimierung
Fakultät Mathematik und Informatik
Friedrich-Schiller-Universität Jena
07740 Jena – Germany
May 2003

ABSTRACT. We describe a retrograde algorithm used to solve Lasker Morris, and present the most important game theoretic results. The game, invented by Emanuel Lasker has its roots in the game of 'Nine Men's Morris'. It tries to give its predecessor more strategical potential to avoid drawish game play without making too drastic changes at the game rules.

1. Introduction

In recent years some non trivial strategic games have completely been solved. Awari, for example was solved by John W. Romein and Henri E. Bal [J.W. Romein, H.E. Bal, 2002]. Qubic was solved by Victor Allis [V. Allis, 92] and Connect Four by Allis [V. Allis, 88] and by James Allen [J. Allen, 89]. Interestingly, these two solutions to Connect Four used complementary approaches: Allis employed a knowledge based approach, whereas Allen used brute force depth first search. The game of Nine Men's Morris was solved by Ralph Gasser [R. Gasser, 1994] using a narrow alpha-beta database for the opening and a complete DTC (depth to conversation) database for the mid- and endgame. In other games, like chess [K. Thompson, 1986] and checkers [J. Schaeffer, 1997] the database approach was applied to solve important endgames
We describe the development of an retrograde algorithm used to create a complete DTW (depth to win) Lasker Morris database comprising 136,476,472,674 positions.

2. Lasker Morris

Nine Men's Morris is a board game for two players with simple rules. It was popular in the 14th century, but earlier versions with fewer than nine pieces have been found dating back to 1400 BC. As with other medieval games, many different rules have evolved over the years. **Lasker Morris** was invented by Emanuel Lasker, chess world champion from 1894 to 1921. It is based on the rules of Nine Men's Morris. In his book 'Brettspiele der Völker' from 1931 [Em.Lasker,1931] he described the rules of his new variant as follows. (translation by the author):

‘One move consists in placing a stone on a vacant point or in sliding an already placed stone to a free neighbour point and the player may do either this or the other. The number of stones in the hand, at the beginning of the game, may be nine or better ten.’

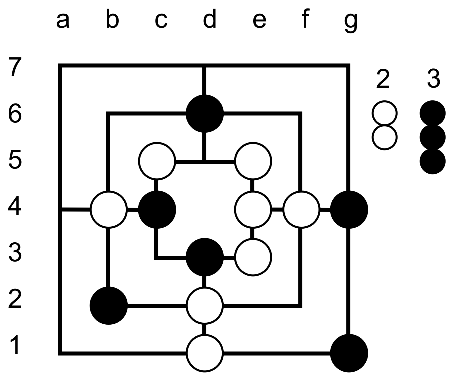


Figure 1. Typical situation for Lasker Morris. White to move has two pieces in his hand, black has three pieces left.

Nine Men’s Morris. The game is played on a square board, as illustrated in Figure 1. There are 24 points (intersections) and pieces may move between them only along the marked lines. Only one piece may be placed on any point. The players start with nine pieces, with a different color for each player.Typically one player is white and the other black. Unlike other games, the board is initially empty. The player with the white pieces starts. During the opening, players alternately place their pieces on vacant points. Each player tries to get three of his pieces in a row while preventing his opponent from doing the same. When a player gets three pieces in a row – this is known as closing a mill – he may take one of his opponent’s men off the board that is not part of a mill. After both players have put all their men on the board a piece is moved by sliding it from its square to a neighbouring empty square. Players keep trying to get three pieces in a row so they can take away an opponent’s man that is not part of a mill. As soon as a player has only three pieces left he may jump one of his pieces to any vacant point on the board.

The game may end in the following ways:

- a player who has less than three pieces loses
- a player who is stalemated loses
-

There are two more subjects that need to be discussed:

1. During the opening it is possible to close two mills simultaneously. Should the player then be allowed to remove two of his opponent’s pieces?
2. When a player has just closed a mill, but there are not enough opponent’s pieces outside a mill. Should the player then be allowed to remove a piece out of a mill?

In our implementation we answer the first question with yes and the second with no - according to the rules of the World Association of Nine Men’s Morris. The difference between NMM and Lasker Morris is that in Lasker Morris not all pieces have to be placed in the very first moves. Placing moves and sliding moves may be executed in arbitrary order.

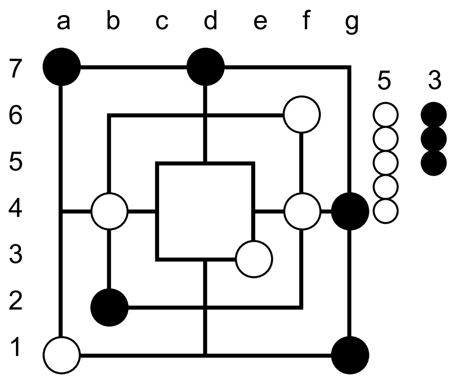


Figure 2. Black to move. g7xb4,f4 (!) is the only move that does not lose the game.

