

**EXPLORING EVOLUTIONARY ALGORITHMS INCORPORATING
TEACHER - STUDENT PREFERENCES FOR UNIVERSITY COURSE
SCHEDULING**

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
INFORMATION SYSTEM

Submitted by:

Aparna Bhutani
2K17/ISY/17

Under the supervision of

Dr. SEBA SUSAN



INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi- 110042

JUNE, 2019

DECLARATION

I, Aparna Bhutani, student of M.Tech (ISY) hereby declare that the project Dissertation titled “Exploring Evolutionary Algorithms incorporating Teacher-Student preferences for University Course Scheduling” which is submitted by me to Department of Information Technology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: New Delhi

Date:

Aparna Bhutani

(Department of Information Technology)

Delhi Technological University

CERTIFICATE

I hereby certify that the project dissertation titled “Exploring Evolutionary Algorithms incorporating Teacher-Student preferences for University Course Scheduling” which is submitted by Aparna Bhutani, 2K17/ISY/17, Department of Information Technology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree of Diploma to this University or elsewhere.

Place: New Delhi

Date:

Dr. Seba Susan

SUPERVISOR

Associate Professor

(Department of Information Technology)

Delhi Technological University

ACKNOWLEDGEMENT

The realization and absolute conclusion of this project involved a lot of guidance and assistance from numerous people and I am really fortunate to have got this all along the project. Whatever I have prepared is because of this guidance and assistance and it is worth mentioning my thanks to them.

My grateful thanks is extended to Delhi Technological University for providing me the platform to go in for this work. I would also like to acknowledge the Information Technology department for providing me with useful and constructive resources.

I am grateful to and privileged enough to get relentless inspiration as well as valuable suggestions from my parents and seniors who aided me to accomplish this project.

APARNA BHUTANI

(Department of Information Technology)

Delhi Technological University

ABSTRACT

Timetable scheduling problem deals with arranging a set of events (classes, exams, courses) into a defined number of timeslots making sure that the conflicts are avoided. In our work, we have proposed a new course scheduling based on mining for students' preferences for Open Elective courses that makes use of optimization algorithms for automated timetable generation and optimization. The Open Elective courses currently running in an actual university system is used for the experiments. Hard and soft constraints are designed based on the timing and classroom constraints and minimization of clashes between teacher schedules. Two different optimization techniques of Genetic Algorithm (GA) and Simulated Annealing (SA) are utilized for our purpose. The generated timetables are analyzed with respect to the timing efficiency and cost function optimization. The results highlight the efficacy of our approach and the generated course schedules are found at par with the manually compiled timetable running in the university.

We have also proposed a novel set of soft constraints for university course timetabling that in addition to conventional constraints incorporate teacher's preferences and

student time management as well. Here, the hard constraints are those that need to be mandatorily satisfied. The soft constraints are the penalties that are sought to be minimized through every iteration of the optimization algorithm. The evolutionary algorithms- Genetic Algorithm, Particle Swarm Optimization, and the heuristic Simulated Annealing algorithm are used for the optimization task. The timetables generated based on actual university data are found to be more humanely optimized than the previous work of the authors due to the incorporation of human factor consideration both from the perspective of teachers and students.

Finally, we have proposed a new memetic algorithm that hybridizes the global search strategies of Genetic Algorithm (GA) with the local search heuristics of Simulated Annealing (SA) and a greedy randomized local search mutation in GA. The basic framework of our memetic algorithm is that of GA. The population of chromosomes in every generation of GA is first refined by a local neighbourhood search for each chromosome as defined by the SA procedure, prior to the selection, crossover and mutation steps of GA. The mutation step in GA is randomized, with a greedy stochastic local search mutation being randomly selected over normal swap mutation of the fittest chromosome in each iteration of GA. The convergence of the fitness function and the stopping criterion are as determined by SA. As proved from the experimental results, this hybridization presents highly optimal solutions with fast convergence. Comparison to the state-of-the-art on a benchmark dataset for university course scheduling proves the efficacy of our approach.

LIST OF FIGURES

| Figure No. | Description | Page No. |
|------------|--|----------|
| Fig. 1 | The proposed memetic algorithm | 13 |
| Fig. 2 | Timetable obtained using GA | 32 |
| Fig. 3 | Timetable generated by Simulated Annealing | 33 |
| Fig. 4 | No of penalties vs Generations for Genetic Algorithm | 35 |
| Fig. 5 | No of penalties vs Iterations for Simulated Annealing | 36 |
| Fig. 6 | No of penalties vs Iterations for Particle Swarm Optimization | 37 |
| Fig. 7 | The number of penalties minimized over generation/iteration number | 38 |
| Fig. 8 | The variation of execution time (ms) over generation/iteration number for GA | 39 |
| Fig. 9 | Execution Time for each iteration vs Iteration Number for PSO | 40 |
| Fig.10 | Execution Time for each iteration vs Iteration Number for SA | 41 |
| Fig. 11 | Performance Analysis for 3 algorithms | 42 |
| Fig 12 | The system GUI for accepting teacher preference against Teacher ID = 5 | 43 |
| Fig 13 | The system GUI for accepting teacher preference against | 43 |

| | Teacher ID = 3 | |
|---------|---|----|
| Fig. 14 | Two instances of timetables generated by GA for the two sections A and B | 44 |
| Fig. 15 | Two instances of timetables generated by SA for the two sections A and B | 45 |
| Fig. 16 | Two instances of timetables generated by PSO for the two sections A and B | 46 |
| Fig. 17 | Comparisons of all the algorithms for Instance 1 | 48 |
| Fig. 18 | Comparisons of all the algorithms for Instance 2 | 49 |
| Fig. 19 | Comparisons of all the algorithms for Instance 3 | 50 |
| Fig. 20 | Comparisons of all the algorithms for Instance 4 | 51 |
| Fig. 21 | Comparisons of all the algorithms for Instance 5 | 52 |
| Fig. 22 | Comparisons of all the algorithms for Instance 6 | 53 |
| Fig. 23 | Comparisons of all the algorithms for Instance 7 | 54 |
| Fig. 24 | Comparisons of all the algorithms for Instance 8 | 55 |
| Fig. 25 | Comparisons of all the algorithms for Instance 9 | 56 |
| Fig. 26 | Comparisons of all the algorithms for Instance 10 | 57 |
| Fig. 27 | Comparisons of all the algorithms for Instance 11 | 58 |
| Fig. 28 | Comparisons of all the algorithms for Instance 12 | 59 |
| Fig. 29 | Comparisons of all the algorithms for Instance 13 | 60 |
| Fig. 30 | Comparisons of all the algorithms for Instance 14 | 61 |

| | | |
|---------|---|----|
| Fig. 31 | Comparisons of all the algorithms for Instance 15 | 62 |
| Fig. 32 | Comparisons of all the algorithms for Instance 16 | 63 |
| Fig. 33 | Comparisons of all the algorithms for Instance 17 | 64 |
| Fig. 34 | Comparisons of all the algorithms for Instance 18 | 65 |
| Fig. 35 | Comparisons of all the algorithms for Instance 19 | 66 |
| Fig. 36 | Comparisons of all the algorithms for Instance 20 | 67 |
| Fig. 37 | Comparisons of all the algorithms for Instance 21 | 68 |
| Fig. 38 | Electives page | 69 |
| Fig. 39 | Teacher selection page | 70 |
| Fig. 40 | Timetable for TW1TF3 using GA | 71 |
| Fig. 41 | Timetable for TW2GF2 using GA | 72 |
| Fig. 42 | Timetable for TW3TF3 using GA | 72 |
| Fig. 43 | Timetable for TW3TF3 using SA | 73 |
| Fig. 44 | Timetable for TW2GF2 using SA | 73 |
| Fig. 45 | Timetable for TW3TF3 using SA | 74 |

LIST OF TABLES

| Table No. | Description | Page No. |
|-----------|---|----------|
| Table 1 | ITC 2007 Benchmark Dataset | 25 |
| Table 2 | Support Count and Support Rank for elective subjects | 29 |
| Table 3 | Associations for Elective Subjects | 30 |
| Table 4 | Comparison of SA and GA | 31 |
| Table 5 | Comparison of Soft Constraint values for all the algorithms | 49 |

LIST OF ABBREVIATIONS

1. GA: Genetic Algorithm
2. SA: Simulated Annealing
3. PSO: Particle Swarm Optimization
4. MA: Memetic Algorithm
5. PRGA: Parallel Recombinative Genetic Algorithm

CONTENTS

| | |
|---|------------|
| Candidate's Declaration | i |
| Certificate | ii |
| Acknowledgement | iii |
| Abstract | iv |
| List of Figures | vi |
| List of Tables | ix |
| List of abbreviations | x |
| Chapter 1 INTRODUCTION | 01 |
| | |
| Chapter 2 LITERATURE REVIEW | 03 |
| 2.1 Timetable Scheduling | 03 |
| 2.2 Optimization using Genetic Algorithm | 04 |
| 2.3 Optimization using Simulated Annealing | 06 |
| 2.4 Optimization using Particle Swarm Optimization | 07 |
| 2.5 Optimization using Hybrid Algorithms | 08 |
| 2.5.1 GA- SA | 08 |
| 2.5.2 Hybrid Algorithm by Mahfoud | 09 |
| 2.5.3 Local Search Algorithm | 10 |
| 2.5.4 Memetic Algorithm | 11 |
| | |
| Chapter 3 PROPOSED WORK | 15 |
| 3.1 Proposed Timetable Scheduling incorporating student preferences using association rule mining for scheduling elective courses using Genetic Algorithm and Simulated Annealing | 15 |
| 3.1.1 Dataset used with initial parameters and assumptions | 15 |
| 3.1.2 Proposed Set of constraints | 16 |
| 3.1.3 Fitness function | 17 |
| 3.1.4 Proposed algorithm using Association Rule Mining and GA | 17 |
| 3.1.5 Proposed algorithm using Association Rule mining and SA | 18 |
| | |
| 3.2 Proposed Timetable Scheduling incorporating teacher preferences for | 19 |

| | |
|---|----|
| Scheduling elective and core courses using Genetic Algorithm, Simulated Annealing and Particle Swarm Optimization | |
| 3.2.1 Dataset used with initial parameters and assumptions | 19 |
| 3.2.2 Proposed Set of constraints | 20 |
| 3.2.3 Fitness function | 21 |
| 3.2.4 Proposed Algorithm using Simulated Annealing for Complete Timetable Scheduling | 21 |
| 3.2.5 Proposed Algorithm using Genetic Algorithm for Complete Timetable Scheduling | 22 |
| 3.2.6 Proposed Algorithm using Particle Swarm Optimization for Complete Timetable Scheduling | 23 |
| 3.3 Proposed new Memetic Algorithm for Course Scheduling | 23 |
| 3.3.1 Dataset used with initial parameters | 23 |
| 3.3.2 Proposed Set of constraints | 25 |
| 3.3.3 Fitness function | 26 |
| 3.3.4 Proposed Memetic Algorithm | 27 |
| Chapter 4 RESULTS | 29 |
| 4.1 Results for Proposed Timetable Scheduling incorporating student preferences using association rule mining for scheduling elective courses using Genetic Algorithm and Simulated Annealing | 29 |
| 4.1.1 Association Rule Mining | 29 |
| 4.1.2 Comparison of timetables generated by GA and SA | 30 |
| 4.1.3 Timetables generated by GA and SA | 32 |
| 4.2 Results for Proposed Timetable Scheduling incorporating teacher preferences for scheduling elective and core courses using Genetic Algorithm, Simulated Annealing and Particle Swarm Optimization | 34 |
| 4.2.1 Performance Analysis | 34 |
| 4.2.2 No of penalties vs Generations/Iterations | 34 |
| 4.2.3 Time taken for execution of each generation/iteration with respect to generation/ iteration number | 39 |
| 4.2.4 Timetables generated by GA and SA | 42 |
| 4.3 Performance analysis of proposed Memetic Algorithm | 46 |

| | | |
|-----|------------------------------|----|
| 4.4 | GuI for Timetable Scheduling | 69 |
| | CONCLUSION | 75 |
| | REFERENCES | 77 |
| | LIST OF PUBLICATION | 81 |

CHAPTER 1

INTRODUCTION

Timetable Scheduling deals with developing course timetables, assigning courses to teachers in the given timeslots and satisfying a set of constraints. Automation of timetable generation has been of increasing interest due to the increase in the elective or optional courses that universities offer to students. It is challenging to schedule the classes for students in fixed time-slots with the available resources of classrooms and instructors.

Some university scheduling problems requires hard and soft constraints to be satisfied completely whereas some require soft constraints to be satisfied as much as possible which eventually controls the quality of the timetable generated. Different fields deal with this timetabling problem like operation research. It includes: Graph Coloring [1] where the vertices represent courses and the edges represent students, Integer Programming [2] and Constraint Satisfaction Programming [3]. The course scheduling problem is a NP Hard problem and requires high amount of computation complexity. This led to the emergence of algorithms like Simulated Annealing and other local search techniques and also metaheuristics like evolutionary and genetic algorithms. The range of optimization algorithms vary from mathematical to evolutionary to heuristic. Genetic Algorithm (GA) is a highly efficient evolutionary algorithm that follows the principle of the evolution of the fittest gene [4]. Its application to the timetable scheduling problem is investigated in [5,6]. Highly optimized schedules are generated by the heuristic Simulated Annealing (SA) algorithm reportedly in minimal time as per research in [7-10]. The different types of schedules generated by SA in [7-10] range from university course timetables, exam timetables, staff job schedules etc. SA and GA are the most investigated optimization algorithms for scheduling problems. Most of the algorithms differ in the initialization

routines or variant of SA and GA used. Several variants of GA are found in literature [11]. Particle Swarm Optimization which is another evolutionary algorithm has been explored in [12, 13] for timetable scheduling. Simulated annealing is used for optimizing in scheduling problems and is very popular because of its less execution and how it generates an optimized timetable. A new work used for elective course scheduling has been seen in [14]. It requires initially the timetable template which makes sure hard and soft constraints are satisfied and then it uses Genetic Algorithm for timetable generation and optimization. Another recent work is [15] which minimizes student conflicts for different courses that uses graphical models and distance-based local search methods.

In first part of our work, we have proposed fully automated course scheduling technique. Our approach uses algorithms GA and SA for generating timetables. We have used association rule mining that mines patterns for the preference of students for Elective courses they wish to attend. We have used our university data for the electives that the students are currently studying. We compare the timetables generated using optimization algorithms like Genetic Algorithm and Simulated Annealing are compared with the manually compiled timetable which is being used by the university.

In second part of our work, we formulate new soft constraint functions in order to generate more human-friendly optimized schedules. GA, SA and PSO optimization algorithms are investigated for the task.

Hybrid approaches which uses the concept of global and local search strategies have emerged that seek to optimize the timetabling problem. These approaches much more fast and optimal solutions. In our final work, we have proposed a memetic algorithm which uses the concept of greedy stochastic local search mutation in order to generate more optimal timetable schedules which offer faster convergence.

CHAPTER 2

LITERATURE REVIEW

2.1 TIMETABLE SCHEDULING

Timetable Scheduling is an optimizing problem that deals with assigning students – courses, teacher - courses, teacher - students in set of given timeslots. Timetable scheduling problem contains a set of hard constraint and soft constraints. The hard constraints have to be followed by all valid solutions and the soft constraints define the quality of valid solutions based on the number of soft constraint satisfied.

Course scheduling is a challenging task these days in schools and university because of the huge amount of elective courses that are offered by the universities, This also causes course scheduling to be a very tedious and time consuming job .

Evolutionary algorithm are population based metaheuristic optimization algorithms. They are inspired and use mechanisms of biological computation. A number of evolutionary algorithms have been used for solving the course scheduling problem to find optimal solution. Genetic Algorithm (GA) is an evolutionary algorithms that is based on Darwin’s theory of “survival of the fittest”. GA and its variants such as NSGA, CSGA etc have been used for automatic timetable generation with different combinations of crossover and mutation steps. These help in generation more efficient solutions. Another meta heuristic approach is a single solution approach such as Simulated Annealing. SA uses a local search process in the neighborhood for continuously refining one single solution. Hybrid approaches that combine evolutionary mechanisms with local search have also emerged as highly efficient alternatives. Some of the hybrid approaches include the multi-population hybrid GA in [16], the hybrid fuzzy evolutionary approach in [17], the fuzzy genetic algorithm with local search in [18].

Memetic algorithms (MA) were introduced by Moscato in 1989. It combines the best features of population based evolutionary algorithms with local search techniques [19]. As per Moscato and Cotta [20], the gene evolution is improved by using various local search techniques. MA is considered to be an amalgamation of GA and SA. Most of these algorithms use the basic framework of Genetic Algorithms and are called as hybridized evolutionary algorithms in [21-23]. Recent work in Memetic Algorithm ensures combination of population based evolutionary approaches with local neighborhood search. Examples of such hybridized approaches are: Whale Optimization with Simulated Annealing [24], Harmony search with the hill climbing algorithm [25], Water wave optimization hybridized with sequential quadratic programming [26].

In our work we have used optimization tools such as genetic algorithm [27,28] , simulated annealing [29,30,31] and particle swarm optimization [12]. We have also proposed a new memetic algorithm that incorporates the goodness of the global search techniques of GA as well as the local search strategies of SA. This memetic algorithm also consists of local search mutation instead of the normal mutation which helps in faster convergence and generating an optimal solution.

2.2 OPTIMIZATION USING GENETIC ALGORITHM

Genetic Algorithm [27,28] is based on the concept of biological evolution that is based on Darwin's theory of "Survival of the Fittest". This states that fittest organism survives and in our case this means that the fittest or the most optimal solution survives in the end . GA is used in optimization problems and produces the most optimal solution.

Operations of Genetic Algorithm are as follows:

2.2.1 Initialize:

This step deals with initializing and defining the parameters that are used in the further GA process. It consists of defining size of chromosome, gene, population, rate of crossover, rate of mutation.

A Chromosome represents one single individual solution. In course scheduling it can represent one timetable instance and so on. Fitness value of chromosomes is calculated at each iteration/ generation of the algorithm.

Population is the entire collection of all the chromosomes.

We define all these in the initialize step of the GA process for the initial random set of chromosomes.

2.2.2 Defining the fitness function:

Fitness function defines what we want to optimize i.e. the function which determines the quality of our solution. It tells the goodness of a chromosome. For example, for a minimizing problem, the lower the fitness value, the more optimal is the chromosome. It helps us in analyzing a single solution and gives a measure to how far it is from its best value.

2.2.3 Selection:

The selection operator helps us in selecting the fittest individuals so as to eventually reach our optimal solution. After calculating the fitness value of each chromosome we select the most fit individuals using various selection techniques such as rank selection, roulette wheel selection, stochastic universal sampling, tournament selection, random selection and so on. It helps us select the parent chromosomes that are further used for crossover. It helps direct the solution in the right direction.

2.2.4 Crossover:

Crossover is the next genetic operator which involves combining genes of the parents to generate a new offspring. The new offspring is retained if it has a higher fitness value than the fitness value of the individual parents else it is discarded. The higher fitness value helps us reach towards the optimal solution. Crossover combines genes and features of the parents to generate a new offspring. It is similar to the concept of biological crossover and reproduction. Crossover can be different types such as: One point crossover, multipoint crossover, uniform crossover, davis' order crossover and so on. A crossover point or points is chosen and crossover is performed at it.

2.2.5 Mutation:

Mutation is a genetic operator that is used to introduce heterogeneousness in the chromosomes. It is similar to the concept of biological mutation where some gene value of the child are drastically different from the parents or family. It is used to introduce genetic diversity or variety in the solution. Mutation can be of various types such as: swap mutation, bit flip mutation, random resetting, scramble mutation, inversion mutation and so on. In mutation, the new chromosome generated is drastically different from the parent chromosomes.

2.3 OPTIMIZATION USING SIMULATED ANNEALING

Simulated Annealing [29,30,31] is a metaheuristic technique which uses the concept of annealing used for metals. In annealing, the metal is heated to very high temperatures. It is then cooled slowly with given conditions and gives a metal which is much more stronger than the original metal. The new metal obtained is more fit than the parent and has less chances of breakage and is hence stronger.

Simulated Annealing is prominently used as an optimization technique. It is used often when the search space is said to be discrete. Temperature(T) gives us the randomness of the solution. The higher the temperature the more is the degree of randomness. At very high temperatures the SA algorithm behaves like random walk. As the temperature decreases the algorithm starts to behave like hill climbing. Higher T, the more exploration is done in the search space for an optimal solution i.e. more random solutions are obtained. As the temperature decreases, the randomness decreases and the solutions start moving towards a particular place in the search space which is eventually the optima.

The algorithm starts with T= very high. For every step, a random neighboring solution is selected. We calculate the gap of energy level between the new and current solution.

$$\Delta E = E(new) - E(current) \quad (2.1)$$

In Equation (2.1),

$E(new)$ =energy value for new solution

$E(current)$ =energy value for current solution

if $\Delta E > 0$: then $E(new)$ is chosen with a probability $P=1$

if $\Delta E < 0$: $E(new)$ is selected with $0 < P < 1$. Here the probability is given as:

$$P = 1 / (1 + e^{-\Delta E/T}) \quad (2.2)$$

In Equation (2.2),

P represents the probability with which the new node is selected.

T represents the temperature

The temperature is decreased for every iteration. Hence, at the minimal temperatures the optimal solution is obtained.

2.4 OPTIMIZATION USING PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization [12] is a heuristic method that uses the concept of swarm and birds flocking. It is used in course scheduling problem and gives us an optimal solution. Particle Swarm Optimization initially starts with randomly generated solutions and tries to eventually converge towards the optimal solution.

The steps of PSO are given as follows:

2.4.1 Initialization:

Each solution in PSO is known as a particle. Every particle has a position and velocity vector. The PSO is initialized with random particles i.e. random solutions.

2.4.2 Finding pBest and gBest:

At every generation fitness of the solution is found as per the fitness function. Each particle is updated using the pBest and gBest values.

lBest denotes the best value of fitness achieved till now. It is also known as the best local fitness value.

tBest denotes the overall best value of fitness i.e. the best global fitness value. It denotes the overall best fitness value obtained taking into account all the particles

2.4.3 Updating velocity:

Using these values, the velocity and position of particles is updated in each generation using following equations:

$$v[i] = v[i - 1] + c1 + \text{random}() * (pBest - [present[i - 1]]) + c2 * \text{random}() * (gBest - present[i - 1]) \quad (2.3)$$

$$present[i] = present[i - 1] + v[i] \quad (2.4)$$

In Equation (2.3) and (2.4),

$v[i-1]$ = The velocity for i-1th particle

$present[i]$ = It denotes the current solution

$\text{random}()$ = It is a random number in (0,1)

$c1, c2$ = These are known as learning factors. Value of $c1, c2$ taken for our work = 2

2.5 OPTIMIZATION USING HYBRID ALGORITHMS

Hybrid optimization techniques involving combination of global search techniques with local search techniques and they give us more optimal solutions with faster convergence. In our work, we have worked with different hybrid algorithms involving GA and SA. These algorithms are as below:

2.5.1 GA-SA[32]

Genetic Algorithm and Simulated Annealing are two popular algorithms. In GA- SA we combine features of GA and SA to generate solutions which are more optimal than our solutions generated with GA and SA individually.

We have used the GA- SA algorithm described in [32] and used it on ITC 2007 dataset to observe the solutions generated and compare it with the other hybrid algorithms.

