Genetic algorithm

Explain the important operations of the algorithm

Selection, crossover and mutation are three important operations that is included in this algorithm.

Selection

My algorithm uses **Roulette selection.** The sum of all the parents' fitness values will first be calculated. Each parent's fitness value is then divided by the sum of the fitness values. This gives the probability of being selected for each individual.

Crossover

Crossover is done by taking two individuals as parents and returns one child. In parent 1 two indexes are randomly selected and the locations that are in that interval and are added to an array. From parent 2, every location that doesn't already exist in the array is put into the array. From this array the new child individual is created.

Mutation

In mutate we check if we are going to mutate that child first with the mutation probability, if the case

is true, I randomly select two indexes and between that interval, flip the locations around. The two selected indexes will swap places with each other, the spots next two them will swap with each other, and the program continue to do this between the interval until it has reached the middle point.

Explain the representation of an individual

For the individual, that represents one solution of the problem, I use real representation. Each individual contains a list of all the locations. A location is represented by X and Y-coordinates to each city and an ID, all those three is represented by real numbers.

Give the equation of the fitness function

$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ – the distance between two cities

$t(I_j) = \sum_{j=1}^{n}(d(i_j))$ – the total distance for all cities for one individual

$fit(i_j) = \frac{1}{t(i_j)}$ – the fitness value for one individual

I_j – one individual

n = the amount of cities

Give the parameters used in your algorithm

The parameters used in the algorithm are **POPULATION_SIZE**, **SURVIVORS**, **NUMBER_OF_PARENTS** and **MUTATION_PROBABILITY**. The population size represents the number of individuals that can be in one generation, and in my algorithm, the population size is 500, so one generation can contain 500 solutions (individuals). Survivors are the number of individuals that survive into the next generation and that is decided depending on their fitness, so I take the 60 bests from the previous generation and put them into the next one. The survivors are used to achieve elitism. The number of parents is 200 and it is those individuals that will be used in crossover and mutation. The mutation probability is used so I don't mutate all. The mutation probability is set to 0.2 and if and if the random number, that are selected to compare with, is lower than the mutation probability, the program will mutate.

Illustrate in a figure how the performance of the population evolves with generations

