# Algorithm 1: Evolutionary Approach using Pure Mutation.

The algorithm was inspired by a paper called a new evolutionary approach to cutting stock problems with and without contiguity [1]. In this approach a population of organisms are created, each one represents a solution.

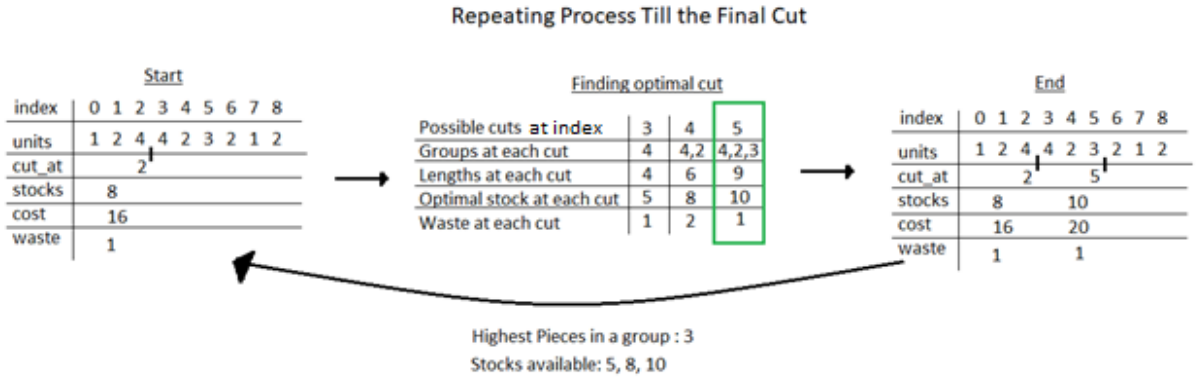
**Chromosomes of an organism (excluding index and cost)**

In the organism, cost refers to the total cost of the solution, and another property is total waste which refers to the total waste of the solution.

| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| units | 1 | 2 | 4 | 4 | 2 | 3 | 2 | 1 | 2 |
| cut_at | | | 2 | | | 5 | | | 8 |
| stocks | 8 | | | 10 | | | 5 | | |
| cost | 16 | | | 20 | | | 10 | | |
| waste | 1 | | | 1 | | | 0 | | |

**Process of choosing cuts.**

The highest possible pieces in a group are calculated. Starting from the unit after the previously cut one, get the lengths of the combination of the next k units, with k increasing by 1 until the maximum possible pieces in a group have been reached or the group length is larger than the largest stock. Find the optimal stock that goes with each group length and retrieve the waste, the cut with the lowest waste and most units in it is chosen. Due to all stock's length being proportional to its cost, being bias to a certain stock was not needed, selecting the highest stock was used to shorten the number of cuts. This method is beneficial as the best possible cuts can be generated.