GA-SA combines the stopping criteria of SA along with the process of GA. The temperature in GA-SA is initially set to be very high and then slowly cooling is done. Initially for high temperatures,

We use a decision function to see if it is good or bad mutation. It is defined as below:

$$D1 = \text{accept} \left\{ \text{if } r_n \leq e^{-\frac{(f_i - f'_i)x^t}{f_i}} \right\} \quad (2.5)$$

$$D1 = \text{reject} \left\{ \text{if } r_n > e^{-\frac{(f_i - f'_i)x^t}{f_i}} \right\} \quad (2.6)$$

In Equation (2.5) and (2.6),

r_n = Real number generated randomly between 0 and 1

f_i = Makespan time before mutation

f'_i = Makespan time after mutation

b = Boltzmann constant

t = Temperature

After mutation step is executed, GA-SA helps to obtain a more fit chromosome. From D1, we can observe that GA-SA ensures there is no chromosome which has a worse fitness value than the current chromosome.

We then select chromosomes and perform crossover similar to GA. After crossover, adaptative mutation is performed as described above. In this, we check if the mutation is a bad mutation or not and new population is selected based on the fittest chromosomes. The final step is selection of a new set of individuals based on elitist roulette wheel selection. After the completion of a cycle, temperature is decreased and crossover, mutation, selection are performed for the next cycle.

2.5.2 Hybrid Algorithm by Mahfoud [33]

Mahfoud introduces the method of parallel recombinative simulated annealing. This algorithm uses convergence properties of simulated annealing and uses GA for the recombinative approach.

We have used the algorithm described in [33] and used it on ITC 2007 dataset to observe the solutions generated and compare it with the other hybrid algorithms.

In Parallel recombinative simulated annealing (PRSA), the convergence of the algorithm is controlled by simulated annealing, that is, the cooling procedure.

In PRSA, the initial temperature = very high. The population is initialized randomly. The stopping criteria is chosen to be $t/2$. Here t gives us size of the population.

We use GA by initially selecting 2 random individuals from the population and assigning them to be parents. 2 children are generated by using single point crossover followed by mutation. Boltzmann trial is held between the children and parents. Finally, the parents are overwritten by the winner of the Boltzmann trials and the process is carried out again.

The Boltzmann trials can be defined as the competition between x and y where E_x represents the combined energy of the parents and E_y represents the combined energy of the children

ΔE is defined in Equation (2.7) as,

$$\Delta E = E_x - E_y \quad (2.7)$$

If $\Delta E > 0$ then x wins with a probability of 1

if $\Delta E < 0$ then x still wins but this time with a probability of P . P can be defined as per Equation (2.8):

$$P = 1/(1 + e^{(E_x - E_y)/T}) \quad (2.8)$$

2.5.3 Local Search Algorithm [34]

The Local Search algorithm combines the properties of GA along with sequential local search. We use this algorithm on the ITC 2007 dataset and compared it with other hybrid algorithms and our proposed memetic algorithm.

The steps of the Local Search Algorithm are described as follows:

- Generation of Initial population: Initial population is generated consisting of N chromosomes. Each chromosome represents a timetable.
- Genetic Operation (Crossover): After generation of the initial population, single point crossover is performed. Crossover point is chosen randomly from $(1, 2, \dots, M)$. Here, M is the length of chromosome
- Genetic Operation (Mutation): After crossover, mutation is performed. Mutation points are also chosen randomly.
- Fitness Measure: In this step, we calculate fitness value of . The fitness function is given as:

$$f = x \quad (2.9)$$

Here in Equation (2.9),

f = fitness function

x = number of penalties

- Genetic Operation (Selection): After fitness calculation, we select chromosomes with highest fitness value using tournament selection method.
- Local Search: After completion of the above steps we use local search to improve the timetable and make its convergence faster.

2.4.4 Memetic Algorithm

Memetic algorithms are known as an extension of Genetic Algorithm. They combine the concept of GA and SA. They are also known as hybrid evolutionary algorithms.

In our work, we have proposed a new memetic algorithm which combines the best features of GA and SA and apply this to university course scheduling problem. We intend to bring together the goodness global solutions of GA and the local neighborhood search of SA. This memetic algorithm also incorporates the concept of greedy stochastic local search mutation instead of the simple mutation step in GA which leads to a more optimal solution with faster convergence.

In our SA implementation for MA, we initially begin with very high temperatures and decrease it for every iteration using the formula.

$$T_{i+1} = T_i * \beta \quad (2.10)$$

In Equation (2.10),

β = Cooling Rate

T_i = Initial Temperature

T_i i.e. the initial temperature and the rate of cooling β is determined by the user depending on the situation. At high temperatures, there is more randomness and the algorithm tries to search the search space more for optimal solution. As the temperature decreases, the randomness decreases and the solution converges towards local minima.

Incorporating SA procedure into GA as a refinement procedure prior to the selection mechanism of the fittest chromosomes in GA in each generation, improves the efficiency and reduces the soft constraint violations, hence improving the quality of the timetables generated. This forms the basis of our method that is described below in more detail in Fig. 1.

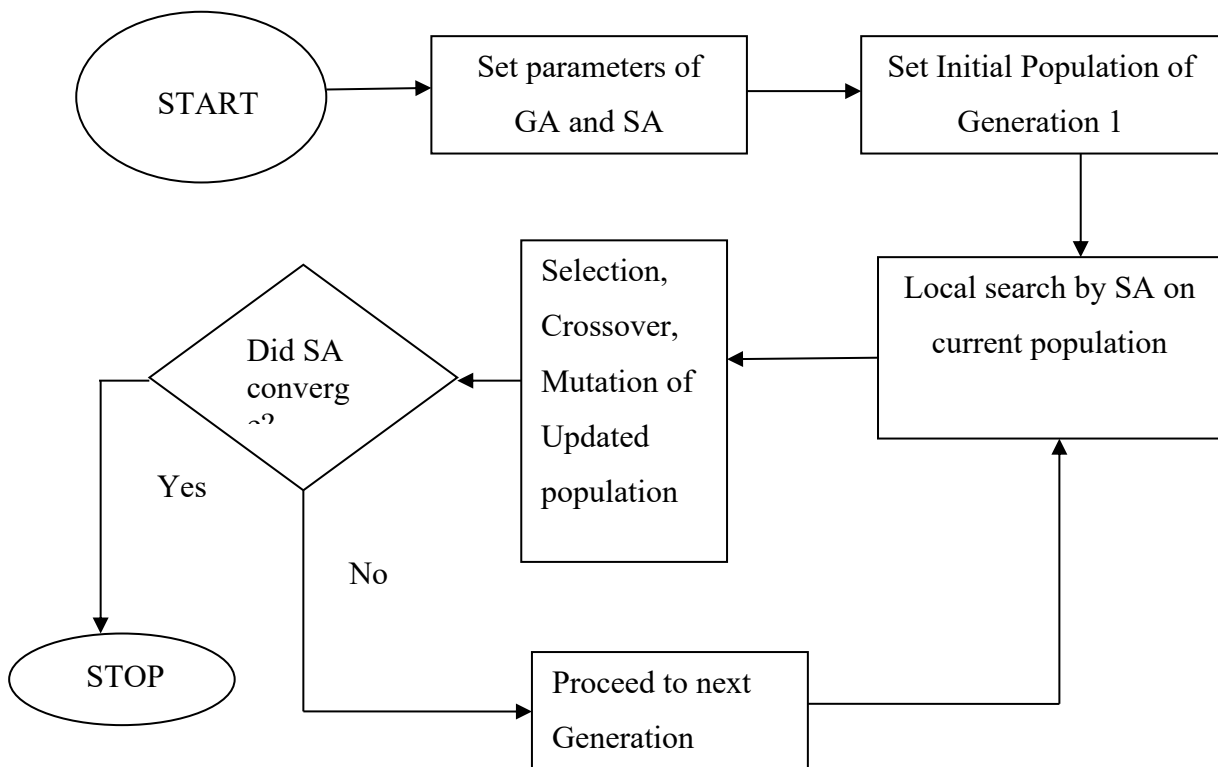


Fig. 1: The proposed memetic algorithm

The proposed memetic algorithm introduced here consists of firstly defining an initial population for generation 1. This is followed by application of local search on the current population. GA operations like Selection, Crossover and Mutation are applied on the population. If algorithm converges we stop, else we proceed to the next generation.

This algorithm further incorporates the concept of greedy stochastic local search mutation. After the selection and crossover of the updated population, the local swap mutation is done. In mutation we introduce a random number “r” which lies between (0,1) . This random number determines whether to perform a simple swap mutation or a local swap mutation. In a simple swap mutation 2 random genes of the chromosome are exchanged. In case of local swap mutation, a random gene is taken and it is swapped with every gene of the chromosome. We determine how fit the chromosome is and compare it with current optimal fitness value and if the new chromosome is

more optimal, it is selected. This algorithm was implemented for the job shop scheduling problem in [35].

The stopping criterion for the memetic algorithm is same as that of SA i.e. temperature value. Hence it brings together the best features of GA and SA and provides a much more optimal solution with faster convergence.

CHAPTER 3

PROPOSED WORK

3.1 PROPOSED TIMETABLE SCHEDULING INCORPORATING STUDENT PREFERENCES USING ASSOCIATION RULE MINING FOR SCHEDULING ELECTIVE COURSES USING GENETIC ALGORITHM AND SIMULATED ANNEALING

In this work, we apply Association Rule Mining to GA and SA, We compare GA and SA based on generation/iteration number and makespan time.

3.1.1 Dataset used with initial parameters and assumptions

We have done Timetable scheduling for 6th semester IT students. Students are divided into 2 sections 6-A and 6-B.

There are 6 elective courses:

- 1 Cyber Forensics and Cyber Crime (CFCC)
- 2 Real Time Systems (RTS)
- 3 Machine Learning (ML)
- 4 Optical networks (ON)
- 5 High speed Networks (HSN)
- 6 Multimedia System Design (MSD)

Two most commonly selected electives are chosen after association rule mining process. These are:

- Cyber forensics and Cyber Crime
- Machine Learning

Assumptions:

There are two electives which are taken by the students of a department for a particular semester. Three teachers are assigned to teach these electives. Elective 1 is taught by one professor and the other 2 teach Elective 2.

There is a theory class of 3 hours every week for both the electives and a tutorial for 1 hour for every group.

Initial Parameters:

- Starting Temperature (for SA) = 1000
- Rate of Cooling (for SA) = 0.05
- Mutation Rate (for Genetic Algorithm) = 0.01
- Population size (for Genetic Algorithm) = 100
- Crossover Rate(for Genetic Algorithm) = 0.9

3.1.2 Proposed Set of constraints

In our work, we have defined a set of hard constraints and soft constraints. Hard constraints cannot be violated and are necessary for timetable generation. Soft constraints can be violated and represent the overall quality of solution. A penalty cost of 1 is assigned for every soft constraint violation.

Hard Constraints:

1. Elective courses can only be taught during 4 hours in the morning from 08:00am to 12:00am.
2. There are 2 electives for a semester.
3. All students of that particular semester are supposed to enroll in these electives compulsorily.

Soft Constraints:

1. Classroom capacity should be greater than the size of the class.
2. No room should be assigned to a course or teacher if its already assigned to another course or teacher for that particular time period.
3. Every course should have a professor to teach it.
4. Each student is assigned to only one classroom at a certain time period.

3.1.3 Fitness function:

The fitness function we are trying to maximize here is:

$$f = 1 / 1 + x \quad (3.1)$$

In Equation (3.1), x denotes the number of penalties.

3.1.4 Proposed algorithm using Association Rule Mining and GA

Stage 1: Use Data Mining to find electives:

1. Student's 2 elective options are taken from them.
2. SUPCOUNT(Support count) is calculated for every elective.
3. A threshold value (T) is chosen.
4. If $SUPCOUNT(elective) < T$, it is eliminated. If $SUPCOUNT(T) = T$ it is said to be strongly associated.
5. FP Growth algorithm is used to find associations between the electives.
6. Table is created for associated electives.
- 4 A count the electives' associations is done and they are arranged in a descending order.
- 5 Associations with maximum count are selected as the two electives.
- 6 Selected electives are used for timetable generation using GA in stage 2.

Stage 2: Applying GA on selected electives to create optimized timetable

1. Initialize a timetable with given number of professors, time slots, classes and electives.
2. Define population size, mutation rate, crossover rate.
3. Form one chromosome, clash_val is calculated. Here, clash_val denotes the penalty value for current chromosome.
4. Calculate fitness of a chromosome (f) using fitness function given in Equation (3.1)

5. Repeat for all generations:
 1. Crossover()
 2. Mutation()
 3. Selection()
 4. Compute new fitness f of a chromosome
6. If (clash_val == 0) or (f==1) STOP
7. Compare factors- generation number, makespan time with SA.

3.1.5 Proposed algorithm using Association Rule mining and SA

Stage 1: Use Data Mining to find electives

1. Student's 2 elective options are taken from them.
2. SUPCOUNT(Support count) is calculated for every elective.
3. A threshold value (T) is chosen.
4. If $SUPCOUNT(elective) < T$, it is eliminated. If $SUPCOUNT(T) = T$ it is said to be strongly associated.
5. FP Growth algorithm is used to find associations between the electives.
6. Table is created for associated electives.
7. A count the electives' associations is done and they are arranged in a descending order.
8. Associations with maximum count are selected as the two electives.
9. Selected electives are used for timetable generation using GA in stage 2.

Stage 2 : Applying SA on selected electives to create optimized timetable

1. Initialize a timetable (number of professors, time slots, classes and subjects).
2. Define temperature, Rate of cooling.
3. While(T is greater than 1)
 - a. Define current node X, Initialize random neighbor P.
 - b. Evaluate $\Delta E = z(P) - z(X)$ (3.2)

$z(P)$ = Total number of penalties for neighboring node for violating soft constraints

$z(X)$ = Total number of penalties for neighboring node for violating soft constraints

- c. If $z(P) < z(X)$ i.e. $\Delta E < 0$ choose N, probability = 1.
- d. If $z(P) > z(X)$ i.e. $\Delta E > 0$ choose N, probability $P = e^{-\Delta E/T}$ (choose N if $P > 0.77$ only)
- e. else Retain C
- f. Temp = temperature * RateOfCooling.
- g. When Temperature < 1 STOP
- h. Compare parameters: Iteration at which solution obtained, makespan time with Genetic Algorithm.

3.2 PROPOSED TIMETABLE SCHEDULING INCORPORATING TEACHER PREFERENCES FOR SCHEDULING ELECTIVE AND CORE COURSES USING GENETIC ALGORITHM, SIMULATED ANNEALING AND PARTICLE SWARM OPTIMIZATION

In the second part of our work, we try and make the timetable more human centric by taking teacher's preferences. We then compare GA, SA and PSO based on Number of penalties vs generations, Time taken for execution of each generation with respect to generation number.

3.2.1 Dataset used with initial parameters and assumptions

Timetable scheduling is done for 6th semester undergraduate B.Tech (Information Technology) students of Delhi Technological University. Students are divided into 2 sections 6-A and 6-B. The number of classrooms are only two- TW2GF2 and TW1TF3 as opposed to three in our previous work in 5.1. The professors are the current staff of Department of Information Technology, Delhi Technological University.

There are 5 theory courses:

Electives:

- Cyber Forensics (CFCC)
- Machine Learning (ML)

Core Courses:

- Compiler Design (CD)
- Software engineering (SE)
- Artificial Intelligence (AI)

There are 2 labs:

- Compiler Design (CD) Lab
- Artificial Intelligence (AI) Lab

3.2.2 Proposed Set of constraints

A collection of constraints is taken from different papers and additional constraints are added to make the timetable more human centric.

- **Hard Constraints (proposed):**

Hard constraints cannot be violated (as per stringent rules of the Delhi Technological University). Hard constraints are as follows:

- Electives are scheduled in the time periods: 08:00am - 10:00am
- Core Courses are taught between time periods: 10:00am - 4:00pm
- All students of the 6th semester from IT department enroll for the theory and practical courses

- **Soft Constraints (proposed):**

A 10 point penalty is assigned if soft constraints are violated. Soft constraints are as follows:

- Classroom capacity should be greater than the size of the class. [2]
- No room should be assigned to a course or teacher if its already assigned to another course or teacher for that particular time period.[2]
- Every teacher can teach only one class at a particular time period. [1] [4]
- Every course should have a professor to teach it.
- Classrooms and lab for separate courses should not overlap one another. [1] [3]
- No student is assigned more than one class at the same time. [2]

- **Additional Soft Constraints (Proposed):**

Set of additional constraints are added in this paper. These are a part of soft constraints and a 10 point penalty is assigned if they are violated. These additional constraints are:

- i. The students of a class are allotted one room in which they will have all classes. This is done to avoid the time consumed for students while traveling between classes.
- ii. There are not a lot of free period gaps between classes in a day for the students.
- iii. The teacher's preference to teach in a particular time slot is considered.

3.2.3 Fitness function

The fitness function we are trying to minimize here is:

$$f = x \tag{3.3}$$

In Equation (3.3), x denotes the number of penalties. We try and minimize the fitness function.

3.2.4 Proposed Algorithm using Simulated Annealing for Complete Timetable Scheduling

- 1 Timetable is initialized with given set of professors, electives, timeslots.
- 2 Take initial temperature to be very high (T=1000).
- 3 Repeat till (Temperature > 1)
 - a. Define current node C, Initialize random neighbor N.
 - b. Evaluate $\Delta E = eval(N) - eval(C)$.

$eval(N)$ = number of clashes (calculated for soft constraint violation) in new node

$eval(C)$ = number of clashes (calculated for soft constraint violation) in current node

- c. If $eval(N) < eval(C)$ i.e. $\Delta E < 0$ choose N, probability = 1.

- d. If $eval(N) > eval(C)$ i.e. $\Delta E > 0$ choose N, probability $P = -\Delta E/T$
- e. Fitness Function (cost function) = x; where x is the number of clashes.
- 4 Temp = temperature * coolingRate.
- 5 Stop when Temperature < 1 or max number of iterations are over.
- 6 Analyse the performance with respect to following factors:
 - f. No of penalties vs iterations.
 - g. Time taken for execution of each iteration with respect to generation number
- 7 Compare these factors with Genetic Algorithm and PSO.

3.2.5 Proposed Algorithm using Genetic Algorithm for Complete Timetable Scheduling

1. Initialize a timetable with given number of professors, time slots, classes and electives.
 2. Define population size, mutation rate, crossover rate.
 3. Form one chromosome, calculate number of clashes as per constraints not satisfied.
 4. Calculate fitness of a chromosome using fitness function=x; where x denotes no of penalties.
 5. Repeat for all generations
 - a. Crossover
 - b. Mutation
 - c. Selection
 - d. Compute new fitness value for new chromosome obtained
 6. Stop when max number of generations are over
 7. Analyze the performance with respect to following factors:
 - a. No of penalties vs generations.
 - b. Time taken for execution of each generation with respect to generation number
- d) Compare these factors with Simulated Annealing and PSO

3.2.6 Proposed Algorithm using Particle Swarm Optimization for Complete Timetable Scheduling

- 1 Initialize random set of particle (solutions)
- 2 Repeat till maximum iterations reached.
 1. Calculate fitness value of each particle. Fitness value= number of penalties
 2. If Current fitness value $>$ pBest , do pBest = current fitness value else keep previous pBest
 3. Assign best particle's pBest to gBest i.e. gBest= best pBest
 4. Calculate velocity of each particle using (1)
 5. Update data values in equations (2) using values from equation (1)
- 3 Analyze the performance with respect to following factors:
 1. No of penalties vs iterations
 2. Time taken for execution of each iteration with respect to generation number
4. Compare these factors with Genetic Algorithm and Simulated Annealing.

3.3 PROPOSED NEW MEMETIC ALGORITHM FOR COURSE SCHEDULING

In the third part of our work, we propose a novel memetic algorithm incorporating the concept of local search mutation. We then compare GA, SA, GA-SA, Mahfoud's Algorithm, Local search Mutation and our Memetic Algorithm.

3.3.1 Dataset used with initial parameters

In this work, we have used the dataset from Track 3 of International Timetabling Competition- 2007: Curriculum-based Course Timetabling. It consists of the following entities.

Days: This gives us the number of days in a week for which the timetable is constructed.

Timeslots: Timeslots are the fixed number of slots in a day and is same for every day of the week.

Periods: A period is composed of a day and a timeslot. The total number of scheduling periods is the product of the days multiplied by the timeslots in a day.

Courses: These specify the total number of courses in a particular timetable instance. Each course contains certain number of lectures. There are certain number of slots in which particular courses cannot be assigned as defined in the dataset.

Teachers: Teachers teach the courses they are assigned to.

Rooms. Each room has a certain capacity as mentioned in the data set. The courses can be assigned to any room provided the number of students are less than the capacity of room.

Curriculum: A curriculum is a set of courses and these courses have students in common. Based on this, we have the conflicts between courses and other soft constraints.

Twenty-one instances were released for this track, seven for each set (early, late, and hidden). All instances are real data and come from the University of Udine.

The number of courses in these instances ranges between 30 and 131, the total number of lectures from 138 to 434, the number of rooms between 5 and 20, and the number of curricula between 13 and 150.

Initial Parameters:

- Initial Temperature = 1000
- Cooling Rate = 0.05

Table 1 below, gives us the details regarding the 21 instances for the ITC 2007 data set.

| Instance Number | Courses | Rooms | Days | Teachers |
|-----------------|---------|-------|------|----------|
| 1 | 30 | 6 | 5 | 24 |
| 2 | 82 | 16 | 5 | 71 |
| 3 | 72 | 16 | 5 | 61 |
| 4 | 79 | 18 | 5 | 70 |
| 5 | 54 | 9 | 6 | 47 |
| 6 | 108 | 18 | 5 | 87 |
| 7 | 131 | 20 | 5 | 67 |
| 8 | 86 | 18 | 5 | 76 |
| 9 | 76 | 18 | 5 | 68 |
| 10 | 115 | 18 | 5 | 88 |
| 11 | 30 | 5 | 5 | 24 |
| 12 | 88 | 11 | 6 | 74 |
| 13 | 82 | 19 | 5 | 77 |
| 14 | 85 | 17 | 5 | 68 |
| 15 | 72 | 16 | 5 | 61 |
| 16 | 108 | 20 | 5 | 87 |
| 17 | 99 | 17 | 5 | 80 |
| 18 | 47 | 9 | 6 | 47 |
| 19 | 74 | 16 | 5 | 66 |
| 20 | 121 | 19 | 5 | 95 |
| 21 | 94 | 18 | 5 | 76 |

Table 1: ITC 2007 Benchmark Dataset

3.3.2 Proposed Set of constraints

A collection of constraints is taken from ITC 2007 curriculum based timetabling problem.

Hard Constraints:

1. **Lectures:** All lectures of a course must be scheduled, and they must be assigned to distinct periods. A violation occurs if a lecture is not scheduled.
2. **Room Occupancy:** Two lectures cannot take place in the same room in the same period. Two lectures in the same room at the same period represent one violation . Any extra lecture in the same period and room counts as one more violation.

3. **Conflicts:** Lectures of courses in the same curriculum or taught by the same teacher must be all scheduled in different periods. Two conflicting lectures in the same period represent one violation. Three conflicting lectures count as 3 violations: one for each pair.
4. **Availabilities:** If the teacher of the course is not available to teach that course at a given period, then no lectures of the course can be scheduled at that period. Each lecture in a period unavailable for that course is one violation.