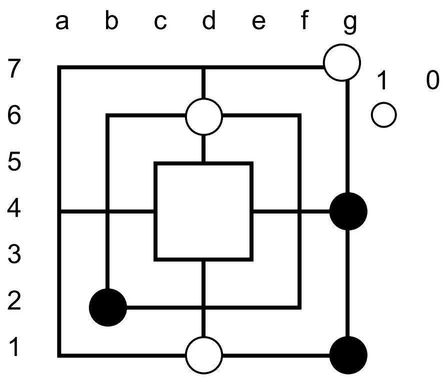


Figure 3. White to move is not allowed to jump.

3. Solving Lasker Morris

State Space
The board consists of 24 points. Each of these points can be occupied by a black piece or a white piece or it can be empty. There are also a number of pieces left in the hand of both players. In a typical game situation we can define four natural numbers: nw,nb,nwh,nbh – where nw/nb is the number of white/black pieces placed on the board and nwh/nbh is the number of white/black pieces left in the hand of the white/black player.
To calculate the number of elements in the complete state space we first can see that the whole state space can be divided into disjoint subspaces. Each subspace is characterized by the four parameters nw,nb,nwh,nbh. When assuming $0 \le nw,nb,nwh,nbh \le 10$ and $0 \le nw+nwh, nb+nbh \le 10$ we have 4356 different subspaces. We can now reduce the number of subspaces by rejecting them when one side has less than three pieces left in the game ($nw+nwh < 3$). So there remain only 3,600 different subspaces.

Unreachable positions
We can eliminate some more subspaces that only contain positions that can never be reached in the course of a legal game. Additionally we can eliminate all non-reachable single positions found in any subspace. In our implementation we do not reject all these positions because we are – of course – interested in them as we are interested in every position where one side can make a legal move.

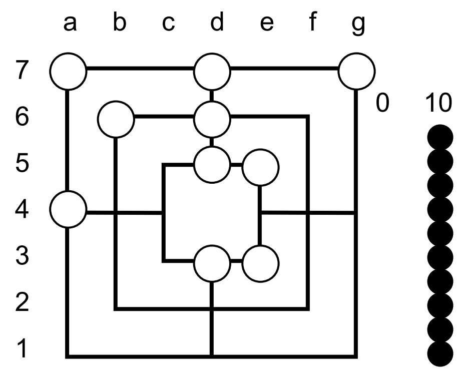


Figure 4. White to move. Win in 65 Paradise (unreachable) position.

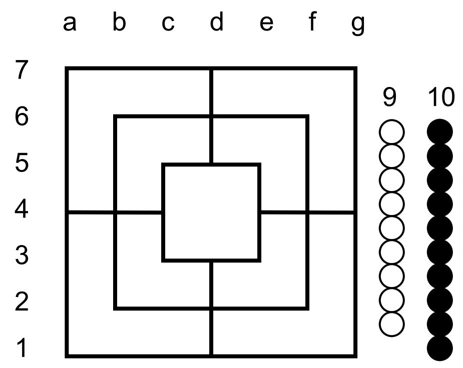


Figure 5. White to move. Loss in 23 Paradise position.

Symmetrical positions

Each subspace contains a lot of symmetrical positions. We have five symmetry axes, but one of these five is redundant, so we can expect at most a 16-fold reduction of the size of the state space. Constructing a function that perfectly solves the 'symmetrical' problem can be difficult. Therefore, as rapid computation is crucial, we only offer a semiperfect solution that reduces the state space by a factor of 15.66 instead of 16. The formula that calculates the size of a subspace is $\binom{24}{nw} * \binom{24-nw}{nb}$. Taking symmetrical positions into account we can reduce the state space including all subspaces from about 2,136 billion to 136.4 billion positions.

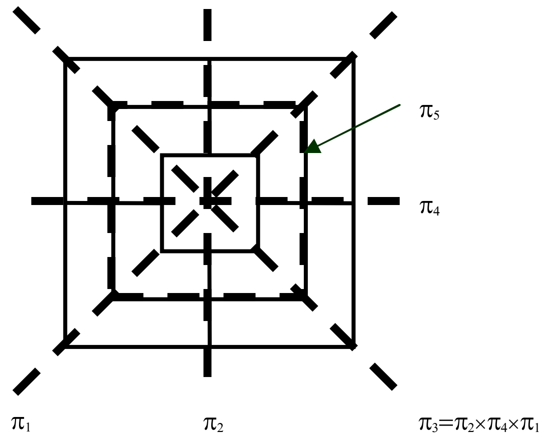


Figure 6. Symmetry axes.

