Rolling Horizon Co-evolution in Two-player General Video Game Playing

Charles Ringer, Cristiana Pacheco, Cristina Dobre and Diego Perez-Liebana

EPSRC
Engineering and Physical Sciences Research Council

Goldsmiths
UNIVERSITY OF LONDON

Queen Mary
University of London

i G G i

Overview

- GVGAI
  - Framework and Competition
  - Two-Player Track
- Algorithms
  - Monte Carlo Tree Search (MCTS)
  - Rolling Horizon Evolutionary Algorithms (RHEA)
  - **Rolling Horizon Co-evolutionary Planning (RHCP)**
- Experimentation
- Results and Discussion
- Future Work

GVGAI (Part 1)

- Many competitions are used as a way to test AI.
  - Super Mario Bros, Go, Starcraft, etc.
- Some framework were developed to tackle General Game Play (GGP).
  - Open AI Gym, Arcade Learning Environment, etc.
- Our focus for this work was in the General Video Game AI (GVGAI).

GVGAI (Part 2)

- Uses forward model unlike other frameworks.
- Agents already created.
- Over 80 publicly available games.
- Based on arcade/puzzle like games.
- Single-player tracks and multi-player track.

Two-Player Track

Main focus of our project: 2-player track.

- Actions are supplied by all agents.
- Perfect for guessing opponent’s moves.
- There are both cooperative and competitive games.
- More interactions, score systems and end conditions.

Example:

GVGAI

Existing approaches

Many approaches have been used and most popular are:

- Monte Carlo Tree Search (MCTS)
- Rolling Horizon Evolutionary Algorithms (RHEA)

Monte Carlo Tree Search (MCTS)

- Overall, it performs 4 steps in each iteration:
  - Selection of a node.
  - Expands its children.
  - Simulation - with a random sequence of moves.
  - Backpropagation – of found reward, back through the tree to update its parent.
- This is repeated until a predefined computational budget is reached.

Rolling Horizon Evolutionary Algorithms (RHEA)

- Statistical Forward Planning Algorithms.
- Each gene in an individual’s chromosome represents an action in the plan.
- Initially, it forms a random population of sequencial actions.
- Evaluation is done by stepping through each action, applying it and using an heuristic function to calculate its fitness.
- A new population is evolved using genetic operators (Elitism, Crossover, Mutation, etc.)
- This is done until the computational budget is hit.

Rolling Horizon Co-evolutionary Planning

Why?

Rolling Horizon Co-evolutionary Planning (RHCP)

Motivation:

What if we think about our opponents moves as well?

Maybe this allows us to find action sequences to counter our opponents plans?

If we assume that our opponent is rational and that RHEA is a reasonable algorithm why not use it to model the opponent?

RHCP

- **Concept:**
  - Develop a plan for the agent and a “best guess” plan for the opponent.
  - Use this plan to evaluate our plan and vice versa.
- **Implementation:**
  - Based off of the default RHEA agent, each iteration we evolve a player plan and an opponent’s plan.

RHCP Overview

INIT:

Player plans = apply shift buffer or random sequences (if first move).
Opponent plans = apply shift buffer or random sequences (if first move).

WHILE BUDGET REMAINS:

Evaluate Player plans w/regards to Opponent plans.
Evolve Player plans.

Evaluate Opponent plans w/regards to Player plans.
Evolve Opponent plans.

RETURN:

First move of best player plan.

Player plan evolution - $\mu + \lambda$ Evolutionary Strategy

Population size: **5**

Sequence length: **20**

μ = **1** (selected through elitism)

λ = **4** (selected through tournament selection, t = 2)

Genetic Operators: **Random Mutation**

Shift Buffer: **On**

Opponent plan evolution - (1+1) RMHC

Used because it is less computation compared to $\mu + \lambda$

Population size: **1**

Sequence length: **20**

Genetic Operators: **Random Mutation**

Shift Buffer: **Off**

Experiment

Akka Arrh	Cooperative	Symmetric	Stochastic
Asteroids	Competitive	Symmetric	Stochastic
Capture the Flag	Competitive	Asymmetric	Deterministic
Cops N Robbers	Competitive	Asymmetric	Deterministic
Gotcha	Competitive	Asymmetric	Stochastic
Klax	Competitive	Symmetric	Stochastic
Samaritan	Competitive	Asymmetric	Deterministic
Sokoban	Cooperative	Symmetric	Deterministic
Steeplechase	Competitive	Symmetric	Deterministic
Tron	Competitive	Symmetric	Deterministic

10 games: variety of Cooperative/Competitive, Symmetric/Asymmetric, and Stochastic/Deterministic.

4 tested algorithms: RHCP, RHEA, MCTS and Random.

10 games, 5 levels, 2 sides (agents play both sides): 100 trials per pairing of agents, so 1000 matches per pair total.

Results

Game ID	RHCP vs RHEA	RHCP vs MCTS	RHEA vs MCTS	RHCP vs Rand.	RHEA vs Rand.	MCTS vs Rand.
0	0 - 0	1 - 1	3 - 3	0 - 0	0 - 0	1 - 1
1	49 - 45	9 - 85	7 - 87	85 - 9	92 - 6	100 - 0
2	56 - 45	6 - 97	3 - 98	88 - 12	90 - 11	100 - 0
3	39 - 31	3 - 66	4 - 70	59 - 8	57 - 13	69 - 1
4	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50
5	54 - 48	35 - 70	23 - 79	100 - 0	97 - 4	100 - 0
6	50 - 50	46 - 54	50 - 50	56 - 38	53 - 44	64 - 34
7	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0
8	1 - 1	1 - 3	4 - 3	2 - 1	3 - 4	12 - 2
9	61 - 39	6 - 94	10 - 90	98 - 2	98 - 2	100 - 0
Total	**360 - 309**	**157 - 520**	**154 - 530**	**538 - 120**	**540 - 134**	**596 - 88**

- All games played across the 6 pairings of agents.
- For each pair, their wins against each other per game.

