# 55-508226 Algorithms and Data Structures

# Project 5 (30%)

# (Individual Work)

The objective of this project is to enable you to implement Genetic Algorithm to the paradigm problem. In this context, you are required to carry out the following tasks:

1. Design your Genetic Algorithm (GA) based on the skeleton program provided.
2. Implement your GA to the paradigm problem and run some experiments to answer the research questions outlined in the research question section.
3. Analyse results and report your findings in the portfolio.

**Note**:

* The deadline for completion is the same as the submission date of your portfolio.
* Please read and digest this document line by line.
* This project requires self-research about the subject and discussions with tutors. Should you have references for the work, **cite** them in the report.

## Background

A **genetic algorithm** is a search heuristic method that is inspired by Charles Darwin’s theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

Diagram

Description automatically generated

Five phases are considered in a genetic algorithm.

* Initial population
* Fitness function

These phases should be in a loop of generations.

* Selection
* Crossover
* Mutation

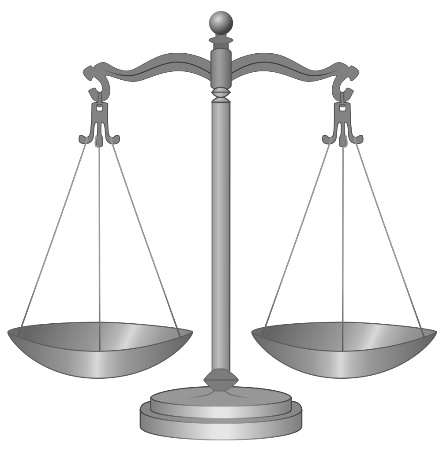
## Timeline Description automatically generated

Graphical user interface, timeline, Excel

Description automatically generated

## The Scales Problem

The scales problem is one of the paradigm problems in heuristics and optimisation used to test algorithms.



Suppose you have *n* objects of various weights, and you are required to split those objects into two equally heavy piles (or as equal as possible). Refer [*Problem Solving - Balance a Scale Problem*](https://youtu.be/-n-A5qcO398)for a video explanation.

In this project, you are given with a list of random numbers as the weights (**Appendix**).

Using Genetic Algorithm, write a program that optimises,

Where,

*L* is the sum of left-hand side weights

*R* is the sum of right-hand side weights

## Genetic Algorithm Representation

We represent the solution as an *n* length binary string/integer where:

* + A zero (0) in position *i* means that weight *i* is on the left side of the scales
  + A one (1) in position *i* means that weight *i* is on the right side of the scales

A picture containing text, clock

Description automatically generated

Each solution has its fitness value that computed from the equation. Those solutions with the least fitness values will survive for the next generation. Example:

If your defined population is 10 candidates. The next generation pool will be 10 + reproduction via cross over (2 candidates) + a mutant. Thus, the next generation will have 10 best candidates from those 13.

This process iterates until the number of generation complete.

## Instructions

With the skeleton codes and sample dataset provided (**Appendix**), write a program to solve the problem using Genetic Algorithm. Answer the questions in the Research Question section.

## Research Questions

1. Design and explain your GA algorithm. Run your algorithm for experimental analysis (sufficient sampling – number of experiments) and demonstrate the results in a table that has maximum, minimum, and mean values of fitness. What are the optimal fitness values for the weights?
2. What are the solutions (chromosomes) for question 1? Tabulate the chromosomes and their fitness values in a table. Verify the fitness using a manual calculation (use excel sheet). Provide evidence in your report.
3. What are the values used for the following parameters in your experiments,

* Number of generations
* Crossover rate
* Mutation rate

What if you increase and decrease the values? Discuss.

1. Demonstrate in writing and/or ADL your crossover strategy for the GA.
2. Demonstrate in writing and/or pseudocode your mutation strategy for the GA. Discuss why mutation is important in GAs?
3. Run empirical experiments to find associate rules of crossover and mutation rates for another dataset of 50 (to be produced from the code provided) weights with the following fitness function,

Where,

*L* is the **product** of left-hand side weights

*R* is the **sum** of right-hand side weights

Analyse and summarise your findings in the report. Plot into graph/s to prove your findings. Discuss your findings.

1. A video recording to demonstrate the work is part of the marks.

# **Appendix**

## Dataset



## Skeleton Code

<https://github.com/zairulmazwan/Project_GeneticAlgo.git>

There are 2 files, *Data.java* and *GeneticAlgo.java*. The *Data.java* file consists of a few methods that you can use to produce a dataset, read data, write data/results. Add more methods to this file to collect experiment results in an efficient way.

Whilst the *GeneticAlgo.java* file is the file that you need to work on. You can write a program with C# if you are more convenient with the language.

You might find the skeleton code is not the best version of GA, however, it can produce good results for the problem. Feel free to add your own method into it or remove/edit if you found necessary.

## Technical Information

The following components might be useful for the project,

* Array
* ArrayList
* Random
* Hashtable
* Modulus operator
* Decimal Format
* Collections
* Sort

## Example of results

Following diagrams/table can be materialised in your report (for a first-class grade).

Diagram 1: Mean of Fitness of 30 Generations

Diagram 1 shows the algorithm was trapped in the local optimum where an early convergence was achieved at 12th generation.

