Lily Jacobi

lwj0001@auburn.edu

COMP 5660 Fall 2021 Assignment 2c

Pac man v Map Co-evolution

Methodology

For the GPac design I decided to implement the gene as a wrapper class for the node class called parseTree. This way I was able to store important information (depth limit, set of sensor and input primitive, etc.) without having to store them in every node, as well as the functions pertaining to the tree. Parsimony pressure was implemented based on the size of the tree.

Species are assigned matches via shuffling both population lists then iterating through them, modulating the index by the length of the smaller list in case of size disparity, so the smaller list does not go out of bounds. This should be sufficiently random, and not assign more than on of the same individual to another unless it goes through the entire array twice. This also ensure that every individual is picked at least once. Support was added for fighting multiple opponents; however, this was computationally intense even matching twice, thus I did not use this functionality in the end product despite testing it thoroughly as two matches provided very little gain over one match, and three took too long. I instead opted to lower the population size of the pac man agents, so that they would have more diverse pool of opponents, though I am not sure how much this helped as it also reduces their evolution rate in a way.

Experiment Setup

New configurations were added for opponent count, operatorPrimitive, sensorPrimitives and constantRange. K-tournament with replacement was chosen for parent selection as it has low computational load, and is relatively fair. Truncation was chosen for survival due to having way lower computational load than k-tournament without replacement, while performing similarly. 1-point crossover was to chosen recombine the map gene due to low computational load. Binary string mutation was chosen to mutate the map gene as it was the only choice. Subtree crossover was chosen as it was the only choice for recombination of trees. Sub tree mutation was chosen as it was the only choice for mutation of trees.

My Config file is as follows:

height	20
width	35
samples	5
pill_spawn	linear
pill_density	0.05
fruit_prob	0.1
fruit_score	10
ghost_type	chase
penalty_coefficient	0.2
opponentCount	1

[pac_GP_configs]	
mu	60
num_children	40
mutation_rate	0.05
individual_class	treeGenotype
parent_selection	k_tournament_with_replacement
survival_selection	truncation
[pac_initialization_kwargs]	
depth_limit	5
operatorPrimitives	['+','-','*','/','RAND']
sensorPrimitives	['G', 'P', 'W', 'F', '#.#']
constantRange	[-10,10]
[pac_parent_selection_kwargs]	
k	5
[pac_recombination_kwargs]	
depth_limit	5
[pac_mutation_kwargs]	
depth_limit	5
[pac_survival_selection_kwargs]	
[map_EA_configs]	
mu	200
num_children	100
mutation_rate	0.2
individual_class	binaryGenotype
parent_selection	k_tournament_with_replacement
survival_selection	truncation
[map_initialization_kwargs]	
length	${fitness_kwargs:height}*${fitness_kwargs:width}
[map_parent_selection_kwargs]	
k	5
[map_recombination_kwargs]	
method	1-point crossover

height	${fitness_kwargs:height}
width	${fitness_kwargs:width}
[map_mutation_kwargs]	
[map_survival_selection_kwargs]	

Results

Species	Standard Deviation of Max per run	Mean of Max per run
Map	0	0
Pac Man	10.47218	28.96667

