Solving CVRP with Random Mutation Hill-Climbing and ERC

Sergio Castillo Hernández
University of Bristol

Background

Among the different variants presented in the hill climbing algorithm, one that provides better results within less time is the random mutation hill climbing optimization.
The benefit of this algorithm is that generates a prospect solution based on a random condition instead of verifying one neighbour or a couple at a time which reduce the amount of time to find an optimal solution.
Different types of mutations can be found when using a genetic algorithms, these mutations can be applied allowing the RMHC a better way to find a lower cost.

Methods

This solution implements 5 algorithms:
1)Nearest Neighbour.- To get a baseline solution.
2)Random Mutation Hill-Climbing.- Which is mainly used to get the lowest cost.
3)Swap.-One of the two algorithms used to generate a prospect solution and find a new one.

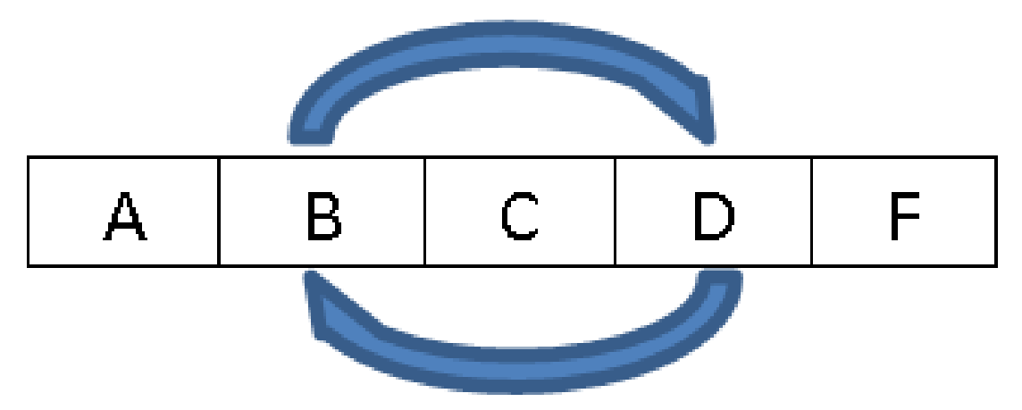

Figure 1. swap mutation representation.

4)Insert.- the second algorithm to mutate and return an offspring.

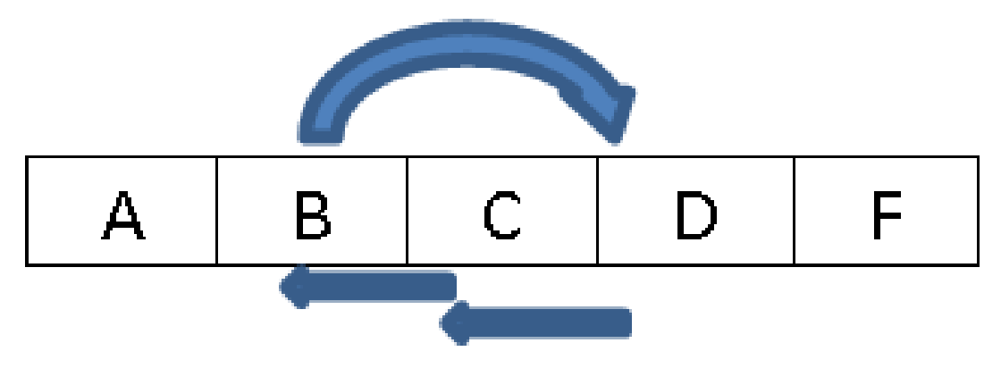

Figure 2. insert mutation representation

5)Edge Recombination Crossover(ERC).- to generate a complete new solution based on a random path and the best solution. Used when a lower cost than the current have not change in certain period of time.

