# Applying Retrograde Analysis to Nine Men's Morris

Ralph Gasser
Informatik ETH
8092 Zürich
Switzerland

## Abstract

This paper presents results gained through retrograde analysis of Nine Men's Morris.
First, I describe some of the methods employed to accelerate the calculations, for example run-length encoding of the databases.
Second, the results themselves are examined. While many human players intuitively regard Nine Men's Morris as a rather simple, drawn game, my results seem to suggest otherwise. Some of the optimal play is clearly beyond human ability, sometimes requiring over 150 plies of seemingly random moves before a win is reached.

# 1. Introduction

Work on this subject originally evolved from my Nine Men's Morris program [Gasser 90]. Due to the different phases of the game, it was evident that the midgame evaluation function alone was insufficient. It was particularly unsuited during the endgame. Instead of imperfectly tuning the evaluation function to these cases, I decided to solve the problem by exhaustively analyzing certain endgame positions.
Retrograde analysis is a technique which has become quite popular recently for solving such exhaustive analysis problems. It is much applied to chess, where some interesting and unsuspected results have turned up. For a summary see [Herik 86]. Some of these results have forced FIDE, the International Chess Federation, to change their rules [Herik 89].
Although many people have already worked on retrograde analysis, it seems to me that the employed algorithms still have the potential for improvement. Some improvements will be rather problem specific, while others may have a broader application.
Along these lines, retrograde analysis for Nine Men's Morris was implemented. I will explain how run-length compression was used to speed up the process and discuss other more problem dependant considerations.
Afterwards, the results gained for Nine Men's Morris are presented. Although these results do not yet have any influence on the question of whether the complete game of Nine Men's Morris is a draw or not, they do suggest that Nine Men's Morris is much richer in possibilities than most human players suspect. Particularly the subgame where both players have only 3 stones on the board is hardly ever (0.2%) a draw, but for humans, finding a win is extremely difficult.

# 2. Nine Men's Morris

Nine Men's Morris is played on a board with 24 points where stones may be placed (Fig.1).

Initially the board is empty and each of the two players receives nine stones. The player with the white stones starts. The game consists of three phases:

- opening phase
- midgame phase
- endgame phase

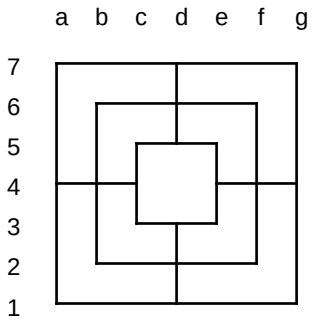


Fig. 1 The empty board

During the **opening phase**, players alternately place their stones on any vacant point. After all stones have been placed, the board might look as in Fig. 2.

Now play proceeds to the **midgame phase**. Here a player may slide any one of his stones to an adjacent vacant point.
If at any time during the game, a player succeeds in arranging three of his stones in a row, also called "closing a mill", he must remove an opponent's stone. Any stone which is not likewise part of a mill may be removed. If all the opponent's stones are part of mills, any stone may be removed. In Fig.3, White has just played to b6 (opening phase) and can now remove Black's stone on a1, but not the one on d2.

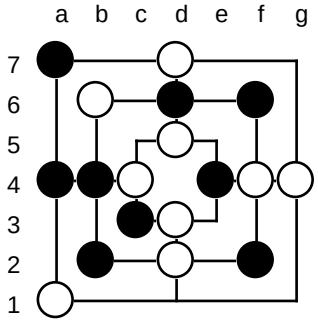


Fig. 2 After the opening phase

As soon as at least one player has only three stones left, we proceed to the **endgame phase**. When it is his turn, the player with three stones may take one of his stones and jump to any vacant point on the board. If he closes a mill, he may, as usual, remove an opponent's stone.

The game ends in the following ways:

- The first player who has less than three stones loses.
- The first player who cannot make a legal move loses (Fig. 4).
- If a repetition of a position occurs, the game is a draw.

