### **Report for Assignment 2: Employee Assignment Optimization Using Genetic Algorithm**

#### **1. Problem Overview**

This assignment focused on the optimisation of employee assignment in a company with 80 employees and 50 departments. Each department receives client requests, and the goal is to minimise the time clients wait for their requests to be processed. Specifically, the aim is to minimise both the **mean waiting time** and the **maximum waiting time**, with an 80% importance placed on the mean and a 20% importance on the maximum. A genetic algorithm (GA) was designed and implemented to find an optimal employee allocation across departments. The GA simulates various employee distributions and evolves these distributions over generations, aiming to achieve the objectives mentioned above.

#### **2. Genetic Algorithm Structure**

The genetic algorithm used in this assignment follows standard GA procedures. Below are the key components:

1. **Chromosome Representation**:
   * Each chromosome represents an employee allocation where the list of 50 integers corresponds to the number of employees assigned to each department.
   * The total number of employees is always adjusted to 80, ensuring no department is left empty.
2. **Fitness Function**:
   * The fitness function evaluates how well a given chromosome (employee allocation) meets the objectives. It is calculated as:

Fitness = 0.8 × Mean Waiting Time + 0.2 × Max Waiting Time  
Lower fitness values indicate better performance.

1. **Crossover**:
   * **Uniform crossover** was used, where for each gene (employee allocation in each department), there is a 50% chance that offspring will inherit it from one parent or the other.
   * **Single-point crossover** was also considered but did not perform as well in early experiments.
2. **Mutation**:
   * The **swap mutation** method was employed, which swaps the employee count between two randomly chosen departments. This mutation was applied with a variable mutation rate between **0.1** and **0.7** during the experiments.
3. **Selection**:
   * **Roulette wheel selection** was used, where chromosomes are selected for mating based on their fitness values. Better chromosomes have a higher probability of being selected.
4. **Parameters**:
   * Population size: 300
   * Generations: 50
   * Mutation rate: 0.7

#### **3. Experiments and Design Decisions**

Throughout the development of the genetic algorithm, several experiments were conducted to fine-tune the parameters and evaluate different design choices, including crossover techniques, mutation rates, selection methods, and population sizes. Below are the most significant experiments and their outcomes:

##### **3.1. Crossover Techniques**

I initially experimented with both **single-point crossover** and **uniform crossover**. The uniform crossover proved to be more effective, producing more diverse offspring due to the randomness of gene exchange at each locus, which helped explore a broader range of solutions.

* **Single-point crossover**: This method often produced similar offspring, limiting the diversity in solutions and leading to premature convergence.
* **Uniform crossover**: This allowed for greater variety in solutions by swapping genes randomly between parents, leading to better performance across generations.

##### **3.2. Mutation Rate**

The mutation rate was varied across different runs to observe its impact on solution quality. Lower mutation rates (0.1-0.3) produced less diversity, resulting in local optima. Increasing the mutation rate to **0.7** introduced more diversity in the population, leading to better overall results. However, rates higher than this led to excessive randomness, degrading performance.

##### **3.3. Selection Methods**

Next I compared **rank selection** with **roulette wheel selection**:

* **Rank selection**: Prioritised higher-ranked chromosomes, but this often reduced the diversity too quickly.
* **Roulette wheel selection**: Balanced the exploration and exploitation process, allowing for better long-term results. Roulette selection was adopted as the primary method for this reason.

##### **3.4. Population Size and Generations**

Initial experiments with smaller population sizes (50, 150) showed slower convergence and less diversity. Increasing the population size to **300** provided more solutions per generation and better overall results, particularly when combined with an increase to 50 generations.

#### **4. Results**

After testing various combinations of GA settings, the final configuration (population size = 300, mutation rate = 0.7, uniform crossover, and roulette wheel selection) produced the best results. The **fitness function** calculated for two of the best results is shown below:

1. **Last Test (8)**:
   * Mean waiting time averaged over 10 simulations: **86.94**
   * Max waiting time averaged over 10 simulations: **603.1**
   * **Fitness** = 0.8 × 86.94 + 0.2 × 603.1 = **190.17**
2. **Test 7**:
   * Mean waiting time averaged over 10 simulations: **58.16**
   * Max waiting time averaged over 10 simulations: **736.9**
   * **Fitness** = 0.8 × 58.16 + 0.2 × 736.9 = **193.91**

**Conclusion**: The last result, with a fitness of **190.17**, is the better solution due to the lower maximum waiting time. Although the mean wait time is higher, the overall fitness is improved because the max wait time is significantly lower, which has a positive impact on client satisfaction.

#### **5. Best Employee Assignment**

The best employee assignment found by the GA is printed below:

[1, 1, 3, 1, 2, 1, 2, 2, 1, 1, 3, 1, 1, 2, 1, 1, 2, 1, 2, 3, 2, 1, 2, 1, 2, 2, 1, 1, 1, 1, 1, 2, 1, 2, 2, 1, 1, 2, 3, 1, 1, 2, 1, 1, 1, 4, 1, 2, 3, 2]

This assignment minimises the mean and max waiting times according to the fitness function described above.

#### **6. Conclusion and Reflection**

This assignment demonstrated the effectiveness of genetic algorithms in solving complex optimisation problems. By experimenting with different GA configurations, I was able to achieve a balance between minimising mean and maximum waiting times, through adjusting many different parameters and functions to achieve my final solution. The final solution provides a practical approach to improving client satisfaction in this dynamic service environment.