Hash function
Our next goal is to define a hash function that compiles each particular position of a given state space with cardinality N into a unique natural number. Having such a hash function makes it unnecessary to store a compressed description of a position along with its game-theoretical value, because the description is encoded in the index of the hash function.

Database
The final database consists of 3600 files. For each subspace we have one file on the hard-disk. The value of a position is stored at the position that is returned from our hash function. The value is zero when the position is a draw or a loss, otherwise the value is equal to the number of (full) moves until white (to move) can force the win. We need 8 Bits to store this value, so the size of the complete database is about 136 billion bytes. To determine whether a position is a loss or a draw we have to apply a mini-max search of one ply. The cost of this search is about 10 to 30 hard-disc accesses per position – that's equal to the number of possible moves in an average position.

Calculation
The first step in retrograde analysis is to initialize all win-in-1 positions. After the initialization has set these wins, it sets the value of all remaining positions to zero. An iterative process then determines the real games-theoretic values of these 'zero' positions. In the n-th step we calculate and mark all win-in-n positions with white to move. Assume all wins in n for white to move are already marked in the database. In the next iteration step we pick up a win-in-n position. All predecessors of this position are potential losses for black (loss in n). If we know for sure that a predecessor q is lost for black regarding to our actual database we can mark all predecessors of q as a win-in-n+1 candidate for white (to move). To check whether q is really lost for black we have to check all the successors of q. If there is a successor that is not jet marked as a win for white, q is not a loss in n for black.

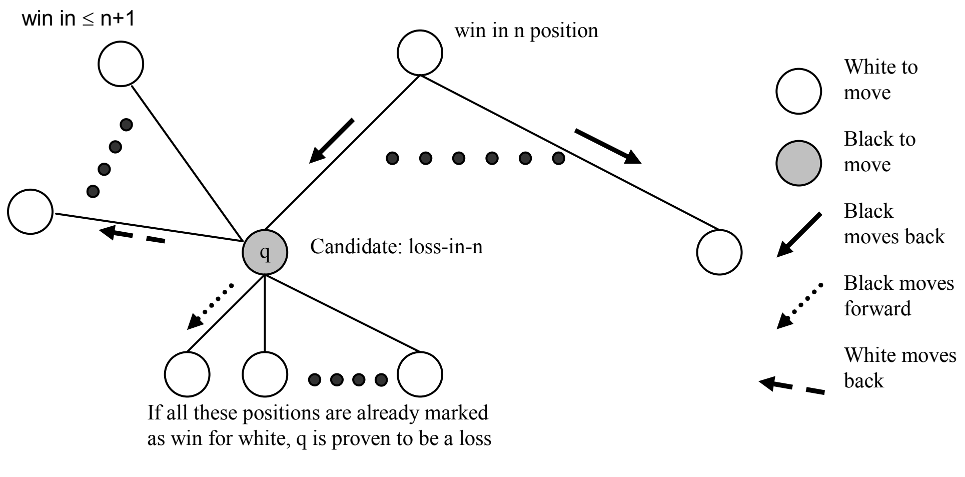


Figure 7. When q is a loss, we can mark all predecessors of q that are not jet evaluated as a win-in-n+1.

This algorithm sounds easy – and in fact, for all games with a small state space that easily fits into the main memory of today's computers computing such a database is an easy job. The state space of Lasker Morris has 2,134 billion

elements. Taking all symmetries into account we can reduce the state space to about 136 billion different positions. To speed up the computation of our database we have to avoid random disc accesses. Having even only one disc access for each position would increase the calculation time to more than 40 years! (136 billion * 10ms). In our implementation we utilize the fact that the database consists of 3,600 files. These files give us the possibility to make use of a very efficient caching method that is described below:
Consider an arbitrary subspace (nw=i, nb=j, nwh=m, nbh=n). After taking back any black move from any position from this subspace, we can find ourself in only five more different subspaces (Figure 8).Moving forward any move from these five subspaces (to prove the loss) throws us into at most thirteen more subspaces. And finally taking back all white moves from the proven loss positions leads also to at most thirteen subspaces.