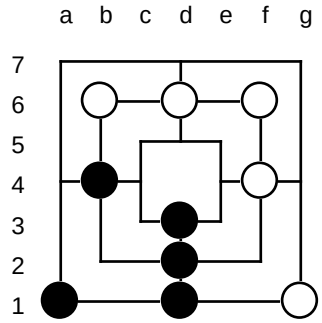


Fig. 3 White moves to b6
Either a1 or b4 is removed

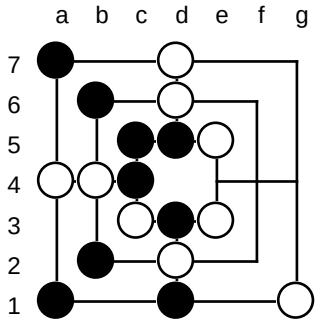


Fig. 4 Black to move
No legal moves available

# 3. Retrograde Analysis

## 3.1. What is retrograde analysis?

The expression "retrograde analysis" is used extensively in computer chess circles, where it stands for a method of calculation which finds the optimal play for all possible board positions in a specific endgame. In computer science, the somewhat analogous term "dynamic programming" is often used.

The characteristic of this method is that a backward calculation is performed. One starts with all positions that are immediate wins or losses. From these, all won or lost positions in 1 ply can be calculated and then with all these positions, the positions that are won or lost in 2 plies etc.

There are two main reasons for applying this method to Nine Men's Morris. First, it seems possible that new insights into the method might be gained that are not as easily visible using chess. Second, Nine Men's Morris is considerably less complex than chess. This smaller domain attracted me, because in the long run there is a chance of solving the game completely.

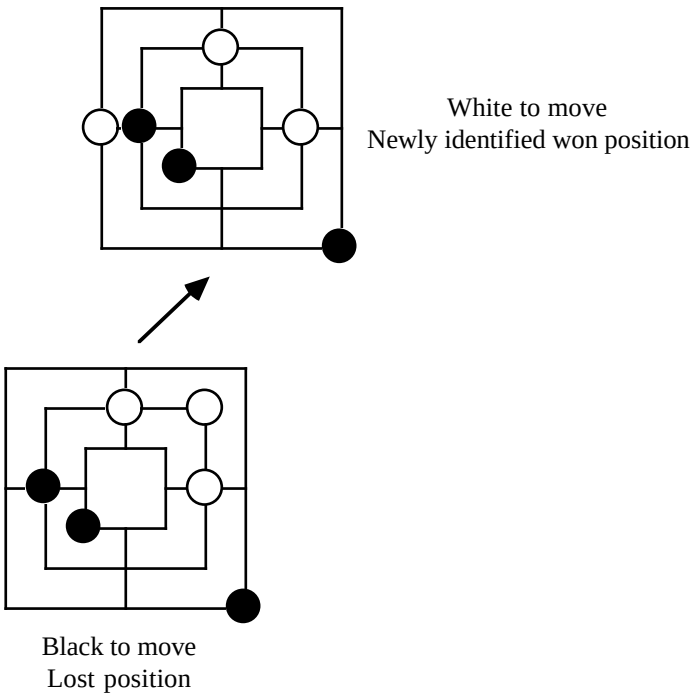
## 3.2. Standard retrograde analysis

Retrograde analysis is accomplished by the following three procedures:

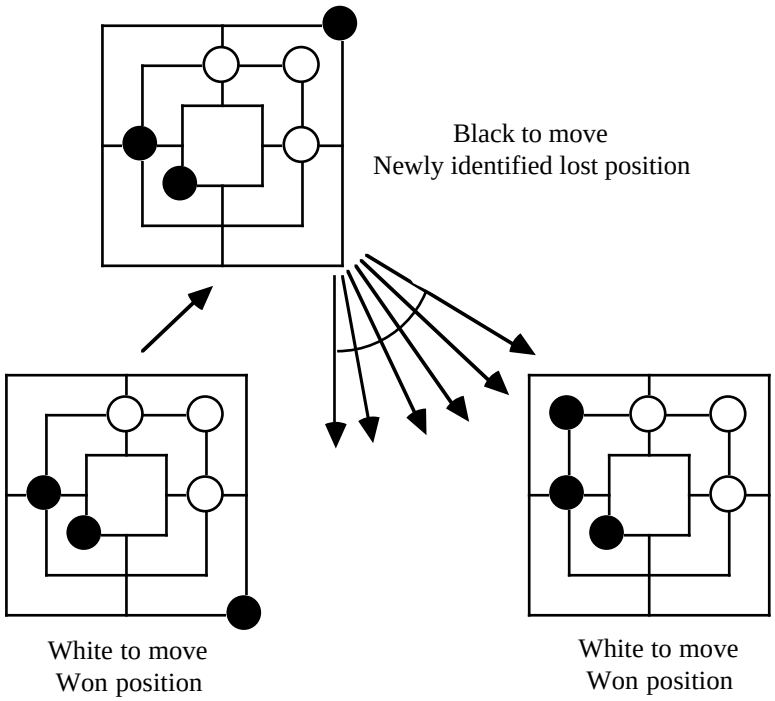
- Initialization
- Loss backup
- Win backup

The initialization loop goes through all boards and identifies terminal positions, where one player has either won or lost. In Nine Men's Morris these terminal position are either positions where the player to move is blocked and therefore loses (Fig. 4) or positions where the player to move can close a mill and thereby reduce his opponent to less than three stones.