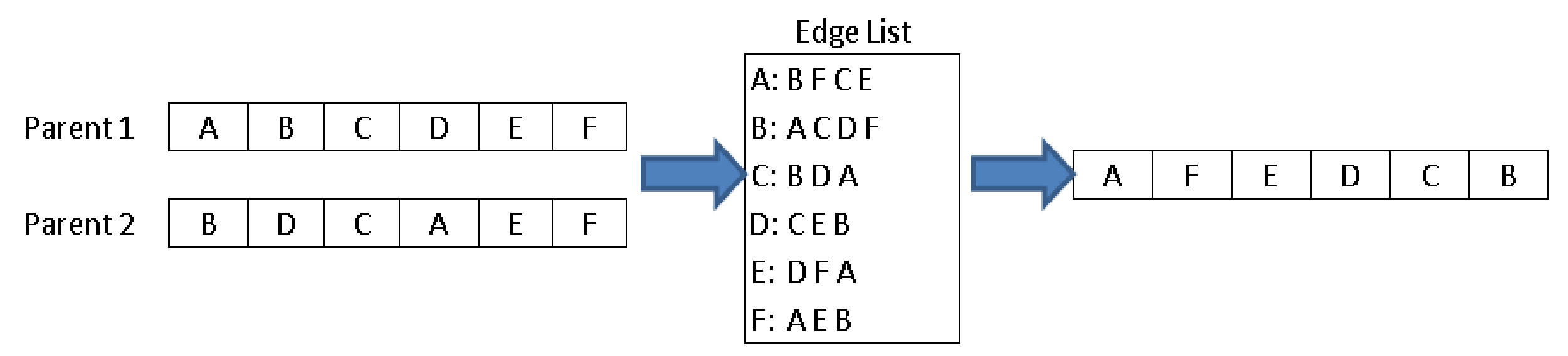

Figure 3. ERC representation.

How the implementation works:

1)Generate a random solution, or a baseline solution with algorithm 1.

2)Start to find the best solution n number of times using algorithm 2.

3)The first solution start to mutate using 3rd or 4th algorithm. Both algorithms have equal opportunity to be chosen.

4)If the mutations have not found a better result in n times; generates a new possible solution using the 5th algorithm.

