CSE 4082 – Project#2

In this project, we used a genetic algorithm to solve minimum weighted vertex cover problem. Our algorithm begins with generating offspring with obtained population number from the user. Then we randomly generated offsprings. To do this, firstly a matrix which size is node_number x population_size filled randomly with 1’s or 0’s to show that this node is in offspring. Then we repair these randomly generated population with repair function. It controls whether these nodes cover all edges or not. If they do not, the program add new nodes into the offspring until getting a nodes which are successfully covers all edges. Now offsprings are ready for mating pool.

In repair function it make a temporal checklist that contains the edges with two nodes for each. And for each off string it enter a while loop until it becomes a feasible offspring which means that all edges are covered with that offsprings 1 bitted nodes. To check this we delete the element of temp_checklist ,if one of the 1 bit nodes in offspring is connected that edge, one by one: If the checklist is empty after the 1 bit nodes in offspring is finished, then the offspring is repaired. Otherwise, we use an array that contains the nodes from the eliminated temp_checklist that contains that uncovered edges and make a random choice to change a zero bit node in offspring that decreases the number of edges uncovered.

In mating pool, randomly chosen two offspring are compared according to their fitness value. The offspring which has lower fitness value than the other is selected and written into as a new member of the population. This is the first time, we need to do this as population size in order to get filled population by winner offsprings.

In crossover function, we form a new population by making child via 1- point crossover with the two different offsprings’s bit set of previous population if the random number generated is smaller than the crossover probability. Otherwise we transform the previous population’s offspring sets directly.

In mutate function we change each offsprings’ bits with opposite value if the generated random number is smaller than the mutation probability which is a very small value.

150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman

Output Table of “003.txt” Graph

Graph “003.txt”					
	Crossover Pr.	**0.5**		**0.9**	
#Generations	**Pop. Size \ Mut. Pr.**	**1/n**	**0.05**	**1/n**	**0.05**
100	**100**	*439.81*	*463.95*	*464.81*	*461.09*
	200	*459.32*	*461.82*	*466.03*	*458.32*
400	**100**	*470.02*	*460.60*	*458.72*	*447.83*
	200	*463.32*	*453.82*	*463.95*	*458.65*

Output Table of “015.txt” Graph

Graph “015.txt”					
	Crossover Pr.	**0.5**		**0.9**	
#Generations	**Pop. Size \ Mut. Pr.**	**1/n**	**0.05**	**1/n**	**0.05**
100	**100**	*501.86*	*506.00*	*506.84*	*505.99*
	200	*506.86*	*506.49*	*506.01*	*506.85*
400	**100**	*499.53*	*504.62*	*504.44*	*505.99*
	200	*504.57*	*504.62*	*504.70*	*504.62*

