Solving University Course Timetabling Problem Using Parallel Genetic Algorithm

Amin Rezaeipanah¹, Zahra Abshirini², Milad Boshkani Zade³

¹Dept. of Computer, Faculty of Computer, University of Rahjuyan Danesh Borazjan, Bushehr, Iran
²Dept. of Applied Chemistry, Faculty of Science, Payame Noor University, Bushehr, Iran
³Dept. of Computer Engineering, Ahram Branch, Islamic Azad University, Ahram, Iran

*Corresponding Author: amin.rezaeipanah@gmail.com, Tel.: +98-9351350563
Available online at: www.isroset.org

Abstract—Scheduling is one of the problems which so many researches have been conducted on it over the years. The university course timetabling problem (UCTP) which is an NP-Hard problem is a type of scheduling problem. The allocation of whole of events in timeslots and rooms performs by the university course timetabling process considering the list of hard and soft constraints presented in one semester, so that no conflict is created in such allocations. In general, it means assigning predefined courses to certain rooms and timeslots under specific constraints. In this paper we establish a hybrid algorithm based on Parallel Genetic Algorithm and Local Search to solve course timetabling problem (PGALS). This combines a direct representation of the timetable with heuristic crossover operators to ensure that the most fundamental constraints are never violated. We see how the algorithm is guaranteed to always produce a feasible solution by hard coding constraints which must not be broken. The proposed algorithm has been applied and evaluated against the latest methodologies in the literature with respect to standard benchmark problems. We demonstrate that the proposed algorithm produces some of the best-known results when tested on BenPaechter competition datasets.

Keywords—Parallel Genetic Algorithm; Local Search; Timetabling Problem; Hybrid Algorithm; BenPaechter Dataset.

I. INTRODUCTION

The process of planning and timetabling courses in universities is a multi-stage process requiring appropriate coordination and communication among several planning groups. These groups include university professors, faculty group heads, educational department manager and planners [1]. The University Course Timetabling Problem (UCTP) is one of the controversial issues presented in various applications. The UCTP goal is to create a timetable for a classroom program, so that there is no interference in the classroom and student timing program. The final program should also be optimal, which means that the best classes and best hours of courses are available. The best choices depend on the parameters specified for the problem [2]. This problem lies in the class of NP-Hard problems in terms of complexity and to find an optimal solution evolutionary algorithms such as the genetic algorithm are very effective.

In a timing issue, there are many constraints. Some of these constraints must necessarily be respected, which are called hard constraints. Observing some of the other constraints will also improve the program. These constraints are soft constraints. However, failure to comply with these constraints does not make the answer to the problem unacceptable, but it is better to follow them. The compliance with these constraints leads to approaching the optimal response. Most systems designed in this field use classical algorithms for classroom programming. Since classical algorithms provide an acceptable answer rather than an optimal answer, new approaches to timing issues are evolutionary and metaheuristic algorithms.

Over the past 40 years, researchers have been using different time-tableing methods using constraint-based methods, population-based methods (eg, genetic algorithms [3], [4], ant colony optimization [5], and memetic algorithms [6]), the metaheuristic methods (eg, tabu search [7], simulated annealing [8], and great deluge [9]), variable neighbourhood search (VNS), and hybrid and hyper-hybrid approaches, etc., have been proposed. In this paper a new hybrid approach based on parallel genetic algorithm and local search is utilized in solving university course timetabling problem. Because of simultaneous studying of several answers and random survey of case space which is perfect for the
scheduling problem. The preferred method in this study is the genetic algorithm.

The remainder of the article is as follows: In section II, it is discussed about the university course timetabling problem, whereas Section III briefly reviews preliminaries the related works. Section IV describes the proposed approach. Section V describes the results and discussions. Finally, in Section VI, the conclusion suggests possible future work is presented.