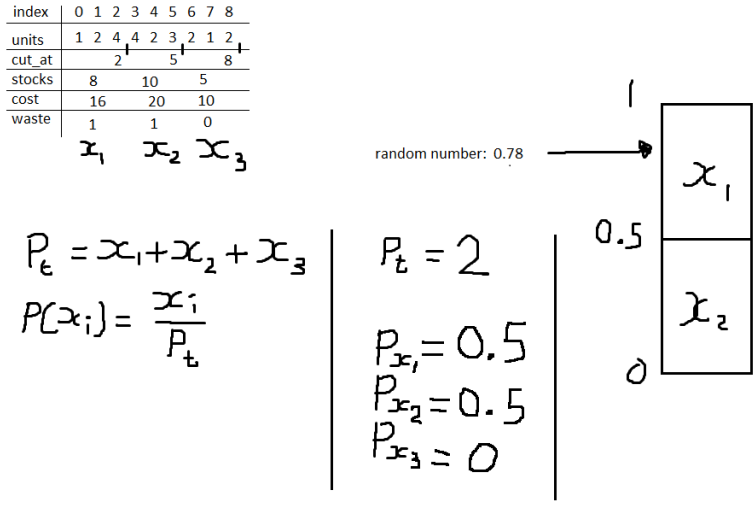

**Evolutionary algorithm**

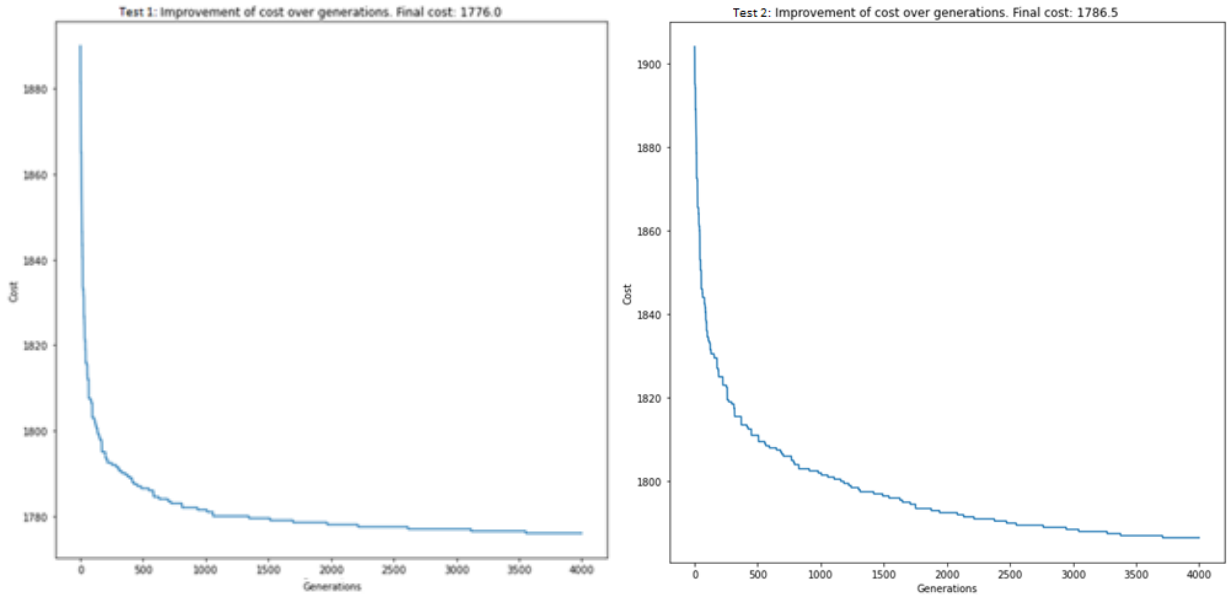
Each generation has all organisms clone themselves and then have those clones mutate twice. Having a clone mutate instead of the original organism allows the algorithm to allow the population to only improve or stabilise. For survivor selection I used as suggested by the paper, tournament selection. This allows the fittest members of the population to advance. Each organism had a chosen random number

of opponents to face off against, the fitness function which represents the waste of a solution was used to choose the winner. Half of the population with the most wins went on to the next generation.

**Three Point Swap (3PS) is used to mutate an organism.**

A cut represents a group of units that have been cut out of a stock. Mutating an organism involves picking a random unit (A) in a random cut. Two different cuts are selected and a random unit (B and C) for each cut is chosen, cuts with no waste will not be chosen. In the paper it suggests giving cuts with low waste a higher possibility of being chosen (Test 2). After experimenting I have found that giving cuts with high waste a higher probability generates better solutions (Test 1). Lastly, A 3PS is completed where A swaps with B, and B swaps with C.




Algorithm 2: Evolutionary Approach using Mutation and Recombination.

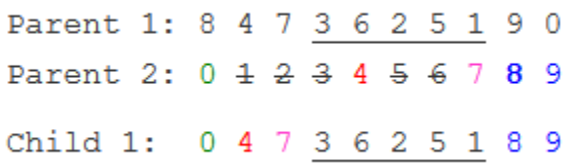
This algorithm is like algorithm 1, but recombination is introduced which replaces cloning with parents.