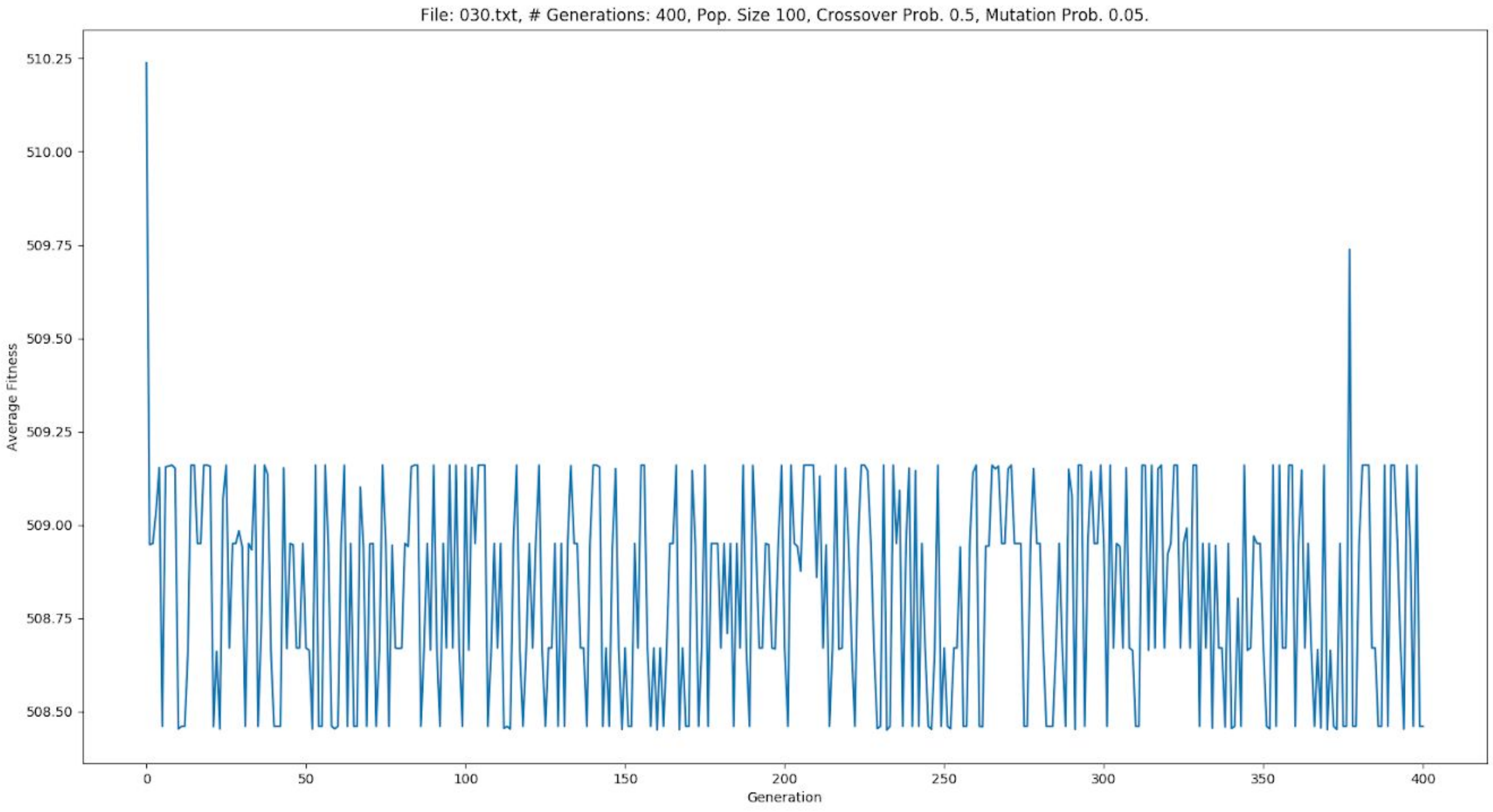
150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman

Output Table of “030.txt” Graph

Graph “030.txt”						
	Crossover Pr.	**0.5**		**0.9**		
#Generations	**Pop. Size \ Mut. Pr.**	**1/n**	**0.05**	**1/n**	**0.05**	
100	**100**	*509.23*	*509.72*	*508.96*	*509.39*	
	200	*509.12*	*509.63*	*508.84*	*509.17*	
400	**100**	*509.16*	*508.34*	*508.62*	*508.45*	
	200	*508.61*	*508.73*	*508.64*	*509.19*	

