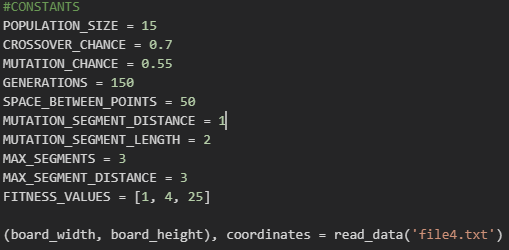
**Artificial Intelligence and Knowledge Engineering Laboratory**

***Vladyslav Gavryliuk 245603***

**Assignment 1**

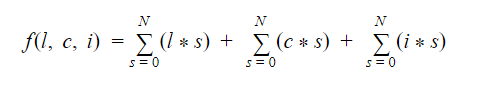
**Genetic Algorithm**

Let's get acquainted with **values** which will be taken into account :



Here we have :

* Chance for some executions for our population.
* Generations, which mean how many time it will be iterated.
* Space between points for drawing our board with space between points.
* Amount of maximum segments and its distance, but with doing mutation it can be expanded.
* The last but not least is the weight of fitness function :
* Total path length = 1
* Total segments count = 4
* Intersections = 25
* Board square and coordinates read by txt file.
* The ***fitness function*** is pefrormed by formula :





The values ordered by FITNESS\_VALUES.

Those values were measured and taken by n-thousands of the testing.

However, there is difference between mine implementation and typical fitness function, because as it was suggested by teacher dr inż Jerzy Sas, I should avoid cases when segments or part of segment goes out of board.

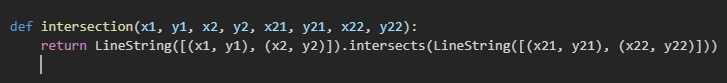
* ***Segment and Path***

Segment has direction and distance for this direction.

Path consists of segments.

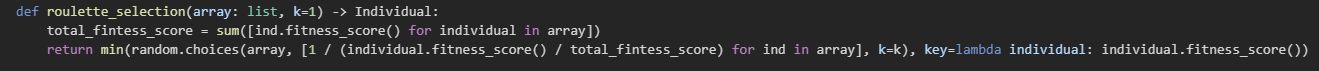
Then I’ve to create and connect segments of the path by methods create\_segments() -> connect\_segments().

To calculate and avoid intersection in further generations we call the method interscetion():



***Selections :***

* ***Roulette*** :



I’m going in approach to decrease the value of fitness score. It means that as lowest value as better it is.

So, then I’ve to power it by (-1) to have more chances for fitness function results which are smaller (lowest value has the biggest chance).

Random.choices exactly works in such way that it has the probability for each value.

* ***Tournament :***



Simply returns individual with the smallest fitness score.

Comparing these both selections, I cay say that tournament is much better, the reason is quit simple, tournament always returns the individual with the lowest value (best fitness) meanwhile roulette has the probability, and not always returns the best one.

Regarding ***crossover and initial population*** there is no something unusual to explain.

So, let’s move to the Mutation.

***Mutation*** works in such way:

Checks if we can split the segment (segment.distance > 2) if so, then the 1st part we just omit and the 2nd will be used in further execution. We check direction and its previous and next move. By checking expressions, we can modify segment. (for example extract segment in such way that another segment will be deleted)

If that mutation does not return empty distance, then we add it, else removing it.

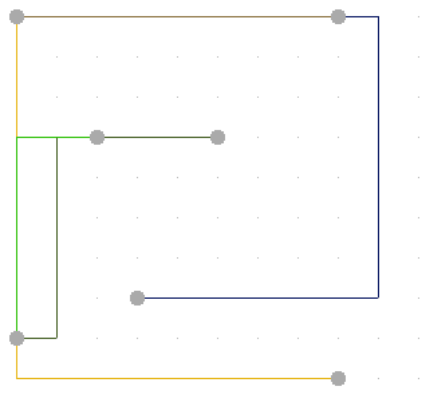
As we did it all, we are looking if nothing was not skipped, for example, if there is no empty segment (distance = 0), if there are no 2 nearest segments with the same direction, if there are no 2 nearest segments with opposite directions.