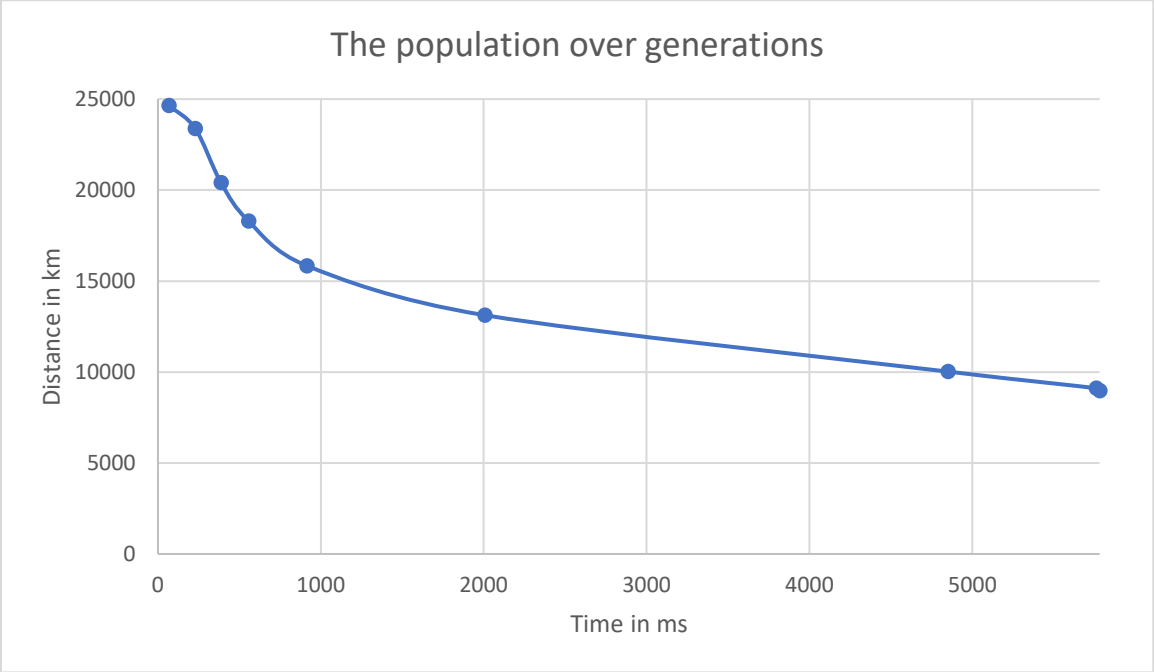
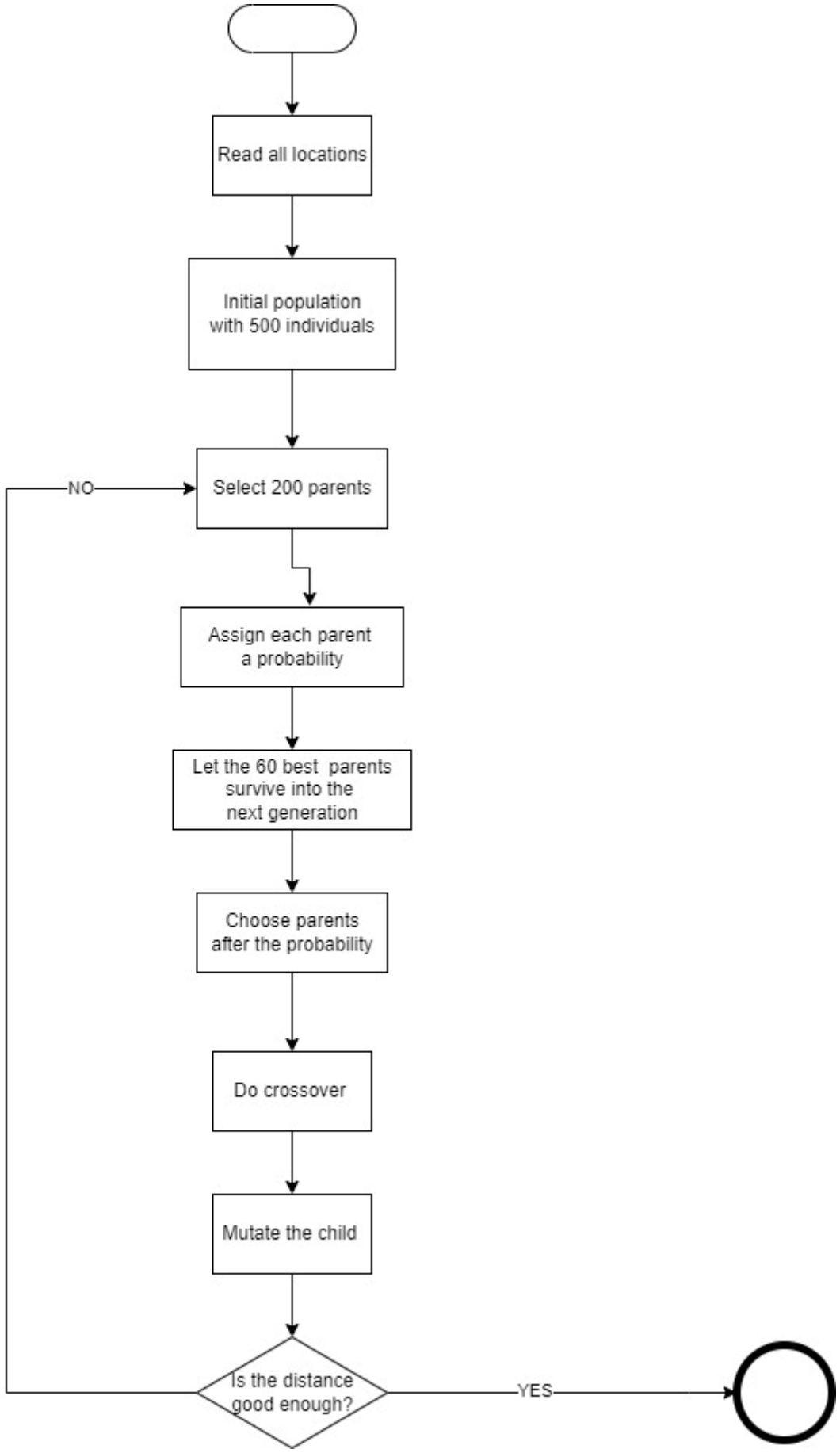


Figure 1: how the population evolves over time.

This shows how the population returns individuals with a high fitness at first, but the more time that goes by, the lower fitness value the population can find.


Read all locations
Initial population with 500 individuals
NO
Select 200 parents
Assign each parent a probability
Let the 60 best parents survive into the next generation
Choose parents after the probability
Do crossover
Mutate the child
Is the distance good enough?
YES

The picture above explains how the genetic algorithm runs. It starts by reading all the locations and splitting the string into id, X- and Y- coordinates. An initial population is created which consists of 500 individuals and 200 of those are selected as parents. Each parent is assigned a probability of being mated. I keep the 60 best parents to continue into the next generation and choose parents to be mated after the assign probability. The children can be mutated if the probability is true that runtime. I check if the distance is correct, if it isn't, I go back to selection and do the hole process over again. If the distance is low enough, I terminate.