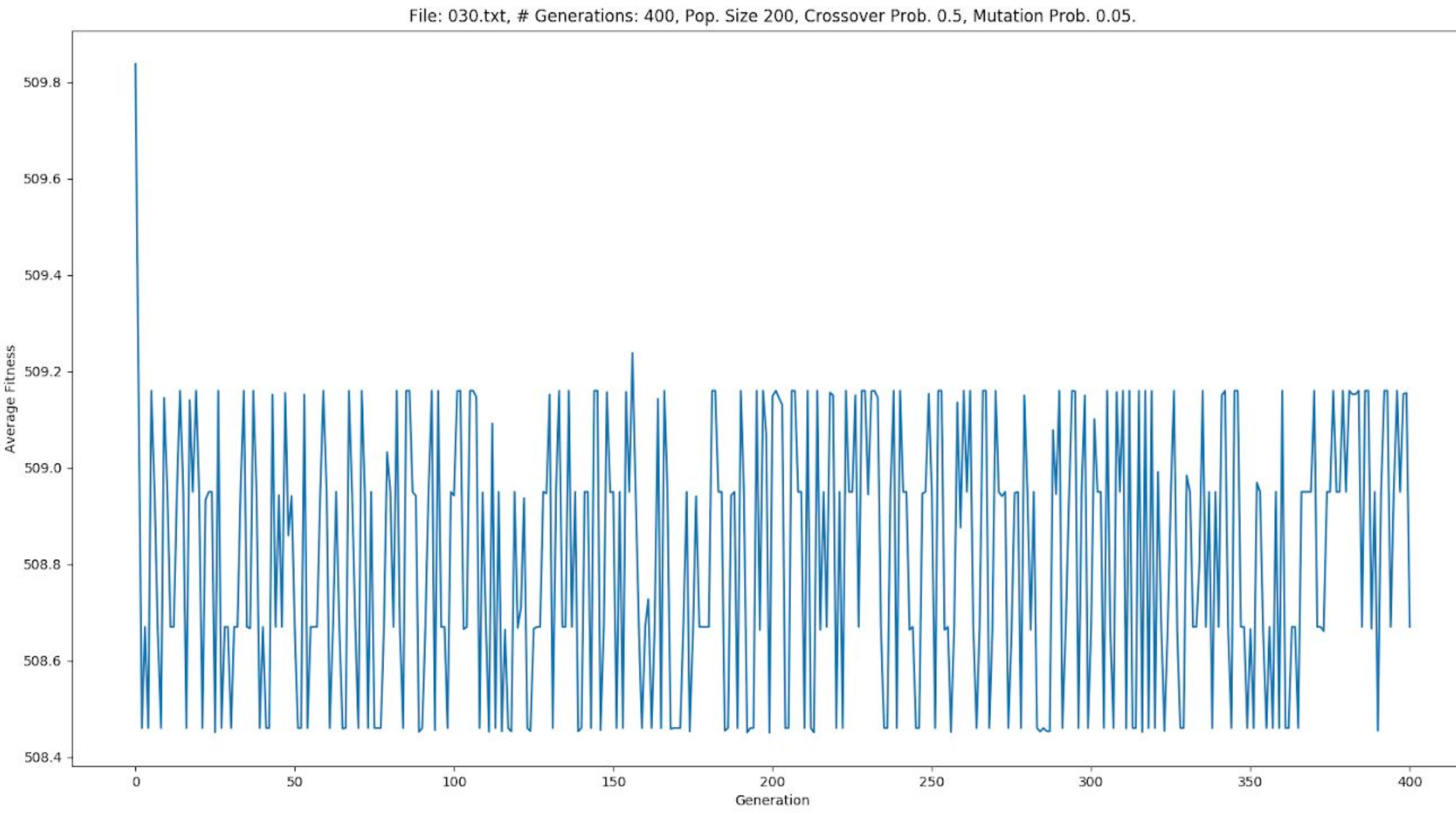
Output Graphs for “030.txt” with Different Configurations

