ALLEGHENY COLLEGE COMPUTER SCIENCE Department

Intelligent Monte-Carlo Tree Search for Perfect Information Games

Lucas Hawk

Presentation Overview

1. Preliminaries
2. Implementation of Game-Playing Framework and Agents
3. Experimental Results
   - Go Experiments
   - Hex Experiments (New!)
4. Future Work and Experimentation

Motivation

Perfect information games provide a very good environment for testing artificial intelligence.

Motivation

Perfect information games provide a very good environment for testing artificial intelligence.

- Extremely controllable
- Clear goal for players
- Simple programming, complex decision-making
- Easy to benchmark agents both against each other and against humans

Monte-Carlo Tree Search (MCTS)

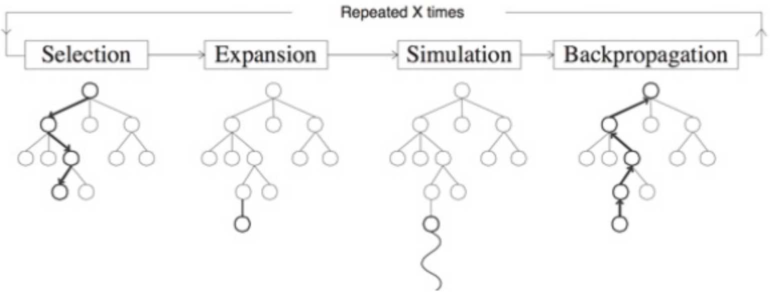
MCTS is currently the golden standard for intelligent game-playing agents

Monte-Carlo Tree Search (MCTS)

MCTS is currently the golden standard for intelligent game-playing agents

- "Anytime algorithm"
- Performance is game-independant — MCTS does not use heuristics
- Converges to optimal strategy

Monte-Carlo Tree Search (MCTS)



Why Go and Hex?

Go and Hex are commonly used games in existing research

Why Go and Hex?

Go and Hex are commonly used games in existing research

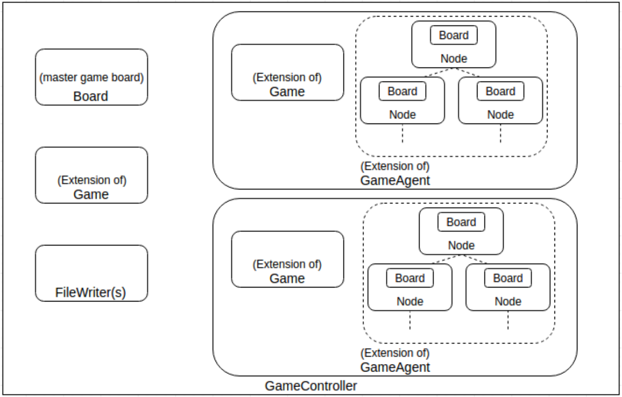
- Very small, simple ruleset
- Very widely played
- Massive branching factor

Why Go and Hex?

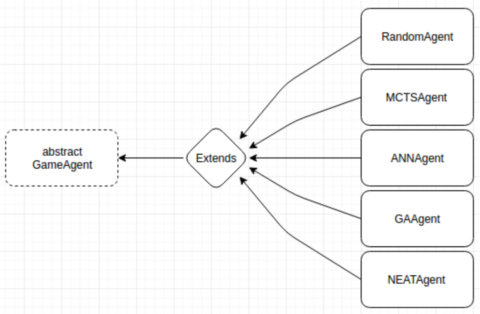
Go and Hex are commonly used games in existing research

- Very small, simple ruleset
- Very widely played
- Massive branching factor
  - 19x19 Go has 2×10^{170} playouts, making it one of the most computationally complex board games ever created
  - Hex has a similar branching factor