The data which will be used : ***File1, File 2***

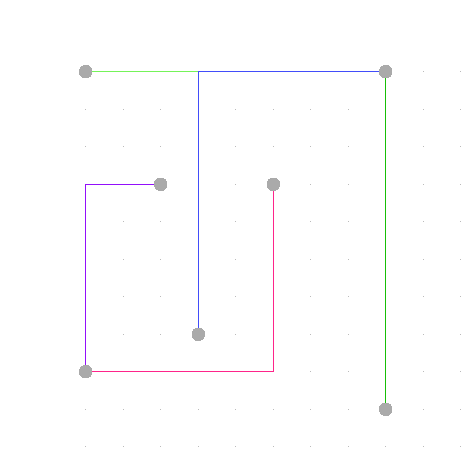
|  |  |
| --- | --- |
| File 1 | File 2 |
| 13;13  0;0;8;9  5;3;0;8  0;0;8;0  2;3;0;8  3;7;8;0 | 13;13  1;1;10;2  2;2;10;3  3;3;10;4  4;4;10;5  5;5;10;6  6;6;10;7  7;7;10;8  8;8;8;1 |

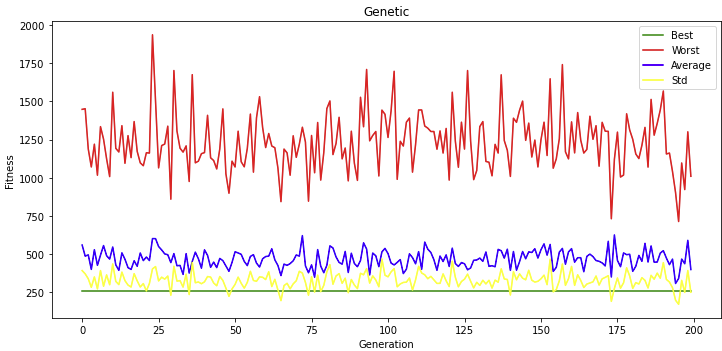
Results :

For ***file1*** with ***Genetic*** algorithm for 400 generations and population size 10.



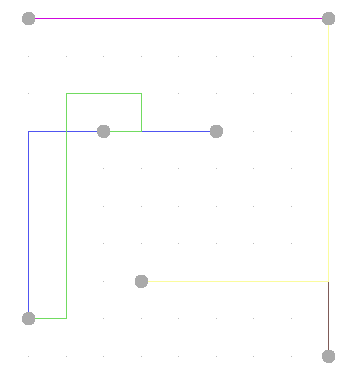
For Generations 200 and population size 20

