

CIS579 Artificial Intelligence

# Genetic Algorithm

Project Report by Kyrylo Krysko

CONTENTS

1. Introduction ..... 3

2. Implementation..... 3

3. Results ..... 4

4. Conclusion ..... 4

## 1. INTRODUCTION

In this assignment, we are implementing a Genetic Algorithm.

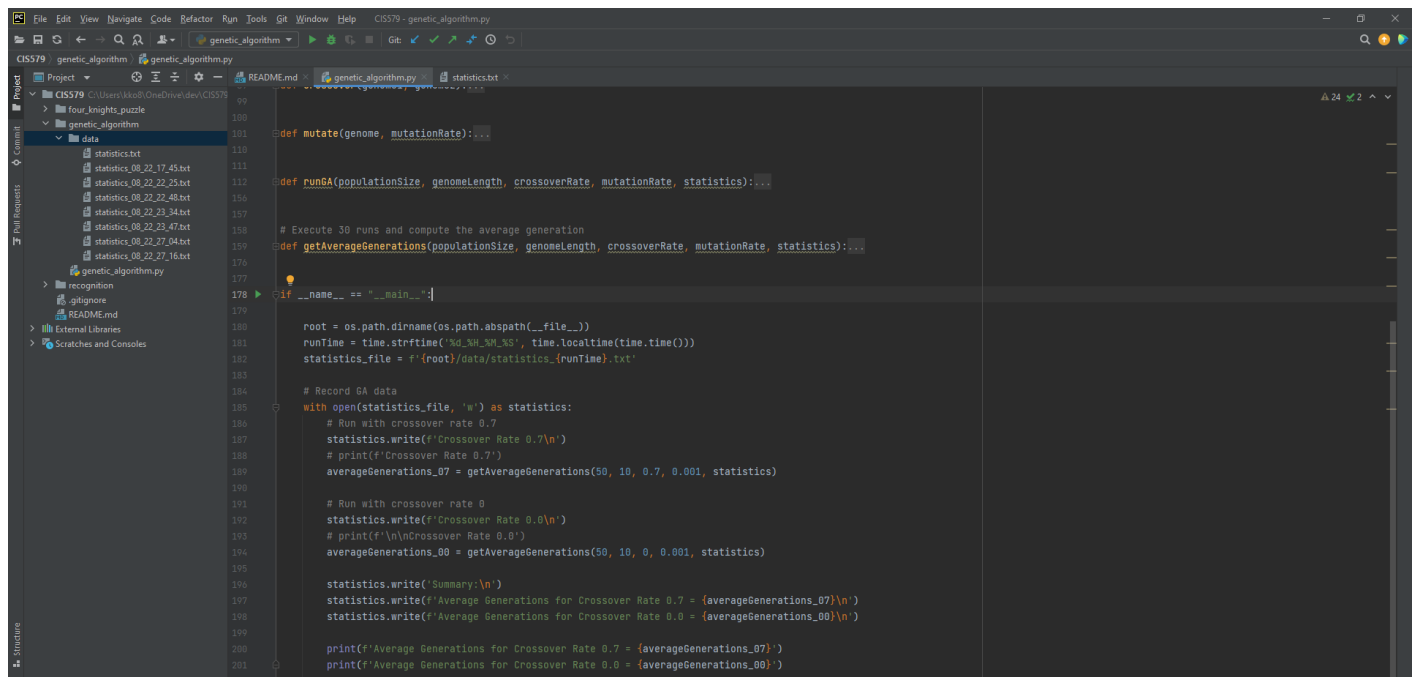
Conditions:

- Roulette-wheel sampling,
- Population size 50,
- Single-point crossover rate 0.7 and 0.0,
- Bitwise mutation rate 0.001.
- Fitness function:  $f(x)$  = number of ones in  $x$ , where  $x$  is a genome of length 10

The goal is to execute 30 runs of the Genetic Algorithm with Crossover Rate 0.7 and 30 runs with Crossover Rate 0.0 and compare results.