Results

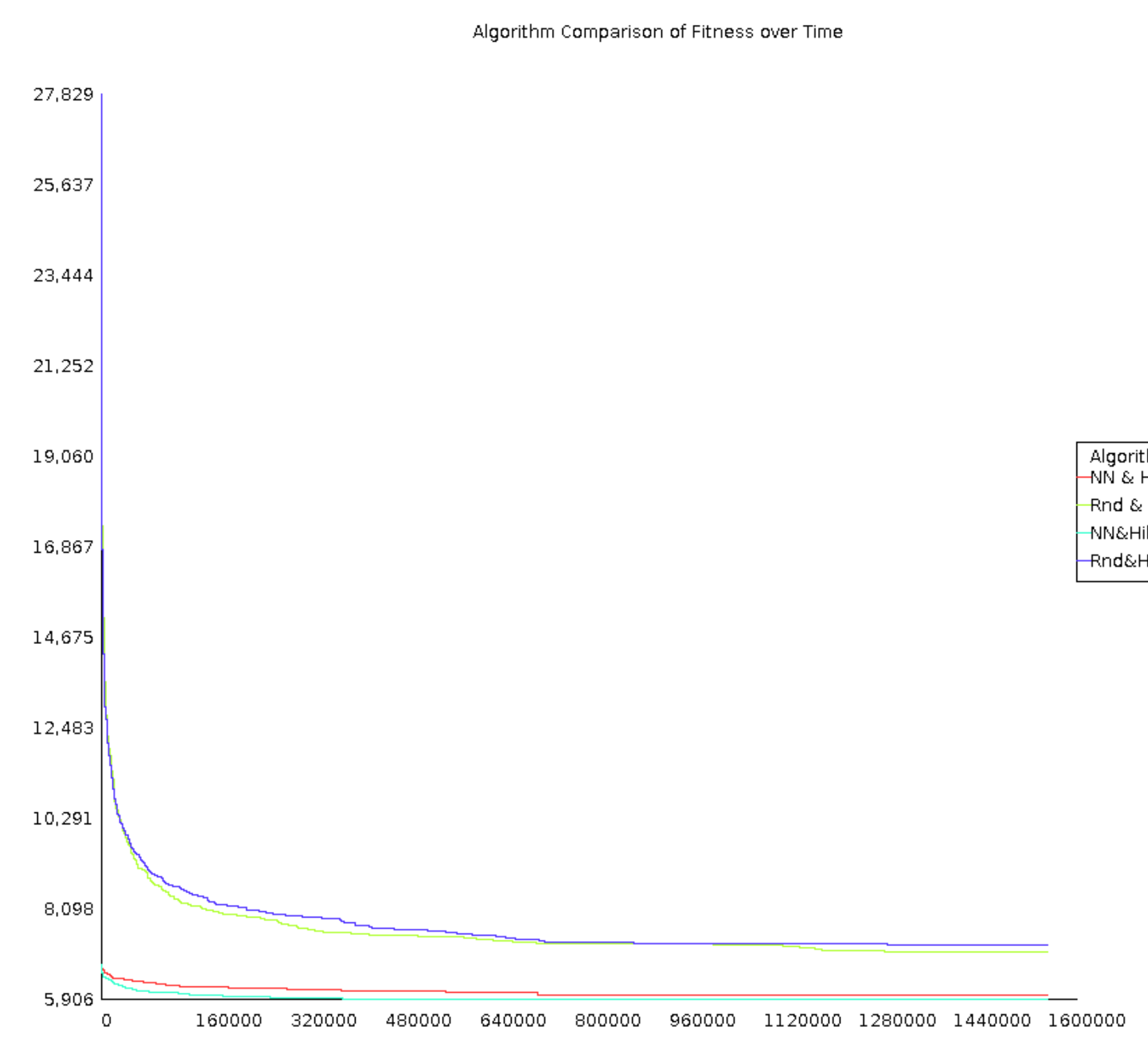

Figure 4. Methods comparison.