1. File: 030.txt, # Generations: 400, Pop. Size 100, Crossover Prob. 0.5, Mutation Prob. 0.05.

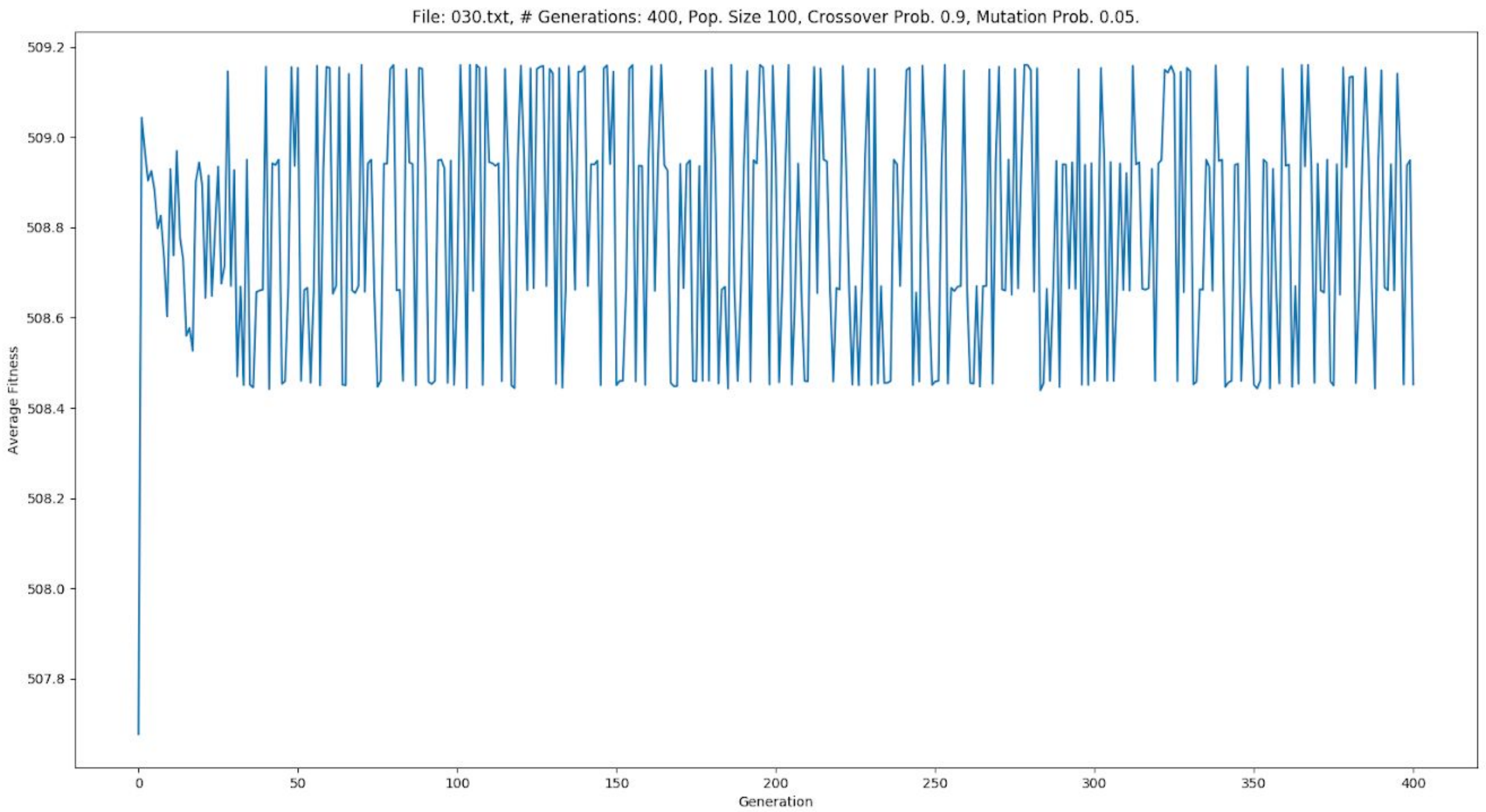


150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman

2. File: 030.txt, # Generations: 400, Pop. Size 200, Crossover Prob. 0.5, Mutation Prob. 0.05.