The loss backup procedure searches for all positions in which the player to move has lost in a specified number of plies. For these positions all legal predecessor positions are generated. All of these predecessor positions are wins.



The win backup procedure is the most processing intensive. First, the won positions for a specified ply-depth are identified, and their predecessor positions generated. For a predecessor position to be a loss, all of its possible successor positions must be wins.




Retrograde analysis is accomplished by repeatedly applying the loss and win backup procedures to all board positions. The algorithm terminates when no new wins or losses are identified.

## 3.3 Possible Improvements

This section deals with possible improvements to the standard method described above. They are not all equally applicable to other games or other hardware configurations, but were specifically tailored to Nine Men's Morris and the Apple Macintosh's operating system.

### 3.3.1. Initialization

In the last section, we saw that the first step in retrograde analysis is to initialize all the immediate won/lost position. As a first approach this can be solved by looking at every possible board position and determining if it is won or lost.
The cost of doing this initialization can be a high percentage of the total run-time of retrograde analysis, especially if most positions are draws (i.e. the win and loss backup loops don't require much time). An example of such a situation occurs in the 4-4 subgame of Nine Men's Morris where only 188 of the 3'225'597 positions are non-draws.

To find an alternative, let's take a look at a specific set of four against four positions.

Fig. 5 shows a possible configuration of four white stones. There are 4845 possible ways of placing the four black stones. If it is White's turn to move we would normally calculate all 4845 possible positions and for each of them test if White is blocked. There is a faster way however. Fig. 6 shows that we need at least 9 black stones to render all the white stones immobile, therefore none of the black stone configurations can block the four white stones. A similar process can be used when it is Black's turn to move. In Fig. 7 we see how the four white stones partition the board into 3 areas. The areas contain 1,2 and 17 empty squares. It is therefore possible to place 1,2,3,17,18,19 or 20 black stones so that they have no adjacent and vacant squares, i.e. they are blocked. But there is no way for White's stones to block four black stones. So once again, we can rapidly determine that none of the 4845 black configurations result in Black being blocked.

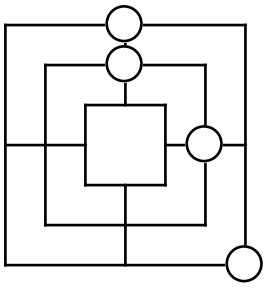


Fig. 5 White stone configuration

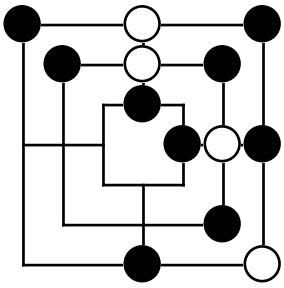


Fig. 6 Minimum amount of black stones needed to block White

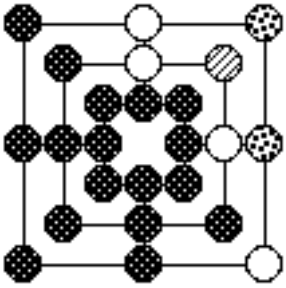


Fig. 7 White cannot block 4 black stones

This method of looking at a multitude of similar positions simultaneously is admittedly rather problem dependent. But the main idea can probably often be used in one form or other. It is certainly also applicable to chess. For instance, after placing all pieces of one color, one can determine a subset of the opponent's 64 possible king positions, so that a mate can only occur by placing the king on a position in this subset.

### 3.3.2. Win backup

In the standard win backup procedure, we go through all boards identifying the positions which win in a specified number of plies. For each of these, we generate their possible predecessor positions. For every predecessor position which has not yet been identified as a win or loss we generate its successor positions. If all the successor positions are wins, we have a newly identified loss. The complexity of this calculation is :

O ( #Wins * #AvgPred * #AvgSucc ) where

# Wins : number of winning positions in specified number of plies

# AvgPred : average number of predecessor positions

# AvgSucc : average number of successor positions

There is another way of achieving the same result. Instead of starting with the won positions, backing them up and checking if they are so far unidentified, we can simply go through the file searching for unidentified positions. For all these positions we then perform the same check with their successor positions. Doing the win backup this way results in the following complexity:

O ( #Unidentified * #AvgSucc )

Roughly speaking, this suggests that the second method is faster if

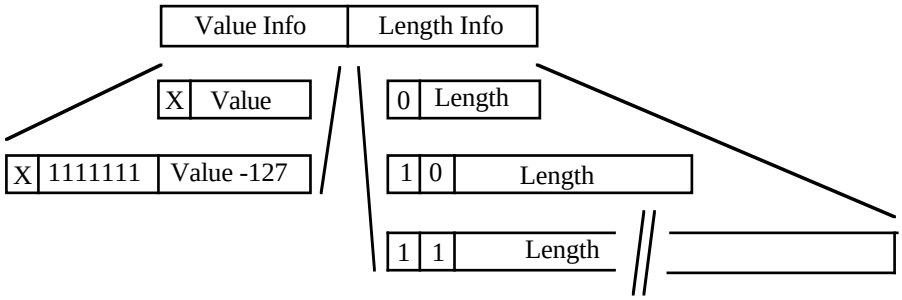
#Win * #AvgPred > #Unidentified.

In Nine Men's Morris such situations can occur. For example in the 7-3 subgame we have some 30 million positions and the number of wins in 3 plies is about 900'000. Since #AvgPred is about 50 it is justified to identify the losses in 4 plies using the second method.

### 3.3.3. Run-length encoding

My calculations were done on an Apple Macintosh, which was also used for other tasks, so that only 30-40 MBytes of hard disk storage capacity was available. Therefore calculating databases with 150 million positions (with a byte per position to differentiate between 256 plies) wasn't possible without some sort of data compression. Run-length compression seems fairly well suited to Nine Men's Morris, because there are many drawn positions which can be strongly compressed. I also tried to translate 'similar' board positions to adjacent file positions. Here 'similar' boards are those which require an identical number of plies until the game ends. If this can be done, run-length compression is more effective.

Standard run-length encoding transforms a sequence of identical values into only one copy of the value plus the length of the sequence. It generates large savings for long sequences, but for short sequences (for example a sequence length of one) it may actually use more memory.
Examining the Nine Men's Morris databases showed that there were many fairly short sequences and also that certain values (those corresponding to low ply numbers) were more frequent. Combining run-length encoding with variable length encoding might therefore be a good idea. These ideas lead to the coding scheme displayed below.




This is basically a run-length encoding, but since most positions will either be drawn or won/lost in a small number of plies, these more frequent values are stored using only one byte. Seven bits are used to store draws and wins/losses from 0 to 125 plies. For up to 381 plies a second byte is used. Since there are many runs of length one, a bit in the value info (bit X) was used to indicate a run-length of one. If the run-length is greater than one, one byte is used for run-lengths from 0-127. Two bytes for 0-16'383 and four bytes for even longer runs. This code has a certain redundancy, but has proved sufficient. Below the resulting compression factors are listed for the calculated subgames. A subgame is characterized by the number of stones on the board, i.e. the 5-3 subgame has 5 stones of one color and 3 of the other.

| Subgame | Uncompressed File Size | Compressed File Size | Compression factor |
|---|---|---|---|
| 3-3 | 210'140 | 89'803 | 2.3 |
| 4-3 | 1'725'960 | 159'387 | 10.8 |
| 4-4 | 3'667'665 | 714 | 5136.8 |
| 5-3 | 5'484'540 | 84'212 | 65.1 |
| 5-4 | 21'938'160 | 37'210 | 589.6 |
| 6-3 | 14'319'168 | 597'471 | 24.0 |
| 5-5 | 32'907'240 | 97'427 | 337.8 |
| 6-4 | 53'696'880 | 12'765'238 | 4.2 |

Run-length encoding has previously been suggested for compressing the final databases [Marris 89]. As far as I am aware, no previous attempts were made to perform retrograde analysis using compressed data.
Any time during retrograde analysis when we perform a sequential search through the file, for example when we need all board positions with a given value, run-length compression speeds our search because the file is smaller. A problem only occurs when we need the value of a specific board position. When using uncompressed databases, we know exactly at which position the value can be found. Using run-length compression, we must basically search through the whole file until we reach the required position. By using an index which points into the compressed file at numerous places we can speed this up. With this index, we can restrict the sequential search to a smaller part of the file. If the compression factor is large enough, this sequential search can still be faster than using uncompressed data, because a larger part of the file can be held in the memory cache, eliminating slow disk accesses. Another fact to keep in mind is that the compressed file size is small at the beginning of retrograde analysis and grows larger the more new positions we identify. This means that for small ply-depths, when most new positions will be found, we still have a small file i.e. fast access. When the compressed file grows larger with increasing ply-depth, the number of positions that we have to check usually grows smaller, so that fast access grows less critical.