**Evolutionary algorithm**

A population is initiated and for each generation pairs of parents are selected to create 2 offspring till the population doubles. Proportional roulette wheel selection is used to select the parents. The probabilities of selecting a parent can be seen as spinning a roulette wheel with the size of the segment for each parent being proportional to its fitness. Waste is used as a fitness function over cost as the cost can never reach 0. Solutions which are closer to the optimal cost will be prioritised more as the effect of the inverse of waste is greater than the inverse of cost. As a parent can be picked multiple times, the better parents will be chosen more often than the poorer ones, thus fulfilling the requirements of survival of the fittest [2]. For survivor selection, the same tournament selection from solution 1 is used.

**Recombination**

With recombination in this problem, it is not viable to take the first half of the units' chromosome from a parent and the send half of that chromosome from another as the offspring will have duplicated or lost units. When accounting for this, the recombination method I choose to create offspring is Order 1 Crossover. A section from the first parents' unit's chromosome is chosen and placed into the offspring in the same position. Next the remaining units are taken the second parent to fill in the missing places in the offspring. This is one of the fastest recombination operators which means a lot of generations can be processed in small amount of time [3].




[3]

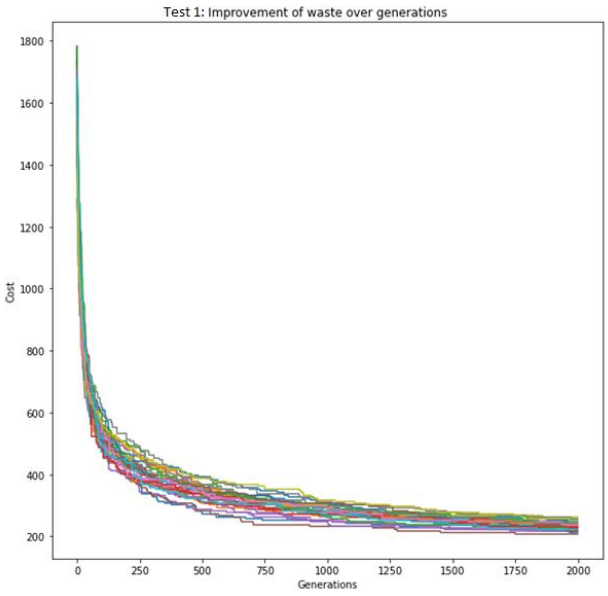
## Methodology

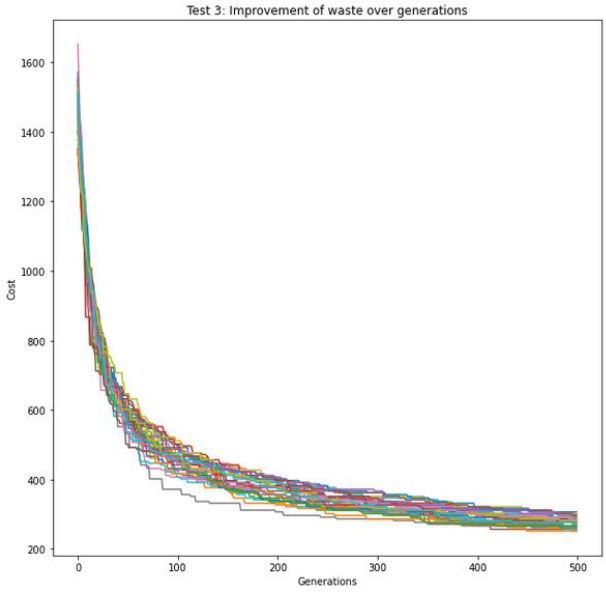
Both algorithm share 3 changeable parameters: generations, size, and opponents. I will be testing to see which combination of values for these parameters produce the best results. As the algorithms could run for days and get an almost optimal solution, a budget of 60 seconds has been set. As the sum of the length of all units is 17633, theoretically, the smallest waste can ever be is 2 (146 stocks of 120, and 1 stock of 115). This will be the target value; the goal would be to get as close of possible to it. Other changeable parameters would be the mutation chance in solution 2. As results may vary when an algorithm with set of parameters is run multiple times, I will be running each set 30 times to get a range of results.