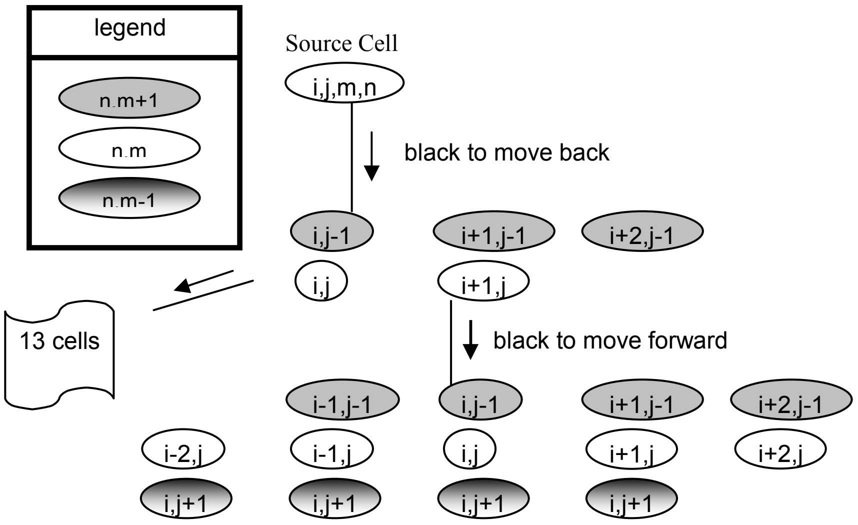


Figure 8. To reduce random disc accesses we operate with cells that are completely loaded temporary into the main memory of the computer.

In our iteration algorithm we arrange a loop over all subspaces to separately find all win-in-n+1 positions of each subspace. This gives us the chance to completely load the few affected subspaces into the main memory of our computer to avoid many disc accesses.

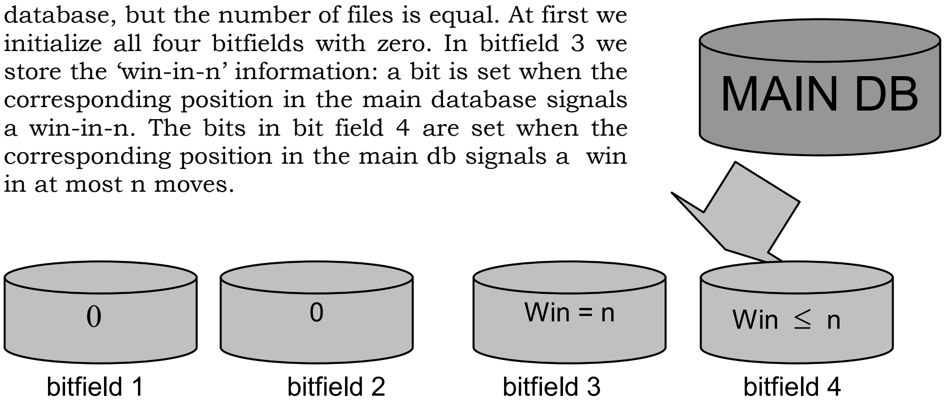
In the following we describe the iteration algorithm in detail that we have used to calculate the whole Lasker Morris database.

Step 0 (Initialization)

This step will be executed only once at the beginning of the iteration process. We create the main database: for each subspace we create one file on the hard disc. The file size is equal to the number of positions in the subspace taking all symmetries into account. All Bytes are set to zero. After this we apply a loop over all positions to explicitly find and mark all win-in-1 position.

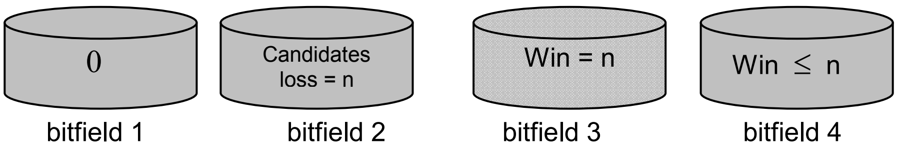
Step 1

In this step we create four bitfields. Each bitfield consists of 3600 files – for each subspace one cell (file). Instead of using one byte per position we now use only one single bit. So the size of one bitfield is only 1/8 of the size of the main database, but the number of files is equal. At first we initialize all four bitfields with zero. In bitfield 3 we store the 'win-in-n' information: a bit is set when the corresponding position in the main database signals a win-in-n. The bits in bit field 4 are set when the corresponding position in the main db signals a win in at most n moves.



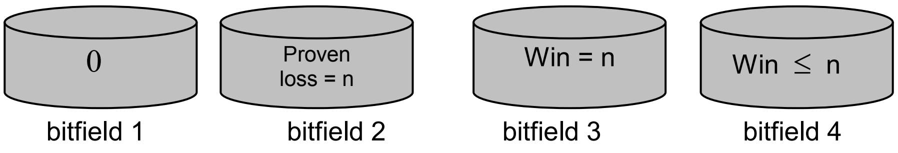
Step 2

In this step we store the candidates for a loss in n (black moves and loses) into bitfield 2. We start a loop over all cells from bitfield 3. We load one cell from bitfield 3 and the five corresponding cells from bitfield 2 into the main memory of our computer. Then we apply another loop over all 'win-in-n' positions, we take back all black moves and mark them as 'loss in n' candidates in one of the five corresponding cells. After finishing the inner loop we store the five cells back to the hard-disk and switch to the next cell from bitfield 3.