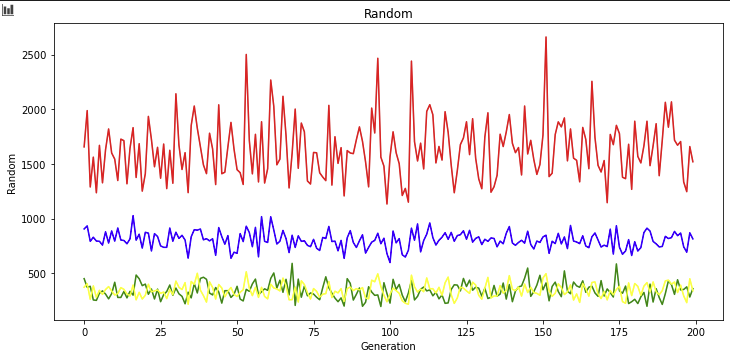




For ***random*** algorithm

Generations 200 and population 20

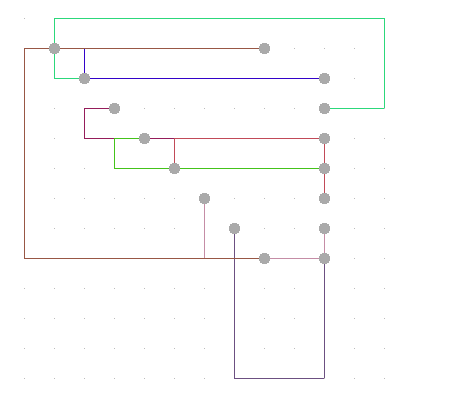


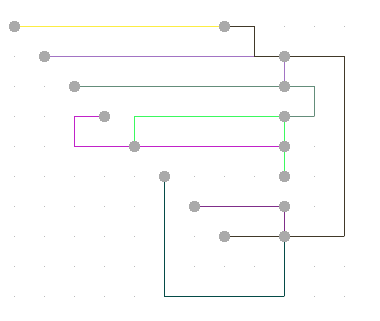


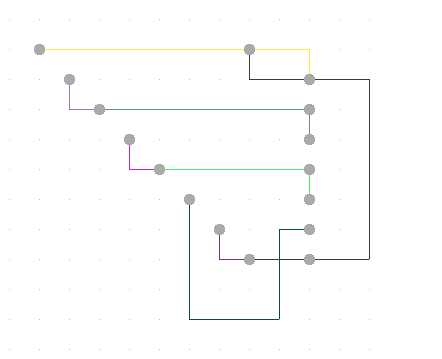
For ***file 2***

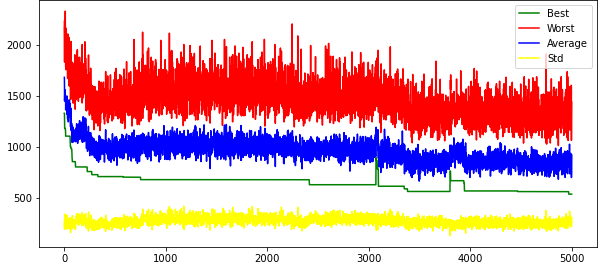
***Genetic*** Algorithm

Generations 5000, population 20





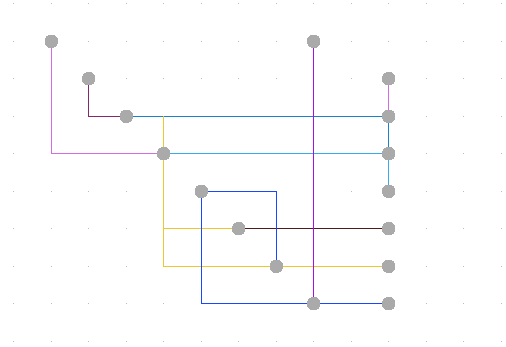


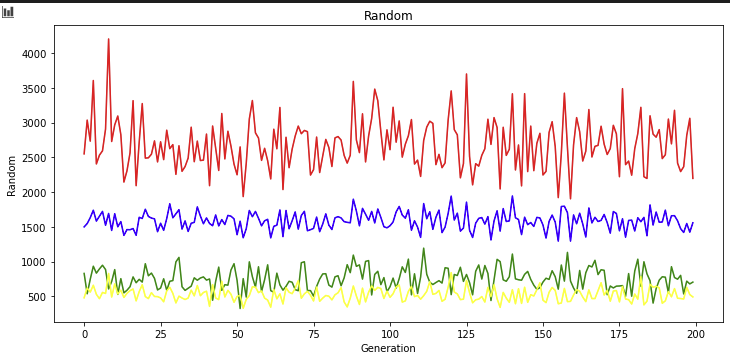


***File 2***

***Random*** algorithm :

Generation 200 and population 20





The calculations will be performed for population size = 10, generations 10:100:1000

Generations = 10

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instance | Generic algorithm[10x] | | | | Random algorithm[N] | | | |
|  | Best | Worst | Avg | Std | Best | Worst | Avg | Std |
| File1 | *270* | *1285* | *604.9* | *422.55* | *479* | *1542* | *845.2* | *349.97* |
| File2 | *831* | *1592* | *1087.1* | *177.40* | *776* | *2291* | *1452.6* | *471.58* |