Algorithm 1

| Test | Generations | Size | Opponents | Minimum waste | Mean waste | Maximum waste | Standard deviation | Average seconds to completion |
|---|---|---|---|---|---|---|---|---|
| 1 | 2000 | 10 | 5 | 207 | 232.8 | 262 | 14.09 | 56.86 |
| 2 | 1000 | 20 | 10 | 217 | 243 | 267 | 12.94 | 55.17 |
| 3 | 500 | 40 | 20 | 252 | 277.7 | 307 | 14.82 | 58.46 |

When testing for the best parameters, I discovered that a trend where a small population with a larger number of generations produced the better result in under 60 seconds. The best results came from test 1 as it had the lowest minimum, mean, and maximum waste.

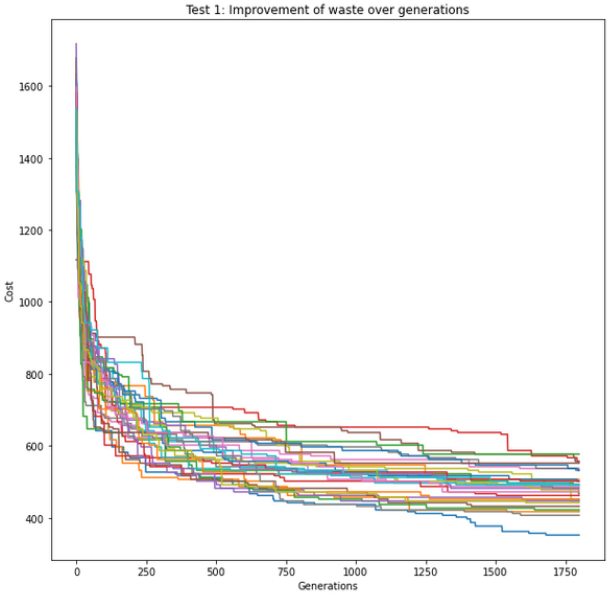


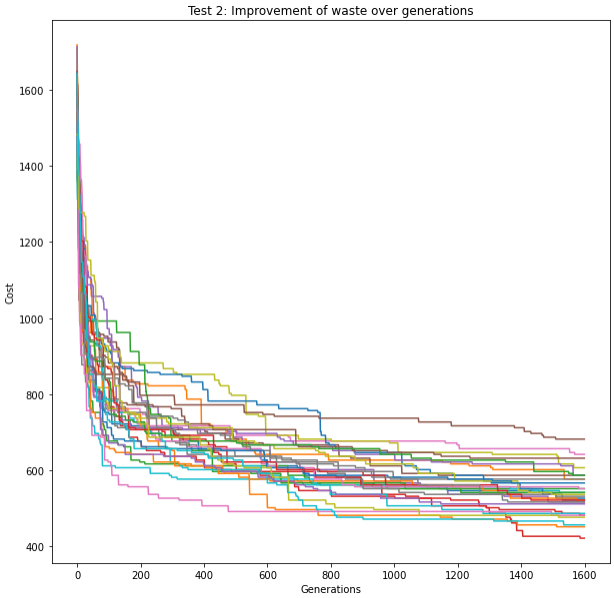