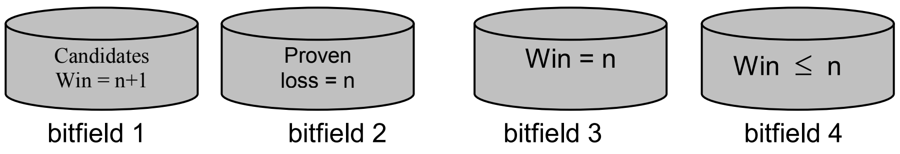
Step 3

In this step we try to prove the loss candidates found in step 2. At the end of this step bitfield 2 contains all proven 'black to move loss in n' positions. Similar to step 2 we apply two loops over all cells from bitfield 2 and over all positions of one cell. We pick up every loss-candidate and generate the set of all successor positions. If one of these successor positions is not marked as a win for white in the corresponding cell from bitfield 4, we can reject the candidate.



Step 4

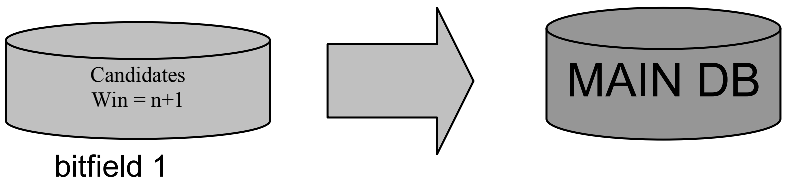
Here we load all cells out of bitfield 2 and the corresponding five cells from bitfield 1. Now we take back all white moves from all ‘loss in n’ positions found in bitfield 2 and mark them as ‘win in n+1 candidate’ in bitfield1.



Step 5

In this step apply an update of the main database. A position is marked as ‘win in n’ when the following two conditions are true

- the current state in the main database is zero
- the corresponding bit in bitfield 1 is set (win-in-n+1 candidate)



Improvements and Verification

In our implementation we applied some more improvements. We connected step 5 and step 1, so the main database had not to be read again after updating. Secondly we managed a special path over the cells of one bitfield to reduce the data traffic when loading and storing the five corresponding cells. The idea is that the sets of the five cells of two neighbouring cells are not disjoint. So we can keep the common cells in the main memory when switching to another cell.

The ten men database was computed between November 2002 and March 2003. The calculation took approximately four month on an Athlon XP 1533MHZ (XP1800+) Computer with 1 GB of main memory. The program is written in C and compiled with GNU C++. For the GUI we used Borland CPP Builder.
During the calculation we experienced some hardware problems. Three of four hard drives involved had malfunctions. The complete database had to be calculated again since the backup hard drive was defective too. After changing all drives the calculation proceeded without any problems. In early and middle of 2002 we had calculated Lasker Morris with seven, eight and nine men. All databases are included in their successors. Since a comparison of files showed no differences we have a good chance that our final database is free from any errors due to defective hardware.

But how can we be sure that there are no software errors? Presently we have no formal proof that our program to create all databases has no errors. But on the other side we have not found any anomaly during our test works in the last three years. This test work include many hundreds of games that have been played in the last time with the Morris-GUI. This GUI always applys a one or two ply minimax search to determine the value of each possible move. In this search it performs a depth check to compare the value stored for the origin position with the values of the positions after each move. Of course, it would be very reassuring when the game was recalculated by an independent database program.

4. Results

Up to the present day Lasker Morris is a relatively unknown game. So we present our results in comparison to the well known mother game Nine Men's Morris. The initial position of Lasker Morris is a draw, if both sides play correctly.
Our results for Nine Men's Morris are based on our own DTM database that was calculated in the beginning of 2000.

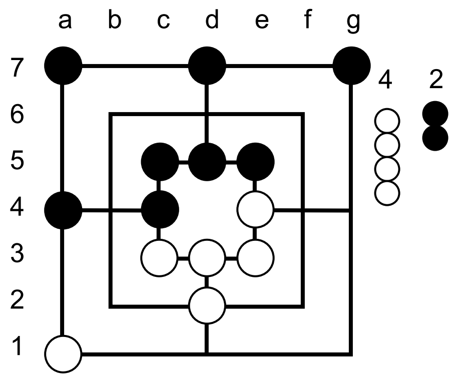


Figure 9. Maximal position in Lasker Morris. White to move. Win in 171.

