# AIML426 Assignment 1 – Debruconn

## Part 1 – Knapsack Problem

### Representation

Individuals in my Genetic Algorithm (GA) are defined as objects containing indices of knapsack items selected (shown in the image). Each of these indices can be used to construct a binary string to represent a GA individual.

### Fitness Function

Text

Description automatically generated with low confidenceThe fitness function I have chosen was shown on the lecture slides. This essentially maximises the value of the items but minimises the weight of the individual. If the total selected items weight is over the bag’s capacity, my fitness function will return a bad fitness result to make the current selection less likely to reproduce.

### Crossover

The crossover function I use will take 2 individuals from the population, select a random index from both individuals and then exchange a section each. As my objects are represented by recording indices of a GA binary string, crossover will exchange indices that aren’t necessarily consecutive.

### Mutation

The mutation function will remove one random item from an individual, and then populate it with another random item. This is achieved using sets in my program to avoid duplicated items.

### Selection

When selecting parents for the next generation, I use the NSGA-II selection algorithm. This algorithm works by selecting a large group of individuals first through tournament selection. Individuals are then selected based on a crowded-comparison operator using a Manhattan distance function to ensure that individuals are all mostly unique. By using the crowd-comparison operator, clustered individuals can be trimmed and therefore not dominate the next generation.

### Parameters

For my parameters I have:

|  |  |
| --- | --- |
| Number of generations | 150 |
| Elitism population | 30 |
| Children in each population | 100 |
| Crossover probability | 65% |
| Mutation probability | 35% |

### Results

#### 10\_269 File

|  |  |  |  |
| --- | --- | --- | --- |
| Run / Seed | Best Value | Weight | Standard Deviation (of best generation) |
| 1 | 295 | 269 | 52.16 |
| 2 | 295 | 269 | 89.89 |
| 3 | 295 | 269 | 88.21 |
| 4 | 295 | 269 | 88.15 |
| 5 | 295 | 269 | 90.14 |
| Mean (from past 5) | 295 | 269 |  |

|  |  |
| --- | --- |
| We can see from my convergence curve that it takes my program roughly 40 generations to reach convergence. The program on average finds an individual with roughly a 282 value after a couple generations. Because the dataset is so small, it is expected to be able to reach convergence quickly as there is only a few combinations possible to reach the optimal value. I think my program performs to standard, there however may be more optimal parameters to reach convergence faster. |  |

#### 23\_10000 File

|  |  |  |  |
| --- | --- | --- | --- |
| Run / Seed | Best Value | Weight | Standard Deviation (of best generation) |
| 1 | 9763 | 9774 | 3200.85 |
| 2 | 9762 | 9763 | 3088.25 |
| 3 | 9756 | 9766 | 3195.16 |
| 4 | 9753 | 9754 | 3144.62 |
| 5 | 9767 | 9768 | 3109.54 |
| Mean (from past 5) |  |  |  |

|  |  |
| --- | --- |
| Due to the high values in the data, I enlarged the section of the curve in the graph.  The program very quickly gets individuals with values of 9700, which is to be expected as there are only 23 items in this dataset. There is a gradual increase in higher value individuals from generation 20 to 150. I suspect that with a higher generation number, that the optimal value will be reached in roughly 100 more generations following the current trend. |  |

#### 100\_995 File

|  |  |  |  |
| --- | --- | --- | --- |
| Run / Seed | Best Value | Weight | Standard Deviation (of best generation) |
| 1 | 1487 | 990 | 482.66 |
| 2 | 1484 | 978 | 471.67 |
| 3 | 1513 | 962 | 492.04 |
| 4 | 1428 | 990 | 457.90 |
| 5 | 1512 | 953 | 473.85 |
| Mean (from past 5) |  |  |  |

|  |  |
| --- | --- |
| This curve is the most realistic convergence curve compared to real-world examples. There is a larger increase in fitness at the start that gradually slows down. This is likely due to the larger dataset and thus higher variability in individuals. The 2nd and 3rd generations remain at 0, this is due to every individual in the population either selecting no items or going over the bag capacity. From the convergence curve I think my program behaves adequately, so I wouldn’t need to adjust the GA parameters further. |  |

## Part 2 – Genetic Algorithm for Feature Selection