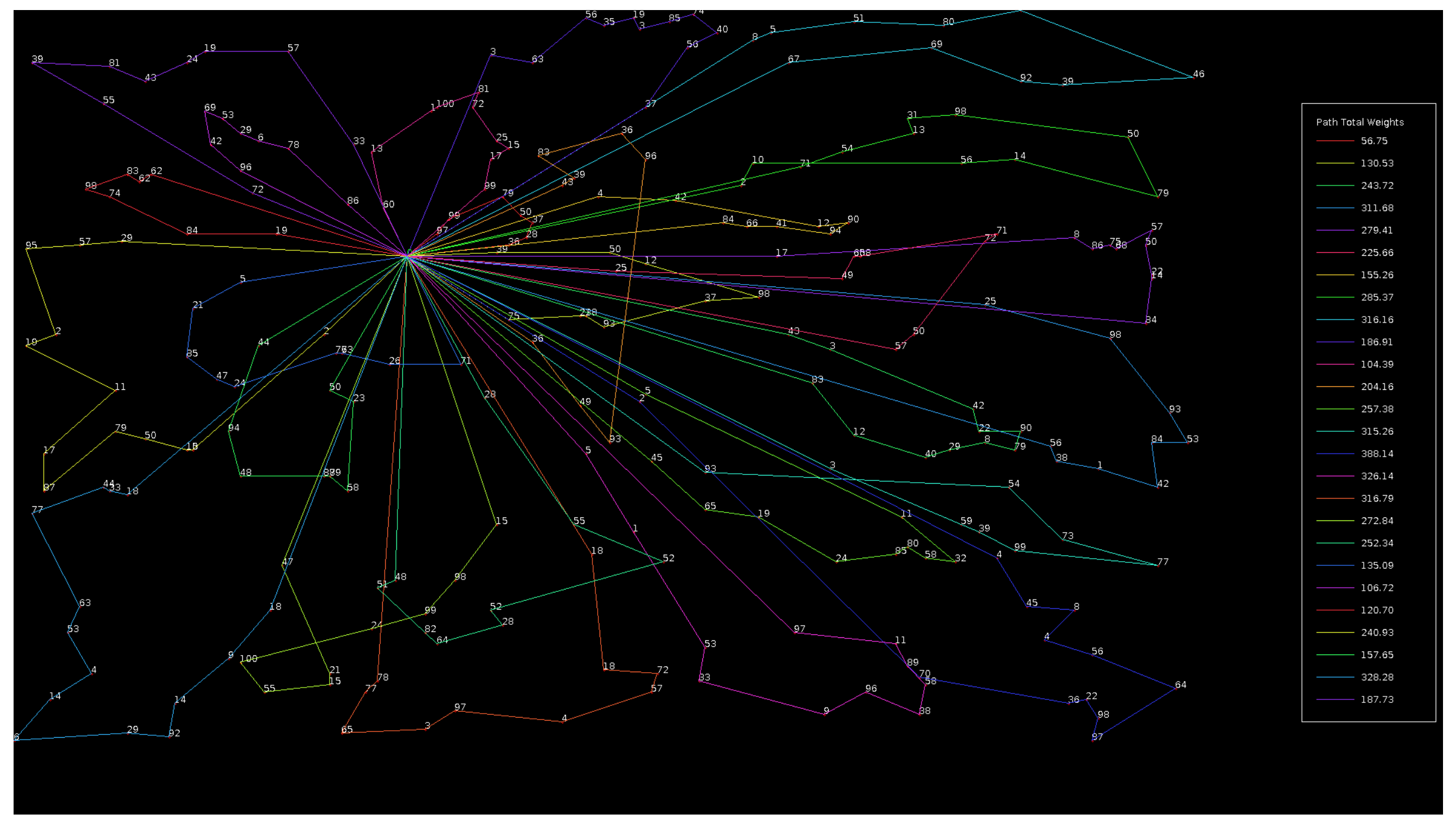

Figure 5 Routes and cost from best result. 26 trucks used.