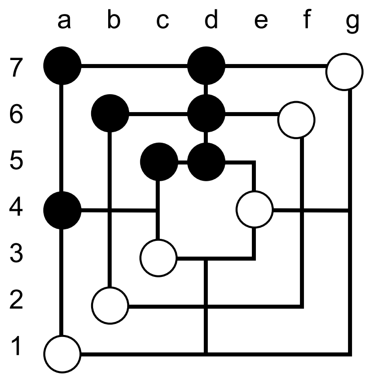


Figure 10. Maximal position in Nine Men's Morris. White to move. Win in 165.

Lasker Morris is a draw. So it is interesting to look how early a mistake can happen. Figure 11 (12) shows the longest winning distance with two (four) men sitting on board after an early mistake of black. Both positions are draws for Nine Men's Morris. The earliest possible error that can occur in Nine Men's Morris is the second move of white, namely 1.c3 f4 2.f2?
Figure 5 (in section 2) shows the power of one single piece. With one piece less, Lasker Morris is won for the side that has the majority.

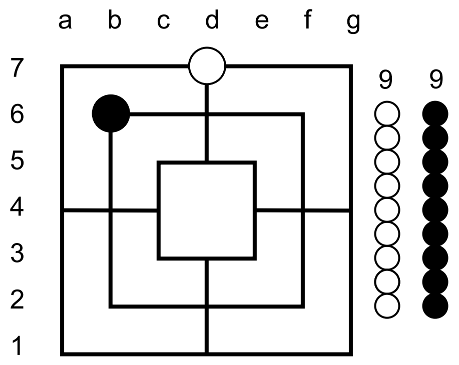


Figure 11. Opening position with white to move. Win in 72.

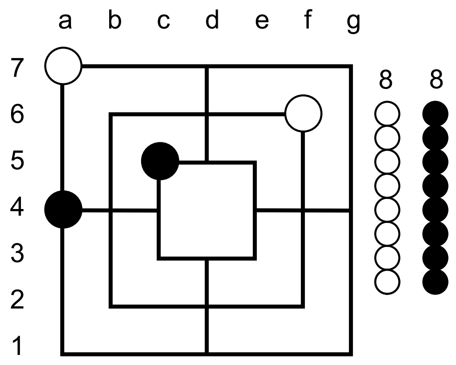


Figure 12. Opening position with white to move. Win in 132.

Figures 13 to 16 show the distribution of the winning distances for all positions with white to move. The horizontal axis displays the distance in (full) moves and the vertical axis is labeled with the number of positions. We can see similarities between Lasker Morris and Nine Men's Morris. Even the lokal peaks are approximately at the same position, the higher distances from Lasker Morris are shiftet about 4 moves rightwards.
In Lasker Morris 51.47 % of all positions are won for white with white to move whereas the winning probability in Nine Men's Morris is only 46.55%. The difference in percentage looks small but is an indicator for our practical observation that Lasker Morris is a much sharper game than Nine Men's Morris.

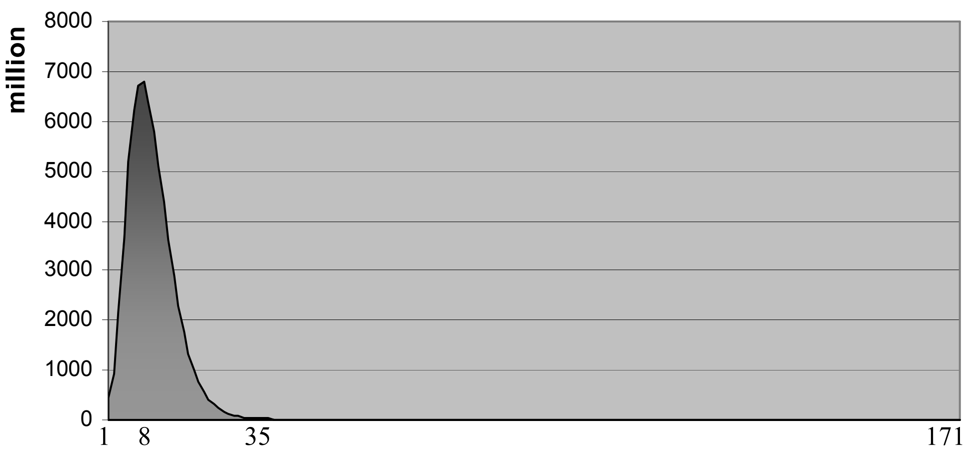


Figure 13. Distribution of the winning distances of Lasker Morris.