II. UNIVERSITY COURSE TIMETABLING PROBLEM

BenPaechter's dataset, one of the most famous collections of university course timetabling, has been used to conduct experiments in this research. This dataset was created in 2005 by Ben Paechter and colleagues and was available at http://code.ub.ac.be. This dataset consists of 12 samples in three small, medium and large sizes. Characteristics of each category are given in Table 1 [10]. This problem includes scheduling 100-400 courses into a timetable with 45 timeslots similar to 5 days of 9 hours as well as satisfying room features and room capacity constraints.

The dataset in Table 1 contains four columns, in which the first column is a list of department events and resource classifications, and the other three columns represent the number of events and resources in small, medium and large sizes. The row number of this dataset is related to each event and the resources available in the department are in ascending order from small to medium to large.

<table>
<thead>
<tr>
<th>Features</th>
<th>Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Small</td>
</tr>
<tr>
<td>Number of events</td>
<td>100</td>
</tr>
<tr>
<td>Number of rooms</td>
<td>5</td>
</tr>
<tr>
<td>Number of room features</td>
<td>5</td>
</tr>
<tr>
<td>Approx. features per room</td>
<td>3</td>
</tr>
<tr>
<td>Percent feature use</td>
<td>70</td>
</tr>
<tr>
<td>Number of student</td>
<td>80</td>
</tr>
<tr>
<td>Max. events per student</td>
<td>20</td>
</tr>
<tr>
<td>Max. student per events</td>
<td>20</td>
</tr>
</tbody>
</table>

Constraints on the UCTP issue are divided into hard and soft constraints. Hard constraints, in contrast to soft constraints, should not be violated in any way. There is no obligation to satisfy soft constraints, but the quality of the timetable increases with increasing observing soft limits. Below is a list of hard and soft constraints taken from the literature of Al-Betar et al. [11].

Hard Constraints:
- H1: A student should not have a single course on a day.
- H2: A student should not have more than two consecutive courses.
- H3: A student should not have a course scheduled in the last time slot of the day.

Soft Constraints:
- S1: A student should not have a single course on a day.
- S2: A student should not have more than two consecutive courses.
- S3: A student should not have a course scheduled in the last time slot of the day.

III. RELATED WORK

In recent years, researchers turn their focus on meta-heuristic, hyper-heuristic and hybridization methods. Several examples on these methods including hill climbing, simulated annealing, great deluge, genetic algorithm, particle swarm optimization, and harmony search. In this paper we are using a multi-stage approach. Although different to hybridized approaches, there are good reasons for using multiple algorithms within an overarching approach. Hybridization has led to good quality results in previous research [12]. More recent work has validated these earlier findings. For example, hybridized mixed integer linear programming, a greedy heuristic, two local search strategies and three meta-heuristics for a vehicle routing problem, reporting positive results [13]. The GA was used as an exploration operator, SA was used to intensify the search. The algorithm was evaluated on 35 cell formulation benchmark instances, producing 24 best results, and two new best results.

Many variants of the high school timetabling problem and their attempts to solve them can be found in respective references [14]. Their fundamental differences are attributed to the uniqueness of each country’s educational system. For example, Moura and Scaraficci solve the high school timetabling problem for three Brazilian high schools using a basic Greedy Randomized Adaptive Search Procedure (GRASP) heuristic followed by a path-relinking [15]. Another hybrid technique which combined two population-based algorithms for real-world course timetabling (i.e. bee colony with hill climbing optimizer) by Bolaji et al. [16]. Daskalaki et al. compared the performance of Genetic Algorithms (GAs) as opposed to Genetic Programming (GP) in solving a set of hard high school timetabling problems [17]. Da Fonseca et al. presented a local search approach to the high school timetabling problem [18]. Ahmed et al. evaluated the performance of a range of selection hyper-heuristics combining different reusable components for high school timetabling [19]. Al-Yakoob and Sherali investigated two decomposition approaches to solve a high school timetabling problem in a case study related to Kuwait’s public educational
system [20]. Lewis and Thompson achieved 100% feasibility on all instances of ITC2007 by using constructive heuristics and followed by their PARTIAL-COL algorithm which uses a tabu mechanism for the remaining unassigned events [21].