Soft Constraints:

The Soft constraints are as follows:

1. **Room Capacity:** For each lecture, the number of students that attend the course must be less or equal than the number of seats of all the rooms that host its lectures. Each student above the capacity counts as 1 point of penalty.
2. **Minimum Working Days:** The lectures of each course must be spread into the given minimum number of days. Each day below the minimum counts as 5 points of penalty.
3. **Curriculum Compactness:** Lectures belonging to a curriculum should be adjacent to each other (i.e., in consecutive periods). For a given curriculum we account for a violation every time there is one lecture not adjacent to any other lecture within the same day. Each isolated lecture in a curriculum counts as 2 points of penalty.
4. **Room Stability:** All lectures of a course should be given in the same room. Each distinct room used for the lectures of a course, but the first, counts as 1 point of penalty.

3.3.3 Fitness function:

The fitness function as per ITC 2007 is given by:

$$f = x \tag{3.4}$$

In Equation (3.4), x denotes the number of penalties. We try and minimize the fitness function.

3.3.4 Proposed Memetic Algorithm with Local Search Mutation:

Algorithm 1: The Genetic Algorithm incorporating local search by SA

1. Initialization the: Initial population of chromosomes, Temperature T , Cooling Rate
2. for each generation do
 1. for each chromosome C in the current population do
 - Procedure(SA): Steps 4(b,c,d,e,f)
 2. end for
 3. for the updated population do
 - Selection
 - Crossover
 - Procedure (Local Search Mutation)
 4. end for
 5. current population=final population;
 6. Update Temperature T as per Eq (2.10)
 7. if $T < 1$ STOP
 8. end for
 9. $S = \text{BEST}(\text{final population})$
 10. return S

Procedure (SA): Applying SA for optimized timetable generation:

1. Initialize a timetable (number of professors, time slots, classes and subjects).
2. Define initial parameters such as Rate of cooling and Temperature.
3. Temperature T is initialized to 1000
4. While(T is greater than 1)
 - i. Define current node X , Initialize random neighbor P .
 - j. Evaluate $\Delta E = z(P) - z(X)$ (3.5)

$z(P)$ = Total number of penalties for neighboring node for violating soft constraints

$z(X)$ = Total number of penalties for neighboring node for violating soft constraints

- k. If $z(P) < z(X)$ i.e. $\Delta E < 0$ choose N, probability = 1.
- l. If $z(P) > z(X)$ i.e. $\Delta E > 0$ choose N, probability $P = e^{-\Delta E/T}$ (choose N if $P > 0.77$ only)
- m. else Retain C
- n. Temp = temperature * RateOfCooling.
- o. When Temperature < 1 STOP

Procedure (LSM): The greedy stochastic local search for mutation

- for entire updated population do
 - select fittest individual f_{min}
- end for
- Generate random number $r \in [0,1]$
- if $r \leq 0.5$ do simple swap mutation by exchanging two random genes of the fittest individual f_{min}
- else
 - select random gene g_i
 - for ($k=0$ to chromosomeSize - 1) do
 - swap g_i and g_k
 - Calculate fitness value of new chromosome f_{new}
 - if $\text{eval}(f_{new}) < \text{eval}(f_{min})$
 - Update $f_{min} = f_{new}$
 - end if
 - end for
- end if

CHAPTER 4

RESULTS

4.1 COMPARISON OF GENETIC ALGORITHM, SIMULATED ANNEALING WITH ASSOCIATION RULE MINING FOR SCHEDULING ELECTIVE COURSES

4.1.1 Association Rule Mining

For sixth semester IT students, there are 6 optional electives. Student give their options for the 2 electives they plan to take. Table 2 gives us the support count and support rank for each subject. Support count (SUPCOUNT) is the count for each subject selected by students. The subjects are ranked (SUPRANK) as per the support count of each subject. A threshold value is fixed. Subject having support count < threshold are eliminated.

In our work, we take threshold=10 so MSD is discarded.

| SUBJECTS | SUPCOUNT | SUPRANK |
|-----------------|-----------------|----------------|
| ML | 216 | 1 |
| CFCC | 184 | 2 |
| ADBMS | 48 | 3 |
| RTS | 41 | 4 |
| ON | 12 | 5 |
| HSN | 11 | 6 |
| MSD | 8 | 7 |

Table 2: Support Count and Support Rank for elective subjects

Here the electives are,

Machine Learning (ML), Cyber Forensics and Cyber Crime (CFCC), Advanced Database Management System (ADBMS), Real Time Systems(RTS), Optical networks (ON), High speed Networks (HSN), Multimedia System Design (MSD).

Also SUPCOUNT represents the support count and SUPRANK is the support rank

The associations between electives is found using the FP growth algorithm and are then counted. Associations having highest association count and rank = 1 are the 2 electives.

As per Table 3, the two electives selected are Machine Learning (ML) and Cyber Forensics and Cyber Crime (CFCC).

| ASSOCIATION_ID | ASSOCIATIONS | ASSOCIATION_COUNT | RANK |
|----------------|--------------|-------------------|------|
| 1 | ML, CFCC | 168 | 1 |
| 2 | ML, ADBMS | 22 | 2 |
| 3 | ML, RTS | 20 | 3 |
| 4 | CFCC, ADBMS | 16 | 4 |
| 5 | RTS, ADBMS | 10 | 5 |
| 6 | RTS, HSN | 8 | 6 |
| 7 | ON, ML | 6 | 7 |
| 8 | ON, RTS | 3 | 8 |
| 9 | ON, HSN | 3 | 9 |

Table 3: Associations for Elective Subjects

4.1.2 Comparison of timetables generated by GA and SA

Finally after the timetables are generated using Genetic Algorithm and Simulated Annealing, they are compared on the basis of following factors. The Timetables generated by SA and GA are automated timetables which require no manual work. They generated timetables have constraint violation (hard constraint or soft

constraint). The comparison is done on the basis of two parameters:

1. **Execution/ Makespan time:** It gives the time taken for timetable generation
2. **No. of Generation/ iteration:** This gives the iteration number (for SA) or generation number (for GA) at which we obtain the solution .

| Best Case Makespan | | Average Case Makespan | | Worst Case Makespan | |
|----------------------------|--------------|-----------------------|--------------|---------------------|--------------|
| Generation No. | Time (in ms) | Generation No. | Time (in ms) | Generation No. | Time (in ms) |
| 1 | 170 | 7 | 265 | 13 | 413 |
| Simulated Annealing | | | | | |
| Best Case Makespan | | Average Case Makespan | | Worst Case Makespan | |
| Iteration No. | Time (in ms) | Iteration No. | Time (in ms) | Iteration No. | Time (in ms) |
| 1 | 143 | 5 | 220 | 10 | 311 |

Table 4: Comparison of SA and GA

From the results we see can see that,

1. Simulated annealing takes lesser time to generate timetable as compared to Genetic algorithm and hence gives a faster result. Both genetic algorithm and simulated annealing are much more faster than manual timetable scheduling.
2. Simulated annealing gives a solution in lesser number lesser number of iterations as compared to genetic algorithm.
3. From the Table 4, we can see that Simulated Annealing takes lesser time as compared to GA for all the best, average and worst case scenario. It also takes lesser number of iterations as compared to GA to reach the optimal solution.

4.1.3 Timetables generated by GA and SA

Timetable obtained after GA

| Room: TW1TF3 | | | | |
|--------------|----------------------------|----------------------------|----------------------------|----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 |
| MONDAY | CFCC- T- A- G2: Prof. B | CFCC- T- A- G1: Prof. B | | |
| TUESDAY | | | | |
| WEDNESDAY | | | | |
| THURSDAY | | | | |
| FRIDAY | | | CFCC- T- B- G1: Prof. B | CFCC- T- B- G2: Prof. B |

| Room: TW2GF2 | | | | |
|--------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 |
| MONDAY | CFCC- B- Theory: Prof. C | | | |
| TUESDAY | | CFCC- B- Theory: Prof. C | CFCC- B- Theory: Prof. C | |
| WEDNESDAY | | | CFCC- A- Theory: Prof. B | CFCC- A- Theory: Prof. B |
| THURSDAY | CFCC- A- Theory: Prof. B | | | |
| FRIDAY | | | | |

| Room: TW3TF3 | | | | |
|--------------|-----------------------|--------------------------|--------------------------|-----------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 |
| MONDAY | | | | |
| TUESDAY | | ML- A Theory: Prof. A | ML- A Theory: Prof. A | |
| WEDNESDAY | ML- T- G3: Prof. A | | | |
| THURSDAY | | | ML- A Theory: Prof. A | ML- T- G4: Prof. A |
| FRIDAY | ML- T- G1: Prof. A | ML- T- G2: Prof. A | | |

Fig. 2: Timetable generated by Genetic Algorithm

Timetable obtained after SA

| Room: TW1TF3 | | | | |
|--------------|-----------------------------|-----------------------------|-----------------------------|-------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 |
| MONDAY | CFCC- A- Theory: Prof. B | | | |
| TUESDAY | CFCC- T- B- G1: Prof. B | CFCC- A- Theory: Prof. B | CFCC- A- Theory: Prof. B | |
| WEDNESDAY | CFCC- T- B- G2: Prof. B | | | |
| THURSDAY | | | | |
| FRIDAY | | | CFCC- T- A- G1: Prof. B | |

| Room: TW2GF2 | | | | |
|--------------|-----------------------------|-----------------------------|-----------------------------|-------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 |
| MONDAY | | | | |
| TUESDAY | | | | |
| WEDNESDAY | | | CFCC- B- Theory: Prof. C | |
| THURSDAY | | CFCC- T- A- G2: Prof. B | | |
| FRIDAY | CFCC- B- Theory: Prof. C | CFCC- B- Theory: Prof. C | | |

| Room: TW3TF3 | | | | |
|--------------|--------------------------|-----------------------|-----------------------|-------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 |
| MONDAY | | ML- T- G2: Prof. A | ML- T- G4: Prof. A | |
| TUESDAY | ML- A Theory: Prof. A | | | |
| WEDNESDAY | ML- A Theory: Prof. A | ML- T- G3: Prof. A | | |
| THURSDAY | ML- A Theory: Prof. A | | ML- T- G1: Prof. A | |
| FRIDAY | | | | |

Fig. 3: Timetable generated by Simulated Annealing

4.2 RESULTS FOR PROPOSED TIMETABLE SCHEDULING INCORPORATING TEACHERS'S PREFERENCES FOR SCHEDULING ELECTIVE AND CORE COURSES USING GENETIC ALGORITHM, SIMULATED ANNEALING AND PARTICLE SWARM OPTIMIZATION

4.2.1 Performance Analysis:

In this work, we generate timetables using GA, SA and PSO algorithms by ensuring that no hard constraints are violated and the soft constraints are minimized to the maximum extent. We analyse the performance of GA, SA and PSO on the basis of 2 parameters:

4.2.2 No of penalties vs Generations/Iterations:

The 3 algorithms are compared with respect to number of penalties and generation/iteration number in Fig. 4.

Performance of GA:

The graph in Fig 4 shows that the number of penalties decrease with the increase in generations in GA. Initially there is a huge decrease in penalties and then it becomes almost constant. The optimized fitness function is when the penalties =0.

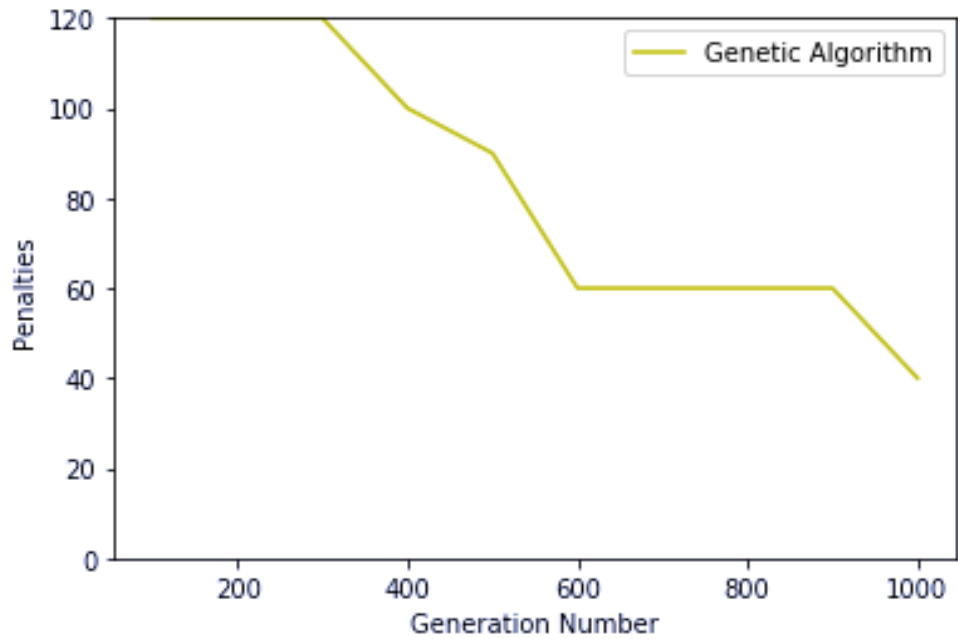


Fig 4: No of penalties vs Generations for Genetic Algorithm

Performance of SA:

This graph in Fig. 5 shows that the number of penalties decrease with the increase in generations in SA. The optimized solution is obtained near iteration = 900 where number of penalties=0 and an optimized timetable is obtained.

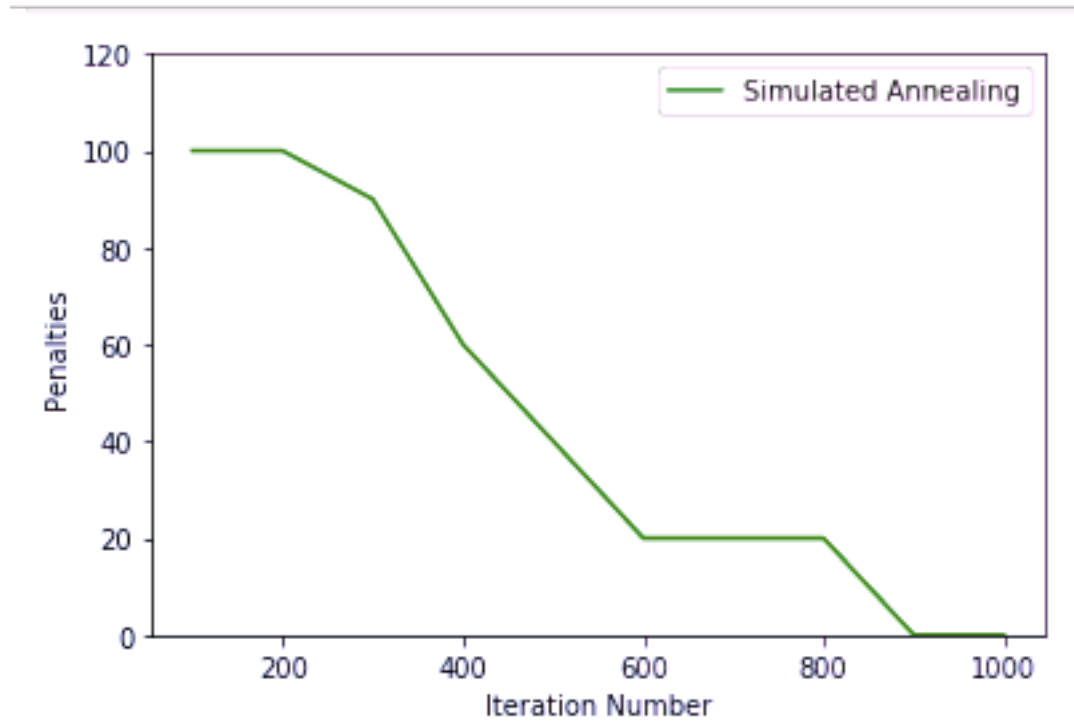


Fig 5: No of penalties vs Iterations for Simulated Annealing

Performance of PSO:

This graph in Fig. 6 shows that the number of penalties decrease with the increase in generations in PSO. The optimized fitness function is when the penalties =0 and then an optimized timetable is obtained.

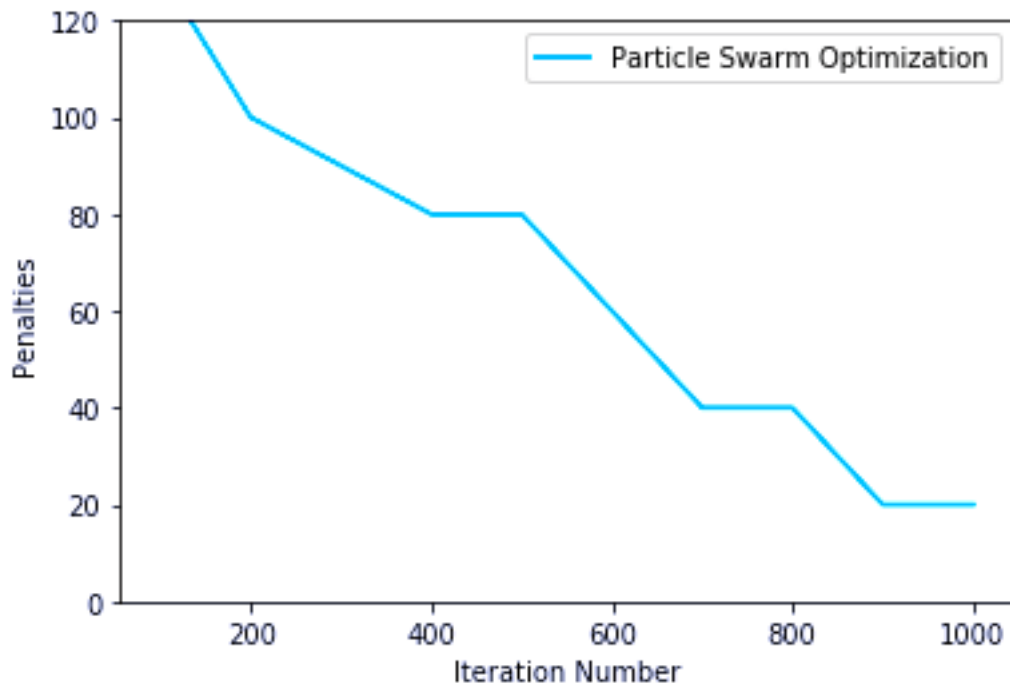
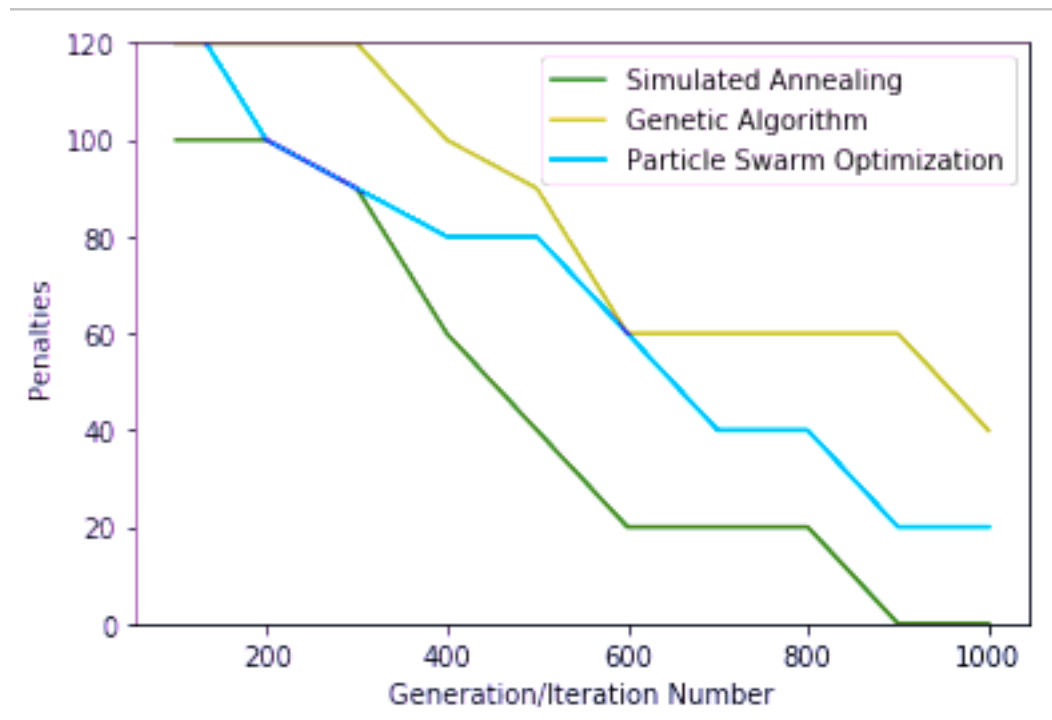


Fig. 6: No of penalties vs Iterations for Particle Swarm Optimization

Comparative graph giving their performance is as below:

The penalties obtained by PSO is smaller than GA at 400th iteration. Simulated Annealing has least amount of penalties and obtains a solution in the least number of iterations.



| | 200 | 400 | 600 | 800 | 1000 |
|-----|-----|-----|-----|-----|------|
| PSO | 100 | 80 | 60 | 40 | 20 |
| GA | 120 | 100 | 60 | 60 | 40 |
| SA | 100 | 60 | 20 | 20 | 0 |

Fig 7: The number of penalties minimized over generation/iteration number

4.2.3 Time taken for execution of each generation/iteration with respect to generation/iteration number

GA, SA and PSO are compared with respect to their execution time (in ms) for every generation/iteration and generation/iteration number.

Performance of GA:

Here we study the execution time of each generation with respect to the generation number. We can observe that for GA, the execution time for consecutive generations decrease with increase in the number of generations. Initial execution time for GA =240 ms and decreases to almost 20ms after 100 generations.

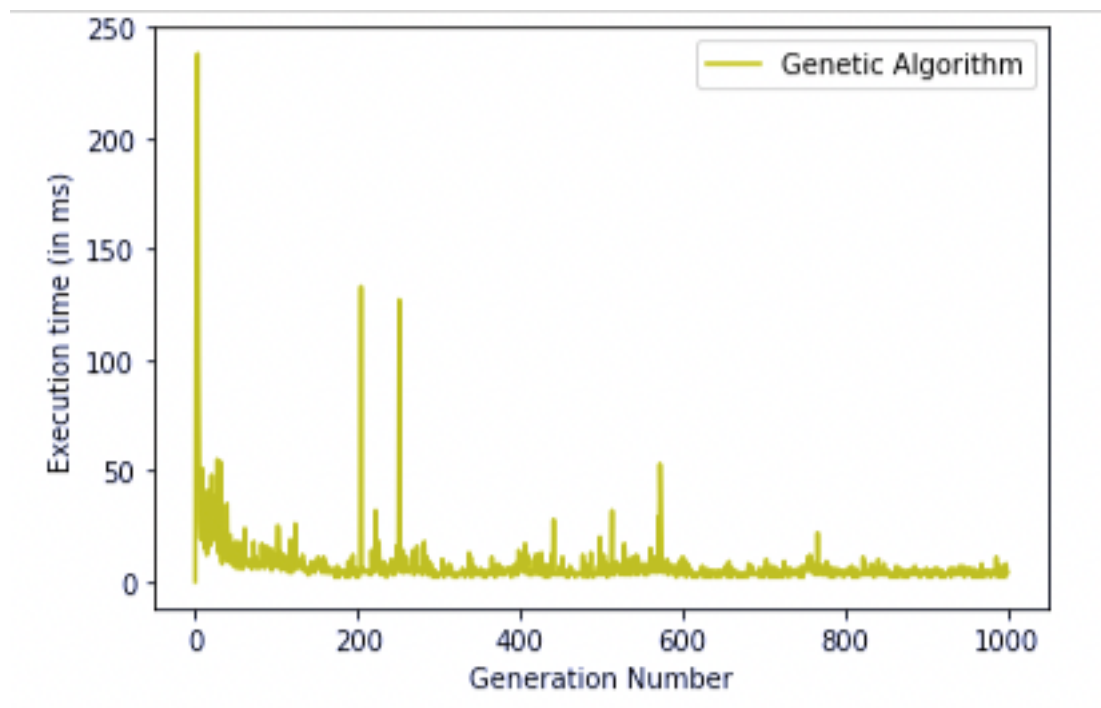


Fig. 8: The variation of execution time (ms) over generation/iteration number for GA

Performance of PSO:

Here we study the execution time of each iteration with respect to the iteration number. We can observe that for PSO, the execution time for consecutive iterations decrease with increase in the number of iterations. We can also observe that execution time for PSO is less than GA and SA (execution time in initial generations = 97ms) and decreases slowly as compared to GA, SA.

