Term Project

Comp 557 Sec1

Kai-Po Lin (kl72), Haoyu Fang (hf21)

Understand the rules of Pac-World

1. Look at the rules that define the behavior of a gene (the different components) and make sure you understand them. Use the simulator to understand the impact of changing the U, V, W, etc genes.

Ans: According to our experiences using the simulator, we understand that:

U→ Changing the U genes of my given species will result in its mite when facing an empty cell, turning to and generating a baby facing the direction which is my assigned value times of turns from the original one.

V→ Changing the V_e genes of my species will result in its mite when facing an enemy mite, whose age is less than the mite's and whose direction is e-time-turn different from its, both replacing the enemy mite by a baby of its own species facing to and turning itself to the direction which is (my assigned value: v* difference of direction: e * the age of the mite: k)=vek times from the original.

W→ Changing the W genes of my species will make the mite, when facing a barrier, have my assigned value times of turns from the original one, corresponding to different age.

X→ Changing the X genes of my species will make the mite, facing a blob, have my assigned value times of turns from the original one, corresponding to different ages.

Y→ Changing the Y_e genes of my species will result in its mite when facing a friendly mite, whose direction is e-time-turn different from the mite's, turning (my assigned value: y* difference of direction: e* the age of the mite: k)=yek times from the original.

Z→ Changing the Z_e genes of my species will result in its mite when facing an enemy mite, whose direction is e-time-turn different from the mite's, turning (my assigned value: z* difference of direction: e* the age of the mite: k)=zek times from the original.

2. How does the all-zeros gene behave when confronting the all-ones or the all-threes genes? Make sure you understand why that behavior results starting from the given rules.

Ans: When confronting the all-ones or the all-threes genes, all-zeros gene is annihilated in 14 rounds. This behavior mainly results from the given rule U. As U genes are assigned to 0, the all-zeros gene behaves like a direct line with its youngest child as a frontier towards the enemy species. As all-ones or the all-threes genes have nonzero U genes assigned, those mites who are facing enemy mites are mostly the strongest ones, in which all-ones or the all-threes genes are hardly beaten. In this case, even though the all-zeros gene encounters all-ones or all-threes genes earlier, it cannot beat either of them with a least-age mite and will soon be replaced in several rounds.

Use the simulator for hand exploration of the space

1. Based on the rules alone, can you come up with good settings for the different gene positions? Try making variations of the all-ones and all-threes, and pit them against the all-ones and all-threes. Can you find a variant of all-ones that beats the all-ones? The all-threes?

Ans: Randomly change all 1's genes in PyPacwarExample.py to find out a super gene that can beat all 1's and all 3's ↓

Example of Gene1:

```
0 393 0 107
Found!
=>  [1, 0, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 3, 1, 1]
Example Python module in C for Pacwar
all ones versus var all ones ...
Number of rounds: 393
Ones PAC-mites remaining: 0
Number of rounds: 107
Threes PAC-mites remaining: 0
```

Example of Gene2:

```
Found!
=>  [1, 0, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1]
Example Python module in C for Pacwar
all ones versus var all ones ...
Number of rounds: 242
Ones PAC-mites remaining: 0
Number of rounds: 96
Threes PAC-mites remaining: 0
```

Example of Gene3:

```
Found!
=>  [1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 3, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
Example Python module in C for Pacwar
all ones versus var all ones ...
Number of rounds: 272
Ones PAC-mites remaining: 0
Number of rounds: 109
Threes PAC-mites remaining: 0
```

Computationally explore the space of genes

1. Which of the algorithms we have studied in class appear relevant to the problem of finding a killer gene?

Ans: Genetic algorithm we studied in class is suitable for us to find out a killer gene since the whole problem is extremely complex (4^{50} possibilities). Though the genetic algorithm doesn't guarantee an global optimal solution, it obsesses time efficiently on complex problems and still be able to yield a decent local optimal solution if we have well-defined an evaluation function as well as selected great selection, crossover, and mutation policies.

2. Implement the simplest among them and use it to find a gene that all beats all-ones and all-threes.

Ans: We randomly change ten of the elements within an all 1's gene in every iteration (like the mutation process in Genetic Algorithm), and check if this gene can both beat all 1's and all 3's.

```
0 393 0 107
Found!
=>  [1, 0, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 3, 1, 1]
Example Python module in C for Pacwar
all ones versus var all ones ...
Number of rounds: 393
Ones PAC-mites remaining: 0
Number of rounds: 107
Threes PAC-mites remaining: 0
```

3. Once you have beaten the all-ones and all-threes, how do you find even better genes?

Ans: This time we force the program to discover a gene that can beat all 1's and all 3's in 300 rounds. Therefore obtain a better gene as a result.

```
Found!
=>  [1, 0, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1]
Example Python module in C for Pacwar
all ones versus var all ones ...
Number of rounds: 242
Ones PAC-mites remaining: 0
Number of rounds: 96
Threes PAC-mites remaining: 0
```

4. What is your plan to find genes that will win the tournament at the end of the term?

Ans: We will apply Genetic Algorithm (GA) to discover super genes. In order to do so, we consider that the definition of the evaluation function may be essential since it is telling the genes in the evaluation process of how strong they are. Currently, we consider the inverse of "Rounds" as the value of the gene. However, perhaps we will add the remaining genes after each battle into account if we think it is useful for our GA's evaluation process.