Hyper-heuristics have emerged as such general purpose, high-level search methodologies which are motivated by the goal of selecting or generating heuristics automatically to solve a wide range of difficult optimisation problems [22]. This work focuses on the selection type of hyper-heuristics.

Most studies conducted in this area are single-objective in which minimizing violation of soft constraints is considered as objective function, and few studies have been focused on optimizing multi-objective models and multi objective optimization approaches. Datta, Deb [23] presented a bi-objective model with the aim of minimizing average number of free time slots between two classes of students and number of consecutive classes for lecturers. Then, they developed a NSGA-II algorithm for solving the model. In another study, Abdullah, Turabieh [24], presented a two-objective model to minimize the number of time slots during which a student waits between two classes as well as the deviations of soft limitations.

Timetabling problems as well as scheduling problems, define a class of hard-to-solve constrained optimization problems of combinatorial nature. In order to solve the model, they developed a NSGA-II algorithm. Thepphakorn, Pongcharoen [25] developed a new multi-objective model by the aim of minimizing the operational costs and the number of inadequate chairs in a course time tabling problem in a university in Thailand. Then, to solve it, they used a multi objective cuckoo search algorithm.

**IV. METHODOLOGY**

In this paper a new approach is provided for the university course timetabling problem. In our proposed PGALS, timetables are created from hybridization of genetic algorithm and local search. The details of each operator and section will be discussed in the following sections. Because of simultaneous studying of several answers and random survey of case space which is perfect for the scheduling problem, The preferred method in this study is the genetic algorithm. After applying the standard genetic algorithm On different samples and answers, It was concluded that this method needs to improve due to the random search on the entire case. As a result, a new approach was created to solve the problem by making some reforms in the algorithm, such as restorative procedures on children, Intelligent operator and the use of parallel structure. The pseudo code for the PGALS algorithm as depicted in Figure 1. The following steps explain the proposed algorithm to solve this problem.

![Pseudo-code of PGALS algorithm](image-url)
A. Genetic Algorithm

Chromosomes length in the university courses scheduling problem is related to the number of events. On the other hand, each event includes two genes characterized by timeslot (time event) and room numbers. Room numbers can be determined according to the number of input sample rooms. According to the BenPaechter standard, class timeslot have values between 1 to 45 (9 hours per day) in 5 days. A sample of proposed chromosomes is shown in Table 2.

In this study we will select the chromosomes in the form of random from search space. The evaluation criteria in the issue of scheduling the university courses are violation of issue constraints.

Table 2. Structure of proposed chromosome (for example)

<table>
<thead>
<tr>
<th>#Events</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Timeslot</td>
<td>-1</td>
<td>34</td>
<td>-1</td>
<td>44</td>
<td>3</td>
</tr>
<tr>
<td>#Room</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

Fitness function measures the value of their violations by studying the limitations of hard and soft issue and according to which determines the suitability of chromosomes. A weight will be determined for each of the limitations which expresses the level of importance for each one. In this study, a criteria such as distance to feasibility is used in order to calculate the quality loss on schedule. Therefore, the required number of students are detected to participate in any infeasible event, and the total number of the students will give us the distance to feasibility schedule. For example, if a chromosome has three unallocated events and the number of students in each of these events are respectively 12, 8 and 5. So, the distance to feasibility is (5+8+12) 25. Eventually, the fitness function is calculated by following equation.

\[
F(X) = \sum_{i=1,n} w^i \times P^{Soft} + \sum_{j=1,m} w^j \times P^{Hard} + w^d \sum_{k=ENS} P^{DKS} \tag{1}
\]

In this equation, \( w^i \) the weight (penalty) can be any constraints (soft and hard). Weight constraints will be determined according to its importance in the timetable. \( P^{Soft} \) is the number of defects in soft constraints, \( P^{Hard} \) is the number of defects in hard constraints, \( P^{DKS} \) is the distance to the feasibility of solution, \( ENS \) (Events Not Specified) is the events that are not allocated to the timetable. Note that the proposed method prevents the creation of events with hard constraints and it considers them as the events which are not allocated. So in the above equation, the number of hard constraints is always zero. In recent years, how to deal with chromosomes that have this type of constraints is one of the main challenges. The aim of this procedure is to repair the chromosomes which have such constraints.