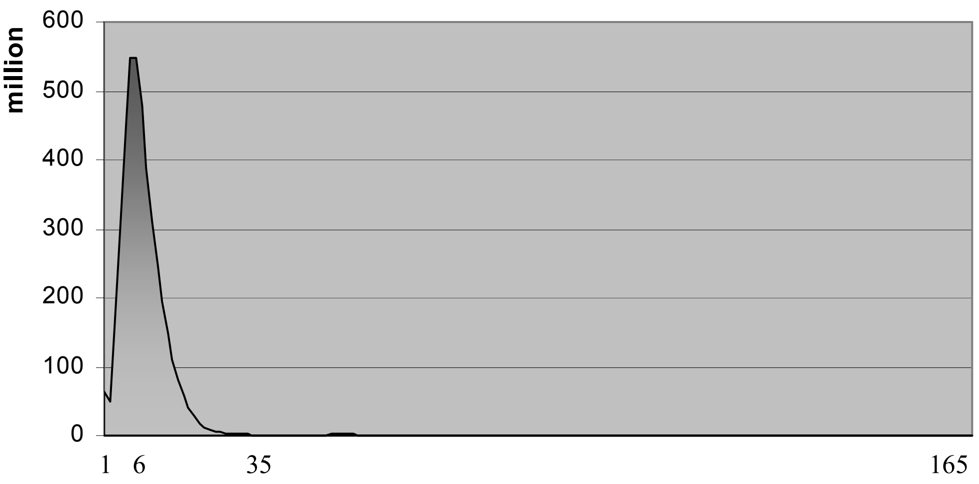


Figure 14. Distribution of the winning distances of Nine Men's Morris .

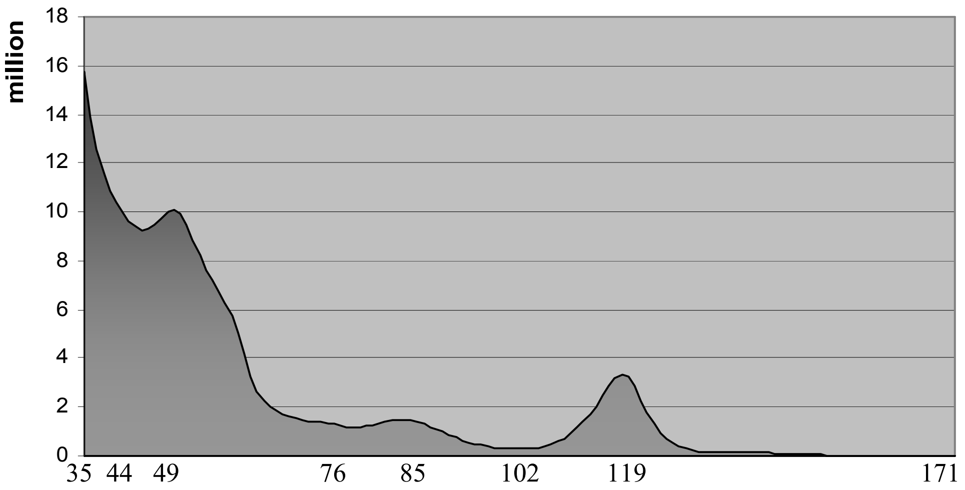


Figure 15. Distribution of the winning distances of Lasker Morris.
Enlargement of the distribution from Figure 13.

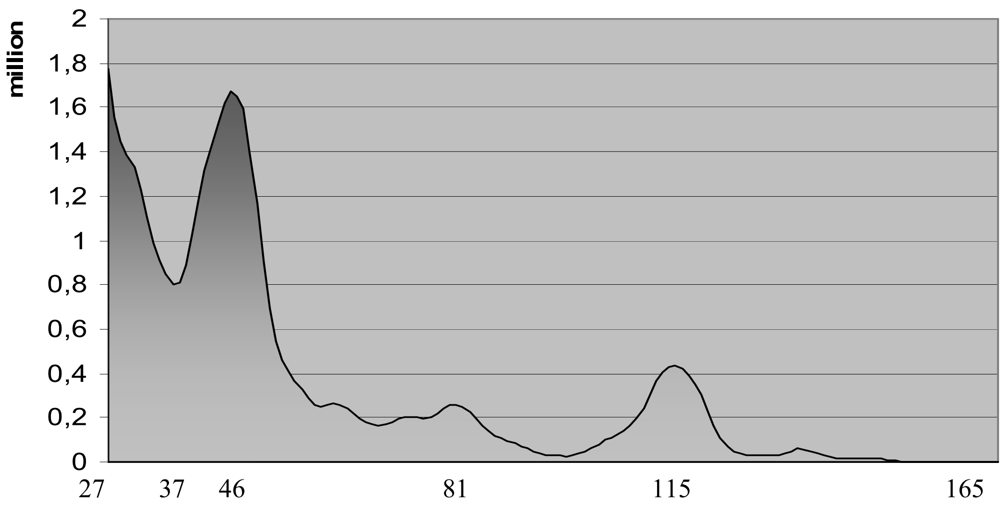


Figure 16. Distribution of the winning distances of Nine Men's Morris. Enlargement of the distribution from Figure 14.