Diagram 2: Mean of Fitness of 30 Generations with 0.3 and 0.7 rates for Crossover and Mutation

Diagram 2 shows the algorithm has a slightly effective search with the parameter values. However, trading off with time, the algorithm has a later convergence (25th generation). The chromosome is 0000010110011011011101100 (Fitness = 0.074).

|  |  |
| --- | --- |
| **Crossover rate = 0.3** | |
| **Mutation rate = 0.4**  **Generations = 30 (each experiment)**  **Experiment sampling = 25** | |
|  |  |
| Experiment | Fitness Mean |
| 1 | 20.096 |
| 2 | 15.709 |
| 3 | 72.041 |
| 4 | 64.199 |
| 5 | 68.719 |
| 6 | 57.199 |
| 7 | 87.704 |
| 8 | 48.573 |
| 9 | 86.698 |
| 10 | 60.946 |
| 11 | 39.348 |
| 12 | 58.511 |
| 13 | 22.201 |
| 14 | 21.073 |
| 15 | 80.697 |
| 16 | 49.325 |
| 17 | 23.918 |
| 18 | 65.755 |
| 19 | 59.118 |
| 20 | 27.279 |
| 21 | 29.603 |
| 22 | 75.229 |
| 23 | 93.600 |
| 24 | 28.988 |
| 25 | 85.615 |
| Mean | ? |
| Stdev | ? |

Which experiment gives the best solution?

Other possible diagrams/statistical values or research questions to consider:

1. Distribution (of reasonable samples) of means for certain crossover and mutation rate.
2. Histogram for (a) to see the pattern and the effect of the parameters to the search.
3. Standard deviation of the fitness mean.
4. What are the best range of crossover and mutation rates for the problem?

Grading Criteria

This work will be graded using the criteria given below.

| **Criterion** | **Marks** | **Artefacts (for each project unless otherwise stated in Notes)** | **What you need to do to get a mark in the first-class range** | **What you need to do to get a pass mark** |
| --- | --- | --- | --- | --- |
| An assessment of understanding of algorithm design | 30 | A set of algorithms expressed in ADL or any format | You are able explain your problem-solving strategy accurately and clearly.  Your problem-solving strategy is effective.  There is a very close mapping between your strategy and your algorithm.  Your algorithms are correct with respect to the specification, i.e., it accomplishes its task unambiguously and accurately.  Your algorithms are accurate in the way they manipulate data stored in your chosen data structures. | Your explanation of your problem-solving strategy lacks clarity.  Your problem-solving strategy is satisfactory but could be more effective.  Your algorithms are correct with respect to the specification, i.e., it accomplishes its task unambiguously and accurately.  Your algorithms are accurate in the way they manipulate data stored in your chosen data structures. |
| Results Analysis and Discussion | 40 | Tabulate experiment results  Analyse statistical values of the results (if applicable)  Elaborate the results and conclude the finding  Summarise the results for the problem  All questions are answered in the report | Present experiment results in tabular format so that empirical analysis can be carried out to justify your algorithms/experiments.  Evidence of good results are produced and comparable to the metric measurement used for the problem.  Critically summarise the result section for findings.  Use graphs or charts to visualise results to support findings.  Improvement to the current experiment is discussed in the report. | Results analysis section is required for projects that entail experiments for problem solving.  Result for each experiment is presented in the report.  Summarise the results for the problem. |
| An assessment of data structure design | 20 | A set of data structures expressed in terms of diagrams and sample data sets (where applicable) | Your choice of data structures is accurate.  You are able to produce a number of alternative data structures by either using data provided to you or by analysing given problem domains.  You are able to provide example data sets to illustrate how data may be stored using your chosen data structures (where appropriate). | Your choice of data structures is mostly accurate.  Alternative data structures are either not considered or not considered consistently.  You are able to provide example data sets to illustrate how data may be stored using your chosen data structures (where appropriate). |
| Incorporation of formative feedback | 5 | Written evidence of how formative feedback was evaluated and acted on | Formative feedback (verbal and written) provided by the teaching team during the development of each project is clearly outlined.  There is clear evidence of the evaluation of feedback received (e.g., in the form of a “to-do-list” with priorities) and recording the action plan using a software tool, such as Trello. | Formative feedback (verbal and written) provided by the teaching team during the development of each project is documented.  There is an action plan in place to address feedback. |
| An assessment of overall software, including testing | 5 | Evidence of how a software system was developed  Evidence of software testing and evaluation of results  A software system and its codebase  A video recording of software developed  Evidence of testing (unit test) | Your software is fit for purpose.  There is clear evidence that you followed a systematic approach to translate your algorithms to corresponding C#/Java programs.  There is clear and light-touch test specification in place for acceptance tests.  There is evidence of successful execution of the test specification and recording of its results.  Video recordings satisfactorily   * demonstrate the functionality of software. * outline any assumptions you made while developing the software | Your software is mostly fit for purpose. However, there is either no evidence of following a systematic approach to translate your algorithms to corresponding C#/Java programs, or evidence is not consistent.  There is a clear and light-touch test specification in place acceptance tests.  There is a video recording for each project, each of which satisfies the expectations laid out fairly adequately, if not completely. |