ALGAV

3DE

1220514 – Liam Saliën

1220528 – Juuso Jalava

1220379 – Benjamin Moreira

1220459 – Patryk Pas

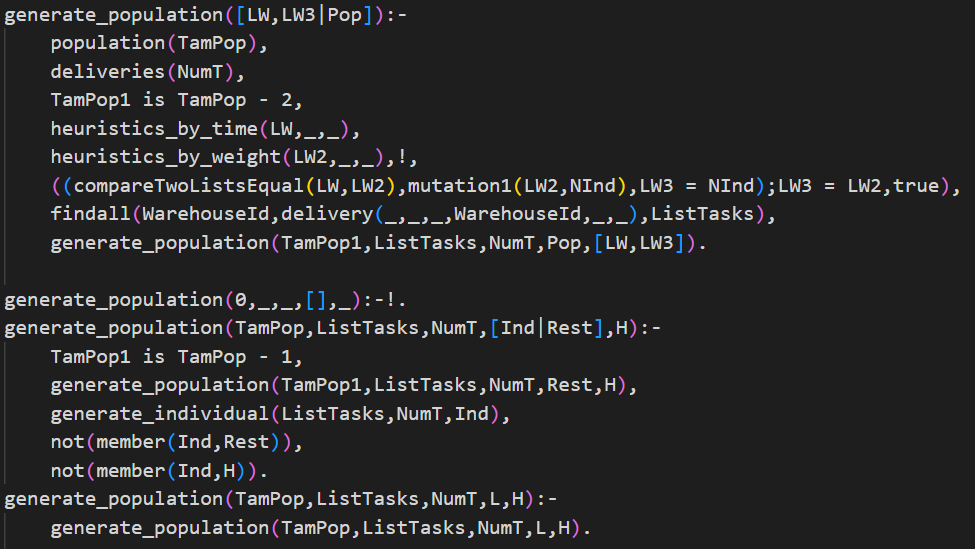
1220448 – Miika Kauppinen

# 1. Introduction

TBD

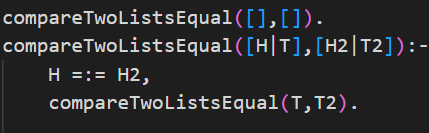
# 2. Creation of the initial population of the Genetic Algorithm (GA)

For this task, the example solution was adapted to our solution, and modified to include 2 different heuristics solution, and then randomly generated deliveries routes (from warehouse to warehouse):



The first generation population predicate (with one variable) will read the amount of deliveries that are in the system, and after that it will subtract the amount of individuals that are expected by 2, as it was required to include two solutions obtained by two different heuristics. The two heuristics used in our solution are by time, and by weight.

Furthermore it checks if the two solutions doesn’t equal each other (which uses a predicate that compares every single item in two lists).

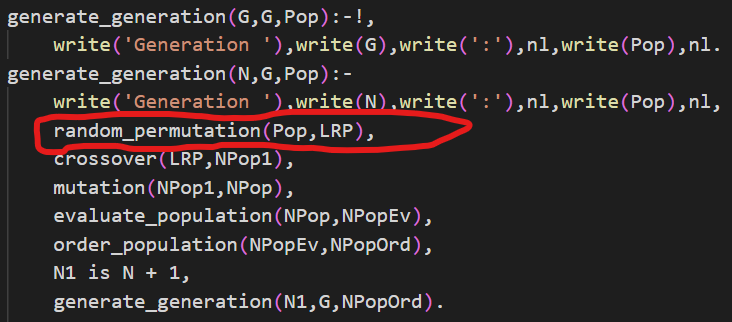


If this is the case, a mutation will be done on the second solution (obtained by the weight heuristic).

After this, for the rest of the individuals, another predicate will be called (and will include a list that already contains the two obtained solutions).

This predicate will be recalled until the TamPop is 0, which means we reached the amount of individuals we had to create. It will also check every single time if the generated solution is not already a part of the randomly generated deliveries routes list, and of the list that contains the two solutions obtained by the heuristics.

# 3. Random Crossover between individuals of the population



For this task the generate generation predicate was modified. The build-in predicate random\_permutation/2 is added after the current generation and their individuals are written to the screen.

Right before the written population will go through crossover and mutation etc, it is shifted by this build in predicate, so that the order will be different. Because of this, crossover doesn’t always happen between the first and second warehouseId, etc., etc.

*Example of this predicate in working:*

If we use random\_permutation on the following list:

*[[9,11,1,8,3]\*463.7758474576272,[11,8,1,3,9]\*486.5362288135594,[11,8,3,1,9]\*508.935593220339,[3,1,8,9,11]\*510.73241525423737,[11,3,9,8,1]\*537.482627118644]*

It might lead to this result:

*[[11, 8, 3, 1, 9]\*508.935593220339, [11, 8, 1, 3, 9]\*486.5362288135594, [9, 11, 1, 8, 3]\*463.7758474576272, [11, 3, 9, 8|...]\*537.482627118644, [3, 1, 8|...]\*510.73241525423737]*

This way our solution works more randomly, as well as the crossover takes place more randomly.

# 4. Selection of the new generation of the population

TBD

# 5. Efficacy Analysis comparing the better individual of the created GA compared with the better from the initial version of the GA.

TBD

# 6. Parametrization of the ending condition of the AG

TBD

# 7. Use of the GA to handle several trucks, representing in the same cromossome the deliveries of the several trucks.

TBD

# 8. Study of methods of Machine Learning

TBD

# 9. From a solution obtained from the GA it is envisaged to be able to allow dynamic changes

TBD

# 10. Conclusions

TBD