Results

Game ID	RHCP vs RHEA	RHCP vs MCTS	RHEA vs MCTS	RHCP vs Rand.	RHEA vs Rand.	MCTS vs Rand.
0	0 - 0	1 - 1	3 - 3	0 - 0	0 - 0	1 - 1
1	49 - 45	9 - 85	7 - 87	85 - 9	92 - 6	100 - 0
2	56 - 45	6 - 97	3 - 98	88 - 12	90 - 11	100 - 0
3	39 - 31	3 - 66	4 - 70	59 - 8	57 - 13	69 - 1
4	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50
5	54 - 48	35 - 70	23 - 79	100 - 0	97 - 4	100 - 0
6	50 - 50	46 - 54	50 - 50	56 - 38	53 - 44	64 - 34
7	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0
8	1 - 1	1 - 3	4 - 3	2 - 1	3 - 4	12 - 2
9	61 - 39	6 - 94	10 - 90	98 - 2	98 - 2	100 - 0
Total	**360 - 309**	**157 - 520**	**154 - 530**	**538 - 120**	**540 - 134**	**596 - 88**

RHCP vs RHEA:

- Operates similarly, despite cost in computational time.
- But, overall, it won a significant amount more of games.

Results

Game ID	RHCP vs RHEA	RHCP vs MCTS	RHEA vs MCTS	RHCP vs Rand.	RHEA vs Rand.	MCTS vs Rand.
0	0 - 0	1 - 1	3 - 3	0 - 0	0 - 0	1 - 1
1	49 - 45	9 - 85	7 - 87	85 - 9	92 - 6	100 - 0
2	56 - 45	6 - 97	3 - 98	88 - 12	90 - 11	100 - 0
3	39 - 31	3 - 66	4 - 70	59 - 8	57 - 13	69 - 1
4	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50
5	54 - 48	35 - 70	23 - 79	100 - 0	97 - 4	100 - 0
6	50 - 50	46 - 54	50 - 50	56 - 38	53 - 44	64 - 34
7	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0
8	1 - 1	1 - 3	4 - 3	2 - 1	3 - 4	12 - 2
9	61 - 39	6 - 94	10 - 90	98 - 2	98 - 2	100 - 0
Total	**360 - 309**	**157 - 520**	**154 - 530**	**538 - 120**	**540 - 134**	**596 - 88**

RHCP/RHEA vs MCTS:

- MCTS remains the dominant algorithm winning over both.
- RHCP does seem to compete more but it is no match.

Results

Game ID	RHCP vs RHEA	RHCP vs MCTS	RHEA vs MCTS	RHCP vs Rand.	RHEA vs Rand.	MCTS vs Rand.
0	0 - 0	1 - 1	3 - 3	0 - 0	0 - 0	1 - 1
1	49 - 45	9 - 85	7 - 87	85 - 9	92 - 6	100 - 0
2	56 - 45	6 - 97	3 - 98	88 - 12	90 - 11	100 - 0
3	39 - 31	3 - 66	4 - 70	59 - 8	57 - 13	69 - 1
4	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50	50 - 50
5	54 - 48	35 - 70	23 - 79	100 - 0	97 - 4	100 - 0
6	50 - 50	46 - 54	50 - 50	56 - 38	53 - 44	64 - 34
7	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0	0 - 0
8	1 - 1	1 - 3	4 - 3	2 - 1	3 - 4	12 - 2
9	61 - 39	6 - 94	10 - 90	98 - 2	98 - 2	100 - 0
Total	**360 - 309**	**157 - 520**	**154 - 530**	**538 - 120**	**540 - 134**	**596 - 88**

RHCP/RHEA/MCTS vs Random:

- Understandably they all performed better than random.

Results

Game	RHEA	MCTS	Random
0	**0.185** (0.016)	0.165 (0.016)	0.166 (0.02)
1	0.169 (0.107)	0.160 (0.167)	**0.173** (0.15)
2	**0.223** (0.01)	0.201 (0.010)	0.2 (0.01)
3	**0.2** (0.032)	0.176 (0.031)	0.18 (0.033)
4	**0.418** (0.013)	0.202 (0.041)	0.199 (0.025)
5	0.375 (0.013)	0.332 (0.013)	**0.418** (0.013)
6	**0.201** (0.034)	0.180 (0.044)	0.181 (0.026)
7	**0.223** (0.009)	0.199 (0.009)	0.199 (0.009)
8	**0.186** (0.008)	0.166 (0.009)	0.168 (0.009)
9	0.245 (0.06)	**0.227** (0.045)	0.197 (0.102)
Avg	**0.223** (0.032)	0.201 (0.038)	0.208 (0.04)

The accuracy of RHCP was also measured.

- It shows fairly low accuracies with the highest ones (in bold) appearing more in games against RHEA and bellow 50%.

Discussion

- RHCP outperforms RHEA providing a small improvement.
- MCTS does remain as the winning algorithm.
    - Rolling Horizon agents are hyperparameter sensitive – results might improve upon testing a wider range of parameters.
- Predictions are fairly low for RHCP which won't help towards it beating this algorithm.
    - A possible reason could be the stochastic nature of the evolutionary algorithm couple with the small number of evolution iterations.

Conclusions

- Thinking about your opponent is great but can show only small improvements.
- This might be because:
  - Evaluations are initially based on a bad opponent plan.
  - We waste resources on the opponent’s plans.
  - It could over-think what a random agent is doing.
- Although, solutions for better results might lie in hyperparameter tuning, increase of resources or smaller rollouts with more generations.