In this study, we select two parents according to the roulette wheel method in order to help convergence of the algorithm and the use of chromosomes of other populations. The first parent is selected from the population of the current genetic algorithm. Furthermore, the second parent is probably selected from the population of other sectors for 50% and is probably selected from the best produced chromosome for 50%.

The pairs which were considered as a parent in the selection part, they exchange their genes In this part and create new members. If the crossover action does not occur on a chromosome pair, the children will be produced by repeating the parents. In this study, uniform crossover operator is designed that each produce one child. In the uniform crossover operator, at first, the child chromosome is generated by ‘0’ and ‘-1’ genes randomly. Then, for every genes with the content of ‘0’ on children chromosome, the equivalent genes are copied from the first parent. In the next step instead of ‘-1’ gene, the genes will be copied from the second parent that do not create hard constraints for child chromosome. Finally, a number of genes may not be assigned any values, these genes have ‘-1’ content and are considered as infeasible events. An example of this operator can see in Figure 2.

Mutation is a random process in which the content of a gene is replaced by another gene in order to produce a genetic structure. This research provides two operators called Local Mutation, Global Mutation. At local mutation operator, one event is randomly selected within the chromosome for infeasible individual events, and the events are swap if fitness improvement is observed.

<table>
<thead>
<tr>
<th>Parent 1</th>
<th>Parent 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3. Initial genes child

<table>
<thead>
<tr>
<th>Initial genes child</th>
<th>Child</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 2. An example of uniform crossover operator

Swap genes are probably 50% of the timeslot and room number, and probably 25% of only timeslot and 25% probably only room number. Infeasible events are those of events that create constraint. Furthermore, in global mutation operator, first a list of all feasible events is determined to join the chromosome. The events can be obtained by sharing the current chromosomes events and all feasible states for the event. In the next stage, an event is randomly selected from the list of possible events for each current and infeasible chromosome event, in a way that does not create hard constraints. The new events are replaced if fitness improvement of the chromosome is observed.
Some of the timetable events in the first stages are not programmable. These events were indicated by ‘-1’ in the genes of a chromosome. The purpose of improvement operator in the proposed algorithm is finding timeslot and room number in the timetable for this type of events so that the proposed solution is infeasible. The operator in fact is designed to improve the timetable and reduce the distance to the feasibility study on child chromosome. The operator randomly allocates timeslot and room number (among all events state) to the events that are contradictory. One of the interesting phenomena in genetic algorithms is that it may be produced chromosomes in median generations which are very suitable and well-being at fitness function quality. The chromosomes may be destroyed. In the result of the combination and mutation operators and will not produced anymore. One way is to identify such cases and also use them in subsequent generations. This technique is called elitism because it has a great influence to answer the question practically. In this study, the best chromosome in each generation is selected among all populations and directly goes to the next generation.

B. Local Search

Local search is introduced to find a solution in order to maximize a criterion among a number of ongoing solutions. These algorithms move from one solution to another one in a space of ongoing solutions (search space) by using limited changes, until a desirable solution is found or a period of time passes. There may be cases in the genes of a chromosome which are less repeated or have not been used at all. For this reason, it requires the operator to enhance the probability of unobserved selection characteristics. The local search, swap randomly some of the genes on unused child chromosome by child chromosome gene. If the quality (fitness function) of timetabling increases, the changes can be done. The termination of the local search algorithm could be based on a timeslot when the best solution has not been optimized by the algorithm in the definite steps.