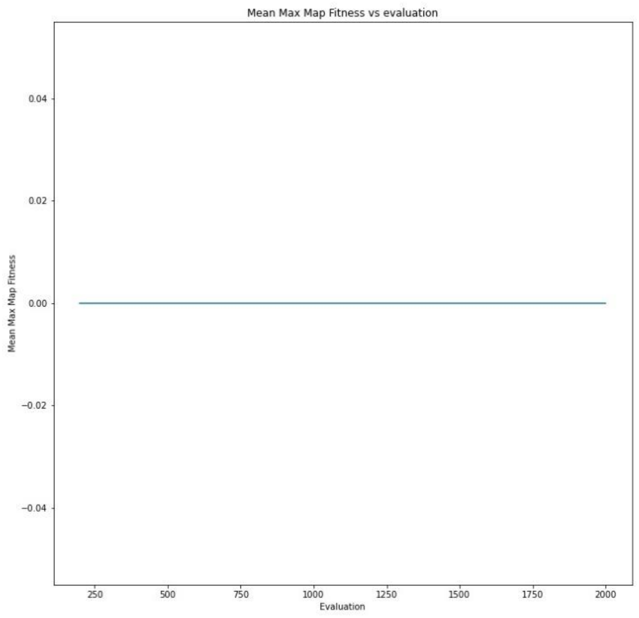


Figure 1 Mean Max Map Fit v Evals

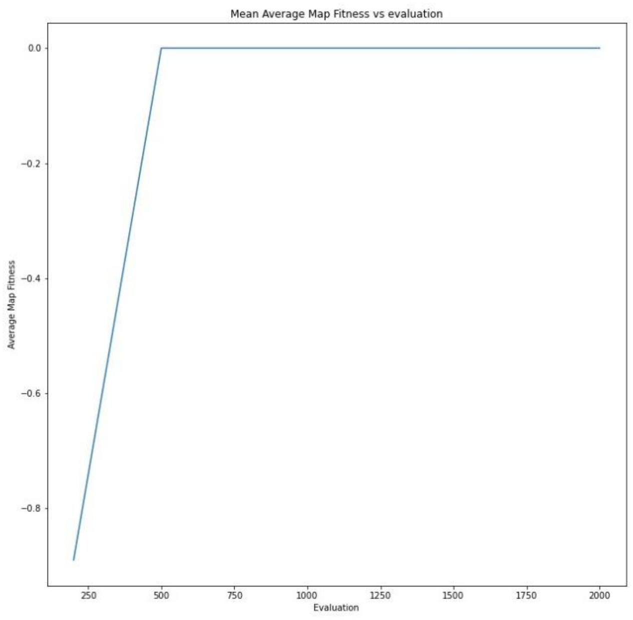


Figure 2 Mean Average Map Fit vs Evals

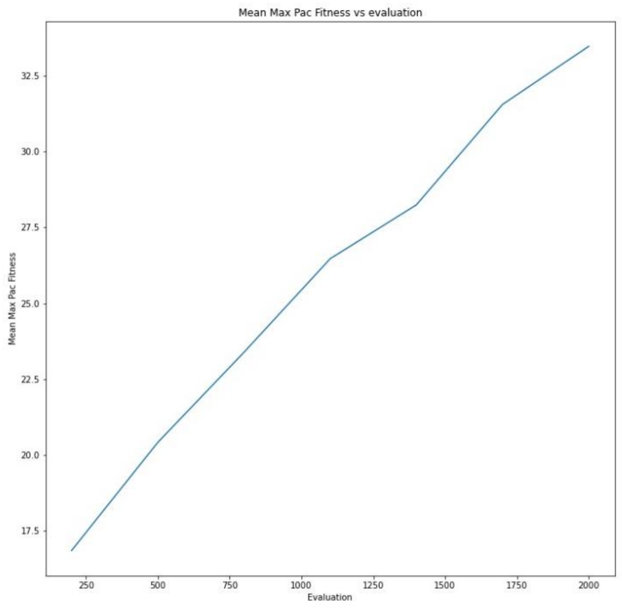


Figure 3 Mean Max Pac Fit vs evals

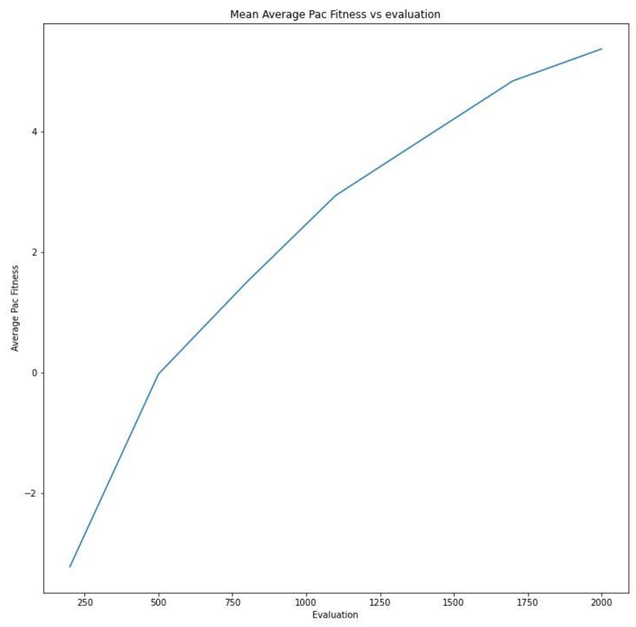


Figure 4 Mean Average Pac Fit vs evals

Discussion

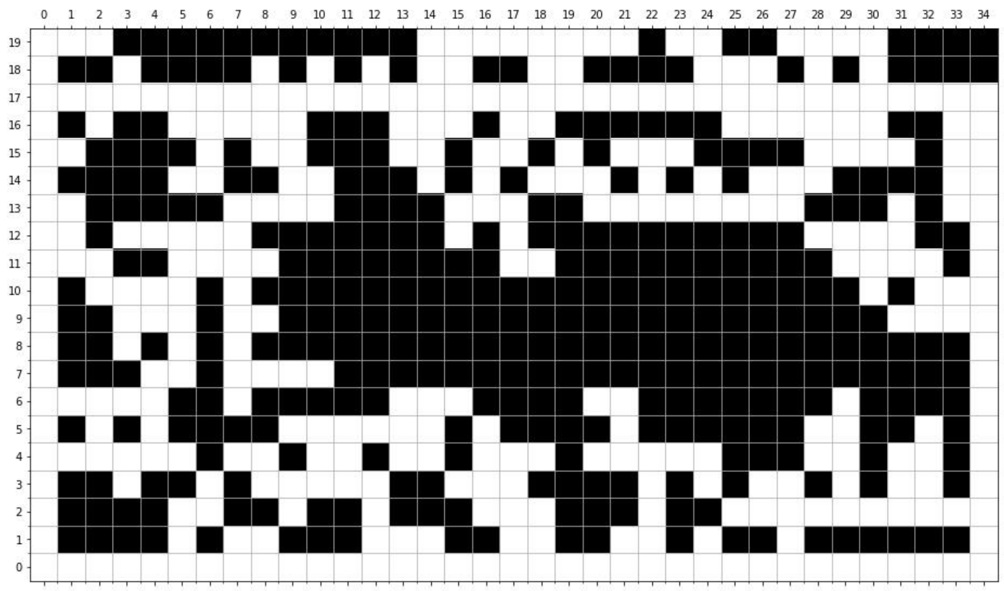
The pac man agents in this co-evolutionary method are extremely prone to getting stuck in dead ends. I noticed in fact, the maps seem to evolve to become a pocked web of dead ends to "trap" the pac man agents so they just sit there until the ghosts come and end the game. Both of the two exhibition matched I did between different full experiments resulted in this webby architecture, and the pac man agent immediately getting stuck, resulting in 0 score. I'm almost certain this is a problem stemming from not directly taking the walls into account when measuring distance to the closest fruit or pill, as it would result in a state score loss to exit the pitfall if the agent is intent on getting pills. The pac man controller still likes to avoid ghosts, in fact the games end by the ghosts getting close enough for the agent to care, and then running directly into them in an attempt to escape. So that quirk remains – it just is used against the controller by the maps developing in such a pocked manner.