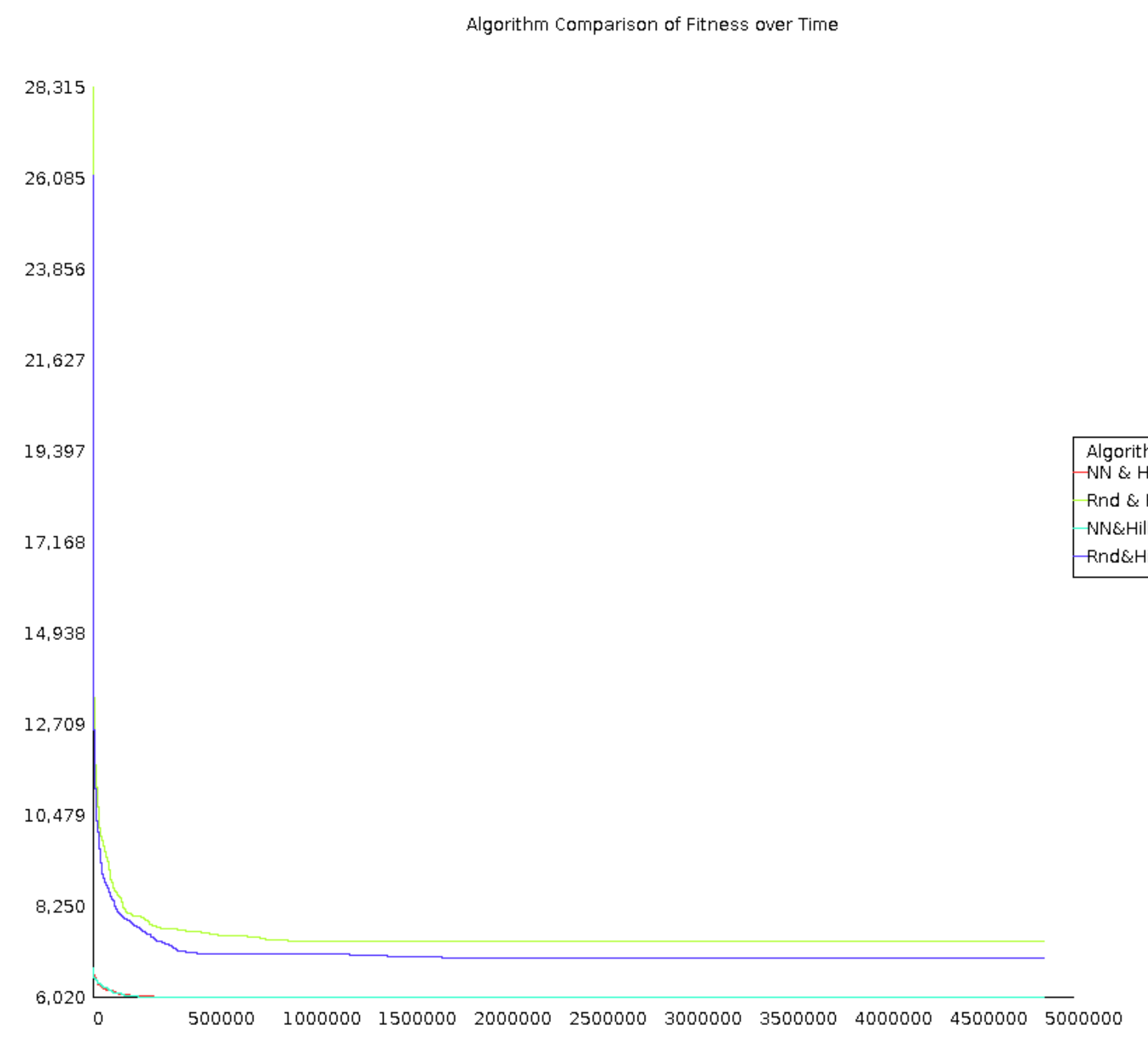

Figure 6. Solution performance on a long term.