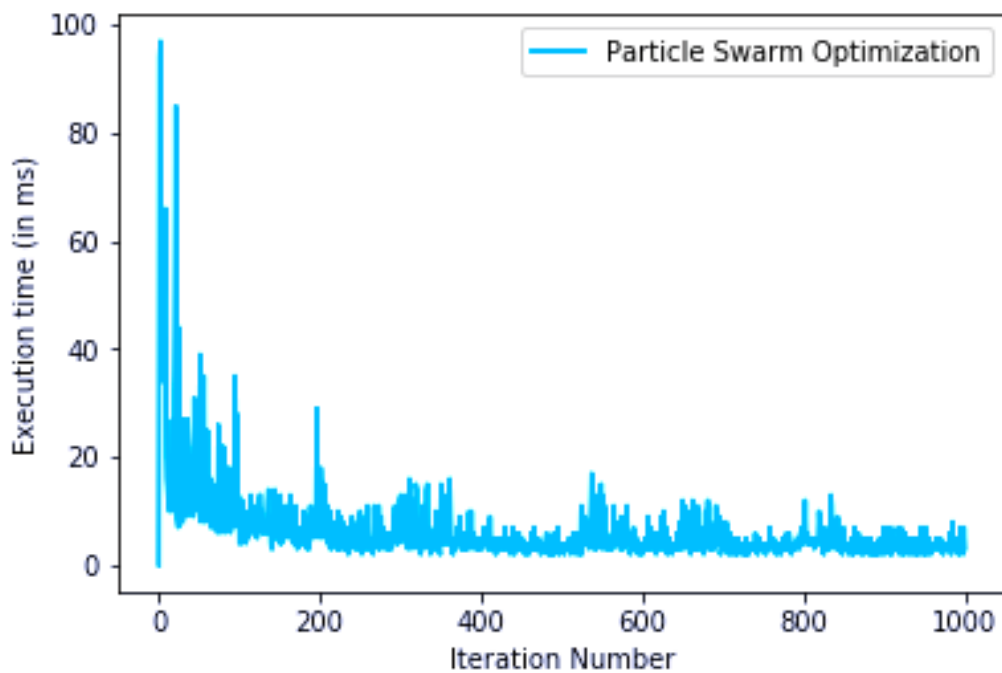


Fig 9: Execution Time for each iteration vs Iteration Number for Particle Swarm Optimization

Performance of SA:

Here we study the execution time of each iteration with respect to the iteration number. We can observe that for SA, the execution time for consecutive iterations decrease with increase in the number of iterations. SA has maximum execution time than GA and PSO in initial iterations i.e. 320 ms. However it decreases very fast to less than 50 ms in less than 100 iterations.

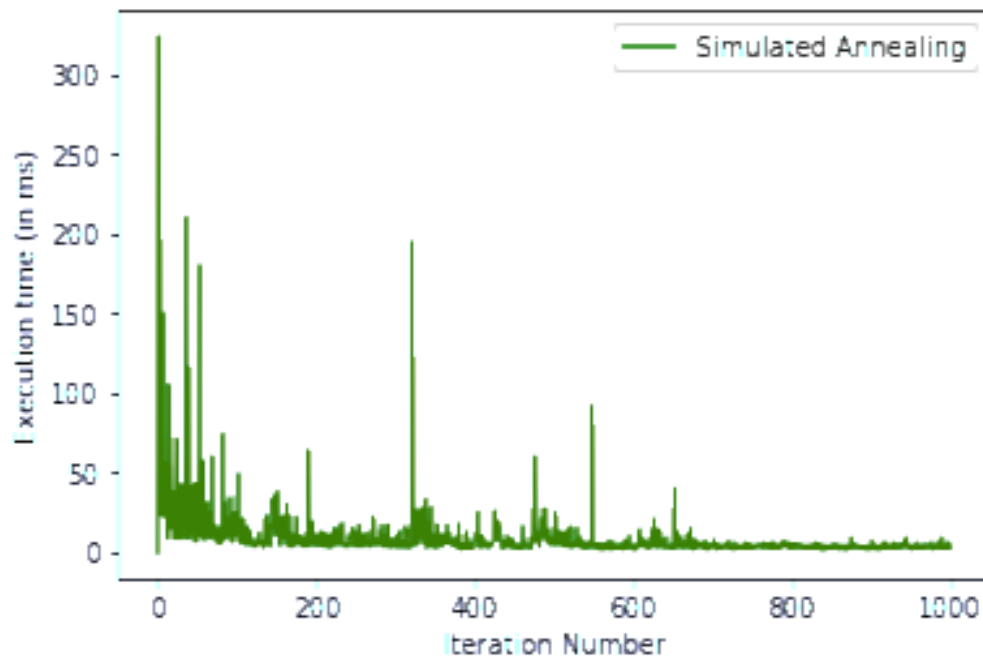


Fig 10: Execution Time for each iteration vs Iteration Number for Simulated Annealing

Comparative performance analysis of GA, SA, PSO

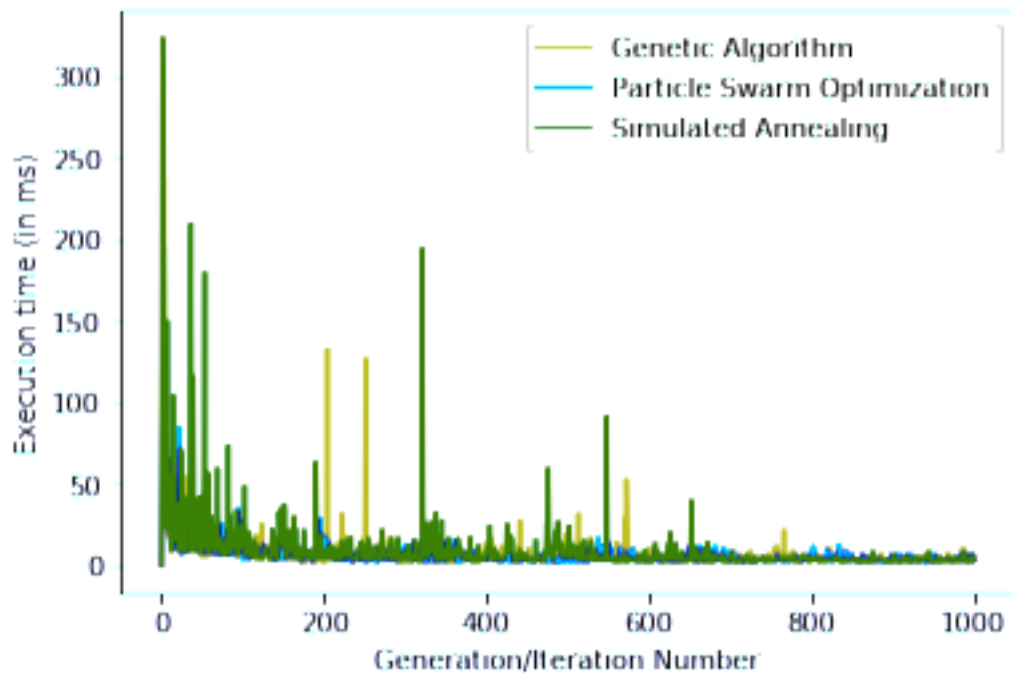


Fig 11: Performance Analysis for 3 algorithms

A comparison of the graphs in Figs. 8-10 indicates that initially, PSO followed by GA has lower execution time. However, overall SA has faster execution as observed from the tail portion of the graph in Fig. 4.

4.2.4 Timetables generated by GA and SA

The teacher preferences are shown in Fig. 12,13. Out of all the professors, the following two professors of Delhi Technological University gave their choice of slots while the other professors indicated that they were comfortable with all slots.

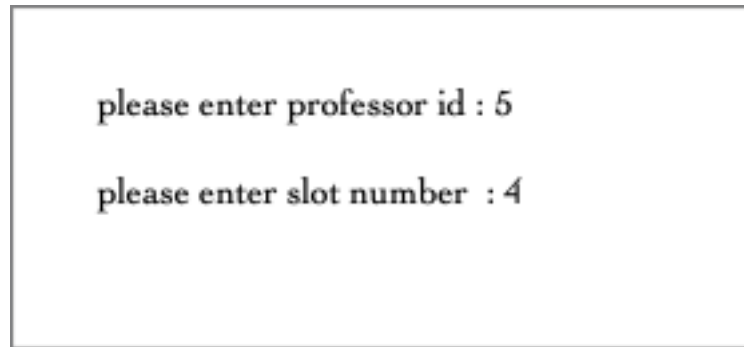


Fig. 12: The system GUI for accepting teacher preference against Teacher ID = 5

Professor 5 is Professor Nidhi and she is given slot 4 i.e. (11:00 - 12:00) and all her classes are allotted in this slot for 6-B.

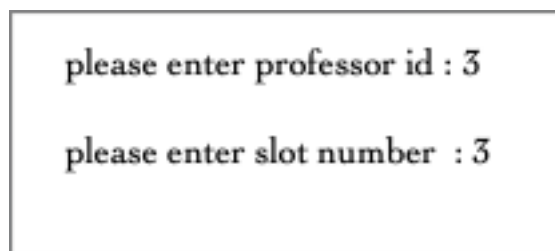


Fig. 13: The system GUI for accepting teacher preference against Teacher ID = 3

Professor 3 is Professor Rahul Katarya and he is given slot 3 i.e. (10:00 - 11:00) and all his classes are allotted in this slot for 6-B.

The timetables generated by are shown below in Figs. 6, 7, 8 for GA, SA and PSO respectively for the two sections 6-A and 6-B.

Timetable obtained after GA

| Room: TW2GF2 | 6-B | | | | | | | |
|------------------|----------------------------|----------------------------|-----------------------------|-------------------------|----------------------------|----------------------------|----------------------------|----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 | 12:00- 13:00 | 13:00- 14:00 | 14:00- 15:00 | 15:00- 16:00 |
| MONDAY | CFCC- B Theory: Prof. C | CFCC- T- B- G2: Prof. B | SE-B Theory: Prof: H | | AI Lab - B- G2: Prof: A | AI Lab - B- G2: Prof: A | CD Lab - B- G1: Prof: D | CD Lab - B- G1: Prof: D |
| TUESDAY | ML- B Theory: Prof. A | CFCC- T- B- G1: Prof. B | SE - B - T - G2: Prof: H | AI-B Theory: Prof: I | | CD Lab - B- G2: Prof: D | CD Lab - B- G2: Prof: D | |
| WEDNESDAY | ML- B Theory: Prof. A | ML- B Theory: Prof. A | SE - B - T - G1: Prof: H | AI-B Theory: Prof: I | CD-B Theory: Prof: J | | | |
| THURSDAY | ML- T- B- G1: Prof. A | CFCC- B Theory: Prof. C | SE-B Theory: Prof: H | | AI Lab - B- G1: Prof: A | AI Lab - B- G1: Prof: A | | |
| FRIDAY | ML- T- B- G2: Prof. A | CFCC- B Theory: Prof. C | SE-B Theory: Prof: H | AI-B Theory: Prof: I | | CD-B Theory: Prof: J | CD-B Theory: Prof: J | |

| Room: TW1TF3 | 6-A | | | | | | | |
|------------------|----------------------------|----------------------------|-------------------------|-----------------------------|-----------------------------|----------------------------|----------------------------|----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 | 12:00- 13:00 | 13:00- 14:00 | 14:00- 15:00 | 15:00- 16:00 |
| MONDAY | CFCC- A Theory: Prof. B | CFCC- A Theory: Prof. B | SE-A Theory: Prof: E | SE-A Theory: Prof: E | | | AI Lab - A- G1: Prof: G | AI Lab - A- G1: Prof: G |
| TUESDAY | ML- A Theory: Prof. A | CFCC- A Theory: Prof. B | | SE-A Theory: Prof: E | SE - A - T - G2: Prof: E | | AI Lab - A- G2: Prof: A | AI Lab - A- G2: Prof: A |
| WEDNESDAY | CFCC- T- A- G1: Prof. B | ML- T- A- G1: Prof. A | AI-A Theory: Prof: F | SE - A - T - G1: Prof: E | | | CD Lab - A- G2: Prof: D | CD Lab - A- G2: Prof: D |
| THURSDAY | CFCC- T- A- G2: Prof. B | ML- A Theory: Prof. A | AI-A Theory: Prof: F | CD-A Theory: Prof: D | | CD Lab - A- G1: Prof: D | CD Lab - A- G1: Prof: D | |
| FRIDAY | ML- A Theory: Prof. A | ML- T- A- G2: Prof. A | AI-A Theory: Prof: F | | CD-A Theory: Prof: D | CD-A Theory: Prof: D | | |

Fig. 14: Two instances of timetables generated by GA for the two sections A and B

Timetable generated using SA

| Room: TW2GF2 | 6-B | | | | | | | |
|------------------|----------------------------|----------------------------|-----------------------------|-------------------------|----------------------------|----------------------------|----------------------------|----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 | 12:00- 13:00 | 13:00- 14:00 | 14:00- 15:00 | 15:00- 16:00 |
| MONDAY | CFCC- B Theory: Prof. C | CFCC- T- B- G2: Prof. B | SE - B - T - G2: Prof: H | | CD Lab - B- G1: Prof: D | CD Lab - B- G1: Prof: D | CD Lab - B- G2: Prof: D | CD Lab - B- G2: Prof: D |
| TUESDAY | CFCC- B Theory: Prof. C | ML- T- B- G2: Prof. A | SE-B Theory: Prof: H | AI-B Theory: Prof: I | CD-B Theory: Prof: J | | CD-B Theory: Prof: J | |
| WEDNESDAY | CFCC- T- B- G1: Prof. B | ML- T- B- G1: Prof. A | SE - B - T - G1: Prof: H | AI-B Theory: Prof: I | | CD-B Theory: Prof: J | | |
| THURSDAY | CFCC- B Theory: Prof. C | ML- B Theory: Prof. A | SE-B Theory: Prof: H | AI-B Theory: Prof: I | | | | |
| FRIDAY | ML- B Theory: Prof. A | ML- B Theory: Prof. A | SE-B Theory: Prof: H | | AI Lab - B- G1: Prof: A | AI Lab - B- G1: Prof: A | AI Lab - B- G2: Prof: A | AI Lab - B- G2: Prof: A |

| Room: TW1TF3 | 6-A | | | | | | | |
|------------------|----------------------------|----------------------------|-----------------------------|----------------------------|-------------------------|----------------------------|----------------------------|-----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 | 12:00- 13:00 | 13:00- 14:00 | 14:00- 15:00 | 15:00- 16:00 |
| MONDAY | ML- A Theory: Prof. A | CFCC- A Theory: Prof. B | CD-A Theory: Prof: D | | AI-A Theory: Prof: F | AI Lab - A- G2: Prof: A | AI Lab - A- G2: Prof: A | |
| TUESDAY | CFCC- A Theory: Prof. B | ML- A Theory: Prof. A | CD-A Theory: Prof: D | CD-A Theory: Prof: D | | CD Lab - A- G2: Prof: D | CD Lab - A- G2: Prof: D | |
| WEDNESDAY | CFCC- T- A- G1: Prof. B | ML- A Theory: Prof. A | | AI-A Theory: Prof: F | AI-A Theory: Prof: F | | | |
| THURSDAY | CFCC- T- A- G2: Prof. B | ML- T- A- G2: Prof. A | SE - A - T - G1: Prof: E | SE-A Theory: Prof: E | SE-A Theory: Prof: E | CD Lab - A- G1: Prof: D | CD Lab - A- G1: Prof: D | |
| FRIDAY | ML- T- A- G1: Prof. A | CFCC- A Theory: Prof. B | AI Lab - A- G1: Prof: G | AI Lab - A- G1: Prof: G | SE-A Theory: Prof: E | | | SE - A - T - G2: Prof: E |

Fig. 15: Two instances of timetables generated by SA for the two sections A and B

Timetable generated using PSO

| Room: TW2GF2 | 6-B | | | | | | | |
|------------------|----------------------------|----------------------------|-----------------------------|-------------------------|----------------------------|----------------------------|----------------------------|----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 | 12:00- 13:00 | 13:00- 14:00 | 14:00- 15:00 | 15:00- 16:00 |
| MONDAY | CFCC- B Theory: Prof. C | ML- T- B- G2: Prof. A | SE-B Theory: Prof: H | AI-B Theory: Prof: I | CD Lab - B- G2: Prof: D | CD Lab - B- G2: Prof: D | | |
| TUESDAY | ML- B Theory: Prof. A | ML- T- B- G1: Prof. A | SE-B Theory: Prof: H | | AI Lab - B- G1: Prof: A | AI Lab - B- G1: Prof: A | | |
| WEDNESDAY | CFCC- T- B- G1: Prof. B | CFCC- T- B- G2: Prof. B | SE - B - T - G1: Prof: H | AI-B Theory: Prof: I | | AI Lab - B- G2: Prof: A | AI Lab - B- G2: Prof: A | |
| THURSDAY | ML- B Theory: Prof. A | ML- B Theory: Prof. A | SE-B Theory: Prof: H | AI-B Theory: Prof: I | CD-B Theory: Prof: J | | | |
| FRIDAY | CFCC- B Theory: Prof. C | CFCC- B Theory: Prof. C | SE - B - T - G2: Prof: H | | CD-B Theory: Prof: J | CD-B Theory: Prof: J | CD Lab - B- G1: Prof: D | CD Lab - B- G1: Prof: D |

| Room: TW1TF3 | 6-A | | | | | | | |
|------------------|----------------------------|----------------------------|-----------------------------|----------------------------|----------------------------|----------------------------|-----------------------------|----------------------------|
| Time | 08:00- 09:00 | 09:00- 10:00 | 10:00- 11:00 | 11:00-12:00 | 12:00- 13:00 | 13:00- 14:00 | 14:00- 15:00 | 15:00- 16:00 |
| MONDAY | ML- T- A- G2: Prof. A | ML- A Theory: Prof. A | | AI Lab - A- G2: Prof: A | AI Lab - A- G2: Prof: A | | CD-A Theory: Prof: D | |
| TUESDAY | CFCC- A Theory: Prof. B | CFCC- A Theory: Prof. B | | SE-A Theory: Prof: E | | CD-A Theory: Prof: D | CD-A Theory: Prof: D | |
| WEDNESDAY | CFCC- A Theory: Prof. B | ML- A Theory: Prof. A | CD Lab - A- G2: Prof: D | CD Lab - A- G2: Prof: D | | AI Lab - A- G1: Prof: G | AI Lab - A- G1: Prof: G | |
| THURSDAY | CFCC- T- A- G2: Prof. B | ML- A Theory: Prof. A | SE - A - T - G2: Prof: E | AI-A Theory: Prof: F | AI-A Theory: Prof: F | | CD Lab - A- G1: Prof: D | CD Lab - A- G1: Prof: D |
| FRIDAY | CFCC- T- A- G1: Prof. B | ML- T- A- G1: Prof. A | SE-A Theory: Prof: E | SE-A Theory: Prof: E | | AI-A Theory: Prof: F | SE - A - T - G1: Prof: E | |

Fig. 16: Two instances of timetables generated by PSO for the two sections A and B

4.3 PERFORMANCE ANALYSIS OF MEMETIC ALGORITHM

The dataset used for memetic algorithm is the ITC-2007: Curriculum based Course Timetabling (ITC-2007) dataset Track 3 for curriculum-based university. This was the benchmark dataset used for international timetable scheduling competition specifically for university timetabling problem.

The proposed memetic algorithm was implemented as per the steps in Section 4.2.3. The results were compared to the course scheduling approaches using GA [28, 29], SA [30, 31], the hybrid GA-SA [32], Mahfoud *et al.*'s method [33] and the local search algorithm [34] both in terms of runtime penalty cost and soft constraint satisfaction.

| S.No. | GA [29] | SA [30] | GA-SA [32] | Mahfoud et al [33] | Local search algorithm [34] | Memetic algo | Memetic +LSM |
|-------|---------|---------|------------|--------------------|-----------------------------|--------------|--------------|
| 1 | 486 | 470 | 461 | 412 | 448 | 411 | 411 |
| 2 | 480 | 474 | 462 | 423 | 461 | 423 | 423 |
| 3 | 149 | 146 | 141 | 131 | 140 | 128 | 128 |
| 4 | 411 | 401 | 400 | 394 | 402 | 394 | 394 |
| 5 | 583 | 546 | 545 | 494 | 500 | 533 | 480 |
| 6 | 489 | 480 | 479 | 470 | 476 | 469 | 469 |
| 7 | 455 | 460 | 440 | 436 | 440 | 436 | 436 |
| 8 | 465 | 430 | 420 | 423 | 419 | 411 | 410 |
| 9 | 430 | 422 | 411 | 405 | 409 | 405 | 405 |
| 10 | 440 | 437 | 416 | 416 | 415 | 414 | 416 |
| 11 | 112 | 111 | 112 | 111 | 111 | 109 | 109 |
| 12 | 544 | 532 | 499 | 484 | 490 | 480 | 480 |
| 13 | 460 | 444 | 426 | 420 | 424 | 420 | 420 |
| 14 | 527 | 527 | 518 | 513 | 512 | 511 | 510 |
| 15 | 591 | 594 | 591 | 589 | 591 | 588 | 588 |
| 16 | 498 | 498 | 491 | 490 | 491 | 490 | 490 |
| 17 | 432 | 428 | 411 | 411 | 411 | 410 | 410 |
| 18 | 489 | 489 | 484 | 485 | 484 | 482 | 482 |
| 19 | 587 | 579 | 577 | 566 | 572 | 566 | 566 |
| 20 | 599 | 598 | 596 | 584 | 592 | 582 | 580 |
| 21 | 523 | 523 | 520 | 511 | 519 | 511 | 510 |

Table 5: Comparison of Soft Constraint values for all the algorithms

As per Table 5, we can observe that the soft constraint violation in minimum for Memetic Algorithm with LSM. This proves that the solutions generated by memetic Memetic Algorithm with LSM are highly optimal as compared to other solutions.