C. Structure of the Parallel Genetic Algorithm

One of the important features of the genetic algorithm is to be run in a parallel way and also to search for spaces that are very complex or large. In general, it can be argued that The issues where the search space is very large and complex, in local optimum the possibility of entrapment for parallel genetic algorithms is so low. This research tries to create different populations instead of using an initial population and processing parallel genetic algorithms. Chromosome structure is identical in all populations. But the population could have the selection operator, crossover and mutation of its own. Another advantage of this method is the possibility of parallel processes in each population, independent of the others. So, in this study producing the initial population and fitness function calculation is done in a parallel way. After a few generations, we will analyze the chromosomes exchange by elitist algorithm. In the exchange of chromosomes, the worst chromosomes in current population are replaced by the best chromosomes in other populations (based on the fitness function).

Care must be taken in the exchange of chromosomes, such that, the number of chromosomes of each population should be greater than the number of new imported chromosomes. By observing this point, we can prevent the incompatible methods to impact on final result negatively. The main problem of convergence genetic algorithm is the local maximum (minimum) which is resulted from similarity of chromosomes (unable to create new and different generations of the same parents). Mutation step due to a random change has little reliability. But the existence of different chromosomes with high suitability (for creating a new and diverse generation) reduces the possibility of convergence to the local maximum (minimum). In this issue, because of variety of constraints, there is no possibility of designing the high-performance genetic algorithm which always solves the problem. Multi-population genetic algorithm tries to resolve (make it ineffective) this problem.

V. RESULTS AND DISCUSSION

In order to evaluate the effectiveness of the proposed approach in timetabling of courses, a program is designed and implemented in MATLAB R2016a language, and the experiments were conducted on a PC with an Intel Core i3, 2.2 GHz, 8GB RAM, Windows 10 ultimate. In the proposed algorithm, the weights of all the Hard constraints have been considered 0.9. This high weight will prevent the selection of chromosomes with hard constraints. Noting that the present issue is a research and soft constraints does not have any implementation priority, so the weight of the three soft constraints is considered identical and 0.1. Propose input parameters of the suggested algorithm (PGALS) as follows: Population size is 50; Crossover rate is 0.8; Mutation rate is 0.15; Max generation is 75; Number of parallel is 5, Maximum local search iteration is 1000.

For showing the amount of proposed method, the suitability of chromosomes (constraints violated) have been investigated compared to the number of successful repetitions. Figure 3 to 5 the show the convergence of the proposed algorithm on small, medium and large instances respectively. As it is obvious, In the first step, the proposed algorithm quickly reduces the accuracy of the DF (distance to feasibility). This is due to the structure of mutation operator and its strategy that is responsible for dealing with such chromosomes. In the small instances examples such as small in which the amount of DF is at first is zero, the structure of algorithm leads to pressure on the soft constraints and causing a rapid decline in their value at the first steps.
The DF value of convergence charts in small instances is zero. In these instances, due to the small size of the search space and the proposed greedy algorithm, the hard limitations are not created genes and the chromosomes are not "-1" gene at the beginning.

Therefore, operators focus on minimizing soft constraints. But the DF values are not zero in the case of the medium and large because there are large dimensions of problem. In this case, according to the provided operators and algorithms Strategy against DF parameter, the proposed algorithm focuses more on reliability of solution. As clearly shown in Figure 4 and 5, DF value compared to other three parameters declines more quickly. When the parameter is zero, the algorithm focuses on soft constraints of the problem. S2 constraints value is higher than any other constraints. Due to the high number of students and lessons for each student, it seems reasonable to say that the constraint compared to other constraints is in a high value.