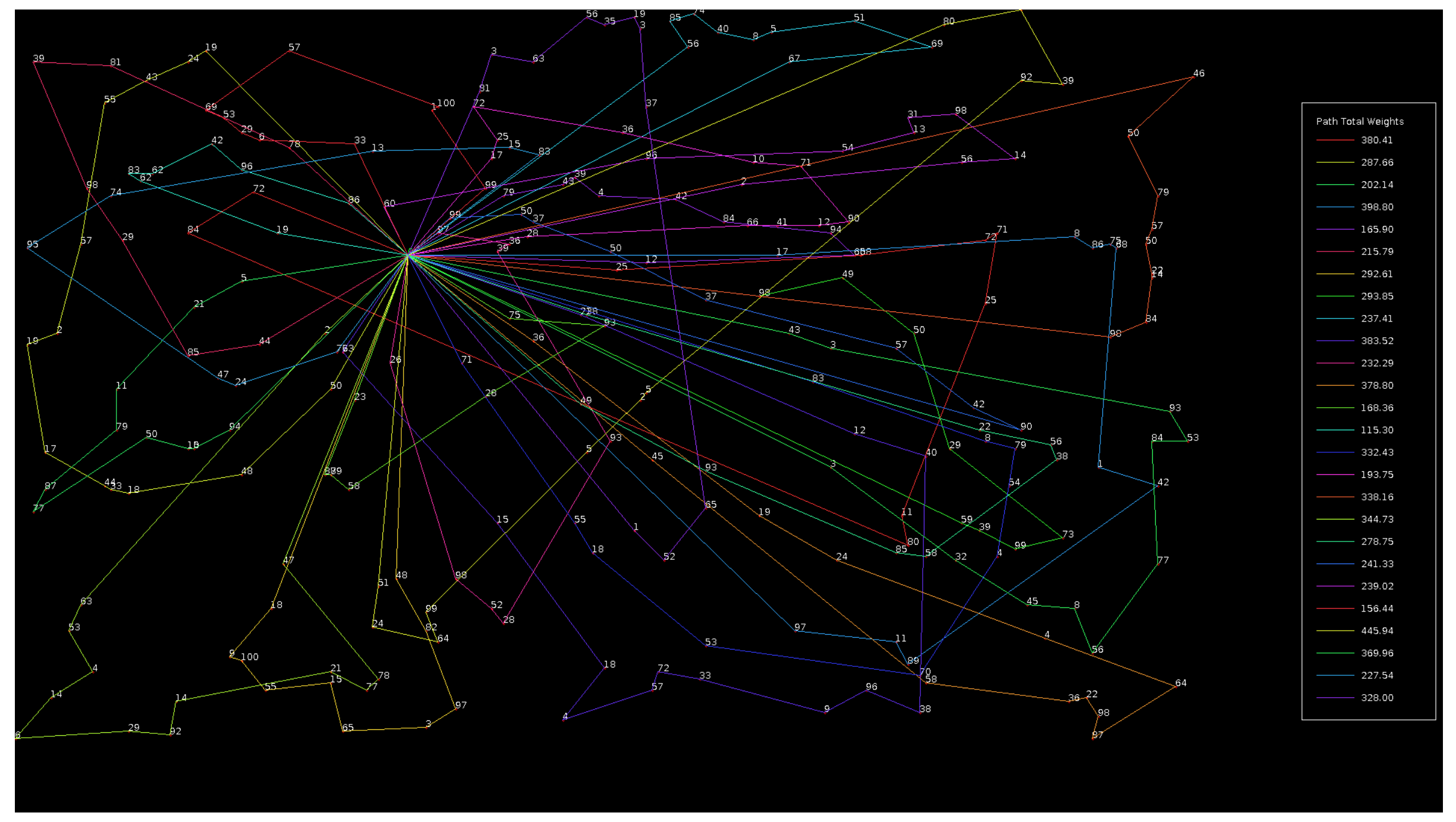

Figure 7. Random routes and cost.

Discussion

As seen on figure 4, starting with a non random solution as baseline helps the RMHC to generate a better result.
Combining these algorithms the cost can be reduce around 20%. This solution return a lowest cost around 6050 ± 100.
Using a random solution as starting point the reduction cost percentage is higher than using the nearest neighbour algorithm, the percentage reduction is approximately 75%, normally a random solution starts with a cost of ~27000 and improves up to ~7000.
Even with a great percentage reduction, starting with a random solution can not get a lower cost than our initial cost using the nearest neighbour algorithm.

When RMHC is not capable to find a better solution during 500 consecutive times the ERC creates a new path based on a random parent and the fittest as second parent, in the short term the ERC did not probe to find a better solution, but during testing, it was the only one with a random initial solution to find a lower cost after iteration 1,500,000 as seen on figure 6.

Comparing figures 5 and 7, nearest neighbour removes long route groups and provides a more orderly initial solution, for instances simulating flower petals, a random initial solution could group customers as far as the most distant customers.

Conclusion

The nearest neighbour algorithm probes to help reducing the time and ordering a solution before RMHC starts running, however, nearest neighbour is not the best deterministic solution to find the lowest cost and some improvements could be taken implementing instead, the Clarke & Wright algorithm for example.
Using swap and insert as mutant agents creates a better optimal than any other tested mutations combined or working as individually.
Another prospect mutation was used(inversion) but only increased the cost, working alone or either with swap or insert.
ERC tries to solve the problem when the algorithm stuck in a local optima, after tested, could found a lower cost but without being reliable or consistent, maybe other algorithms like 2-Opt could prevent this problem.

References

1.http://www.lcc.uma.es/~eat/pdf/evocop04.pdf.
2.http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.8466&rep=rep1&type=pdf
3.http://papers.nips.cc/paper/836-when-will-a-genetic-algorithm-outperform-hill-climbing.pdf
4.http://www.rubicite.com/Tutorials/GeneticAlgorithms/CrossoverOperators/EdgeRecombinationCrossoverOperator.aspx
5.https://github.com/daentech/CVRP/blob/master/src/com/github/daentech/algorithms/SimpleGA.java