The fitness functions are plotted with respect to time for all 21 instances as shown in Fig. 9,10,11,12,13,14,15,16,17. The graphs indicate a considerably low value for Memetic Algorithm with LSM when compared to all the other methods. We can also observe that Memetic Algorithm with LSM takes lesser time to reach the optimal solution.

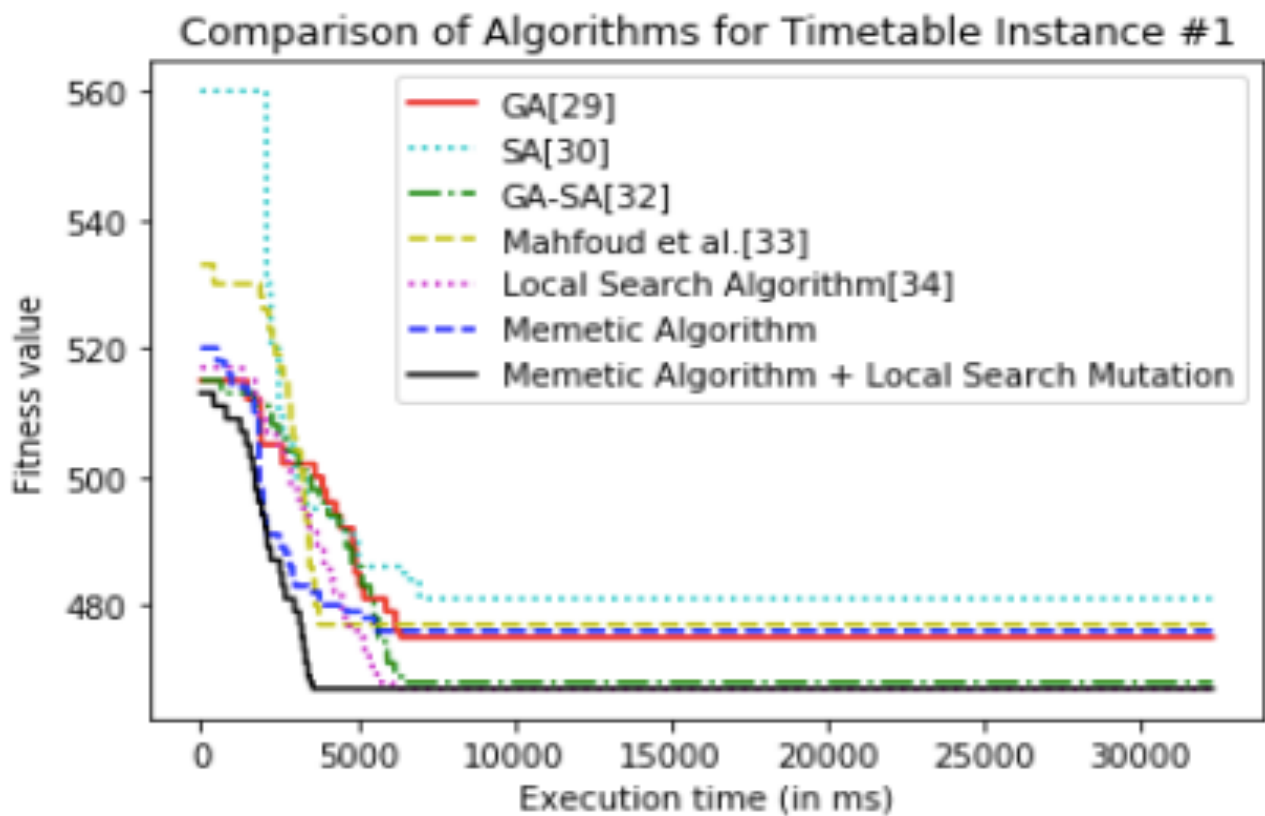


Fig. 17: Comparison of all the algorithms for Instance 1

Comparison of Algorithms for Timetable Instance #2

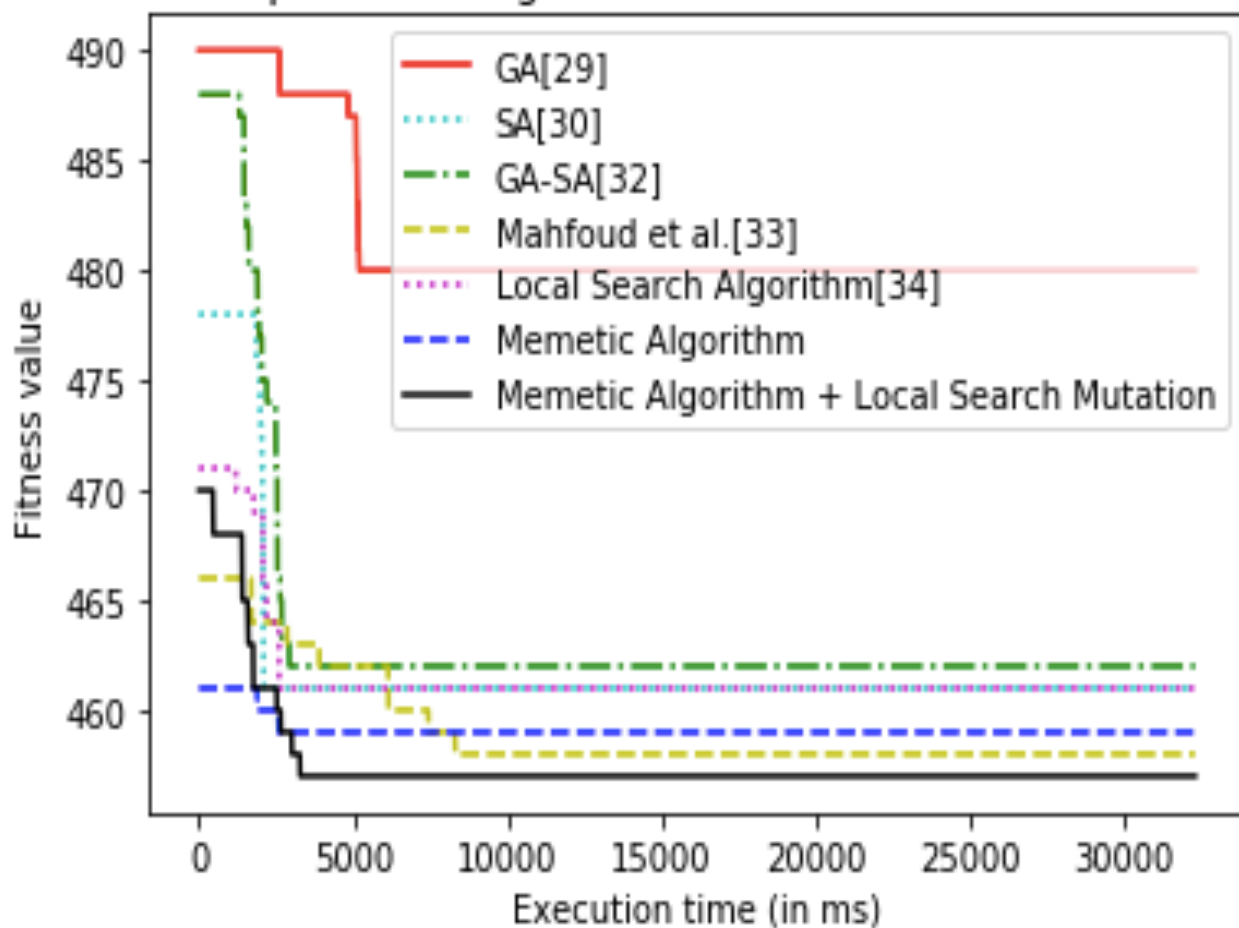


Fig. 18: Comparison of all the algorithms for Instance 2

Comparison of Algorithms for Timetable Instance #3

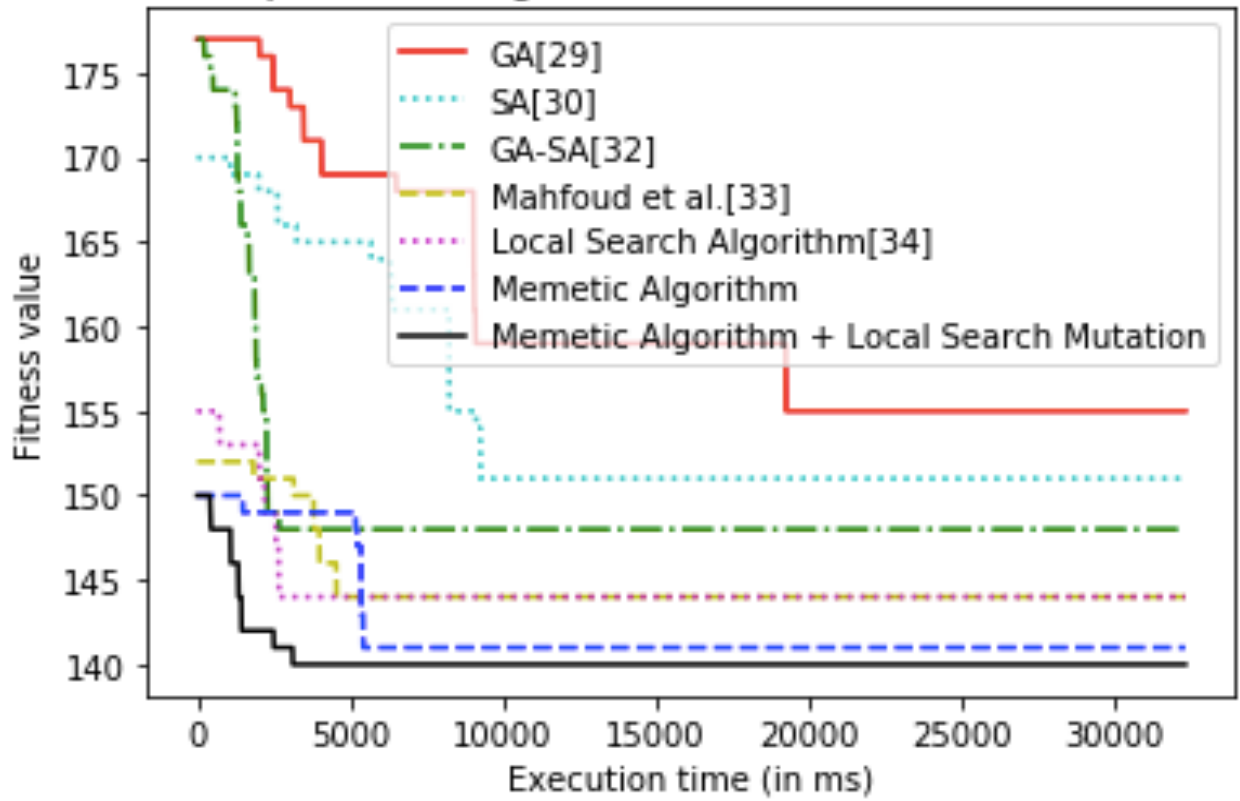


Fig. 19: Comparison of all the algorithms for Instance 3

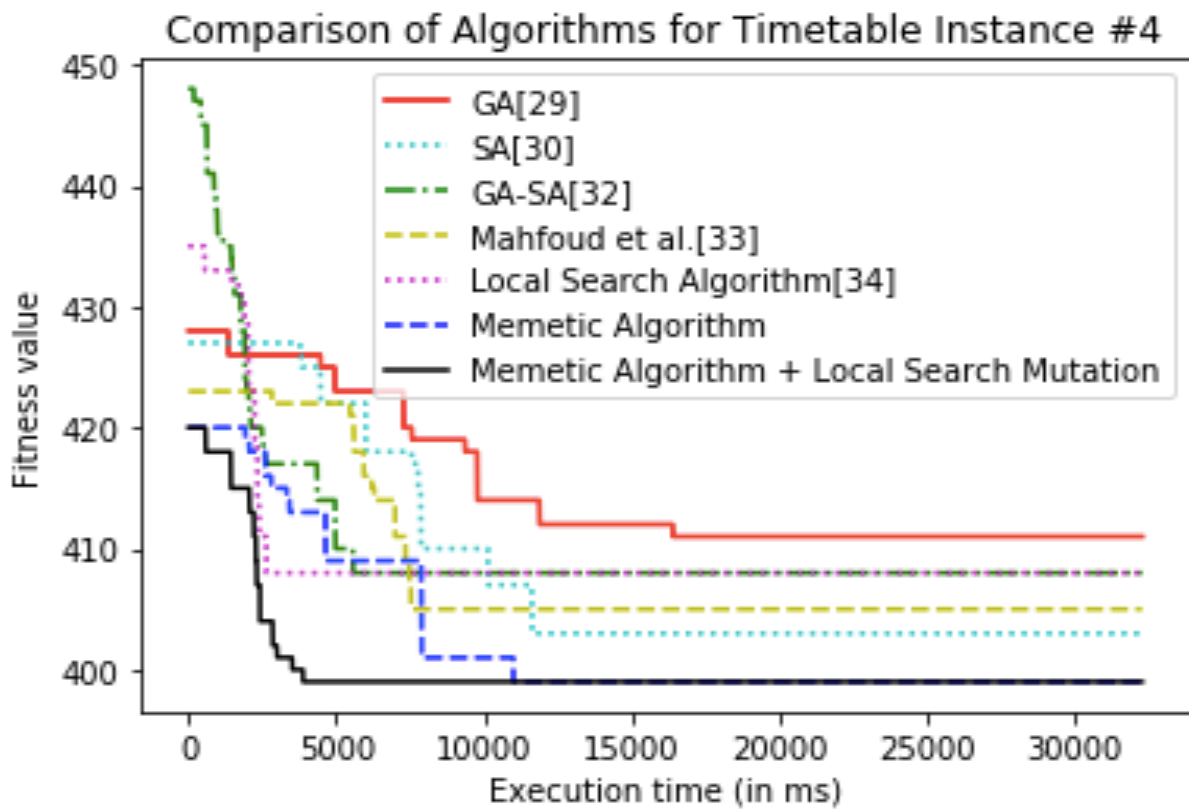


Fig. 20: Comparison of all the algorithms for Instance 4

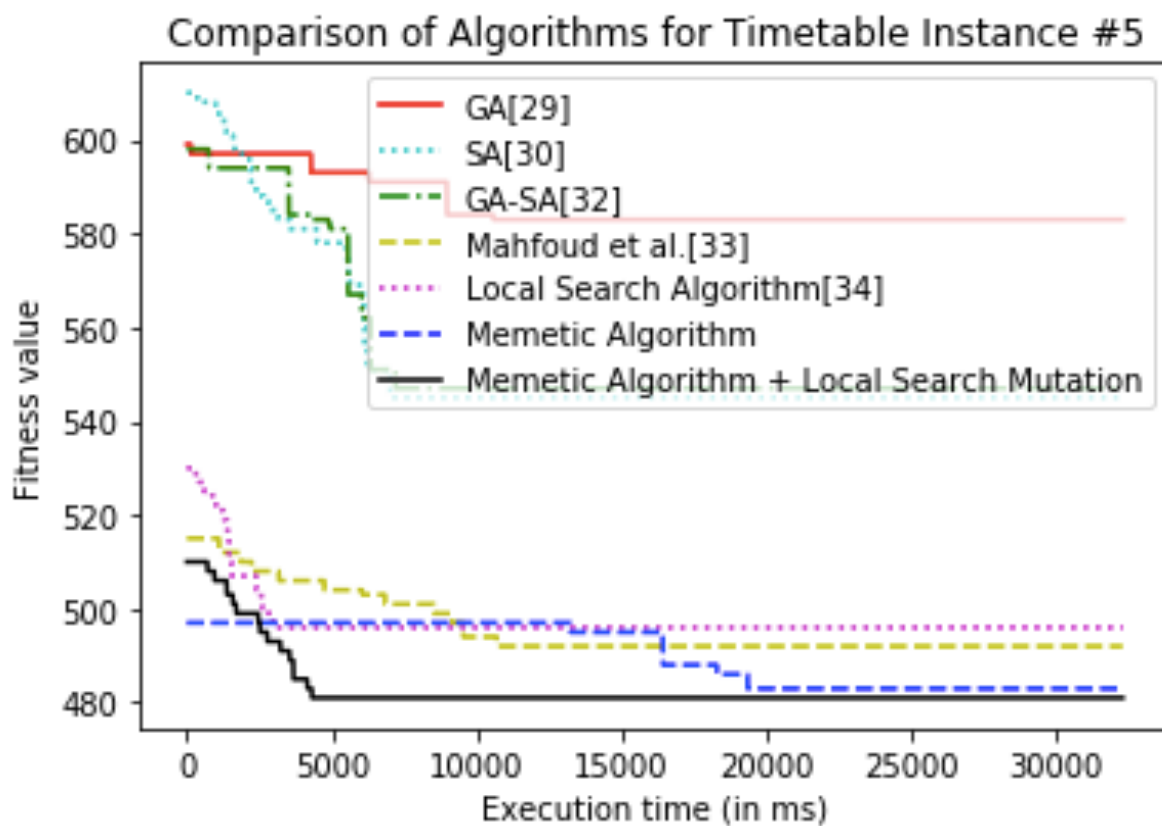


Fig. 21: Comparison of all the algorithms for Instance 5

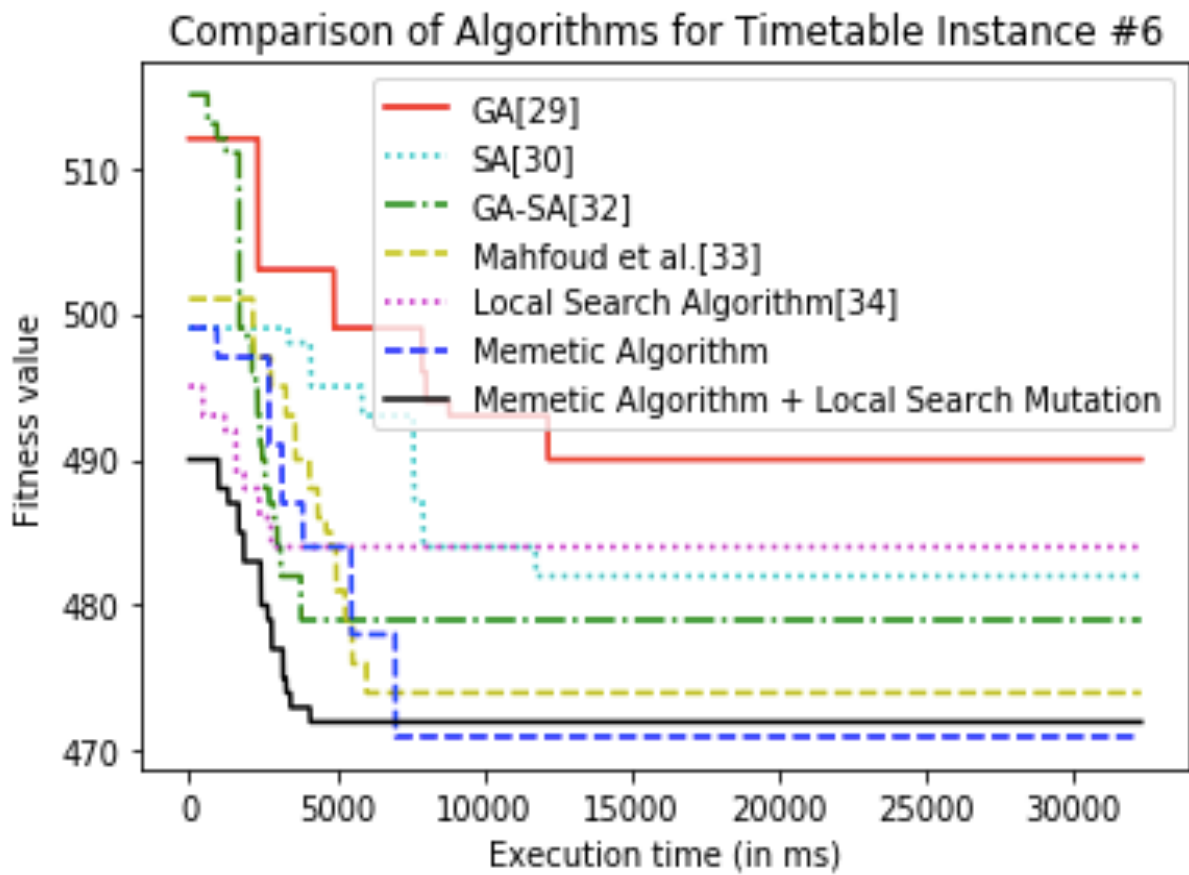


Fig. 22: Comparison of all the algorithms for Instance 6

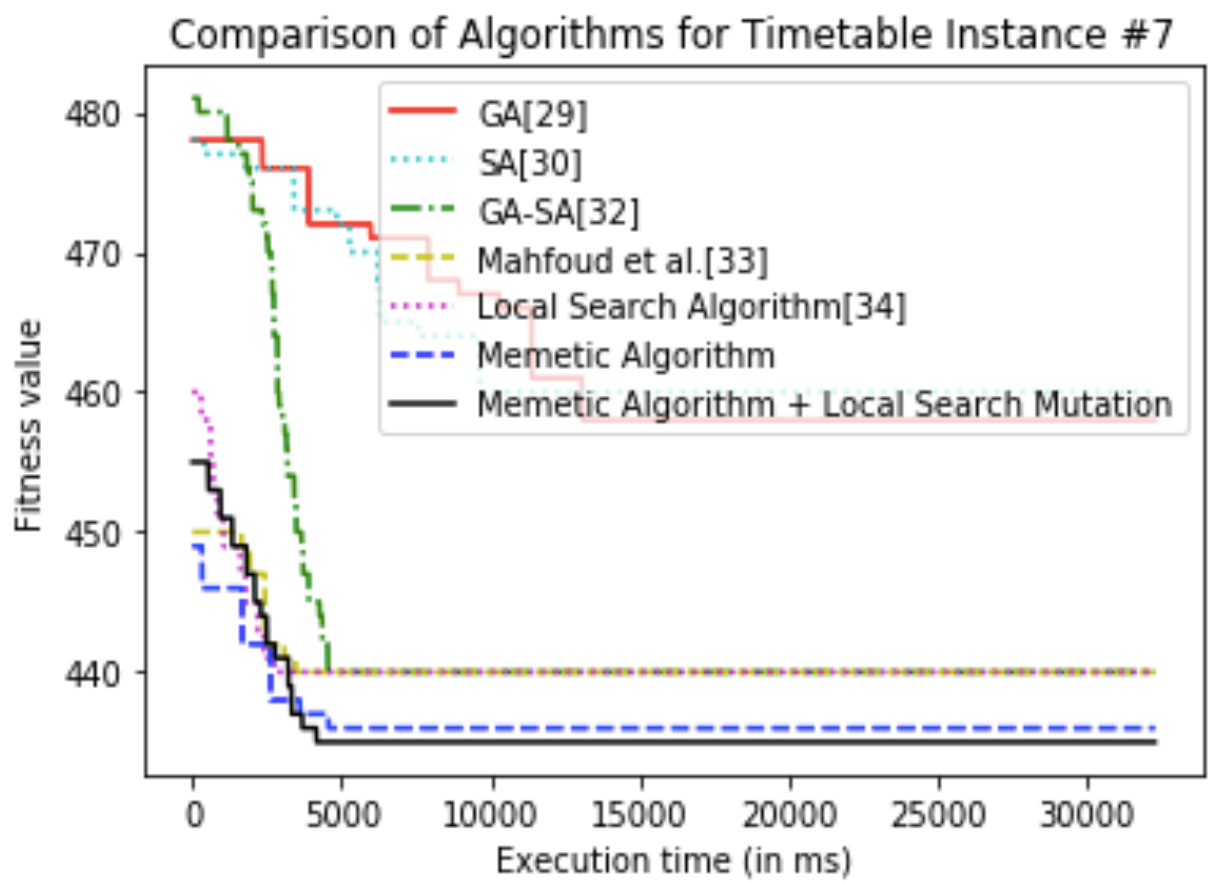


Fig. 23: Comparison of all the algorithms for Instance 7

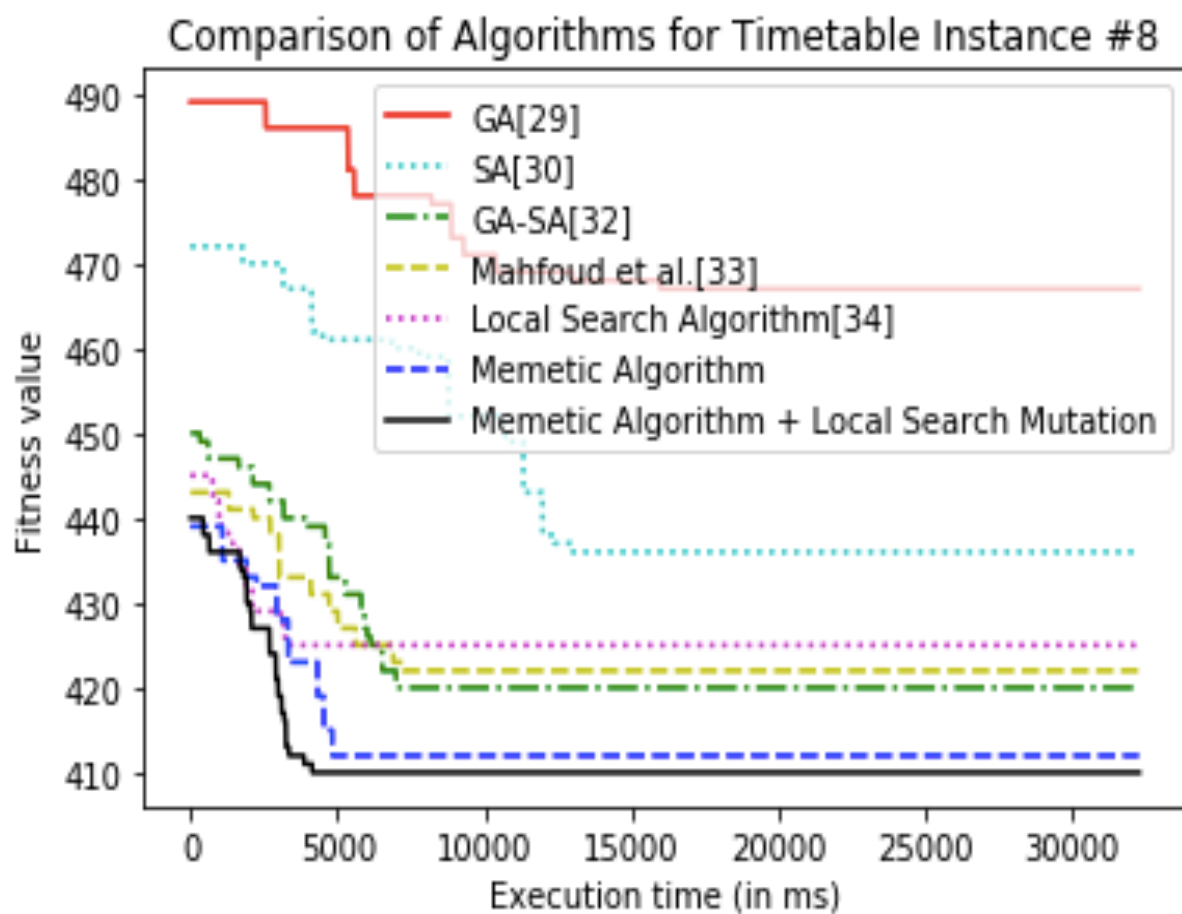


Fig. 24: Comparison of all the algorithms for Instance 8

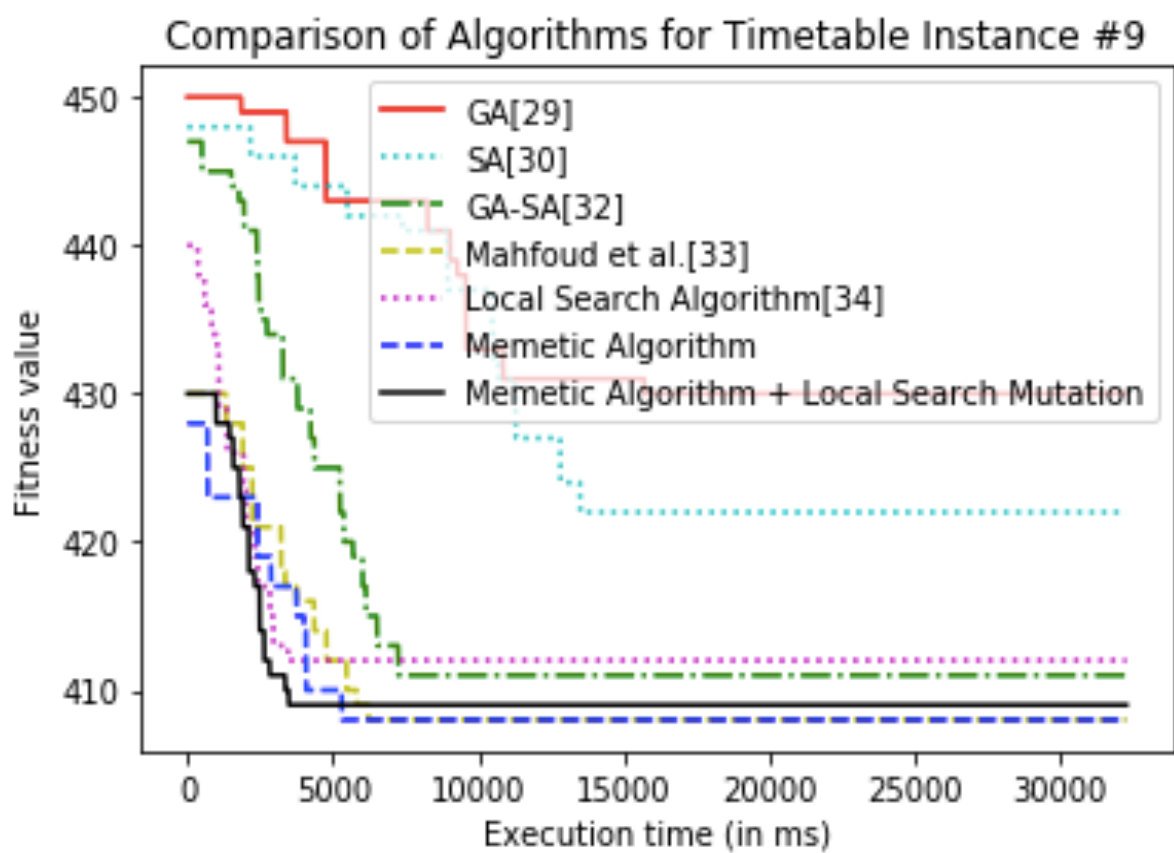


Fig. 25: Comparison of all the algorithms for Instance 9

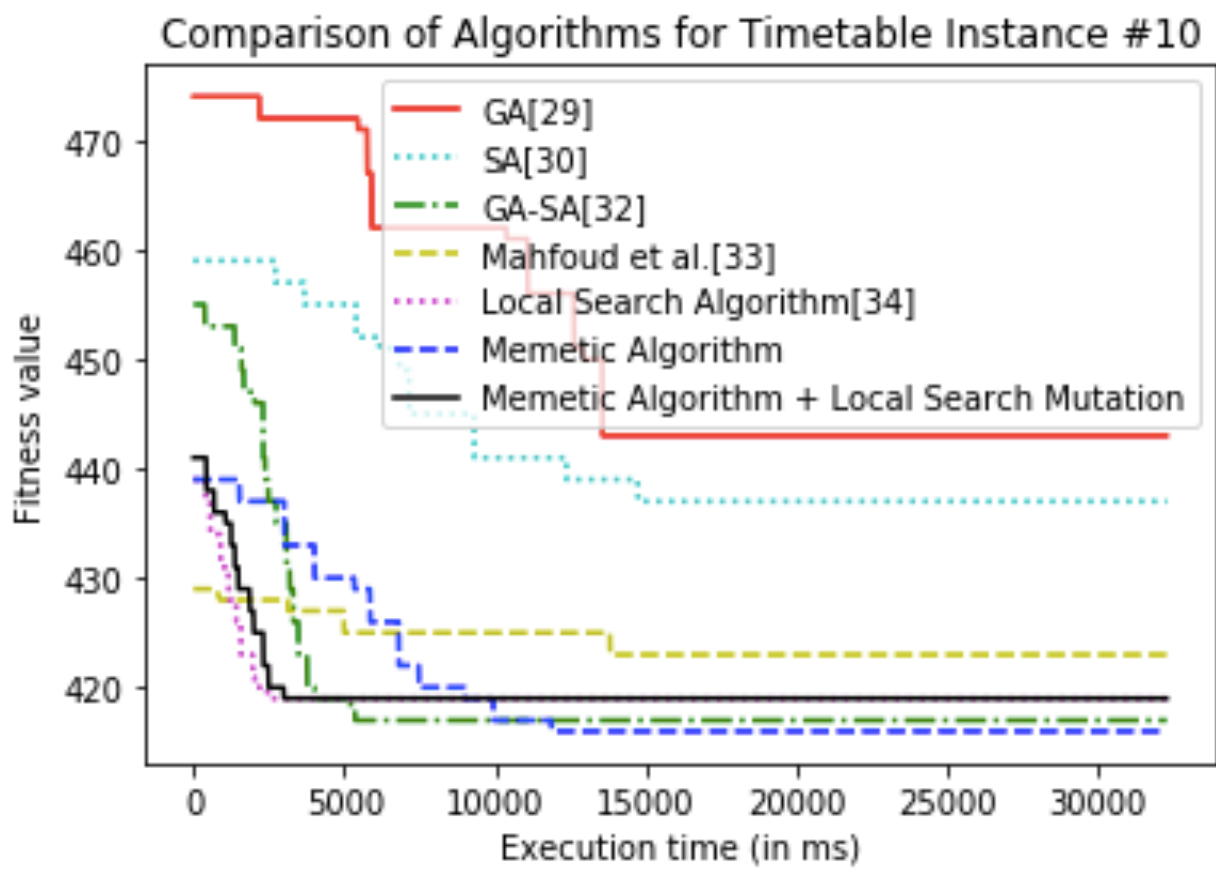


Fig. 26: Comparison of all the algorithms for Instance 10

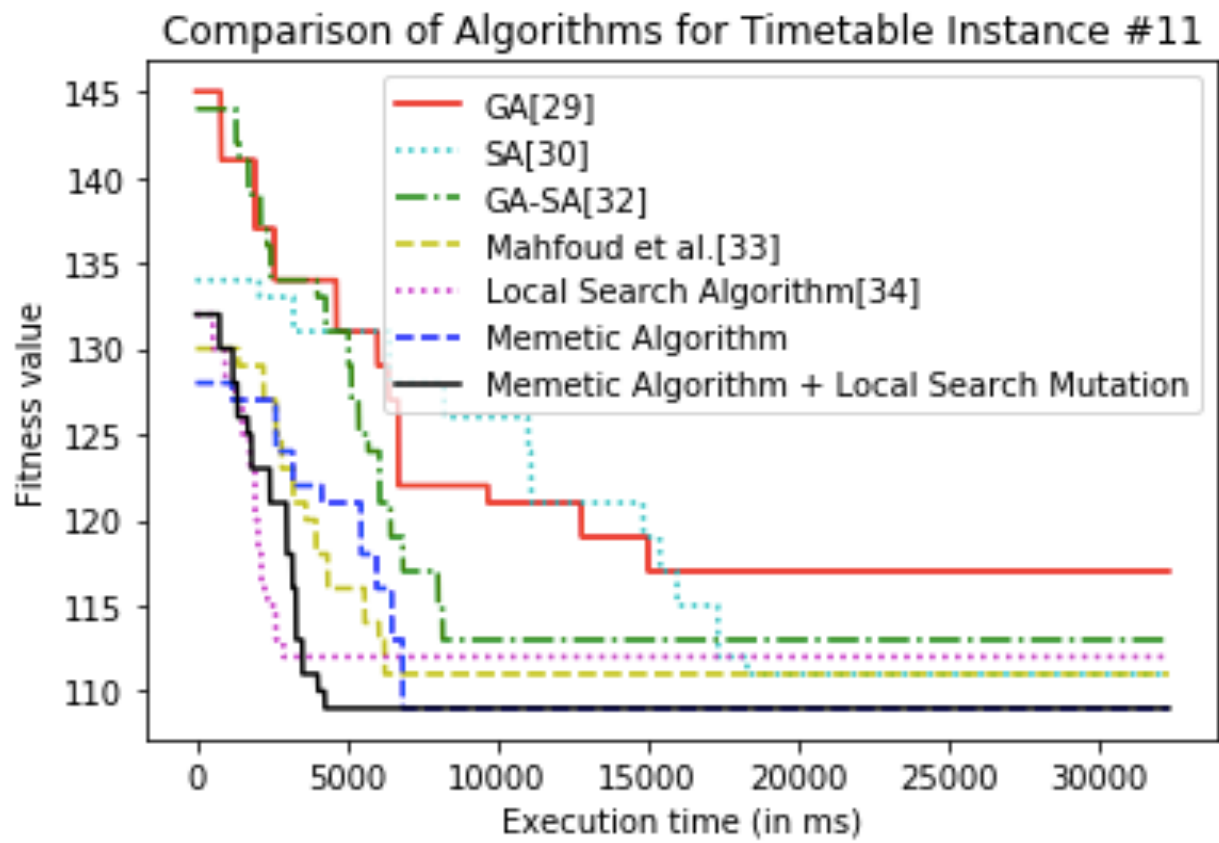


Fig. 27: Comparison of all the algorithms for Instance 11

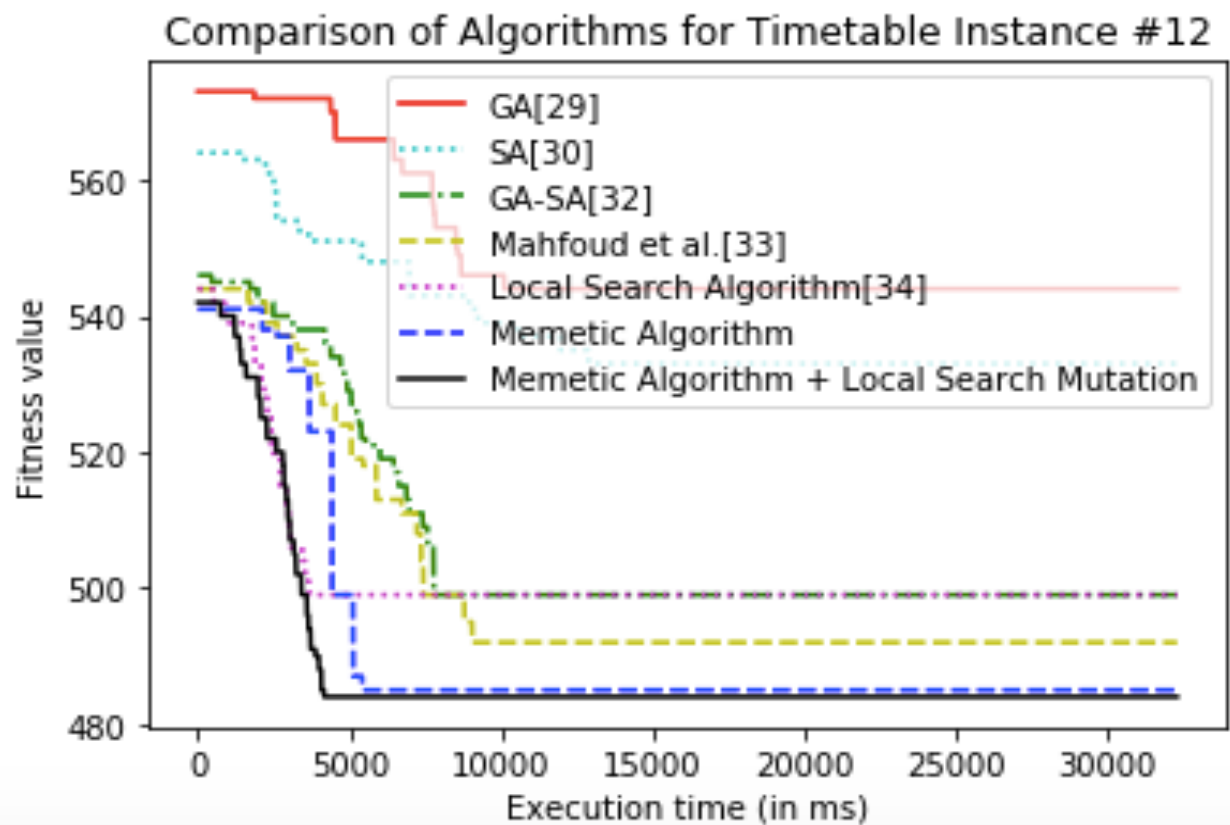


Fig. 28: Comparison of all the algorithms for Instance 12

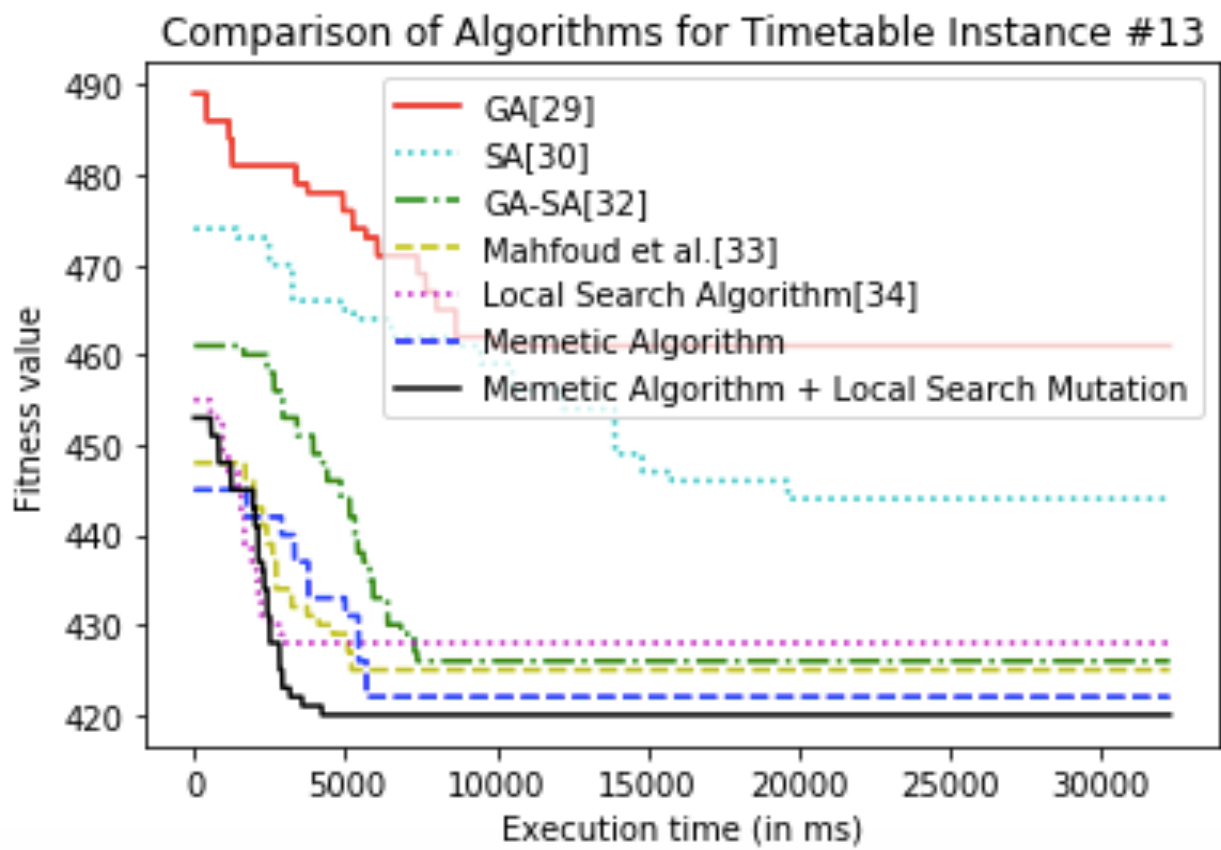


Fig. 29: Comparisons of all the algorithms for Instance 13

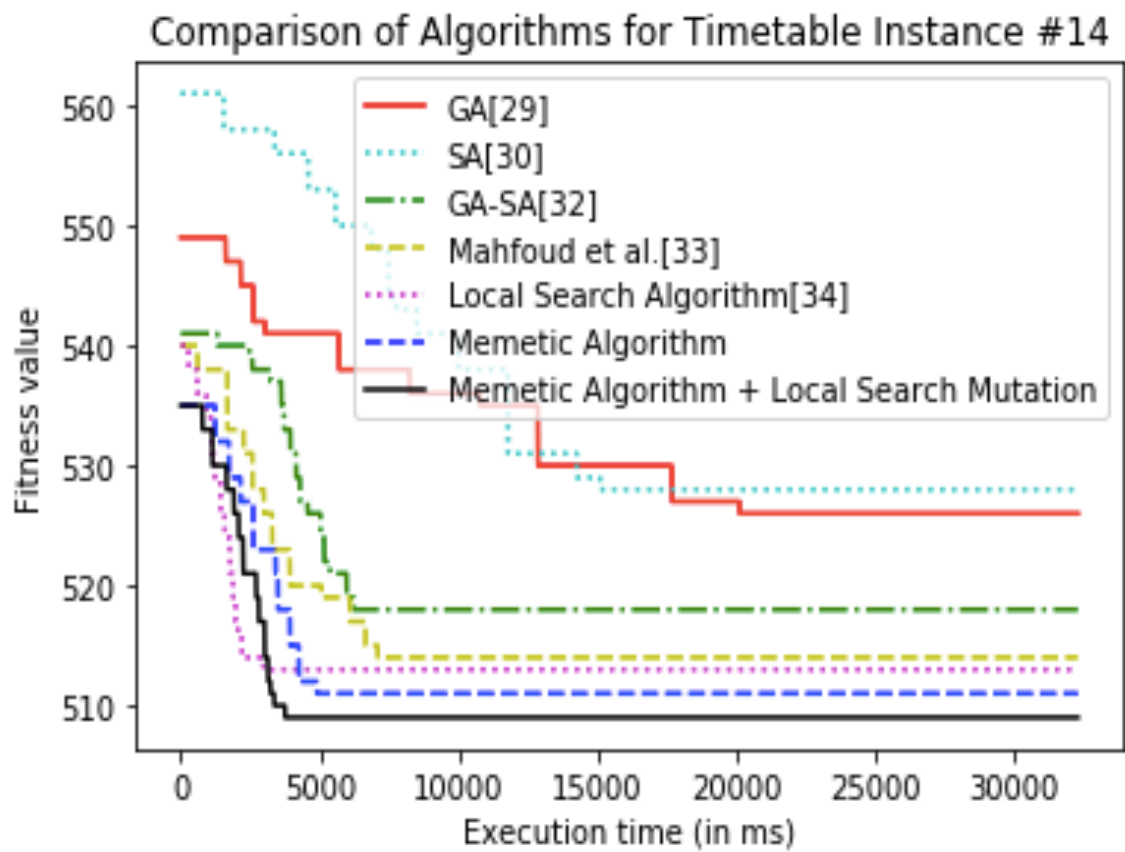


Fig 30: Comparisons of all the algorithms for Instance 14

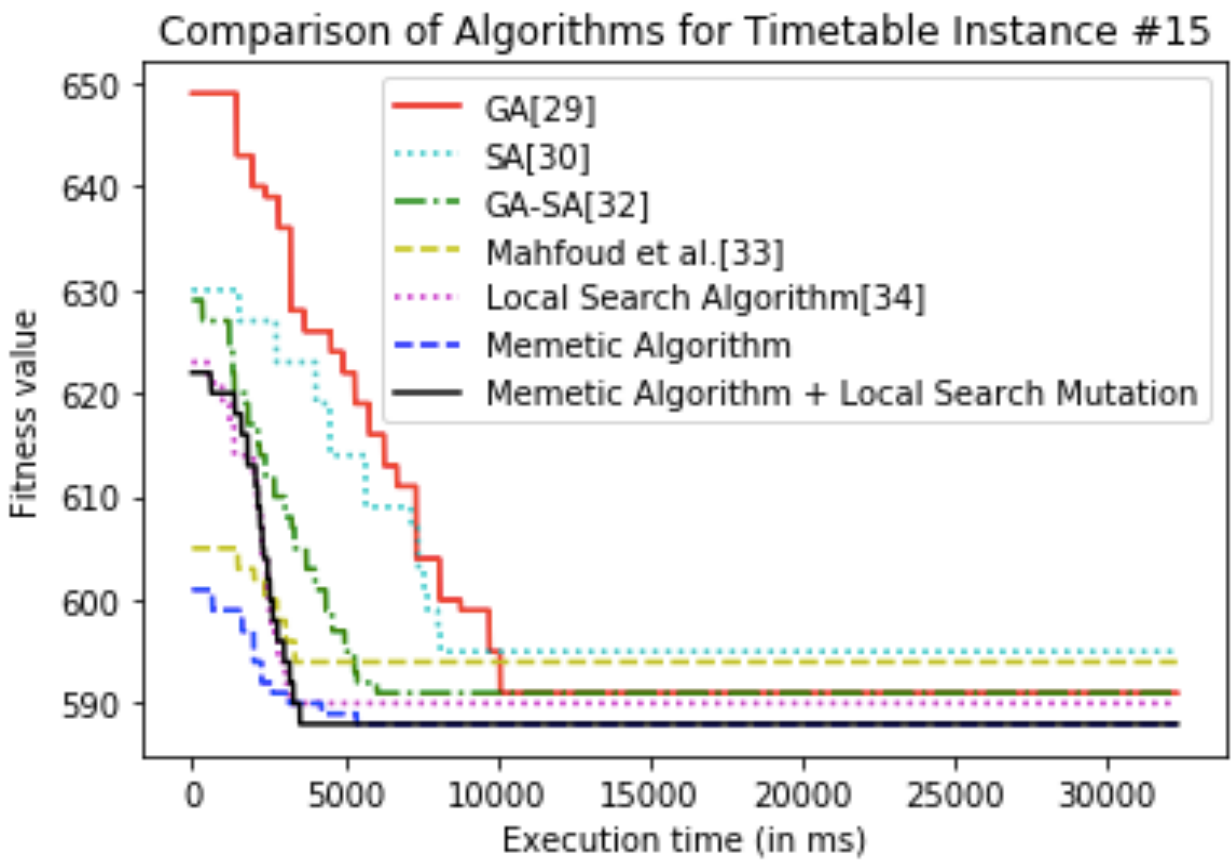


Fig. 31: Comparisons of all the algorithms for Instance 15

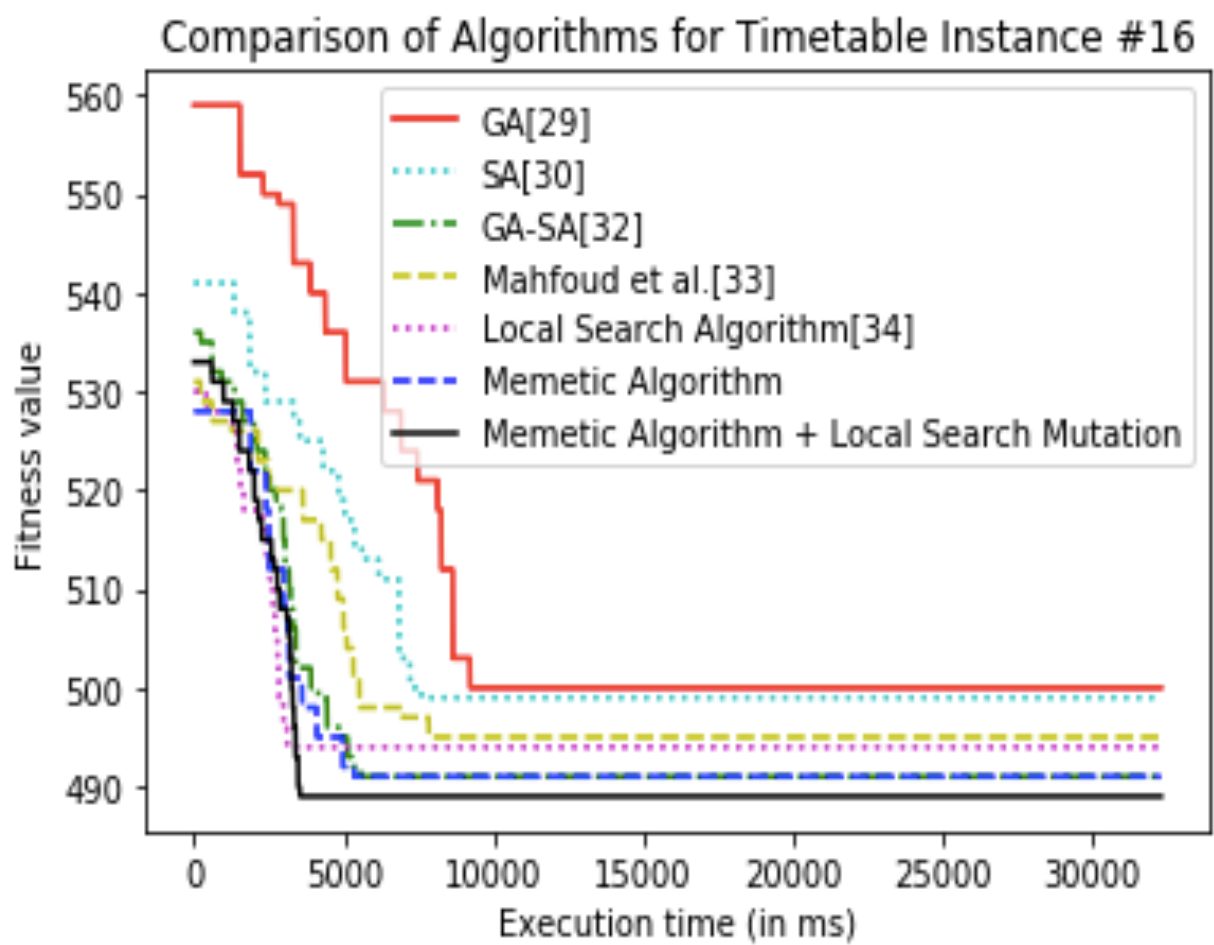


Fig. 32: Comparisons of all the algorithms for Instance 16

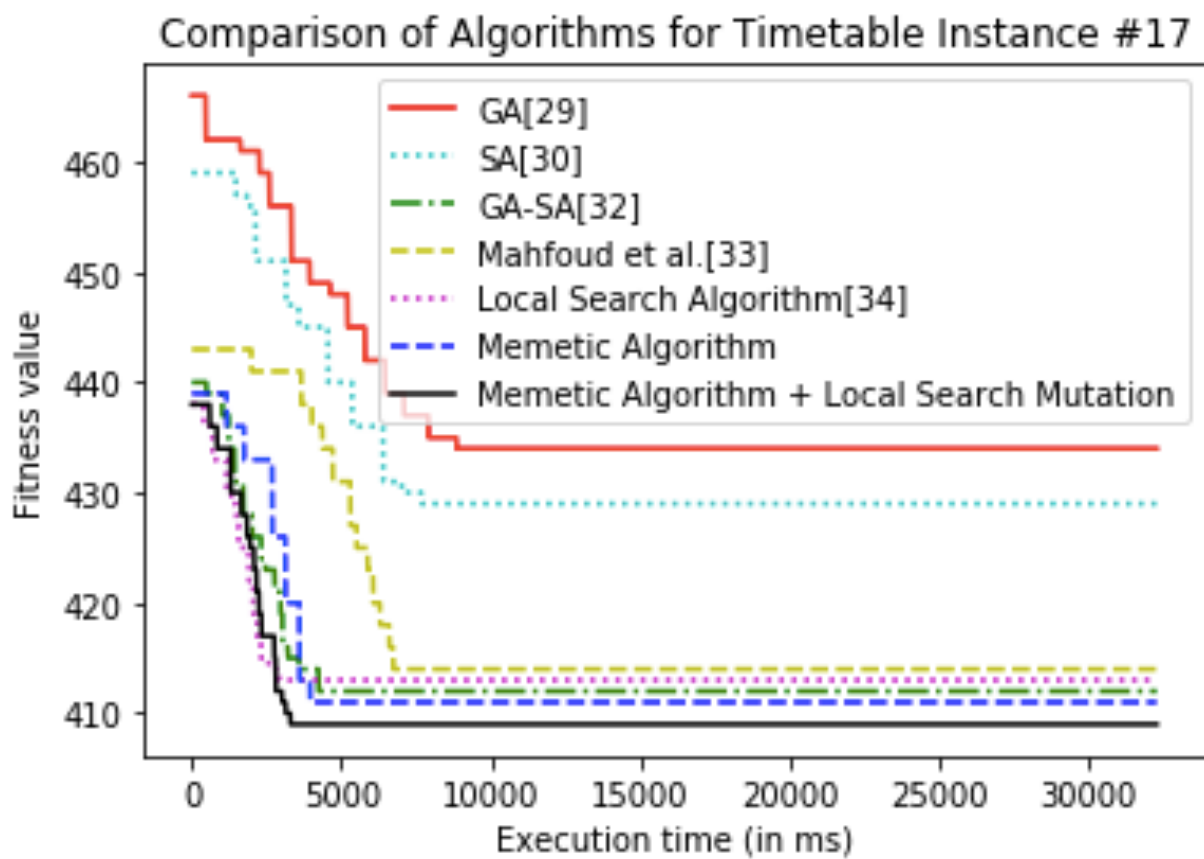


Fig. 33: Comparisons of all the algorithms for Instance 17

Comparison of Algorithms for Timetable Instance #18

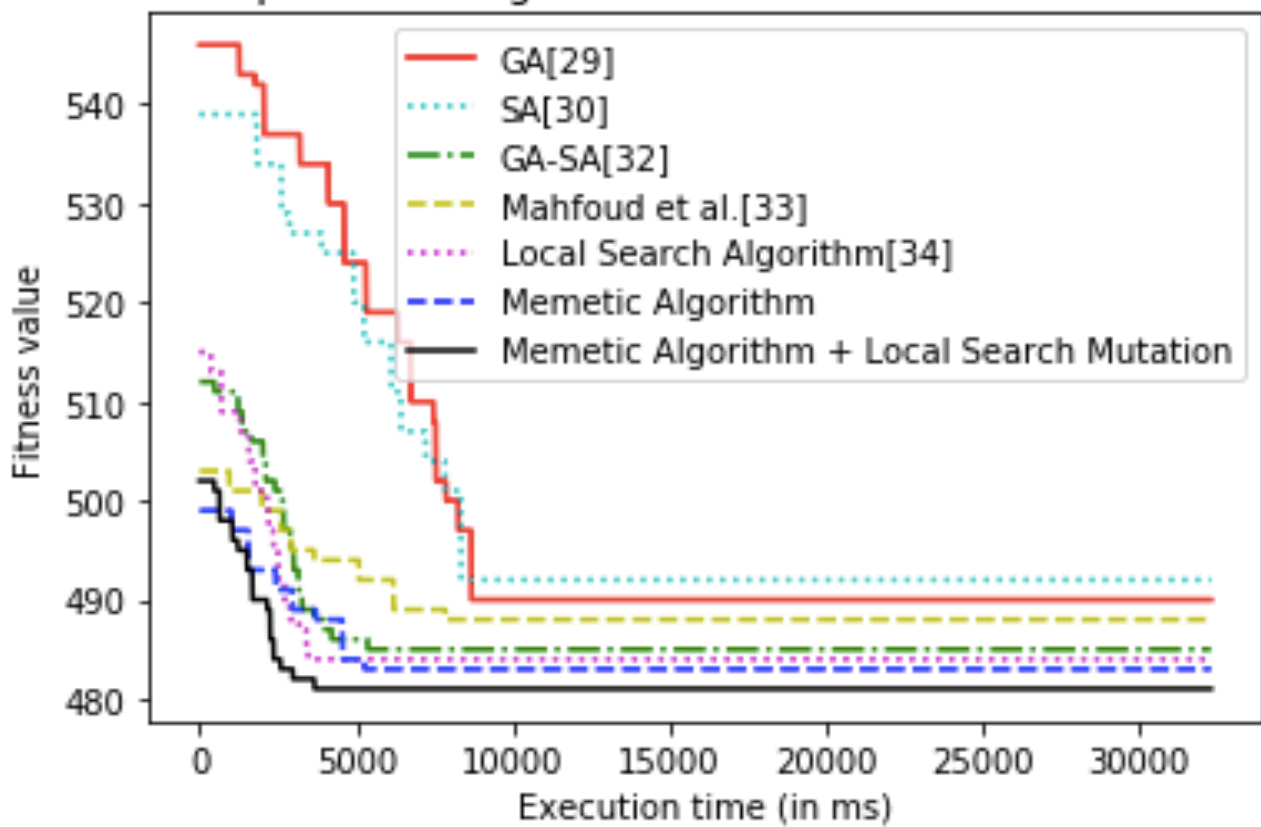


Fig. 34: Comparisons of all the algorithms for Instance 18

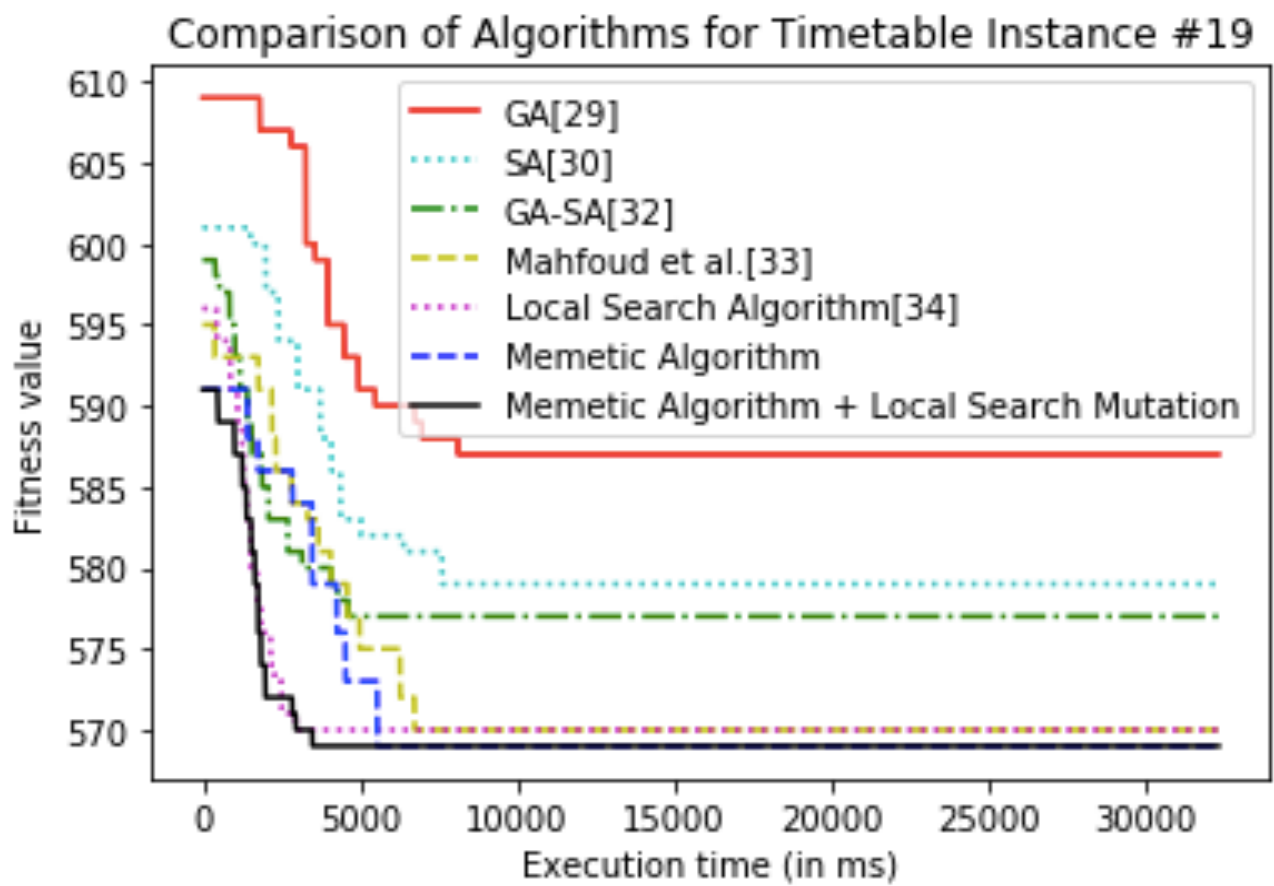


Fig. 35: Comparisons of all the algorithms for Instance 19

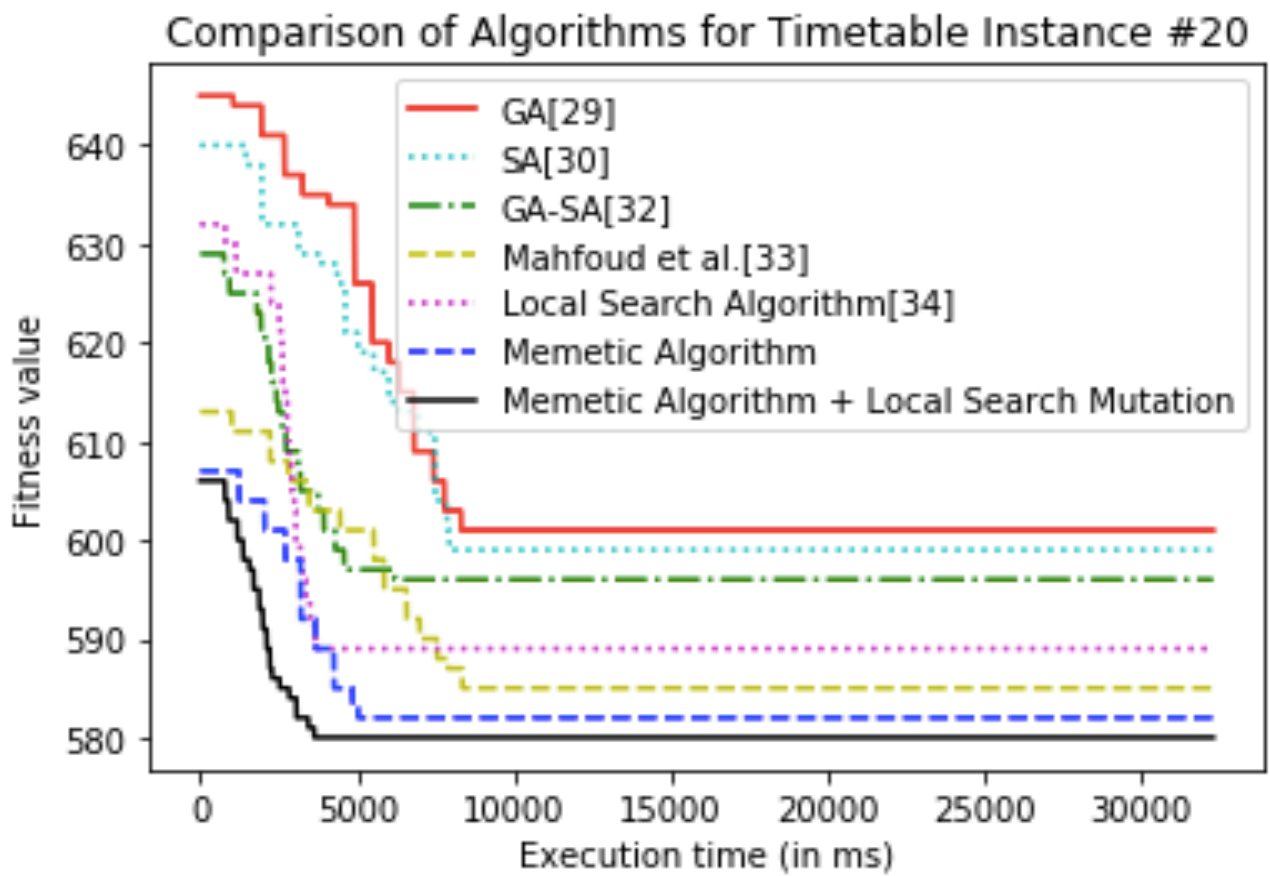


Fig. 36: Comparisons of all the algorithms for Instance 20

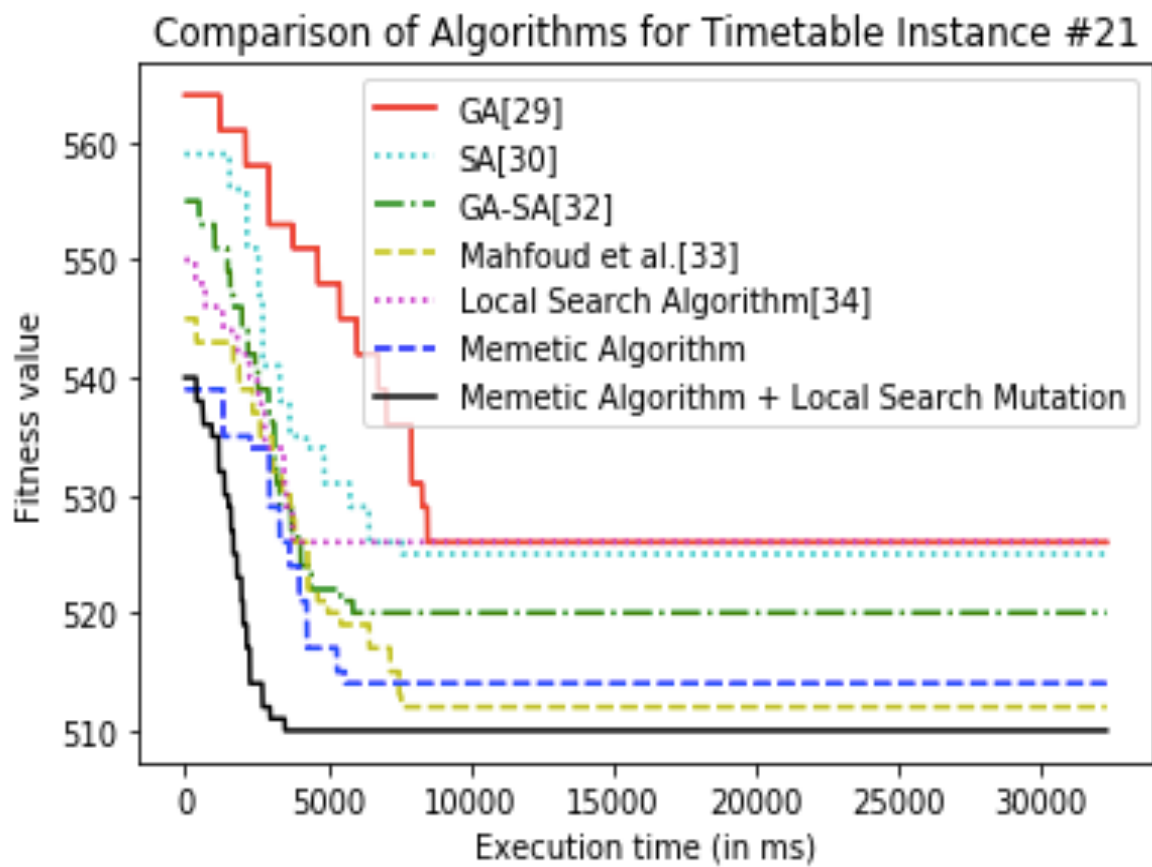


Fig. 37: Comparisons of all the algorithms for Instance 21

4.4 GUI FOR TIMETABLE SCHEDULING

- **Start Page**

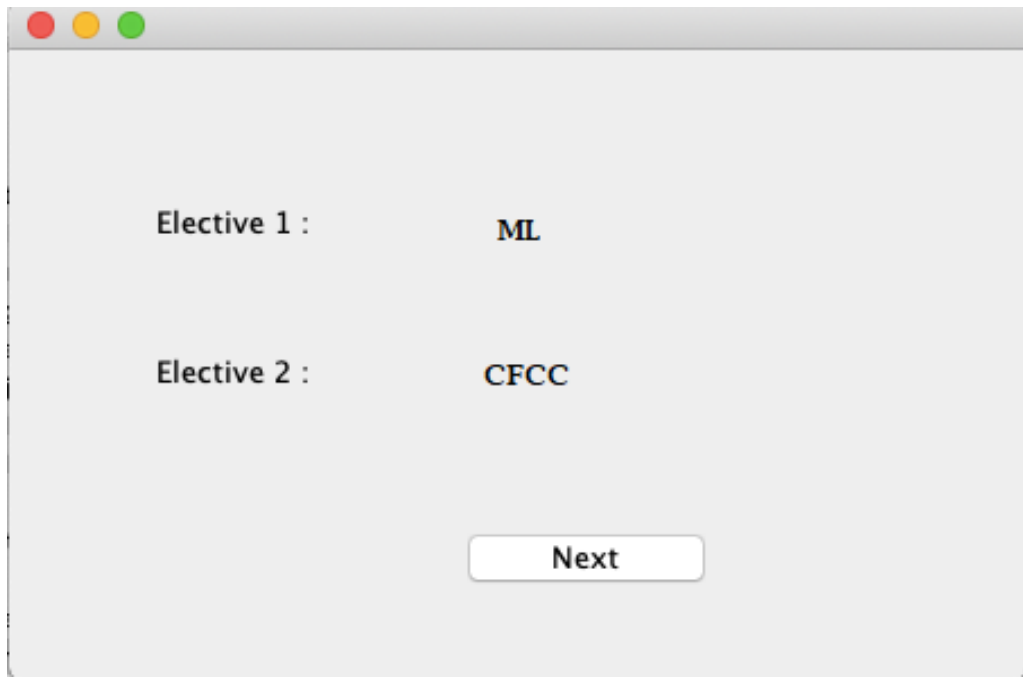
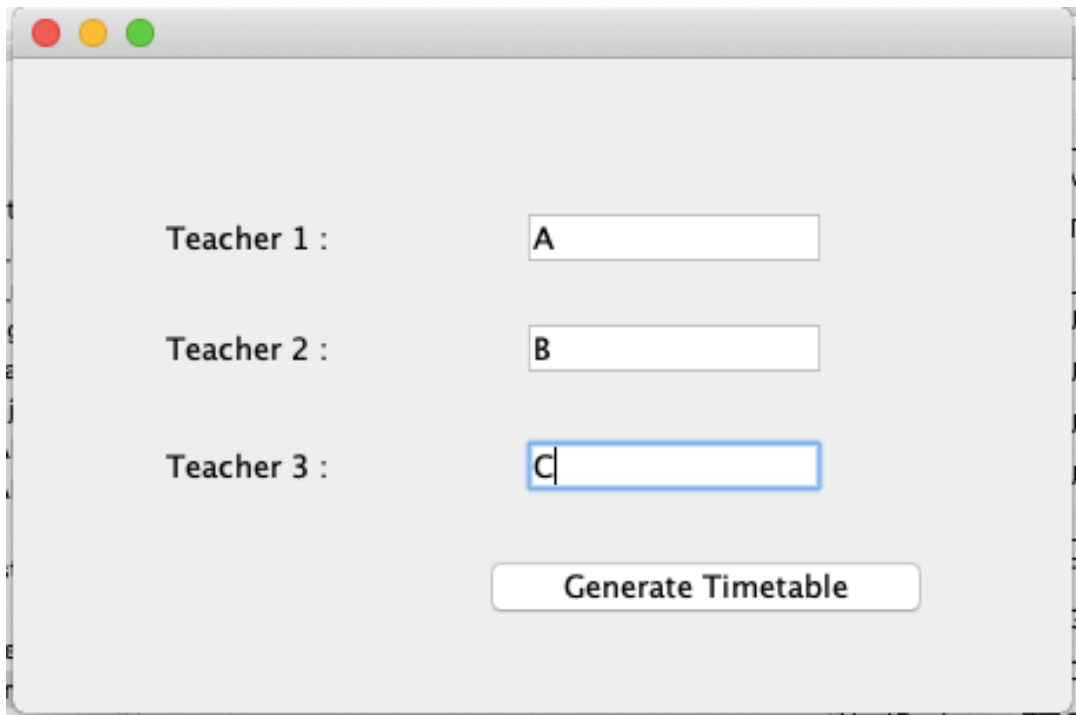


Fig. 38: Electives page

This page gives us the first page the 2 electives are displayed which are chosen after association rule mining

- **Teacher selection page**



Teacher 1 : A

Teacher 2 : B

Teacher 3 : C

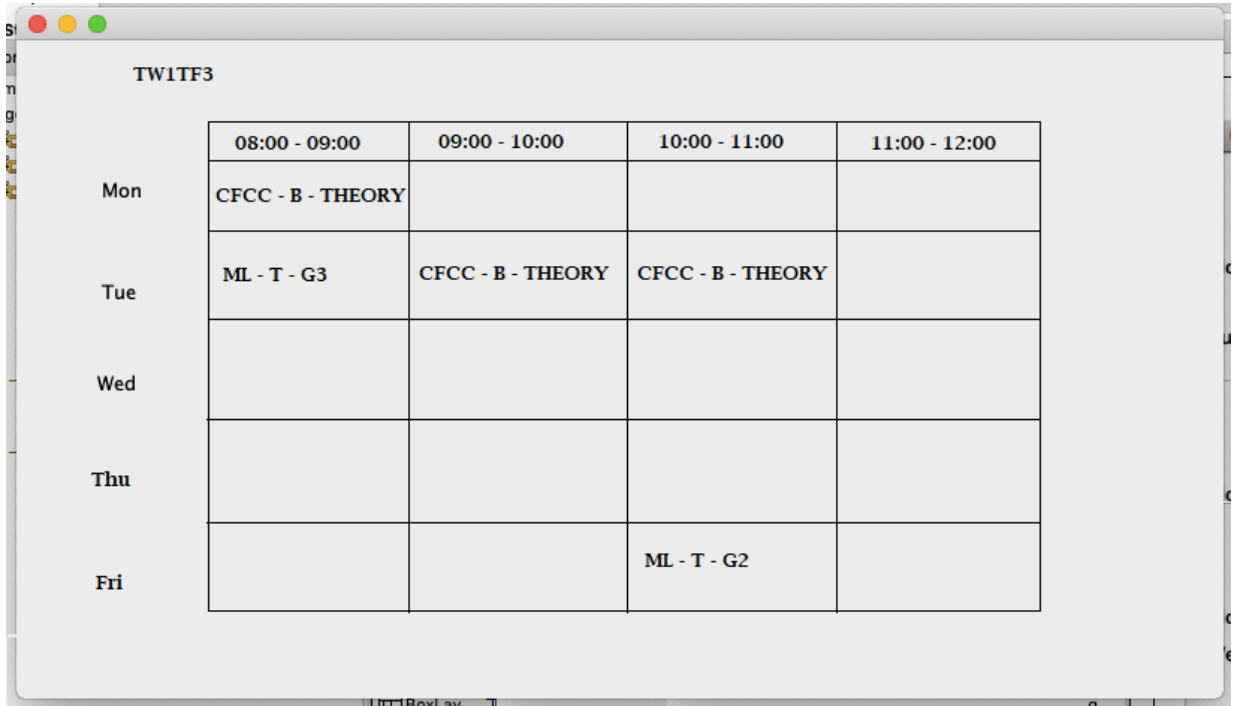
Generate Timetable

Fig. 39: Teacher selection page

This page is where we enter the names of 3 teachers who will be teaching the 2 electives. We click on “Generate Timetable” button to generate the timetable

- **Timetable generation with GA**

This gives us the final timetable schedule for the chosen electives using Genetic Algorithm



| | 08:00 - 09:00 | 09:00 - 10:00 | 10:00 - 11:00 | 11:00 - 12:00 |
|-----|-------------------|-------------------|-------------------|---------------|
| Mon | CFCC - B - THEORY | | | |
| Tue | ML - T - G3 | CFCC - B - THEORY | CFCC - B - THEORY | |
| Wed | | | | |
| Thu | | | | |
| Fri | | | ML - T - G2 | |

Fig. 40: Timetable for TW1TF3 using GA

| | 08:00 - 09:00 | 09:00 - 10:00 | 10:00 - 11:00 | 11:00 - 12:00 |
|-----|---------------|-------------------|-------------------|-------------------|
| Mon | | CFCC - A - THEORY | CFCC - A - THEORY | |
| Tue | | | | ML - T - G4 |
| Wed | ML - T - G1 | | CFCC - A - THEORY | |
| Thu | | | | |
| Fri | ML THEORY | CFCC - A - T - G1 | | CFCC - B - T - G1 |

