Laboratory Assignment 6
Applying Genetic Algorithms

CSC372-M72: Optimisation

2022-23

1 Objectives.

- To apply Genetic Algorithms (GAs) to solve an interesting combinatorial problem known as the knapsack problem.

2 Background: The Knapsack Problem

In this assignment, you will deal with the famous Knapsack problem: given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is at most a predefined limit, and the total value is maximised. This is a common optimisation problem that many of us face when we are going on a trip where we have to decide what we can take in a case that meets the weight restrictions of our carrier.

The most common description of this problem is known as the $0-1$ Knapsack problem. Imagine that we have n items that we can pick from, and whether the ith item is picked or not is represented by a binary variable $x_i \in \{0, 1\}$, with the natural meaning that if $x_i = 0$ then the ith item has not been picked, and *vice versa*. We, therefore, have a binary decision vector with n elements $\mathbf{x} = (x_1, \ldots, x_n)^\top$.

Furthermore, we are given that the ith item has a weight w_i and a value v_i. Thus, we have the weight vector $\mathbf{w} = (w_1, \ldots, w_n)^\top$ and the value vector $\mathbf{v} = (v_1, \ldots, v_n)^\top$.

Now, the maximisation problem is formally defined as:

$$\max_{\mathbf{x}} f(\mathbf{x}) = \sum_{i=1}^{n} v_i x_i, \tag{1}$$

subject to:

$$g(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i \leq W, \tag{2}$$

$$x_i \in \{0, 1\}. \tag{3}$$

Of course, we can apply the static penalty approach that we have used in one of the laboratory exercises and transform the objective function as follows:

$$\max_{\mathbf{x}} \phi_{static}(\mathbf{x}, f(\cdot), g(\cdot)) = f(\mathbf{x}) - r \times \max[0, g(\mathbf{x}) - W]^\beta, \tag{4}$$

subject to:

$$x_i \in \{0, 1\}. \tag{5}$$

For this assignment, imagine that you have gone to do your weekly fruit shopping with a bag that allows you to carry up to 1200 gms. Now, you have made a list of things that you might need. The list is given below:

Item ID	Description	Weight (gms)	Value (£)
1	Banana	600	1.35
2	Raspberries	180	2.25
3	Blueberries	250	2.5
4	Medjool Dates	500	4.50
5	Pineapple	400	1.75
6	Mango	250	2.5
7	Watermelon	380	2
8	Strawberries	400	3.5

Your goal is to spend as much money as possible while adhering to the limit of carrying 1200 gms. You will determine the items that you ought to buy to meet this goal using a Genetic Algorithm.

Before attempting the tasks below, it would be useful for you to revise the materials covered in Lectures 7 (part 2) and Lecture 8.

3 Tasks

1. Write a function in Python that takes a decision array $\mathbf{x}$, a weight array $\mathbf{w}$, a value array $\mathbf{v}$, a limit on weight W, a penalty multiplier r and an exponent β, and returns the result of $\phi_{static}(\mathbf{x}, f(\cdot), g(\cdot))$ in equation (4).

2. Implement a function for generating a random population (as a 2-dimensional binary array). This function should take the number of rows and number of columns as parameters.

3. Implement the truncation selection function with $k = 4$.

4. Implement the one-point crossover function.

5. Implement the bitwise mutation.

6. Implement the Genetic Algorithm discussed in the lecture, and run it using the following hyper-parameters:

Hyperparamter	Value
Population size, m	10
Number of generations, K	1000
Crossover probability, p_c	0.8
Mutation probability, p_m	0.2

For the function in equation (4), use $r = 2$ and $\beta = 1$.

7. Plot the best individual's fitness vs the number of generations.

8. Report the best solution that you have found.

Warning: Please be careful that your implementation is appropriate for the problem at hand.