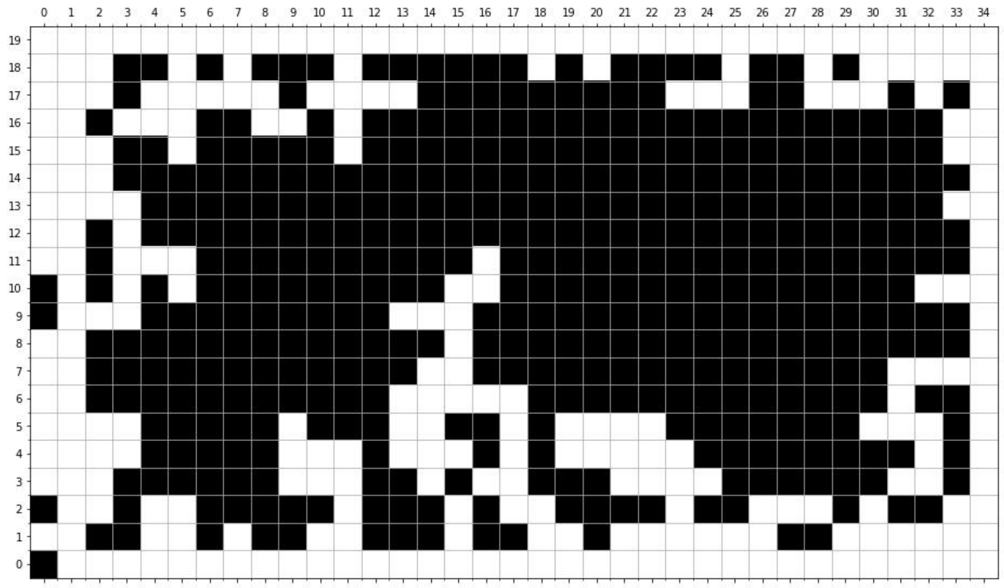
Conclusion

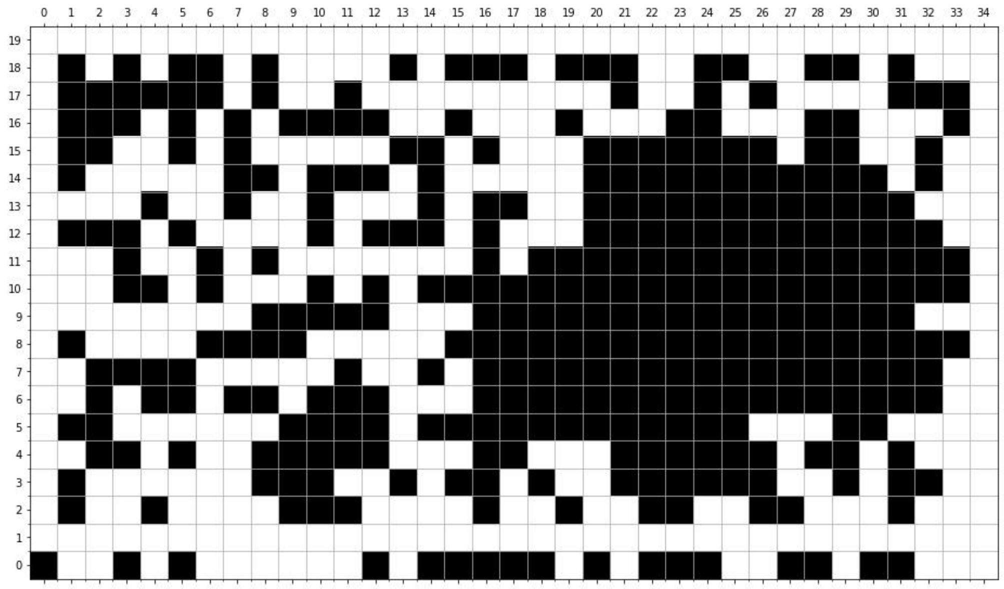
Competitive co-evolution of Pac Man controller and map appears to be very unfair to the Pac Man controller, as the map develops traps that can essentially end the game if the agent falls in one.

Appendix: Map images

As a demonstration of how pocked these maps can be here are a few.







Pac Man and Ghost Co-Evolution

Methodology

For this I added another sensor to my configuration, distance to pac man. I also modified some previously pac man specific ones (walls, and ghosts) to work with ghosts as well, which was mostly just changing it from 'm' to the player variable which was passed from the play_GPac function to evaluate trees, and then to the terminals in question.

Experiment Setup

For this the map subset was removed and replaced with the ghost subset of configurations. The ghost set was mostly the same except for a different set of sensor primitives. The reasoning behind this is that the ghosts probably don't care where the pills or fruit are, but do care where pac man is, so pills and fruit sensors were removed and the new pac man sensor was added. Reasonings for algorithmic choices are the same as in the other experiment.