Fig. 41: Timetable for TW2GF2 using GA

| | 08:00 - 09:00 | 09:00 - 10:00 | 10:00 - 11:00 | 11:00 - 12:00 |
|-----|---------------|---------------|---------------|-------------------|
| Mon | | | | |
| Tue | | | | |
| Wed | | | | CFCC - A - T - G2 |
| Thu | ML THEORY | ML THEORY | | CFCC - B - T - G2 |
| Fri | | | | |

Fig. 42: Timetable for TW3TF3 using GA

- **Timetable generation with SA**

This gives us the final timetable schedule for the chosen electives using Simulated Annealing.

| | 08:00 - 09:00 | 09:00 - 10:00 | 10:00 - 11:00 | 11:00 - 12:00 |
|-----|-------------------|-------------------|-------------------|---------------|
| Mon | | | CFCC - B - THEORY | |
| Tue | | | CFCC - B - THEORY | |
| Wed | CFCC - A - THEORY | CFCC - A - THEORY | | ML THEORY |
| Thu | CFCC - A - THEORY | | | |
| Fri | | | | ML THEORY |

Fig. 43: Timetable for TW1TF3 using SA

| | 08:00 - 09:00 | 09:00 - 10:00 | 10:00 - 11:00 | 11:00 - 12:00 |
|-----|-------------------|-------------------|---------------|---------------|
| Mon | CFCC - B - T - G1 | CFCC - A - T - G1 | | |
| Tue | CFCC - A - T - G2 | | | |
| Wed | | | | |
| Thu | | ML - T - G4 | | ML THEORY |
| Fri | CFCC - B - T - G2 | | | |

Fig. 44: Timetable for TW2GF2 using SA

| | 08:00 - 09:00 | 09:00 - 10:00 | 10:00 - 11:00 | 11:00 - 12:00 |
|-----|---------------|---------------|-------------------|---------------|
| Mon | | | | ML - T - G2 |
| Tue | | | | ML - T - G3 |
| Wed | | | | |
| Thu | | | | |
| Fri | | ML - T - G1 | CFCC - B - THEORY | |

Fig. 45: Timetable for TW3TF3 using SA

CONCLUSION

In this work, we have compared the algorithms Simulated annealing and genetic algorithm for timetable optimization. The generated timetables generate optimal solutions and the data association rule mining help us determine the electives as per students preferences, The optimization algorithms are compared on the basis of their makespan time and generation/iteration number. SA is found to have better efficiency than GA with respect to both the computational time as well as timetable generation.

We have also investigated genetic algorithm, simulated annealing and particle swarm optimization algorithms for the task of automatically creating an entire timetable (electives+ core courses+ lab) for third year undergraduate students for our University. We have proposed a novel set of hard constraints and soft constraints suited to the task that incorporate teacher's preferences for slots and minimize student movement between classes. The three optimization algorithms are compared on the basis of Execution Time of each generation/iteration with respect to generation/iteration number which decreases most in Simulated Annealing. We also compared the algorithms on the basis of No of penalties vs Generations/Iterations. The penalties can be ranked as $SA < PSO < GA$. Simulated Annealing has least amount of penalties and obtains a solution in the least number of iterations.

Finally, we have proposed a novel memetic algorithm which uses the goodness of both GA and SA and introduces the concept of greedy stochastic local search mutation that finds optimal solutions with faster convergence. In the memetic algorithm SA helps in providing the stopping criteria for MA which helps in converging of the solution, GA helps in providing optimal solution and local search mutation in GA helps in obtaining optimal solution in lesser time and better convergence. ITC 2007 timetabling dataset is used for performing comparisons between all the methods and gives us that performance of Memetic Algorithm along with Local Search Mutation is better in terms of faster converging and optimal solution. The approach has good potential to yield optimum results when applied to other real-world optimization problems, and that forms the future scope of our work.

REFERENCES

- [1] Dandashi, Amal, and Mayez Al-Mouhamed. "Graph coloring for class scheduling." In *ACS/IEEE International Conference on Computer Systems and Applications-AICCSA 2010*, pp. 1-4. IEEE, 2010.