# 4. Results

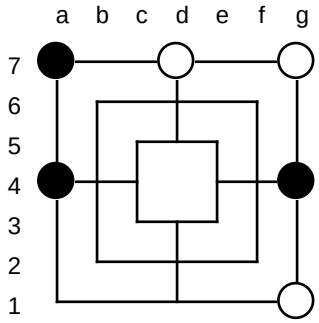
The following results were all obtained using a Macintosh IIx with 8 MBytes of memory. The run-times varied from a few minutes (4-4 subgame) to weeks (6-4 subgame). More exact timing is not available because this program was only run when the computer was not otherwise used.
For each database, some brief statistics along with an example of one of the most difficult positions are given. When there is more than one optimal move, the alternatives are indented. The terminology is as follows: A W-B subgame includes all board positions where there are W white stones and B black stones i.e. the 5-4 subgame comprises all positions with 5 white stones and 4 black stones. The positions with 5 black and 4 white stones can be transformed to 5-4 positions by switching stone colors. A position is called 'won' if the player to move can win, likewise for losses. The 'Max Win Ply' and 'Max Loss Ply' columns give the longest winning or losing sequence for a position in the subgame. Playing this sequence can result in closing mills and entering different subgames. 'Max Win Ply' and 'Max Loss Ply' measure the number of plies until a win/loss is finally reached and not the number of plies until the subgame changes.
These results disprove many statements found in [Müller 87]. It seems that humans are not especially good at these combinatorial games.

## 4.1. 3-3 Subgame

This was the first subgame to be examined and to date it is also the most interesting, i.e. the percentage of draws is extremely low (about 0.2%). This contradicts human intuition, which suggests that the 3-3 subgame is almost always a tie.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 47'167 | 96 | 9'659 | 25 | 26 |

An example of a win position in 25 plies is shown below along with a possible winning sequence:




White to move
win in 25 plies

○ 1 g7-a1
● 2 a7-d1
○ 3 a1-d6
● 4 d1-d5
○ 5 g1-b6
● 6 g4-f6
○ 7 d7-b4
● 8 d5-b2
  ● 8 a4-b2
○ 9 b6-d5
● 10 b2-d7
○ 11 d6-c5
● 12 d7-e5
○ 13 d5-b6
● 14 f6-b2
○ 15 b4-c4
● 16 a4-c3
○ 17 c5-a4
● 18 c3-b4
  ● 18 b2-b4
  ● 18 e5-b4
○ 19 c4-a7
● 20 e5-a1
  ● 20 b4-a1
  ● 20 b2-a1
○ 21 a4-d7
● 22 a1-g7
  ● 22 b4-g7
  ● 22 b2-g7
○ 23 a7-d6
● 24 g7-d5
  ● 24 g7-f6
  ● 24 b4-d5
  ● 24 b4-f6
  ● 24 b2-f6
  ● 24 b2-d5
○ 25 d7-f6() xb4

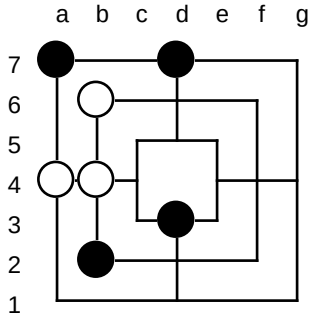
## 4.2. 4-3 Subgame

This subgame is somewhat less interesting than 3-3, but we can still gain some new insights from this database. First, observe that the player with 4 stones has a maximum win ply of 1 i.e. he can only win if it's his turn and he has an open mill, a rather pathological case which probably won't occur during actual game playing. Second, if it's Black's (3 stones) turn, he can't

lose! Obviously the additional mobility gained by having the ability to jump (i.e. 3 stones left) is quite valuable.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 75'397 | 681'906 | 3'095 | 1 | 32 |
| Black To Move | 102'281 | 658'117 | 0 | 33 | - |

An example of a win position in 33 plies is shown below along with a possible winning sequence:




White to move win in 33 plies

○ 1 b6-g7
● 2 d7-d6
○ 3 a4-g1
○ 3 b4-g1
● 4 a7-d7
○ 5 b4-f6
● 6 d3-d2
● 6 d6-b6
● 6 d6-d5
● 6 d3-e3
○ 7 g1-f2
○ 7 g7-f2
● 8 d7-a7
● 8 d6-b6
● 8 d2-d3
● 8 d2-d1
○ 9 g7-f4() xd6
○ 9 g7-f4() xa7

● 10 a7-b6
● 10 d2-b6
○ 11 f4-b4
● 12 d2-f4
○ 13 f2-a4
● 14 f4-c4
○ 15 f6-a7
● 16 b2-a1
● 16 b6-a1
○ 17 a4-g7
● 18 b6-d7
● 18 c4-d7
○ 19 a7-g1
● 20 c4-g4
○ 21 g7-a4
● 22 g4-c4
● 22 d7-c4
○ 23 a4-b2

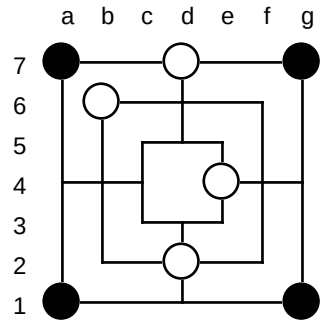
● 24 c4-b6
○ 25 b4-f2
● 26 a1-d2
○ 27 b2-d1
● 28 b6-a1
● 28 d7-a1
○ 29 d1-f4
● 30 a1-f6
● 30 d2-f6
● 30 d7-f6
○ 31 f2-g4
● 32 f6-e4
● 32 f6-g7
● 32 d7-e4
● 32 d7-g7
● 32 d2-e4
● 32 d2-g7
○ 33 f4-g7() xd2

## 4.3. 4-4 Subgame

This is a rather boring subgame. There are hardly any winning positions (about 0.005%) and even fewer losing positions. But the wins that can be reached are of a different nature than before. Wins are not achieved by removing an opponent's stone, but rather by blocking his pieces.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 159 | 3'225'409 | 29 | 9 | 8 |

The following example shows that even the longest winning sequences are easily recognized by a human.




White to move
win in 9 plies

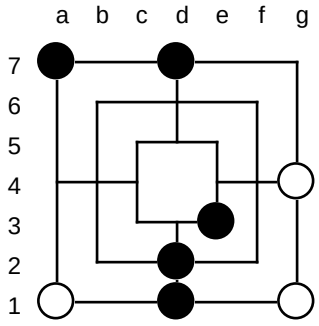
○ 1 d2-d1
● 2 a1-a4
  ● 2 a7-a4
  ● 2 g1-g4
  ● 2 g7-g4
○ 3 b6-b4
● 4 g1-g4
  ● 4 g7-g4
  ● 4 a4-a1
○ 5 e4-f4
● 6 a4-a1
  ● 6 g4-g1
○ 7 b4-a4
● 8 g4-g1
○ 9 f4-g4

## 4.4. 5-3 Subgame

The 5-3 subgame is similar to the 4-3 subgame. The player with 5 stones has many winning positions, but since they are at such shallow depths, it seems unlikely that they will be reached in actual play. The player with 3 stones has interesting winning sequences, but there are so few of them that this subgame is essentially a tie.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 580'660 | 1'999'730 | 0 | 3 | - |
| Black To Move | 6'301 | 2'564'412 | 9'677 | 31 | 2 |

The following example shows that to win this subgame, the opponent must first be reduced to 3 stones. This suggests a method of winning a drawn position against an imperfect opponent: the player with 3 stones should try to reduce his opponent to 3 stones, because correct 3-3 (or 4-3) play is more difficult than correct 5-3 play.