My Configuration file is as follows:

[fitness_kwargs]	
map_gene	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
height	20
width	35
samples	5
pill_spawn	linear
pill_density	0.05
fruit_prob	0.1
fruit_score	10

ghost_type	genetic
penalty_coefficient	0.2
opponentCount	1
[pac_GP_configs]	
mu	200
num_children	100
mutation_rate	0.05
individual_class	treeGenotype
parent_selection	k_tournament_with_replacement
survival_selection	truncation
[pac_initialization_kwargs]	
depth_limit	5
operatorPrimitives	['+','-','*','/','RAND']
sensorPrimitives	['G', 'P', 'W', 'F', '#.#']
constantRange	[-10,10]
[pac_parent_selection_kwargs]	
k	5
[pac_recombination_kwargs]	
depth_limit	5
[pac_mutation_kwargs]	
depth_limit	5
[pac_survival_selection_kwargs]	
[ghost_GP_configs]	
mu	200
num_children	100
mutation_rate	0.05
individual_class	treeGenotype
parent_selection	k_tournament_with_replacement
survival_selection	truncation

[ghost_initialization_kwargs]	
depth_limit	5
operatorPrimitives	['+','-','*','/','RAND']
sensorPrimitives	['G', 'PAC', 'W',#.#']
constantRange	[-10,10]
[ghost_parent_selection_kwargs]	
k	5
[ghost_recombination_kwargs]	
depth_limit	5
[ghost_mutation_kwargs]	
depth_limit	5
[ghost_survival_selection_kwargs]	

Results

Species	Standard Dev of Max per run	Average of Max Per run
Ghost	29.17102	-17.5
Pac man	17.52348	179.8333

