

---

# **Playing Nine Men's Morris: An AI Agent Using Adversarial Search**

*Dushime Mudahera Richard*

*Masters in Data Science*

*Faculty of Mathematics, Natural Sciences, and Information Technologies*

*University of Primorska*

---

# Introduction & Background

Nine Men's Morris is a classic two-player strategy game

Three phases: placing, moving, flying

Goal: form "mills" (three in a row) to capture opponent pieces

Why game AI? Good testbed for search algorithms and strategic reasoning



## Objectives

Build an intelligent AI agent for Nine Men's Morris

Use adversarial search (Minimax + Alpha-beta Pruning)

Support multiple difficulty levels

Measure and analyze AI performance

## Game Mechanics & Phases

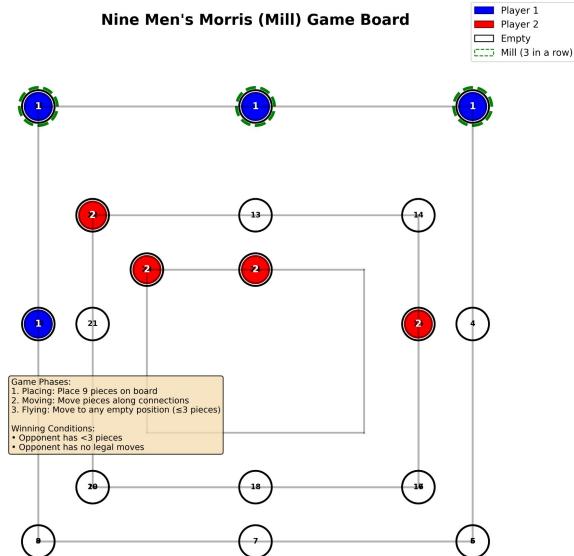
Placing phase: Players alternate, placing pieces

Moving phase: Players slide pieces to adjacent positions

Flying phase: When player has 3 pieces, can move anywhere

Mills allow a player to remove an opponent's piece

Nine Men's Morris (Mill) Game Board



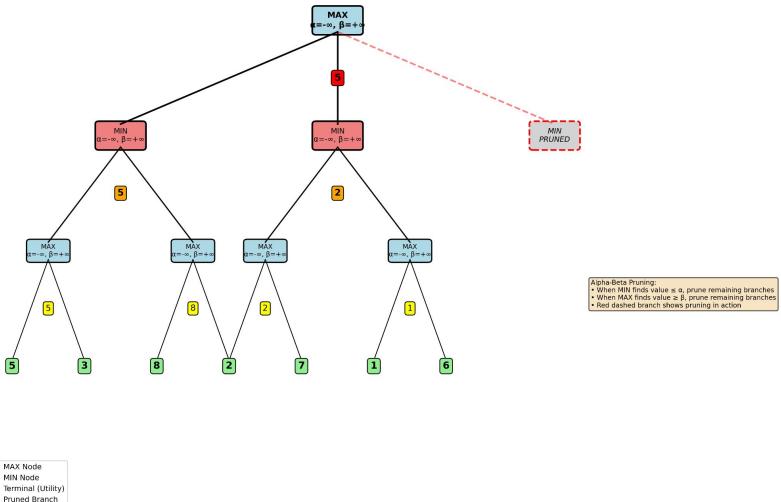
# AI Algorithm

Minimax: Recursively simulates all possible moves and countermoves

Alpha-beta pruning: Skips unnecessary branches, accelerates search

Depth limiting: Ensures fast, real-time play

Minimax Algorithm with Alpha-Beta Pruning



# Utility Function Design

Evaluates board positions using:

Piece count

Mills (3 in a row)

Mobility (number of possible moves)

Threats (potential mills)

Phase bonus

Weighted sum produces total score

## Utility Function Components

$$\text{Utility} = \text{Piece Score} + \text{Mill Score} + \text{Mobility Score} + \text{Phase Bonus} + \text{Threat Score}$$

All scores evaluated from Player 1 perspective (higher = better)

Piece Count	$(\text{my\_pieces} - \text{opp\_pieces}) \times 10$ Difference in pieces on board	Weight: 10.0
-------------	---	--------------

Mill Count	$(\text{my\_mills} - \text{opp\_mills}) \times 50$ Number of completed mills (3 in a row)	Weight: 50.0 (highest)
------------	--	------------------------