- [2] Daskalaki, Sophia, Theodore Birbas, and Efthymios Housos. "An integer programming formulation for a case study in university timetabling." *European Journal of Operational Research* 153, no. 1 (2004): 117-135.
- [3] Zhang, Lixi, and SimKim Lau. "Constructing university timetable using constraint satisfaction programming approach." In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, vol. 2, pp. 55-60. IEEE, 2005.
- [4] Holland, John Henry. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.
- [5] Yu, Enzhe, and Ki-Seok Sung. "A genetic algorithm for a university weekly courses timetabling problem." *International transactions in operational research* 9, no. 6 (2002): 703-717.
- [6] Rozaimée, Azilawati, Adibah Nabihah Shafee, Nurul Anissa Abdul Hadi, and Mohamad Afendee Mohamed. "A Framework for University's Final Exam Timetable Allocation Using Genetic Algorithm." *World Applied Sciences Journal* 35, no. 7 (2017): 1210-1215.
- [7] Duong, Tuan-Anh, and Kim-Hoa Lam. "Combining Constraint Programming and Simulated Annealing on University Exam Timetabling." In *RIVF*, pp. 205-210. 2004.
- [8] Thompson, Jonathan, and Kathryn A. Dowsland. "General cooling schedules for a simulated annealing based timetabling system." In *International Conference on the Practice and Theory of Automated Timetabling*, pp. 345-363. Springer, Berlin, Heidelberg, 1995.

- [9] Zheng, Shuang, Long Wang, Yueyue Liu, and Rui Zhang. "A simulated annealing algorithm for university course timetabling considering travelling distances." *International Journal of Computing Science and Mathematics* 6, no. 2 (2015): 139-151.
- [10] Brusco, Michael J., and Larry W. Jacobs. "A simulated annealing approach to the cyclic staff-scheduling problem." *Naval Research Logistics (NRL)* 40, no. 1 (1993): 69-84.
- [11] Sastry, Kumara, David E. Goldberg, and Graham Kendall. "Genetic algorithms." In *Search methodologies*, pp. 93-117. Springer, Boston, MA, 2014
- [12] Chu, Shu-Chuan, Yi-Tin Chen, and Jiun-Huei Ho. "Timetable scheduling using particle swarm optimization." In *Innovative Computing, Information and Control, 2006. ICICIC'06. First International Conference on*, vol. 3, pp. 324-327. IEEE, 2006
- [13] Adrianto, Dennise. "Comparison Using Particle Swarm Optimization and Genetic Algorithm for Timetable Scheduling." *Journal of Computer Science* 10, no. 2 (2014): 341
- [14] Wang, Yao-Te, Yu-Hsin Cheng, Ting-Cheng Chang, and S. M. Jen. "On the application of data mining technique and genetic algorithm to an automatic course scheduling system." In *2008 IEEE Conference on Cybernetics and Intelligent Systems*, pp. 400-405. IEEE, 2008.
- [15] Chu, Shu-Chuan, Yi-Tin Chen, and Jiun-Huei Ho. "Timetable scheduling using particle swarm optimization." In *Innovative Computing, Information and Control, 2006. ICICIC'06. First International Conference on*, vol. 3, pp. 324-327. IEEE, 2006
- [16] Kohshori, Meysam Shahvali, and Mehrnaz Shirani Liri. "Multi Population Hybrid Genetic Algorithms for University Course Timetabling." *Annals of the University Dunarea de Jos of Galati: Fascicle: I, Economics & Applied Informatics* 18, no. 2 (2012).
- [17] Rachmawati, Lily, and Dipti Srinivasan. "A hybrid fuzzy evolutionary algorithm for a multi-objective resource allocation problem." In *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, pp. 6-pp. IEEE, 2005.
- [18] Kohshori, Meysam Shahvali, Mohammad Saniee Abadeh, and Hedieh Sajedi. "A fuzzy genetic algorithm with local search for university course timetabling."

- In *The 3rd International Conference on Data Mining and Intelligent Information Technology Applications*, pp. 250-254. IEEE, 2011.
- [19] Moscato, Pablo. "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms." *Caltech concurrent computation program, C3P Report 826* (1989): 1989.
- [20] Moscato, Pablo, and Carlos Cotta. "A gentle introduction to memetic algorithms." In *Handbook of metaheuristics*, pp. 105-144. Springer, Boston, MA, 2003.