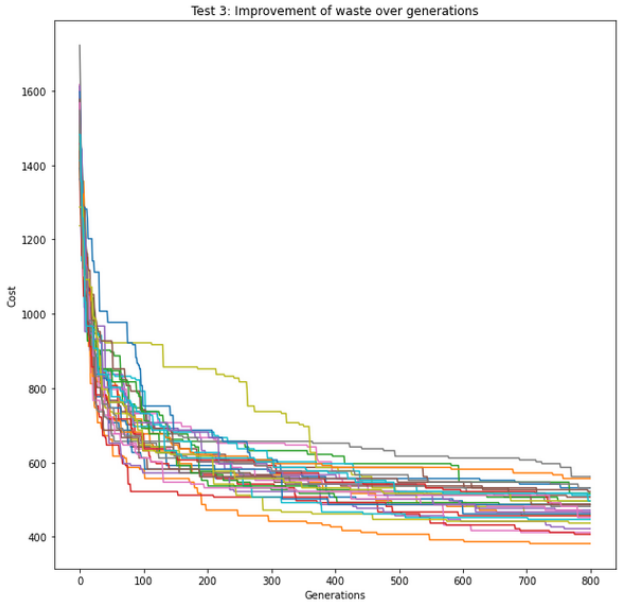

Algorithm 2

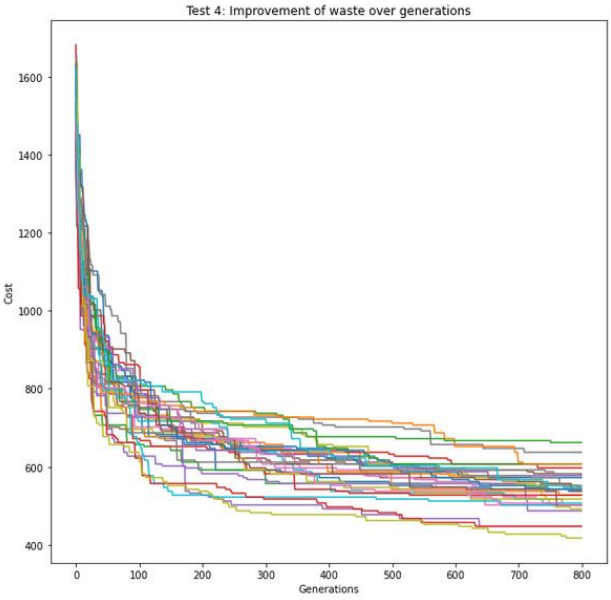
| Test | Generations | Size | Opponents | Mutation chance | Minimum waste | Mean waste | Maximum waste | Standard deviation | Average seconds to completion |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1800 | 10 | 5 | 0.3 | 352 | 481.33 | 577 | 49.83 | 61.36 |
| 2 | 1650 | 10 | 5 | 0.7 | 422 | 540 | 682 | 57.45 | 58.45 |
| 3 | 800 | 20 | 10 | 0.3 | 382 | 478.8 | 562 | 41.92 | 59.91 |
| 4 | 800 | 20 | 10 | 0.7 | 417 | 536.67 | 662 | 54.65 | 57.83 |
| 5 | 450 | 40 | 20 | 0.3 | 387 | 483.33 | 612 | 54.04 | 62.18 |
| 6 | 450 | 40 | 20 | 0.7 | 432 | 535.5 | 697 | 64.18 | 63.72 |