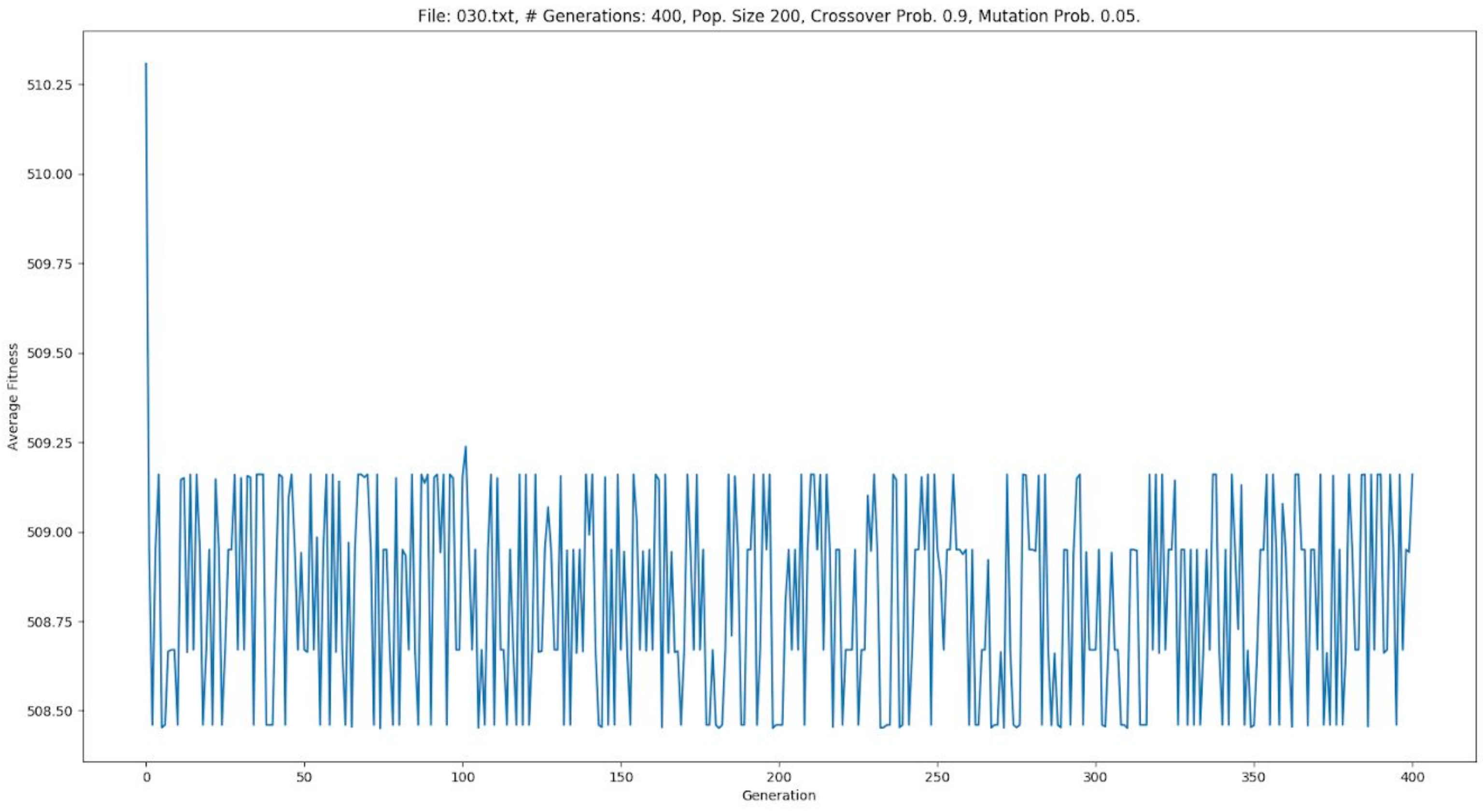


3. File: 030.txt, # Generations: 400, Pop. Size 100, Crossover Prob. 0.9, Mutation Prob. 0.05.

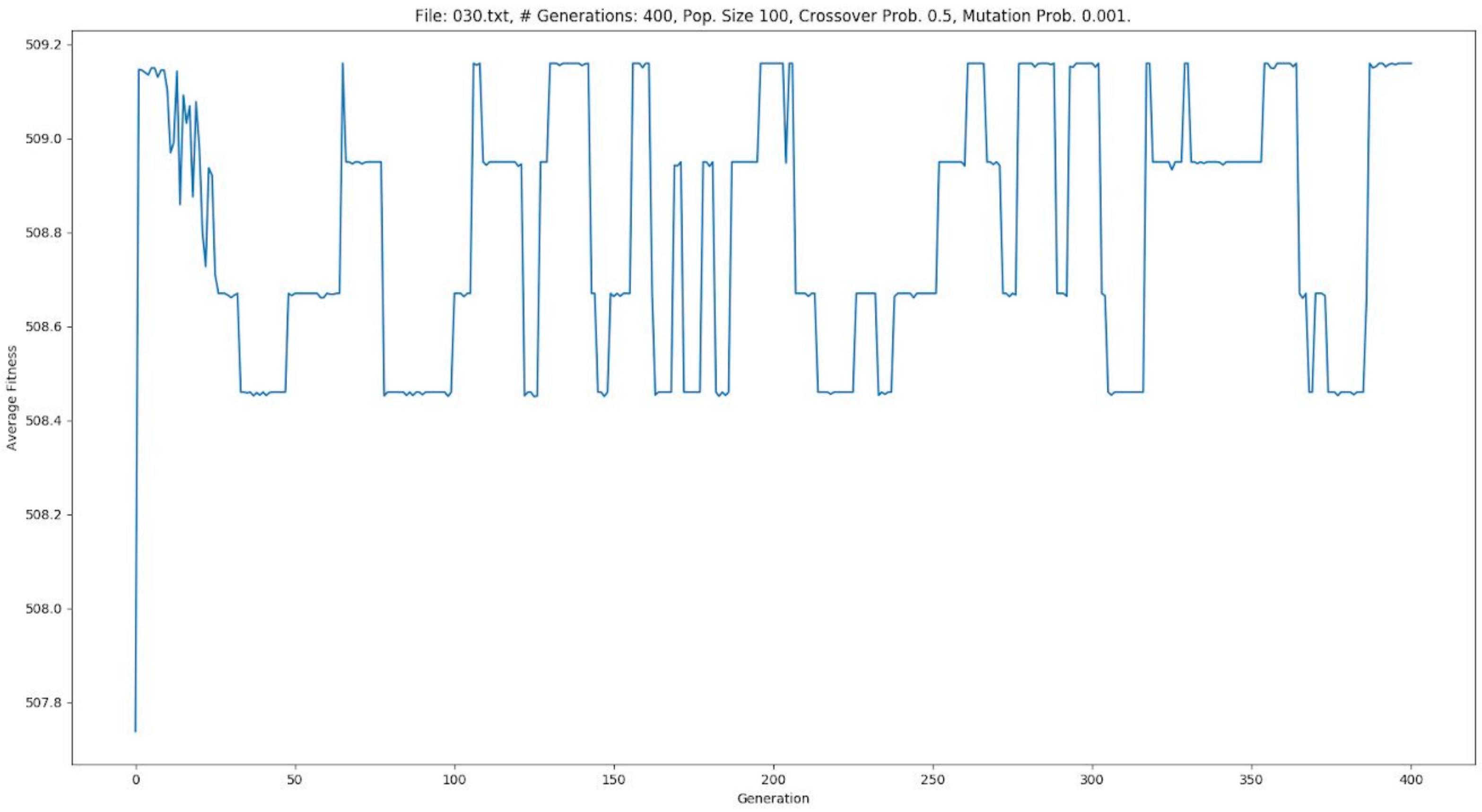


150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman

4. File: 030.txt, # Generations: 400, Pop. Size 200, Crossover Prob. 0.9, Mutation Prob. 0.05.

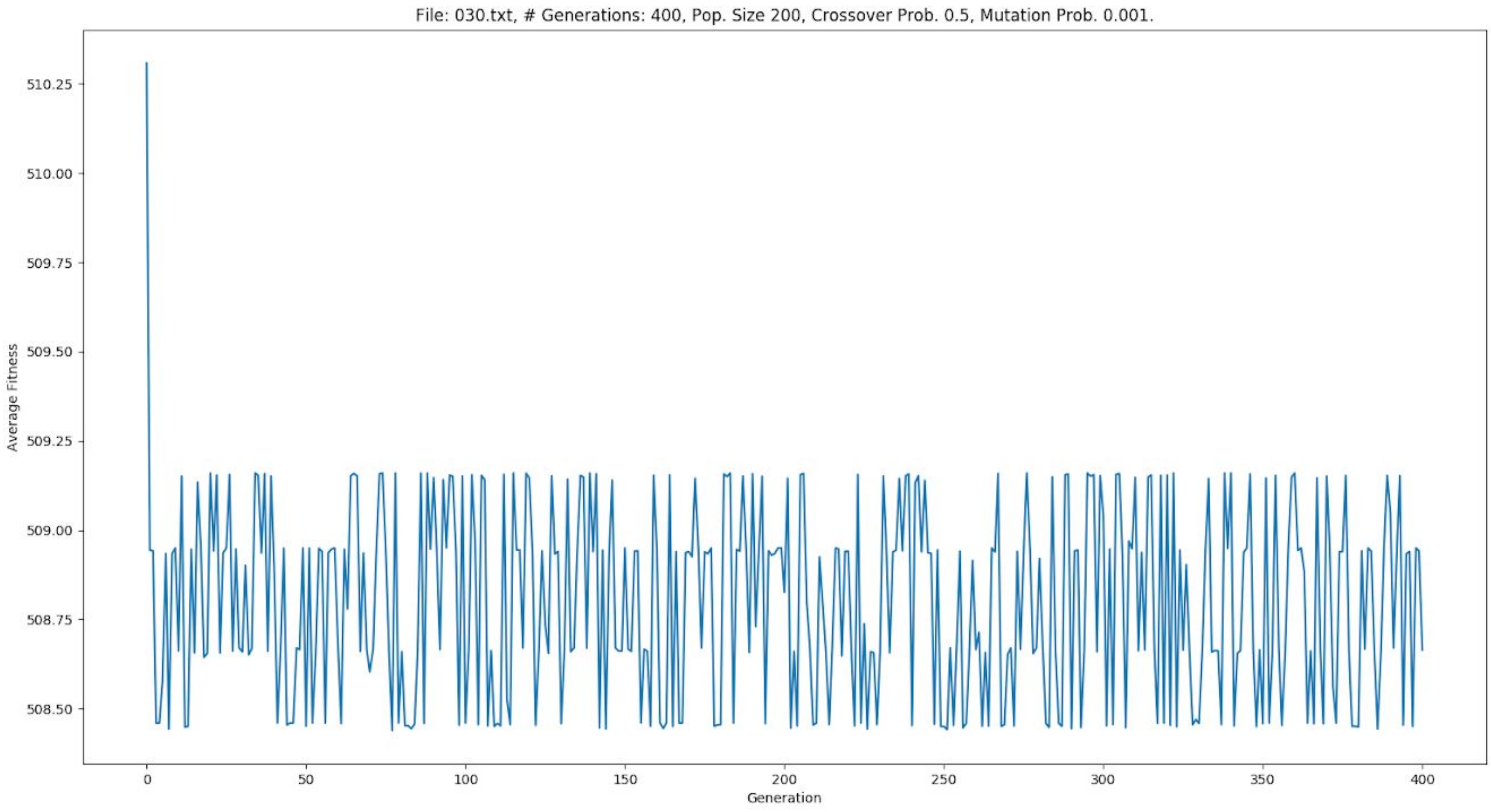


5. File: 030.txt, # Generations: 400, Pop. Size 100, Crossover Prob. 0.5, Mutation Prob. 1/n.

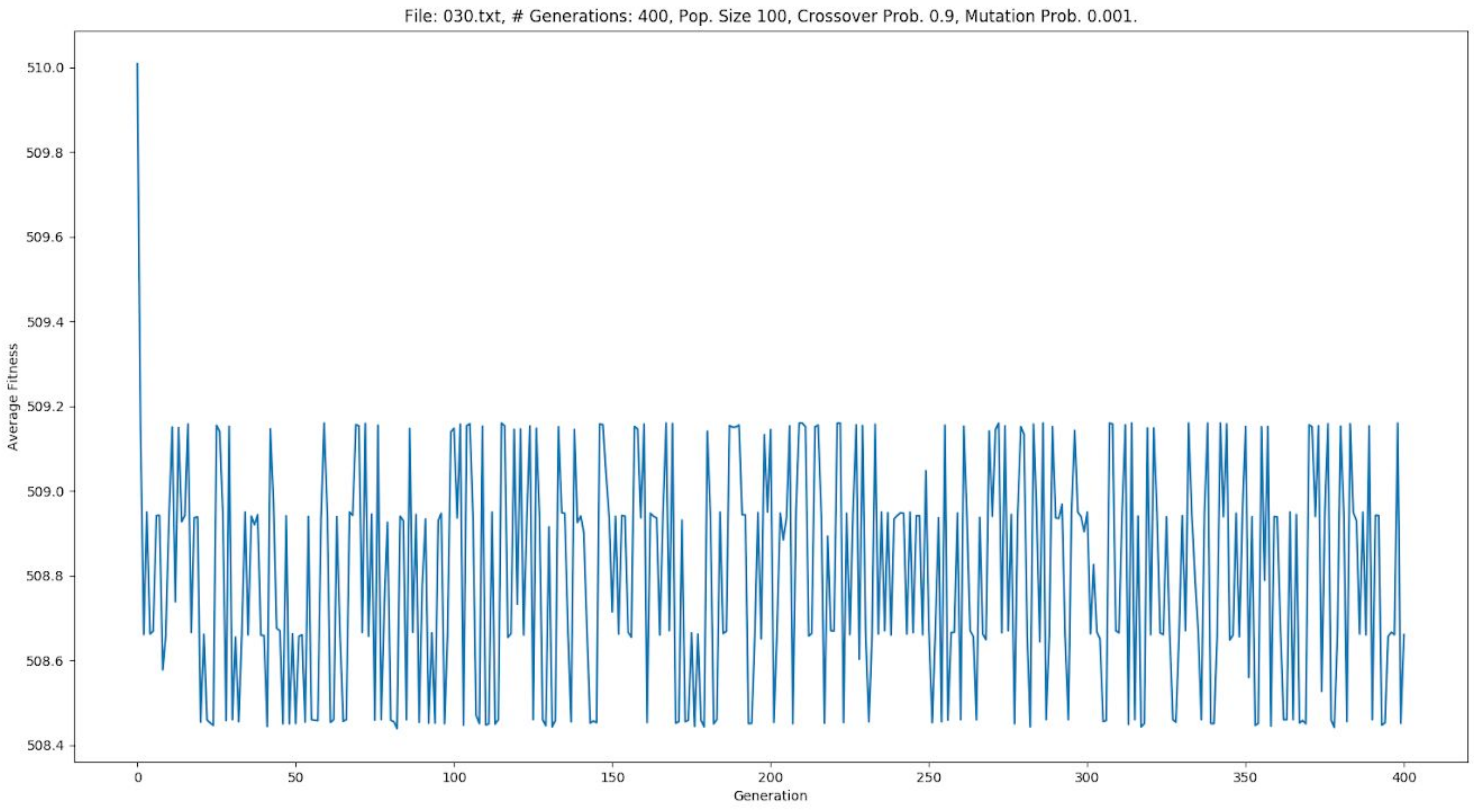