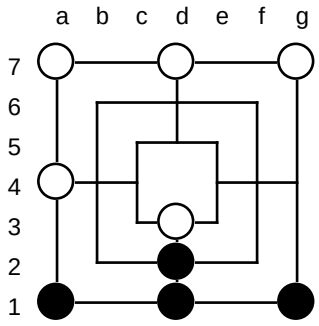

White to move
win in 31 plies

○ 1 a1-g7() xd1
● 2 a7-a4
  ● 2 d2-f2
○ 3 g4-a7
● 4 a4-b4
  ● 4 d2-b2
○ 5 g7-d1
  ○ 5 a7-d1
  ○ 5 a7-a1
● 6 d7-g7
○ 7 a7-a1() xd2
● 8 e3-a7
  ● 8 b4-a7
○ 9 d1-d7
● 10 b4-d1
○ 11 a1-d6
● 12 d1-d5
○ 13 g1-b6
● 14 g7-f6
  ● 14 a7-f6
○ 15 d6-b2
● 16 a7-b4
  ● 16 d5-b4
○ 17 b6-f2
● 18 d5-d2
○ 19 b2-d6
● 20 b4-d5
  ● 20 d2-d5
○ 21 d6-a7
● 22 f6-g7
  ● 22 d5-g7
○ 23 d7-a4
  ○ 23 d7-f6
● 24 g7-a1
  ● 24 d5-a1
○ 25 a7-b4
  ○ 25 a7-f6
  ○ 25 a4-f4
● 26 a1-c4
  ● 26 d5-c4
○ 27 a4-b6
  ○ 27 a4-f6
● 28 c4-b2
  ● 28 d5-b2
  ● 28 d2-b2
○ 29 b4-f6
● 30 b2-d6
  ● 30 b2-f4
  ● 30 d5-d6
  ● 30 d5-f4
  ● 30 d2-f4
  ● 30 d2-d6
○ 31 b6-f4() xd6

## 4.5. 5-4 Subgame

The 5-4 subgame is also basically a tie. All possible wins are reached by blocking the opponent.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 9'889 | 10'300'599 | 8 | 29 | 4 |
| Black To Move | 51 | 10'308'935 | 1'510 | 5 | 28 |



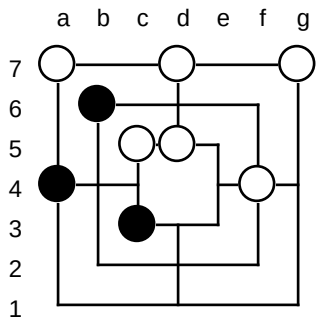

White to move
win in 29 plies

○ 1 g7-g4
● 2 d2-b2
  ● 2 d2-f2
○ 3 d3-d2
● 4 b2-b4
○ 5 d7-d6
● 6 b4-b6
  ● 6 b4-c4
  ● 6 b4-b2
○ 7 a7-d7
  ○ 7 d6-f6
  ○ 7 d6-d5
● 8 b6-b4
○ 9 d6-d5
  ○ 9 d6-b6
  ○ 9 d6-f6
● 10 b4-c4
○ 11 d5-c5
  ○ 11 d7-d6
● 12 c4-c3
○ 13 d7-d6
● 14 c3-d3
○ 15 d6-d5
  ○ 15 d6-f6
● 16 d3-e3
○ 17 d5-e5
● 18 e3-e4
○ 19 c5-d5
● 20 e4-e3
○ 21 e5-e4
● 22 e3-d3
○ 23 d5-c5
● 24 d3-c3
  ● 24 d3-e3
○ 25 c5-c4
● 26 c3-d3
○ 27 c4-c3
  ○ 27 e4-e3
● 28 d3-e3
○ 29 c3-d3

## 4.6. 6-3 Subgame

The 6-3 subgame is similar to the 5-3 subgame. But thanks to the additional stone, White cannot lose. The possible wins are at shallow depths, but strangely enough humans seem to find optimal play rather difficult.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 2'752'371 | 4'122'949 | 0 | 7 | - |
| Black To Move | 0 | 6'683'320 | 192'000 | - | 6 |



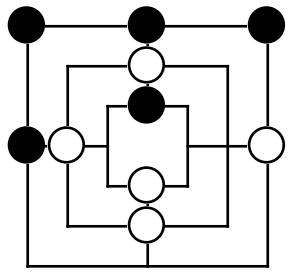

White to move
win in 7 plies

○ 1 g7-g4
● 2 c3-g7
  ● 2 b6-g7
  ● 2 a4-g7
○ 3 d7-d6
● 4 g7-d7
  ● 4 b6-d7
  ● 4 a4-d7
○ 5 d5-e5
● 6 b6-d5
  ● 6 d7-d5
  ● 6 d7-e4
  ● 6 b6-e4
  ● 6 a4-d5
  ● 6 a4-e4
○ 7 e5-e4() xd7

## 4.7. 5-5 Subgame

All wins are achieved by blocking the opponent.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 28'819 | 30'881'800 | 4'295 | 57 | 56 |



