An Investigation of AI Tree Search Methods and Their Effectiveness at Playing the Card Game Gwent

Jason Chalom (711985)[1]
Supervisor: Professor Clint Van Alten[1]
[1]University of the Witwatersrand, South Africa

UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG

Objectives

1. **Simulate** the Game of Gwent
2. Build **AI Agents** for Gwent
3. **Empirically Compare** AI agents
4. If Possible **Optimize** Agents Using **Parallel Programming**

Motivation

Artificial Intelligence (AI):

- Not all games are the same [1]
- Games can be **difficult** for AI to solve in **reasonable time** [1]
- The use of data structures and sampling techniques to **model games**.

The Domain (Gwent):

- **New** domain
- **Interesting** rules
- **Hidden information** (unknown hands) [2]
- It's a **fun** game

PLAYER 1
Siege Combat
Ranged Combat
Close Combat
Weather Cards
Close Combat
Ranged Combat
Siege Combat
PLAYER 2

Example game board [2]

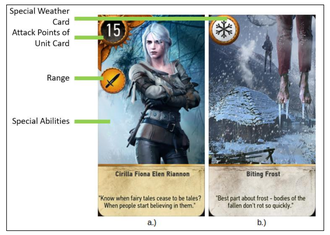
References

[1] Stuart Russell and Peter Norvig. Artificial Intelligence: A Modern Approach (3rd Edition). Pearson, 2009.
[2] CD PROJEKT S.A., CD PROJEKT RED. GWENT Guide. pages 1–4. Poland, 2015.
[3] Chaslot, Guillaume and Bakkes, Sander and Szita, Istvan and Spronck, Pieter. Monte-Carlo Tree Search: A New Framework for Game AI. In Christian Darken and Michael Mateas, editors, AIIDE. The AAAI Press, 2008.

Background

Gwent

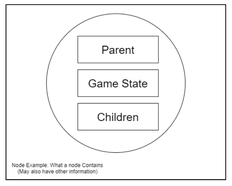
- Rules (Excluding deck building) [2]:
  - 2 Player **turn-based**
  - Best of **three** rounds
  - Players can **pass** a round
  - Only draw **10 random cards** before 1st round
  - Units are placed in **specific rows**
  - Round **wins** calculated from total **attack points**

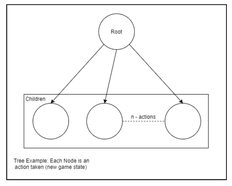


Special Cards	Abilities
Weather - Reduce attack points to 1 (for each card in a specific row)	Spy – add to opponent side but draw 3 new cards into hand
Commander's Horn – doubles attack points of row	Medic – revive discarded unit cards
Hero – a special unit which is not affected by other special cards	Tight bond – multiplies sibling card attack points by number of cards on board

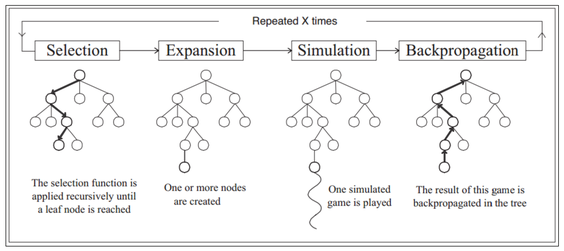
Monte-Carlo Tree Search (MCTS) [3]

- Uses game-state trees





- Has 4 phases (Adapted from references [3])



Implementation

Random Agents:

- **3 Agents** using pseudo-random number generators
  - Mersenne Twister
  - Marsaglia's Xorshift
  - Fast Rand

Heuristic Agent:

- Pick a card with the **maximum reward** from the player's hand using this linear combination:

$$reward = 0.2 * (AttackPoints - EnemyAttackPoints) + 0.2 * (HandSize - EnemyHandSize) + 0.6 * (RoundWins - EnemyRoundWins)$$

Random Rollout:

- **Iterate** through hand
- **simulate** n-number of games with each card
- Pick card with **most wins**
- Compute rollouts in **parallel**

MCTS Agent:

- **Selection** initially samples randomly based on hand
- **Expansion** based on selection
- **Simulation** uses random rollout
- **Back-propagation** reward:
  - Win – 1.0
  - Draw = 0.5
  - Loss = -1.0

MCTS2 Agent:

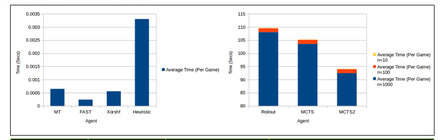
- **Prunes** away potential **paths** which are worse than the best
- Found during **back-propagation**

Results

Method:

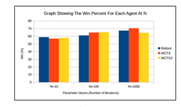
- AIs are pitted against each other in a **round-robin** style tournament
- Each "game" is run **twice** with the players and decks swapped (for x – runs)

Agent Times:

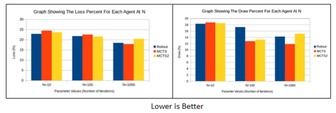


Agent	N=10 (secs)	N=100 (secs)	N=1000 (secs)
Rollout	0.111459	1.47849	108.0554215
MCTS	0.092066	1.52643	103.5874541
MCTS2	0.089473	1.50069	92.45517475

Agents VS Random (Over all random Agents):



Higher is Better



Lower is Better

Table of Agents VS Random:

Agents (P1 vs P2)	P1 Wins (%)	P2 Wins (%)	Draws (%)
Heuristic vs MT	31.8	34.73	33.46
Heuristic vs Fast Rand	99.34	0.48	0.18
Heuristic vs XorShift	31.97	34.67	33.36
Rollout vs MT	51.875	28.125	20
Rollout vs Fast Rand	97.5	0	2.5
Rollout vs XorShift	55	28.125	16.875
MCTS vs MT	63.83	25.45	12.27
MCTS vs Fast Rand	99.09	0	0.91
MCTS vs XorShift	60.45	25	14.55
MCTS2 vs MT	49.5	32	18.5
MCTS2 vs Fast Rand	99	0	1
MCTS2 vs XorShift	54	25	21

Table of Agents VS Agents:

Agents (P1 vs P2)	N	P1 Wins(%)	P2 Wins(%)	Draws(%)
Heuristic vs Rollout	10	33	36	31
Heuristic vs MCTS	10	21	40	39
Heuristic vs MCTS2	10	26	42	32
Heuristic vs Rollout	100	34	36	30
Heuristic vs Rollout	1000	35	33	32
Heuristic vs MCTS	100	28	54	18
Heuristic vs MCTS	1000	16	63	21
Heuristic vs MCTS2	100	27	54	19
Rollout vs MCTS	10	35	33	32
Rollout vs MCTS2	10	29	39	32
Rollout vs MCTS	100	26	58	16
Rollout vs MCTS	1000	26	51	23
Rollout vs MCTS2	100	16	64	20
MCTS vs MCTS2	10	46	40	14
MCTS vs MCTS2	100	51	39	10

Future Work

1. Logic based agent
2. Hybrid MCTS (Mixture with Minimax algorithm)
3. More Complex Rules for Gwent
4. Machine learning and statistical sampling techniques