150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman

6. File: 030.txt, # Generations: 400, Pop. Size 200, Crossover Prob. 0.5, Mutation Prob. 1/n.

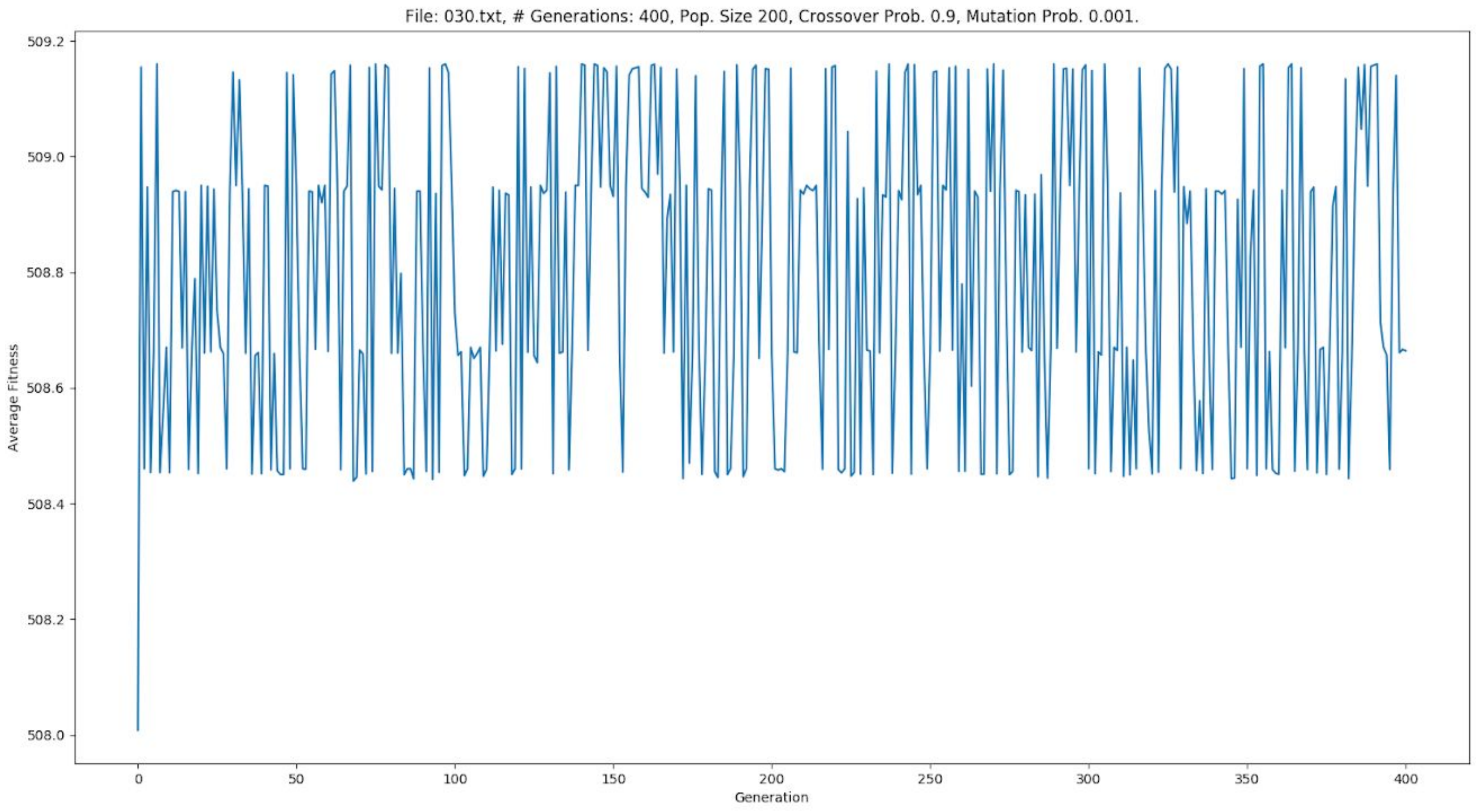


7. File: 030.txt, # Generations: 400, Pop. Size 100, Crossover Prob. 0.9, Mutation Prob. 1/n.



150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman

8. File: 030.txt, # Generations: 400, Pop. Size 200, Crossover Prob. 0.9, Mutation Prob. 1/n.



150114022 - Oğuzhan Bölükbaş
150114025 - Faruk Furkan Şişman