Table 3 compares the performance of the proposed algorithm to other similar ones. In this table, the terms B and M, respectively, meaning the best and the average solutions among 30 different performance that show the sum number of constraints violated by algorithms. In these sections $A_i$ (i = 1,2,...,24) are algorithms which have been used in mentioned datasets and presented in detail in Appendix 1.
In the Table 3 at first the performance of algorithms A1, A5, A12, A23, A24 and A24 are evaluated on small (1–5) dataset, where the ranking of algorithms would have better performance in terms of efficiency over this dataset so that here A23 and A24 algorithms have good performance. We could obtain the performance of the above algorithms on Medium (1–5) size dataset where the ranking process of algorithms would have better performance in terms of efficiency over this data A12 so that here the efficiency of the algorithm A24 is the best and A15 and A23 algorithms have good performance. Finally, we could obtain the performance of mentioned algorithms on large (1–2) and very large dataset where the ranking process of algorithms would have better performance in terms of efficiency over these two datasets so that in the large dataset, the efficiency of the algorithm A24 is the best and A12 and A23 algorithms have good performance. However, the worst performance belongs to both A11 and A13 algorithms. In general, we find that the performance of the algorithm A23 and A24 (proposed algorithm) in all Categories is better than other algorithms.

The studies performed on the issue of run-time of the compared hybrid algorithms reveal that the algorithms A15, A17 and A24 have less average run-time compared to other methods. The run-time of A15 and A24 algorithms is 825s and the 780s for the BenPaechter dataset (average for all instances). This comparison is performed only among some algorithms, since many algorithms do not investigate the run-time factor in their studies.

### VI. CONCLUSION AND FUTURE SCOPE

Scheduling is one of the problems so many researches have been worked on it over the years. In this study, a new method was presented for solving timetabling problems of University courses based on genetic algorithm. In the proposed method, hard and soft constraints are used to determine suitable solutions and the criterion of the distance to feasibility of solutions have also been used. Regarding the large instance, our PGALS was not always able to produce a feasible timetable. Our method like some approaches, cannot always produce a feasible timetable for the large instance. Future work includes further analysis of the contribution of individual components (local search and guided search) toward the performance of PGALS. We believe that the performance of the genetic algorithm for the UCTP can be improved by applying advanced genetic operators and heuristics. Future work includes further analysis of the contribution of individual components (simulated annealing and guided search) toward the performance of PGALS.

### APPENDIX I.

Neighborhood Search [38]), A20: (TSH : Tabu-Search Hyperheuristic [39]), A21: (FMH: Fuzzy Multiple Heuristic [30], A22: (GSGA: Guided Search Genetic Algorithm [35]), A23: (HA: Hybrid Algorithm [39]), A24: (PGALS: Parallel Improved Genetic Algorithm and Local Search (The proposed algorithm)).

REFERENCES


Authors Profile

Mr. Amin Rezaeipanah received her B.E. in Computer Science & Engineering from University Ferdows College of Engineering, Mashhad, Iran in 2010 and M.E. in Software Engineering and Artificial intelligence from Shiraz University, Shiraz, Iran in 2013. He is a postdoctoral researcher at Ph.D. He is also a graduate teaching assistant at Rabjuyan Danesh Borazjan University. His main research interests are recommender systems, social network analysis, WSN and large-scale data mining.

Ms. Zahra Abshirini received her B.E. in chemical engineering from Shahid Rajaee Teacher Training University, Tehran, Iran in 2009 and M.E. in chemical engineering from Payame Noor University, Bushehr, Iran in 2013. He is a postdoctoral researcher at Ph.D. He is also a graduate teaching assistant at Education. His main research interests are synthesis pyrazole derivatives, production of chromones and nano catalysts.

Mr. Milad Boshkani Zadeh received received her B.E. in Computer Science & Engineering from Islamic Azad University - Bushehr Branch, Bushehr, Iran in 2008 and M.E. in Information Technology from Islamic Azad University - Ahram Branch, Bushehr, Iran in 2011. His main research interests are data mining, Parallel computing and Simulation Optimization.