**Report: Exam Schedule Generation by Searching (Local and Evolutionary)**

# **Criteria for Assignment Evaluation:**

Explain the following items clearly:

1. How did you represent a chromosome **in your solution?** (40%)
2. How does crossover happen between chromosomes **in your solution?**(20%)
3. How does mutation change the chromosome **in your solution?** (15%)
4. How do you calculate fitness of any chromosome **in your solution?** (15%)
5. How do you run local search in conjunction with GA algorithm **in your solution**? (10%)

# **Introduction:**

In this programming assignment, genetic algorithm combined with local search was used to solve the scheduling problem.

A **genetic algorithm** is a search heuristic 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.

There were five phases that were used to compute results.

1. Initial population
2. Fitness function
3. Selection
4. Crossover
5. Mutation
6. Local Search

# **Techniques used:**

# **Population:**

In the file general.info there were three days and seven slots per day given to us, so we made 2d array of [3][7] and at every index of it there was an array list representing the courses ids whose exams were to be held. That was our single chromosome which was populated using random numbers.

# **Fitness Function:**

# **Selection:**

Since our population consisted of numerous chromosomes, we choose the best for further evaluation using their fitness values.

# **Crossover:**

Crossover was done using two of the best chromosomes, their courses present in the array lists were swapped with each other.

# **Mutation:**

Using some random probability mutation was done on the fittest chromosome by swapping courses within.

# **Fitness Function:**

While computing evaluation of given chromosome we kept in mind following points and assign fitness value according to its importance

* All exams must be scheduled within the given number of days.
* Total students taking exam in one given slot must be less than the total room capacity.
* Number students having two exams in one given slot must be minimized.
* Number of students having exams on two consecutive slots must be minimized.
* Not even a single student can have more than two exams in one slot.
* Not even a single student can have more than two exams in consecutive lot.
* Not even a single student can have more than three exams in one given day.

## **Approach:**

1. In case number 1 mention in above points, we have set the limit to by making of array of given number of days in GeneralInfo.txt and populate it accordingly and we make sure it that each and every exam must be scheduled in given days.
2. In case number 2, as I discuss above our approach to solve this problem. We basically make List type array which contain students of given courses List at given time slot we find the count of total student and compare the room capacity with it.
3. In case 3 and case 5 we basically use same approach. We use same technique as mention in point 2, get the list of students of each course in given slot and then simply compare it.
4. In case 7, we simply count the student id count in given day, if it exceeds then 3, we simply assume that this particular student have three exams in given day.

## 

All these points which are mentioned above are taken into consideration while computing fitness of chromosome.

# **Experimental Result:**

Here we generate random population of almost 40 and represent it in the given table below with its fitness value.

|  |
| --- |
| **Initial Population:**  Generation 1 Fitness value 1 = 2942  Generation 1 Fitness value 2 = 4513  Generation 1 Fitness value 3 = 4436  Generation 1 Fitness value 4 = 3755  Generation 1 Fitness value 5 = 3866  Generation 1 Fitness value 6 = 3636  Generation 1 Fitness value 7 = 5674  Generation 1 Fitness value 8 = 4276  Generation 1 Fitness value 9 = 4594  Generation 1 Fitness value 10 = 6241  Generation 1 Fitness value 11 = 3847  Generation 1 Fitness value 12 = 3668  Generation 1 Fitness value 13 = 4426  Generation 1 Fitness value 14 = 4023  Generation 1 Fitness value 15 = 2874  Generation 1 Fitness value 16 = 1683  Generation 1 Fitness value 17 = 3581  Generation 1 Fitness value 18 = 4591  Generation 1 Fitness value 19 = 2648  Generation 1 Fitness value 20 = 5297  Generation 1 Fitness value 21 = 3496  Generation 1 Fitness value 22 = 3675  Generation 1 Fitness value 23 = 4996  Generation 1 Fitness value 24 = 3054  Generation 1 Fitness value 25 = 4641  Generation 1 Fitness value 26 = 4104  Generation 1 Fitness value 27 = 3702  Generation 1 Fitness value 28 = 3576  Generation 1 Fitness value 29 = 3635  Generation 1 Fitness value 30 = 3326  Generation 1 Fitness value 31 = 4067  Generation 1 Fitness value 32 = 4146  Generation 1 Fitness value 33 = 3823  Generation 1 Fitness value 34 = 3520  Generation 1 Fitness value 35 = 5671  Generation 1 Fitness value 36 = 3526  Generation 1 Fitness value 37 = 3064  Generation 1 Fitness value 38 = 3750  Generation 1 Fitness value 39 = 1426  Generation 1 Fitness value 40 = 4298 |

We set our criteria of picking of chromosome from given population. If chromosomes from given population didn’t meet the criteria then we generate another set of population by using crossover and mutation. This process repeats until we get our desired chromosome with minimum number of clashes.