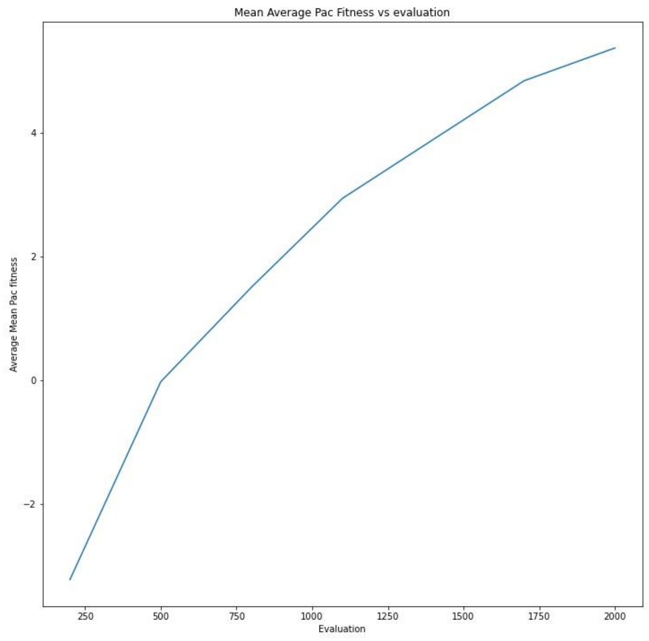


Figure 5 Mean Average Pac Fitness vs Evals

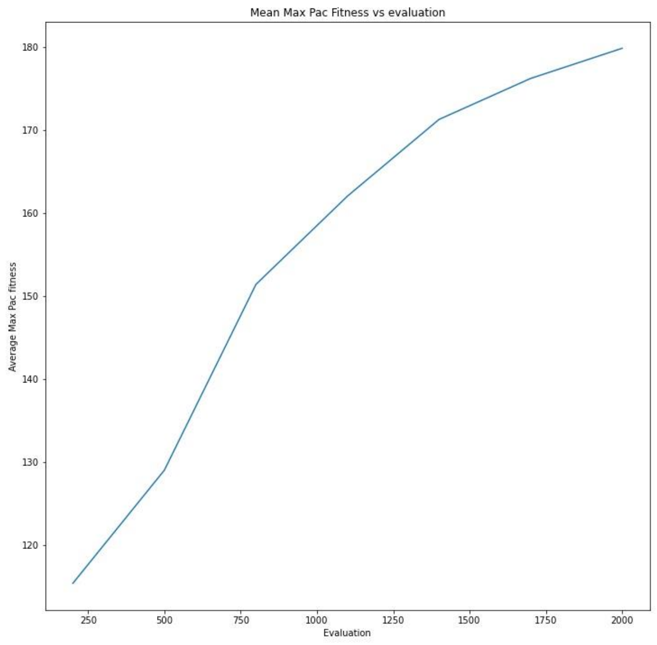


Figure 6 Mean Max Pac Fitness vs Evals

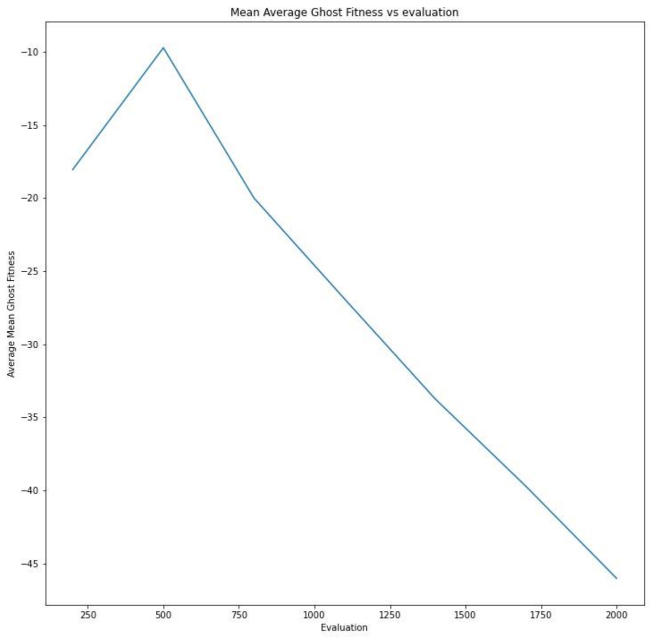


Figure 7 Mean Average Ghost Fitness vs Evals

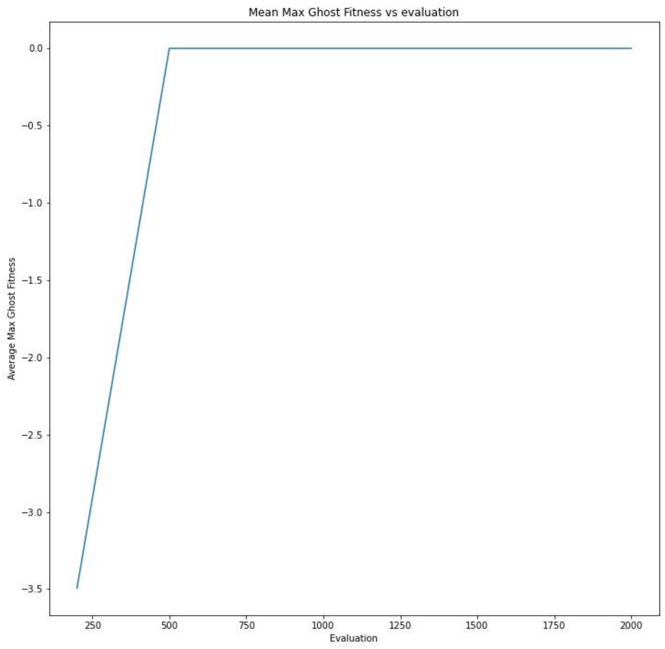


Figure 8 Mean Max Ghost Fitness vs Evals

Discussion

Pac man vs ghost seems like a more level playing field, the ghosts didn't really develop any technique similar to the maps, instead they mostly act like pac man agents. That is to say that they like to move back and forth a lot and go for pac man when able. In fact, it was sort of infuriating to watch the exhibition match, as pac man and the ghosts were, for a large majority of it, on opposite sides of the map moving back and forth at each other until someone made the first proper move.

Conclusion

Co-evolving Pac Man and ghosts competitively seems to be more fair for both sides than evolving Pac Man against maps. The average of max per run of both is decent, and the exhibition match actually played out unlike the map vs Pac scenario.