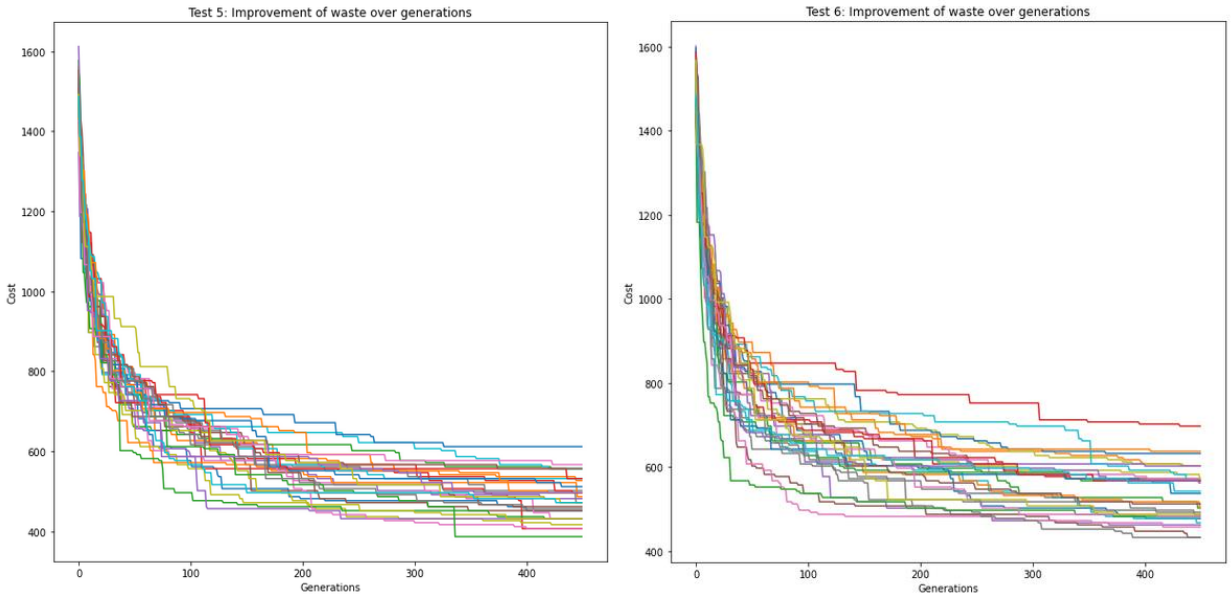
When testing for the best parameters, I discovered that a trend where a lower mutation chance produced a lower waste, this excludes test 2, 4, and 6 from being selected to represent the algorithm. Tests 1, 3, and 5 had very similar means but test 3 had a much lower standard deviation then the rest.


Test 1: Improvement of waste over generations
Cost
Generations
1600
1400
1200
1000
800
600
400
0
250
500
750
1000
1250
1500
1750


Test 2: Improvement of waste over generations
Cost
Generations
1600
1400
1200
1000
800
600
400
0
200
400
600
800
1000
1200
1400
1600


Test 3: Improvement of waste over generations
Cost
Generations
1600
1400
1200
1000
800
600
400
0
100
200
300
400
500
600
700
800


Test 4: Improvement of waste over generations
Cost
Generations
1600
1400
1200
1000
800
600
400
0
100
200
300
400
500
600
700
800




## Conclusion

Solution 1 is algorithm 1 with test 1 parameters, solution 2 is algorithm 2 with test 3 parameters, as previously stated, these produced the best results. The two parts of the results to consider are mean and standard deviation. Solution 1 produced the lowest minimum, mean, and maximum waste, the minimum waste for solution 2 did not even come close to the maximum waste of solution 1. A low standard deviation means a solution is very closely related to the average which makes it very reliable. Therefore, a person does not have to chance whether they get an extremely bad or good solution as all runs will be very similar. The standard deviation in solution 1 is about a third of the size of solution 2, which makes it more reliable. With comparison completed, as solution 1 outperformed solution 2 in both categories, it is the best solution.

## References

[1] K. Liang, X. Yao, C. Newton and D. Hoffman, "A new evolutionary approach to cutting stock problems with and without contiguity", *Computers & Operations Research*, vol. 29, no. 12, pp. 1641-1659, 2002. Available: https://www.cs.bham.ac.uk/~xin/papers/COR_LiangYaoNewtonHoffman.pdf.

[2] N. Razali and J. Geraghty, "Genetic Algorithm Performance with Different Selection Strategiesin Solving TSP", Proceedings of the World Congress on Engineering, 2011.

[3] "Order 1 Crossover Operator", *Rubicite*. [Online]. Available: https://www.rubicite.com/Tutorials/GeneticAlgorithms/CrossoverOperators/Order1CrossoverOperator.aspx. [Accessed: 23- Apr- 2021].