Framework Implementation

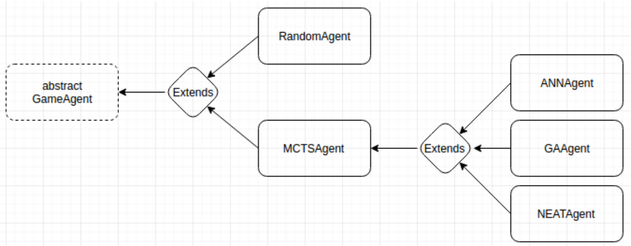


Agent Implementation



Agent Inheritance Structure

Agent Implementation



How it Should Have Worked

Intelligent Agents

- ANNAgent

Intelligent Agents

- ANNAgent
- GAAgent

Intelligent Agents

- ANNAgent
- GAAgent
- NEATAgent

Experiments

- Compare performance of each agent when given a maximum time-allowed-per-move

Experiments

- Compare performance of each agent when given a maximum time-allowed-per-move
- Limiting time rather than MCTS iterations gives a more accurate indication of the tradeoff between complexity and performance

Experiments

Experimental Setup

Experiments

Experimental Setup

- Each pair of agents competed head-to-head in Hex and Go

Experiments

Experimental Setup

- Each pair of agents competed head-to-head in Hex and Go
  - 9x9, 11x11, 14x14 Hex — 5x5, 7x7, 9x9, 11x11, 13x13 Go
  - Time-allowances of 500ms, 1000ms, 2000ms, 4000ms, 8000ms

Experiments

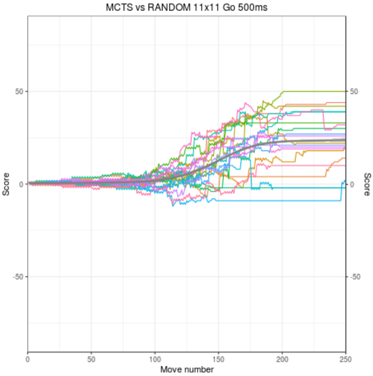
Experimental Setup

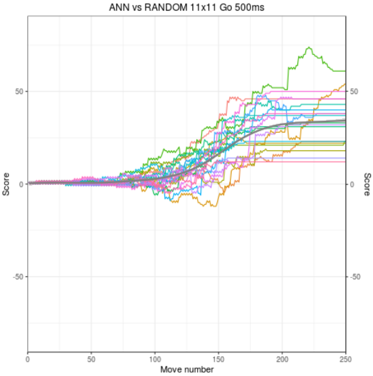
- Each pair of agents competed head-to-head in Hex and Go
  - 9x9, 11x11, 14x14 Hex — 5x5, 7x7, 9x9, 11x11, 13x13 Go
  - Time-allowances of 500ms, 1000ms, 2000ms, 4000ms, 8000ms
- 20 games of each combination of board size and time-allowance

Go Results vs RandomAgent

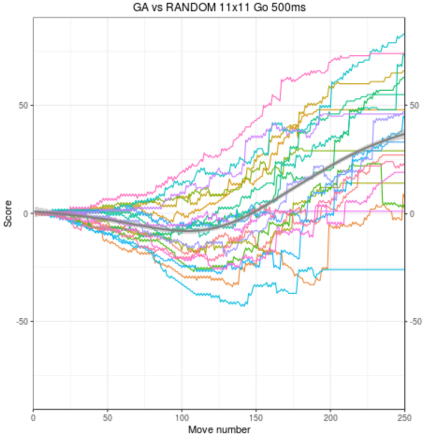
Agent (vs Random)	win rate	mean score (turn 250)
MCTS	98.8	33.1
ANN	98.4	32.5
GA	98.2	31.516

Go Results vs RandomAgent

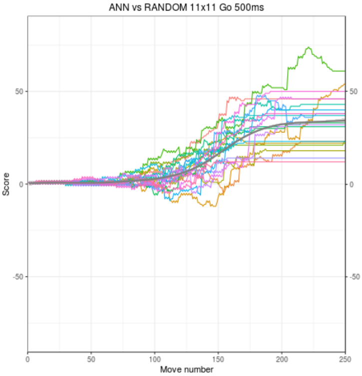


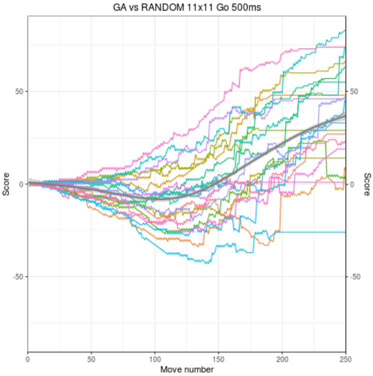


Go Results vs RandomAgent



Go Results vs RandomAgent





Go Results vs MCTSAgent

Agent (vs MCTS)	win rate	mean score (turn 250)
ANN	61.4	4.31
GA	56.9	1.8

Go Results vs MCTSAgent

Mean Score by time-allowance

time-allowance	GAAgent mean score	ANNAgent mean score
1000	-2.475	5.875
2000	-0.125	5.0
4000	2.65	4.21
8000	4.425	4.35