5. Conclusions

The Game of Lasker Morris is a draw. A DTW (depth to win) database with the game theoretic values of all possible 136 billion positions is now available. For the future we are planning to find and implement a strategy that improves the chances of achieving more than the game theoretic value against a fallible opponent.

6. Appendix

Table 1 shows an optimal movesequence to achive the game theoretic value of Figure 11 with perfect play of both sides.

1.	g4!	a7	26.	d2-f2	e4-e3	51.	e5-e4	d2-d3
2.	d2!	a1	27.	g4-g7	e3-e4	52.	c5-d5	d3-d2
3.	a4!	f4	28.	c5-d5	c4-c3	53.	d5-e5	d2-d3
4.	d3!	a1-d1	29.	g7-d7xe4	b4-b2	54.	a7-d7	d3-d2
5.	c3!	f6	30.	a4-b4	b2-d2	55.	e4-e3	d2-d3
6.	e3xf6	f6	31.	d5-c5	d2-d1	56.	e5-e4	d3-d2
7.	c3-c4!	d6xa4	32.	d7-a7	d1-a1	57.	e3-d3	a4-a1
8.	c4-c3xd1!	b6-b4	33.	d3-e3	c3-d3	58.	e4-e5	a1-a4
9.	d1xd6!	c4	34.	e3-e4	d3-d2	59.	d7-g7	a4-a1
10.	e3-e4!	a7-a4xc3	35.	c5-d5	a1-d1	60.	g7-g4	a1-a4
11.	d2-f2!	b4-b6	36.	a7-d7xd1	f4-g4	61.	g4-g1	d2-d1
12.	f2-d2xc4!	a4-b4	37.	f2-f4	g4-g7	62.	d3-d2	a4-a1
13.	d2-b2	d6xb2	38.	e4-e3	d2-d3	63.	f4-g4	a1-a4
14.	d3-d2!	d6-d5	39.	d5-c5	d3-c3	64.	g4-g7	a4-a1
15.	d7-d6!	b2xd6	40.	c5-c4	c3-d3	65.	e5-e4	f6-f4
16.	d6!	b4-c4	41.	c4-c3	g7-g4	66.	g7-d7	a1-a4
17.	b4!	c4-c3	42.	d7-g7	g4-g1	67.	d7-a7	f4-g4
18.	d1-a1	c3-d3	43.	e3-e4	g1-d1	68.	e4-f4	g4-g7
19.	a1-a4	d3-e3	44.	c3-c4	d3-c3	69.	f4-g4	a4-a1
20.	e4-e5	e3-d3	45.	e4-e5	d1-a1	70.	a7-a4	g7-d7
21.	b4-c4	b2-b4	46.	g7-d7	a1-a4	71.	g4-g7	d7-a7
22.	c4-c3	d5-c5	47.	d7-a7	c3-d3	72.	g7-d7#	
23.	e5-d5	c5-c4	48.	e5-d5	d3-d2			
24.	d5-c5	d3-e3	49.	c4-c5	d2-f2			
25.	c3-d3	e3-e4	50.	d5-e5	f2-d2			

Table 1. Optimal move sequence. A move is marked by an exclamation mark when it is the only move that keeps the win. In his first 17 moves white has to find the unique win for 15 times.

References

[R. Gasser, 1995] Harnessing *Computational Resources for Efficient Exhaustive Search*. PhD thesis, ETH, Swiss Federal Institute of Technology, Zurich. *ICCA Journal*, Vol. 18, No. 2, pp. 85-86. ISSN 0920-234X.

[J. Schaeffer, 1997] *One Jump Ahead: Challenging Human Supremacy in Checkers*. Springer-Verlag, New York, N.Y. ISBN 0-3879-4930-5.

[J.W. Romein, H.E. Bal, 2002] *Awari is Solved.* ICGA Journal, Vol. 25, No. 3, pp. 162-165

[J. Allen, 1989]. A Note on the Computer Solution of Connect-Four. *Heuristic Programming in Artificial Intelligence: the first computer olympiad* (eds. D.N.L. Levy and D.F. Beal), pp. 134-135. Ellis Horwood Ltd., Chichester, England. ISBN 0-7458-0778-X.

[L.V. Allis, 1988] *A Knowledge-Based Approach of Connect Four: The Game is Over, White to Move Wins.* M.Sc. Thesis, Vrije Universiteit. Report No. IR-163, Faculty of Mathematics and Computer Science, Vrije Universteit, Amsterdam

[K. Thompson, 1986). Retrograde Analysis of Certain Endgames. ICCA Journal, Vol. 9, No. 3, pp. 131-139.

[Em. Lasker, 1931] *Brettspiele der Völker*