Genetic Algorithms

EX2

14/05/2020

Or Zinger 200687572

Abstract:

One of the interesting methods that has been raises during the recent years to solve complex problems, was the Genetic Algorithms family. This family taken from the biological world to solve complex problems in the field of computer science.

We introduce here a solution with GA to the 4-Coloring graph, one of the famous NP-complete problems in Computer Science field. The problem is to paint with 4 colors a graph $G(V, E)$ where V is the set of the graph nodes and E is the set of the graph edges. We were given a map with 12 polygons and we try to paint it in 4 colors with GA.

Model:

We took the 12-polygons map and convert it to 12-nodes undirected connectivity graph. Each polygon described as a node and a boundary between two polygons as an edge. We took boundaries that overlap with other polygons only if they shared a common side. To represent abstraction of the graph, we represent it as a one dimensional vector with 12 cells, where each cell is filled with a random number between 1 to 4, each one of this values represent a color. Side to it, we held an adjacency matrix.

Algorithm:

We have started with a random population in size $N \times 12$ where each individual from the population P has a 12 cells randomly painted. After we have created the initial population, we started to run over the graph with N solutions at each generation, until it founds a satisfactory solution.

1) **Fitness -** Calculate the fitness of each individual. The fitness defined as the number of prohibited boundaries in all over the graph. The less number of prohibited, the better. It's was defined by the following formula:

$$F(g)_{g \in P} = \sum_{i,j \in E} \delta(i,j)$$

where $\delta(i,j)$ is the $Kroncker\ delta$:

$$\delta(i,j) = \begin{cases} 1 \; c(i) = c(j) \\ 0 \; c(i) \neq c(j) \end{cases}$$

Where $c(x)$ is the color of node x.
After we have calculated for each individual, we got a list with all the fitness values for the current population, and normalized it by the following formula:

$$F_{normalize}(g)_{g \in Population} = \frac{\sum_{i,j \in E} \delta(i,j)}{\sum_{g \in P} \sum_{i,j \in E} \delta(i,j)} = \frac{F(g)_{g \in P}}{\sum_{g \in P} F(g)}$$

After it, we have sorted the list and calculate for each the individual its part of the whole part (inspired by the probability roulette). The better fitness it has, the better chances of being selected during the selection period.

2) **Selection** – select random number and check the first parent who has greater number. Do the same with the second parent. They cannot be the same parents.

3) **Crossover** – Cut both parents at the selected crossover split point and doing a crossover between them:

101100101001
000011100101

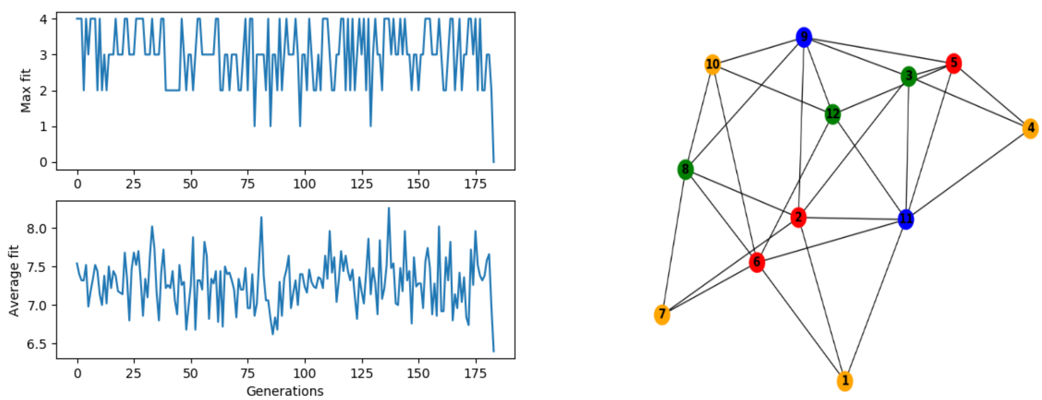
101111100101
000000101001

4) **Mutation** – For each cell of the individual vector, select random number and check if its lower than the given mutation probability parameter. If so, change its color to one of the three possible colors that left.