Go Results vs MCTSAgent

Mean Score by board size

board size	GAAgent mean	ANNAgent mean
5	2.86	1.41
7	2.23	2.18
9	1.81	4.39
11	-2.45	9.31

Go Results Head2Head

GAAgent vs ANNAgent mean score by time-allowance and board size

time-allowance	mean score	board size	mean score
500	-1.53	5	1.8
1000	-1.7	7	2.47
2000	-5.16	9	-1.26
4000	1.41	11	-5.8
8000	3.5	—	—

Go Results Summary

- GAAgent shows a positive correlation with time-allowance, and a negative correlation with board size

Go Results Summary

- GAAgent shows a positive correlation with time-allowance, and a negative correlation with board size
- ANNAgent performance is more independant of time-allowance, board size than standard MCTSAgent

Go Results Summary

- GAAgent shows a positive correlation with time-allowance, and a negative correlation with board size
- ANNAgent performance is more independant of time-allowance, board size than standard MCTSAgent
- NEATAgent results still to come

Hex Results (new!)

- Against the RandomAgent, both ANNAgent and MCTSAgent won 100% of their games

Hex Results (new!)

- Against the RandomAgent, both ANNAgent and MCTSAgent won 100% of their games
- GAAgent only won about 60% of its games against RandomAgent

Hex Results vs MCTSAgent

Agent (vs MCTS)	win rate
ANN	66.6
GA	19.3

Hex Results Head2Head

- ANNAgent won 88% of the 300 total games played

Hex Results Head2Head

- ANNAgent won 88% of the 300 total games played
- No trends regarding board size or time-allowance — ANNAgent was simply dominant all around

Hex Results Summary

- ANNAgent seems to perform similarly as in Go, providing a moderate performance boost with little correlation to board size or time-allowance

Hex Results Summary

- ANNAgent seems to perform similarly as in Go, providing a moderate performance boost with little correlation to board size or time-allowance
- GAAgent SUCKS (with the heuristic features we used)

Hex Results Summary

- ANNAgent seems to perform similarly as in Go, providing a moderate performance boost with little correlation to board size or time-allowance
- GAAgent <u>SUCKS</u> (with the heuristic features we used)
  - Highlights the agent's dependency on a valid, high-performance heuristic

Results Summary

- Even with a “lightly trained” network, an agent using ANN pruning is able to outperform an agent using standard MCTS by 20-25%
- ANN pruning performance seems to be independant of board size and time-allowance

Results Summary

- An agent utilizing a rapidly evolving GA is *capable* of outperforming an agent using standard MCTS, or even the agent using ANN pruning

Results Summary

- An agent utilizing a rapidly evolving GA is *capable* of outperforming an agent using standard MCTS, or even the agent using ANN pruning
- Agent's performance strongly correlates with both board size and time-allowance

Results Summary

- An agent utilizing a rapidly evolving GA is *capable* of outperforming an agent using standard MCTS, or even the agent using ANN pruning
- Agent's performance strongly correlates with both board size and time-allowance
- Agent's performance is almost entirely dependant on a well-defined heuristic strategy of the game it is playing

Future Work

By the end of the semester

- NEATAgent evaluation

Future Work

By the end of the semester

- NEATAgent evaluation
- Analyze differences in computational complexity by looking at number of iterations per second each agent is able to complete

Future Work

By the end of the semester

- NEATAgent evaluation
- Analyze differences in computational complexity by looking at number of iterations per second each agent is able to complete

Beyond the end of the semester

- Go board resolution optimization
- Effect of better-trained networks on ANNAgent performance