## 2. IMPLEMENTATION

The program implemented with Python3. Each time we execute the program, it runs Genetic Algorithm 30 times with crossover rate 0.7 and 0 and records statistics to a separate file: <program root>/data/statistics\_<time stamp>.txt



```
100 def mutate(genome, mutationRate):...
101
102 def runGA(populationSize, genomeLength, crossoverRate, mutationRate, statistics):...
103
104 # Execute 30 runs and compute the average generation
105 def getAverageGenerations(populationSize, genomeLength, crossoverRate, mutationRate, statistics):...
106
107
108 if __name__ == '__main__':
109
110     root = os.path.dirname(os.path.abspath(__file__))
111     runTime = time.strftime("%d_%M_%S", time.localtime(time.time()))
112     statistics_file = f'{root}/data/statistics_{runTime}.txt'
113
114     # Record GA data
115     with open(statistics_file, 'w') as statistics:
116         # Run with crossover rate 0.7
117         statistics.write(f'Crossover Rate 0.7\n')
118         # print(f'Crossover Rate 0.7')
119         averageGenerations_07 = getAverageGenerations(50, 10, 0.7, 0.001, statistics)
120
121         # Run with crossover rate 0
122         statistics.write(f'Crossover Rate 0.0\n')
123         # print(f'\n\nCrossover Rate 0.0')
124         averageGenerations_00 = getAverageGenerations(50, 10, 0, 0.001, statistics)
125
126         statistics.write('Summary:\n')
127         statistics.write(f'Average Generations for Crossover Rate 0.7 = {averageGenerations_07}\n')
128         statistics.write(f'Average Generations for Crossover Rate 0.0 = {averageGenerations_00}\n')
129
130     print(f'Average Generations for Crossover Rate 0.7 = {averageGenerations_07}')
131     print(f'Average Generations for Crossover Rate 0.0 = {averageGenerations_00}')
```

The program code attached to submission.

### 3. RESULTS

The program was executed 7 times and results gathered in the table below. Full list of statistics files attached to submission.

```
1 Crossover Rate 0.7
2 GA execution number: 0. GA data:
3 Generation 0: average fitness 5.38, best fitness 9.00
4 Generation 1: average fitness 5.58, best fitness 8.00
5 Generation 2: average fitness 5.96, best fitness 9.00
6 Generation 3: average fitness 6.16, best fitness 8.00
7 Generation 4: average fitness 6.20, best fitness 8.00
8 Generation 5: average fitness 6.44, best fitness 9.00
9 Generation 6: average fitness 6.44, best fitness 10.00
10 Number of generations to get best genome at execution number 0: 6
11
12 Crossover Rate 0.0
13 GA execution number: 1. GA data:
14 Generation 0: average fitness 5.32, best fitness 8.00
15 Generation 1: average fitness 5.36, best fitness 9.00
16 Generation 2: average fitness 5.50, best fitness 9.00
17 Generation 3: average fitness 6.00, best fitness 9.00
18 Generation 4: average fitness 6.64, best fitness 9.00
19 Generation 5: average fitness 5.72, best fitness 9.00
20 Generation 6: average fitness 5.50, best fitness 8.00
21 Generation 7: average fitness 6.42, best fitness 9.00
22 Generation 8: average fitness 6.24, best fitness 9.00
23 Generation 9: average fitness 6.64, best fitness 8.00
24 Generation 10: average fitness 6.70, best fitness 9.00
25 Generation 11: average fitness 6.88, best fitness 8.00
26 Generation 12: average fitness 6.80, best fitness 10.00
27 Number of generations to get best genome at execution number 1: 12
```

Program Execution Attempt	Average Generations for Crossover 0.7	Average Generations for Crossover 0.0
1	16.6	30
2	15.9	29
3	14.2	25.9
4	17.9	28.7
5	17.0	27.2
6	15.7	25.5
7	15.2	27.6

### 4. CONCLUSION

As we can see from 7 program execution with crossover rate = 0 we get best genome with much more generations.

If crossover rate is 0, this means the genetic algorithm will not use the crossover operation at all. In this case, the genetic algorithm will rely only on mutation (and the selection of existing good solutions) to explore the search space. This may slow down the rate of improvement in the population, as potentially beneficial combinations of genes from different individuals cannot be explored via crossover.