Mobility	$(\text{my\_moves} - \text{opp\_moves}) \times 2$ Number of legal moves available	Weight: 2.0
----------	--	-------------

Phase Bonus	+5 if in placing phase Early game advantage	Weight: 5.0
-------------	--	-------------

Threat Detection	$(\text{my\_threats} - \text{opp\_threats}) \times 10$ Potential mills (2 pieces, 1 empty)	Weight: 10.0
------------------	---	--------------

Example Calculation:  
Piece:  $(7-6) \times 10 = +10$  | Mill:  $(1-0) \times 50 = +50$  | Mobility:  $(5-4) \times 2 = +2$  | Phase: +5 | Threat:  $(2-1) \times 10 = +10$   
Total Utility =  $10 + 50 + 2 + 5 + 10 = +77$



## Difficulty Levels & Implementation

three difficulties: Easy, Medium, Hard using famnit-gym

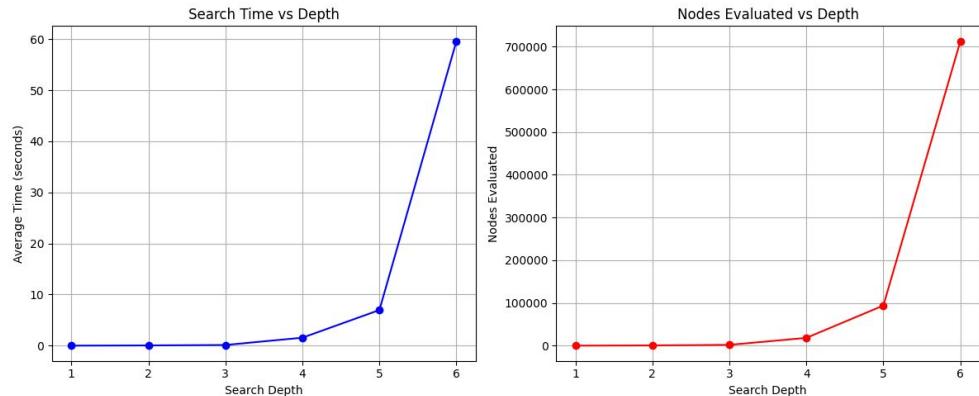
Difficulty	Depth	Randomness
Easy	2	30%
Medium	4	15%
Hard	6	0%

## Depth Analysis & Key Results

Search depth vs time: Move time grows steeply at higher depths (>6 is impractical)

Nodes evaluated: Alpha-beta pruning cuts down search space by ~60–80%

Best for play: Depth 5–6 gives good moves in under 5 seconds



# Tournaments & Self-Play Analysis

AI agents compete at different levels

```
(famnit) rdm@rdm nine_mens_morris_mill_game % python3 examples/self_play.py
Running self-play analysis (Hard AI vs Hard AI)...
This may take a while...
```

Self-Play Results:

```
-----  
Total games: 4  
Draws: 0  
Draw rate: 0.00%  
Average game time: 804.49s
```

Note: With perfect play, games should end in draws.  
Draw rate of 0.00% indicates AI quality.

```
(famnit) rdm@rdm nine_mens_morris_mill_game %
```

```
Problems Output Debug Console Terminal Ports
(famnit) rdm@rdm nine_mens_morris_mill_game % python3 examples/tournament.py
Starting tournament...
=====
Game 1/2... AI1 (easy) wins
Game 2/2... AI1 (easy) wins
Game 1/2... AI2 (hard) wins
Game 2/2... AI2 (hard) wins
Game 1/2... AI1 (medium) wins
Game 2/2... AI2 (hard) wins
==== Tournament Results ===
easy vs medium:
  AI1 wins: 2
  AI2 wins: 0
  Draws: 0
  Avg game time: 26.37s

easy vs hard:
  AI1 wins: 0
  AI2 wins: 2
  Draws: 0
  Avg game time: 961.92s

medium vs hard:
  AI1 wins: 1
  AI2 wins: 1
  Draws: 0
  Avg game time: 780.21s

Results saved to results/statistics/tournament_results.json
(famnit) rdm@rdm nine_mens_morris_mill_game %
```

---

## Conclusions & Discussion

Alpha-beta pruning enables fast, deep searches

Custom utility function ensures strategic strength

Difficulty levels make game fun for all users

Self-play confirms near-optimal play

---

Thank You All

Questions?

Else: All the best