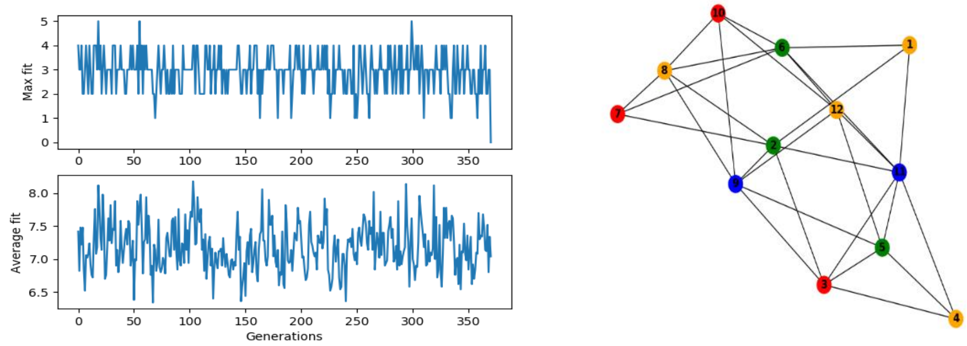
The stopped criterion is to find a satisfactory solution to the problem, it means that it found an individual with fitness of 0.

Results:

In our experiments we have examine number of hyperparameters, such as: crossover percentage, number of the population, and mutation probability. The graphs describe the accuracy change of the population fitness during the generations. Because the evaluation of the solutions involved a stochastic process, the curves look like a zigzag form, but eventually the solutions family shared an average fitness between them, closely to the optimal solution. The above graph describes the change of the best solution (have the minimal fitness) during the time. The below graph describes the change of the average fitness value of the population during the time.

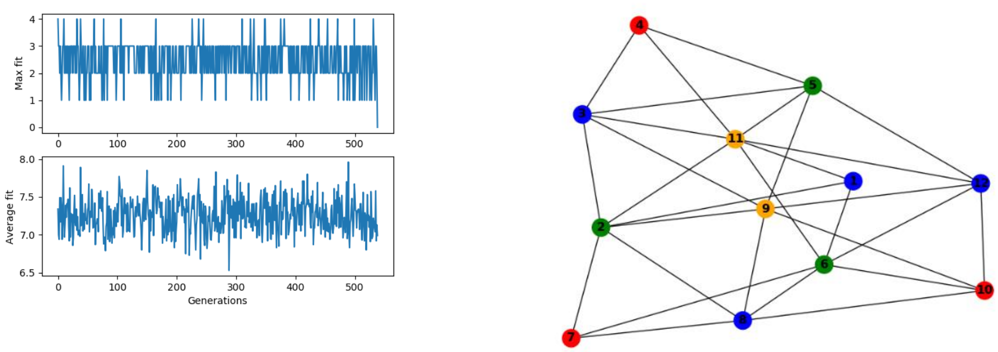


A simulation of 25% crossover splitting, 50 individuals in population and p_m = 0.5 probability of mutation after 183 generations



A simulation of 25% crossover splitting, 50 individuals in population and p_m = 0.2 probability of mutation after 370 generations

We can see here that the density of the best solution is very tight and this is because we have reduced the mutation probability to 0.2, so it having troubles to keeping on its average fitness during the time and has a lot of changes of its fitness.



A simulation of 30% crossover splitting, 100 individuals in population and p_m = 0.7 probability of mutation after 538 generations

Conclusions:

The model success to generating a solution for the 4-color map problem, with variance of generations. To avoid earlier convergence, we set a few strategies for preventing it, like preventing from same individuals to merge in with themselves, increasing populations size, and a high rate of mutation.