- [21] Cotta-Porras, Carlos. "A study of hybridisation techniques and their application to the design of evolutionary algorithms." *AI Communications* 11, no. 3, 4 (1998): 223-224.
- [22] Grosan, Crina, and Ajith Abraham. "Hybrid evolutionary algorithms: methodologies, architectures, and reviews." In *Hybrid evolutionary algorithms*, pp. 1-17. Springer, Berlin, Heidelberg, 2007.
- [23] Zeb, Alam, Mushtaq Khan, Nawar Khan, Adnan Tariq, Liaqat Ali, Farooque Azam, and Syed Husain Imran Jaffery. "Hybridization of simulated annealing with genetic algorithm for cell formation problem." *The International Journal of Advanced Manufacturing Technology* 86, no. 5-8 (2016): 2243-2254.
- [24] Mafarja, Majdi M., and Seyedali Mirjalili. "Hybrid Whale Optimization Algorithm with simulated annealing for feature selection." *Neurocomputing* 260 (2017): 302-312.
- [25] Assad, Assif, and Kusum Deep. "Harmony search based memetic algorithms for solving sudoku." *International Journal of System Assurance Engineering and Management* 9, no. 4 (2018): 741-754.
- [26] Singh, Gurmukh, Munish Rattan, Sandeep Singh Gill, and Nitin Mittal. "Hybridization of water wave optimization and sequential quadratic programming for cognitive radio system." *Soft Computing* (2018): 1-21.
- [27] Yu, Enzhe, and Ki-Seok Sung. "A genetic algorithm for a university weekly courses timetabling problem." *International transactions in operational research* 9, no. 6 (2002): 703-717.
- [28] Rozaimée, Azilawati, Adibah Nabihah Shafee, Nurul Anissa Abdul Hadi, and Mohamad Afendee Mohamed. "A Framework for University's Final Exam Timetable Allocation Using Genetic Algorithm." *World Applied Sciences Journal* 35, no. 7 (2017): 1210-1215.

- [29] Duong, Tuan-Anh, and Kim-Hoa Lam. "Combining Constraint Programming and Simulated Annealing on University Exam Timetabling." In *RIVF*, pp. 205-210. 2004.
- [30] Thompson, Jonathan, and Kathryn A. Dowsland. "General cooling schedules for a simulated annealing based timetabling system." In *International Conference on the Practice and Theory of Automated Timetabling*, pp. 345-363. Springer, Berlin, Heidelberg, 1995.
- [31] Zheng, Shuang, Long Wang, Yueyue Liu, and Rui Zhang. "A simulated annealing algorithm for university course timetabling considering travelling distances." *International Journal of Computing Science and Mathematics* 6, no. 2 (2015): 139-151.
- [32] Bettemir, Önder Halis, and Rifat Sonmez. "Hybrid genetic algorithm with simulated annealing for resource-constrained project scheduling." *Journal of Management in Engineering* 31, no. 5 (2014): 04014082.
- [33] Mahfoud, Samir W., and David E. Goldberg. "Parallel recombinative simulated annealing: a genetic algorithm." *Parallel computing* 21, no. 1 (1995): 1-28
- [34] Abdullah, Salwani, and Hamza Turabieh. "Generating university course timetable using genetic algorithms and local search." In *2008 Third International Conference on Convergence and Hybrid Information Technology*, vol. 1, pp. 254-260. IEEE, 2008.
- [35] Ombuki, Beatrice M., and Mario Ventresca. "Local search genetic algorithms for the job shop scheduling problem." *Applied Intelligence* 21, no. 1 (2004): 99-109.

LIST OF PUBLICATIONS

- [1] Susan, Seba, and Aparna Bhutani. "Data Mining with Association Rules for Scheduling Open Elective Courses Using Optimization Algorithms." In *International Conference on Intelligent Systems Design and Applications*, pp. 770-778. Springer, Cham, 2018 [06-08 December 2018, Vellore, India]
- [2] Susan, Seba, and Aparna Bhutani. "A Novel Memetic Algorithm incorporating Greedy Stochastic Local Search Mutation for Course Scheduling." In *International Conference on Computational Science and Engineering*, IEEE, 2019[01-03 August 2019, New York, USA]
- [3] Susan, Seba, and Aparna Bhutani. "Incorporating Teacher's Preferences and Student Time Management in University Course Timetabling". In *International Journal of Computer Information Systems and Industrial Management Applications*, 2019 (Scopus Indexed, Article in Press)

Project page: <https://github.com/users/aparna-bhutani/projects/1>