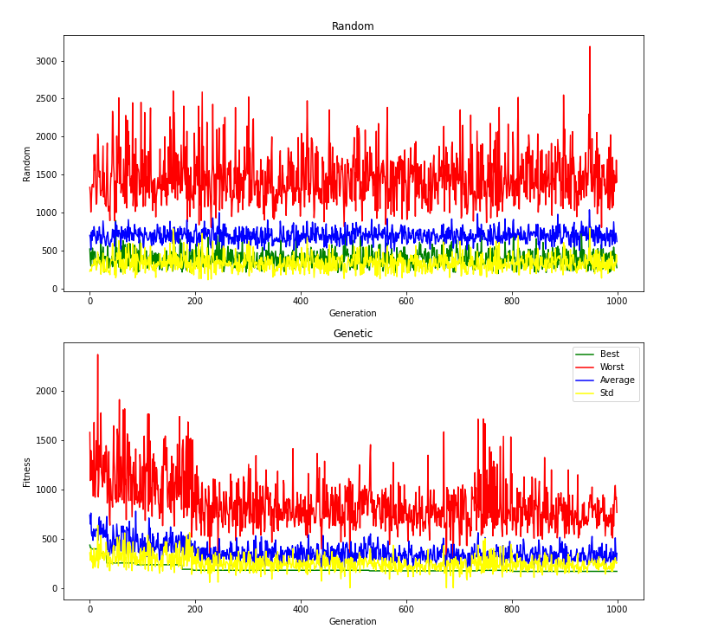
Generations = 100

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instance | Generic algorithm[10x] | | | | Random algorithm[N] | | | |
|  | Best | Worst | Avg | Std | Best | Worst | Avg | Std |
| File1 | *284* | *1022* | *423.4* | *269.75* | *348* | *1921* | *809.3* | *529.36* |
| File2 | *563* | *1833* | *1039.0* | *563.55* | *677* | *2141* | *1546.2* | *496.85* |

Generations = 1000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instance | Generic algorithm[10x] | | | | Random algorithm[N] | | | |
|  | Best | Worst | Avg | Std | Best | Worst | Avg | Std |
| File1 | *167* | *765* | *350.0* | *254.13* | *279* | *1404* | *621.9* | *356.61* |
| File2 | *511* | *2162* | *1190.7* | *658.50* | *849* | *1869* | *1330.9* | *393.40* |

Visualization comparing of plots for case : Generations = 1000 populations = 20, File 1

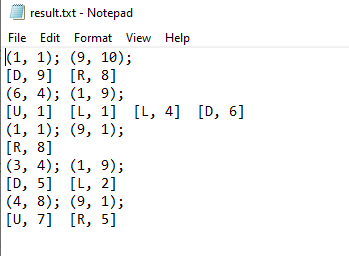


***Conclusion***

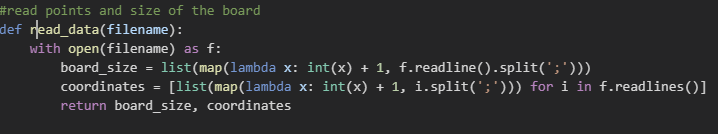
As we can see from tables above, random algorithm has some results which with comparing to genetic algorithm looks better, but if looking with analytics, then it’s obvious that with bigger set of population the “best” takes lower amount of fitness function with newer individuals which are better than previous ones, so it means that our executions on the individuals give us better one, also there is option when for example “worst” has more score than in random one, but this is can be explained easily : mutation, crossover, selections has random option, which sometimes may take the way we didn’t want or expect.

Nevertheless, the genetic algorithm gives more precisely result than random one.

At the end it creates txt file with all paths and their moving:



The values are increased by 1 to have a point which is located on (0;0). Also, the size of board increased by 1.



***Smile, because this is the end 😊***

***With best wishes,***

***Vladyslav Gavryliuk.***