**Assignment 2**

**Objective:**

The goal of this assignment is to develop and implement a Genetic Algorithm (GA) to optimize the assignment of multiple robots to a set of tasks in a dynamic production environment. Our primary objectives are to minimize the total production time, ensure a balanced workload across robots, and prioritize critical tasks effectively. Additionally, we will create a detailed visualization to illustrate the final task assignments, robot efficiencies, and task priorities.

Here, I have processed mock data for duration and priorities of task, and efficiency of robots. Mock data is artificially generated or simulated data that has the structure and features of genuine data but does not originate from actual observations or events. The fitness function aims to minimize the total production time while ensuring a balanced workload across robots and prioritizing critical tasks.

**Fitness Function:**

There are three functions which are total\_production\_time, workload\_balance, and fitness\_function.

**total\_production\_time:** This function iterates over each robot and calculates the total time required for that robot to complete all tasks assigned to it. It sums up the product of task duration, task priority, and divides by the efficiency of the robot. The maximum total time among all robots is returned.

Tr = ∑ n∈tasks(r)

Ttotal = max (T1, T2, ... , TR)

**workload\_balance:** The workload balance penalizes uneven distribution of work among robots. This function is used to calculate the standard deviation of the total production times. A lower standard deviation indicates a more balanced workload among the robots.

B=σ(T1,T2,...,TR)

**fitness\_function:** The fitness score is calculated by adding the entire production time and the workload balance. The goal of this fitness function appears to be minimizing overall production time while maintaining a balanced workload among robots and prioritizing important activities. The method searches for assigned tasks that result in faster completion of tasks and more evenly distributed workloads among the robots by minimizing the fitness function.

F(I) = Ttotal + B

**Selection, Crossover, and Mutation:**

**Selection:** For selection, I have chosen the tournament selection. The Tournament Selection Process is an effective method in genetic algorithms for selecting individuals from a population to act as parents for crossover. The function takes parents for crossover by arranging competitions in which individuals compete depending on their fitness. The winners of these competitions become parents and produce kids. This guarantees that parents with better fitness are more likely to contribute to the following generation, hence promoting the spread of advantageous genetic features. The function iterates over each individual in the population. For each individual, it randomly selects a subset of individuals from the population. Then selects the individual with the highest fitness value from this subset as the winner and adds it to the list of selected parents.

**Crossover:** Crossover is a genetic operation used to combine the genetic information of two parents to generate new offspring. Single point crossover method is used in implementing this genetic algorithm. Single point crossover is one of the simplest and most commonly used crossover methods because of its simplicity and efficacy in producing different offspring. Single-point crossover involves random genetic exchange between two individuals to produce new offspring that introduce new genetic combinations, promoting diversity, and exploring solution space in genetic algorithms. For each iteration, it randomly selects two parents from the parents’ array. Then, it randomly selects a crossover point. After selecting the crossover point, it creates a new offspring by combining parts of the genomes of the two parents.

Suppose Parent 1: 110110010011

Parent 2: 110011101101

Offspring 1: 1101100 | 01101

Offspring 2: 1100111 | 10011

**Mutation:** Mutation introduces random genetic variations, providing new genetic structures for exploration and helping the algorithm escape local optima. In this assignment, the function iterates over offspring and genes in the population. Then check if a random number is less than the mutation rate for each gene. If so, the gene is mutated by assigning a new random value.

**Genetic Algorithm:**

The genetic algorithm iteratively explores the solution space by combining and mutating candidate solutions to find an optimal or near-optimal solution to the task allocation problem. It is similar to the process of natural selection and evolution, in which the fittest individuals are more likely to survive and reproduce, progressively enhancing the total population's fitness over time.

**Visualization:**

The visualization aims to represent a task allocation scenario involving robots, where each row of the grid corresponds to a robot and each column represents a task. This function generates a visual representation of task assignments, where task durations and priorities are annotated in a grid layout. It provides a clear overview of which tasks are assigned to which robots, their durations, and their priorities. Additionally, it includes a legend for task priorities to enhance the interpretability of the visualization. I also have added a plot which shows the histogram of the total working hours of each robot.