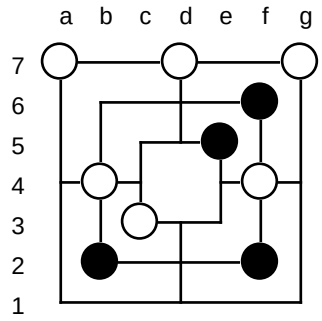
White to move
win in 57 plies

○ 1 d3-c3
● 2 d5-c5
○ 3 c3-c4
● 4 c5-d5
○ 5 d2-f2
● 6 d5-e5
○ 7 f2-f4
  ○ 7 c4-c3
● 8 e5-e4
○ 9 c4-c3
● 10 e4-e5
○ 11 f4-e4
● 12 e5-d5
○ 13 e4-e5
  ○ 13 c3-c4
● 14 d5-c5
○ 15 c3-c4
● 16 c5-d5
○ 17 c4-c5
● 18 a4-a1
○ 19 b4-a4
● 20 a1-d1
○ 21 c5-c4
● 22 d1-a1
  ● 22 d1-d2
  ● 22 d1-g1
  ● 22 d5-c5
○ 23 e5-e4
  ○ 23 c4-b4
● 24 a1-d1
  ● 24 d5-c5
  ● 24 d5-e5
○ 25 c4-b4
● 26 d1-a1
  ● 26 d1-d2
  ● 26 d1-g1
  ● 26 d5-c5
  ● 26 d5-e5
○ 27 b4-b6
  ○ 27 e4-f4
● 28 d5-e5
  ● 28 d5-c5
  ● 28 a1-d1
○ 29 e4-f4
● 30 e5-d5
  ● 30 e5-e4
  ● 30 a1-d1
○ 31 f4-f6() xd5
● 32 a1-d1
○ 33 b6-b4
● 34 d1-d2
○ 35 f6-f4
  ○ 35 b4-c4
  ○ 35 b4-b2
● 36 d2-d3
○ 37 b4-b2
  ○ 37 b4-c4
● 38 d3-c3
○ 39 f4-e4
  ○ 39 b2-d2
● 40 c3-d3
○ 41 e4-e5
  ○ 41 b2-d2
● 42 d3-d2
  ● 42 d3-c3
  ● 42 d3-e3
○ 43 e5-d5
● 44 d2-f2
  ● 44 d2-d3
○ 45 b2-d2
● 46 f2-f4
○ 47 d5-e5
● 48 f4-e4
○ 49 d2-d3
● 50 e4-f4
○ 51 e5-e4
● 52 f4-f2
○ 53 d3-d2
● 54 f2-f4
○ 55 d2-f2
● 56 f4-f6
○ 57 e4-f4

## 4.8. 6-4 Subgame

This is the subgame that to date has the longest winning sequence, namely 157 plies. The tests I have performed indicate that finding a winning sequence is clearly beyond human ability.

| | # Wins | # Draws | # Losses | Max Win Ply | Max Loss Ply |
|---|---|---|---|---|---|
| White To Move | 5'985'293 | 19'780'495 | 4 | 157 | 2 |
| Black To Move | 22 | 24'115'798 | 1'649'972 | 3 | 156 |




White to move
win in 157 plies

Unfortunately, the winning sequence is much too elaborate to be shown here, but can be obtained from the author.

# 5. Conclusion

One purpose of this work was to better understand the abilities and limitations of retrograde analysis. Using the improvements described earlier and other Macintosh specific adjustments, the speed of the retrograde calculations was increased by more than a factor 10. Further work will focus on generalizing the experience gained in Nine Men's Morris to other applications of retrograde analysis.

A second, fairly long term, goal of this work is the complete analysis of Nine Men's Morris. Whether or not this can be achieved in the near future depends a great deal on the game itself. If Nine Men's Morris is a draw, then many or even most positions will be draws and the run-length encoding scheme will prove effective in both speeding calculations and in reducing memory space requirements. At present I seem to be approaching the limits imposed by the Macintosh. In the future I will either have to port my code to a faster machine or speed up my calculations. A further speedup may be achieved by not differentiating between the number of plies required to win, but rather to classify positions only as wins,losses or draws.

# 6. References

[Gasser 90] Gasser, R., "Heuristic Search and Retrograde Analysis: their application to Nine Men's Morris", Diploma thesis, ETH Zürich, February 1990.

[Herik 86] van den Herik H.J. and Herschberg I.S.,"A data base on data bases", ICCA Journal 9(1), p. 29 (March 1986).

[Herik 89] van den Herik H.J. and Herschberg I.S.,"The 50-move rule revisited", ICCA Journal 12(3), p. 192 (September 1989).

[Marris 89] Marris, C.A.,"Compressing a chess-endgame database", ICCA Journal 12(1), p. 22 (March 1989).

[Müller 87] Müller, R.F., "Mühle", ECON Taschenbuch Verlag, October 1987.