Solving Timetabling Problem as Undirected Graph using Genetic Algorithm

Younis R. Elhaddad, and Mohamed A. Alshaari

Abstract— The timetable of any faculty is difficult to create due to having to coordinate many schedules. This is even more challenging task when done manually. In the industry, timetabling problem is commonly referred to as 'NP hard problem', as it is hard to solve using conventional programming. Therefore, different types of heuristics search, such as genetic algorithms, simulated annealing algorithm, etc. are currently used, as they provide solution to this problem in reasonable time. In this work, new genetic algorithm incorporating chromosome encoded and fitness function techniques is used to solve the faculty timetable problem. More specifically, in the proposed approach, the mapping of the timetabling problem is performed in terms of undirected graph.

Keywords- *Genetic algorithm, Timetable, Undirected graph*

I. Introduction

THE timetable of any faculty is difficult to create due to having to coordinate many schedules. This is even more challenging task when done manually. In the industry, timetabling problem is commonly referred to as 'NP hard problem', as it is hard to solve using conventional programming. Therefore, different types of heuristics search, such as genetic algorithms, simulated annealing algorithm, etc. are currently used, as they provide solution to this problem in reasonable time. In this work, new genetic algorithm incorporating chromosome encoded and fitness function techniques is used to solve the faculty timetable problem. More specifically, in the proposed approach, the mapping of the timetabling problem is performed in terms of undirected graph.

II. Input Data

The faculty timetable creation is contingent on unifying and scheduling information pertaining to the lecturers, classrooms, courses, hard constraints, soft constraints, maximum number of time periods per day, etc. This information can be represented in a tabular form, as shown in Table 1-4. In this example, Table 1 contains lecturer data, with each lecturer name associated with a unique identifier.

Younis R. Elhaddad, Faculty of information Technology University of Benghazi, Benghazi, Libya. Email: Younis.haddad@benghazi.edu.ly

Mohamed A. Alshaari, Faculty of information Technology University of Benghazi, Benghazi, Libya. Email: mohamed.shaari@gmail.com

TABLE I
LECTURERS

Lecturer Key Value	Lecturer Name
1	Ali M.
2	Salem R.
.	
.	
..	
.	
30	Waled F.

Table II contains data on all the classrooms that can be booked during the semester and their capacity.

TABLE II
CLASSROOMS

Classroom Key value	Classroom Name	Capacity
1	C1	40
2	Hall2	70
3	C5	30
….		
20	L2	25

Table III contains data pertaining to all the courses that will be taught during the semester, the name of the lecturer and the classroom where the course will be held.

TABLE III
COURSES

Course No.	Course Code & Group	Lecturer	Classrom
1	CS 431	Elhaddad	C5
2	CS 331	Wafa	D3
3	IT 101 A	Maha	C4
4	IT 101 B	Ali	C4
5	IT 202 A	Hietham	C5
….			
165	CS 254	Elsharri	D2

Finally, Table IV contains the keys of selected time-periods of the timetable

TABLE IV
TIME-PERIOD

Key value	Period time
1	8:00 - 9:30
2	9:30 - 11:00
3	11:00 - 12:30
4	12:30 - 2:00
5	2:00 - 3:30
6	3:30 - 5:00
7	5:00 - 6:30

As Table 1-4 provide all the necessary information for creating the faculty timetable, we can use them to generate the main input table, which will have twice as many rows as there are courses taught during the semester, and six columns. The first column is used for the course key values, and because each course will be taught twice during the week, there will be twice as many classes. The second column contains all the lectures to be taught during the semester, taking into consideration that, if any subject is taught to more than one group of students, each group is treated as single course. For example, if the course *IT101* has to be taught to three different groups i.e. Group A, group B and Group C, then *IT101A*, *IT101B,* and *IT101C* will be considered as different courses in Table 3. Moreover, because each subject will be taught in two different time periods during the week, each one will be denoted with different number. The reason for applying this technique stems from the fitness function design, which will be explained later. The third column shows the names of lecturers teaching the courses, with the classroom booked for the lecture given in the fourth column. The fifth column indicates the year to which the course corresponds, and the sixth column is used to indicate the similarity between the subjects, whereby the same subjects indicates by same number.

TABLE V
MAIN INPUTS

Course No.	Course Code & Group	Lecturer	Classroom	Year	Similarity
1	CS 431	Elhaddad	C5	3	1
2	CS 431	Elhaddad	D3	3	1
3	IT 101 A	Maha	C4	1	2
4	IT 101 A	Maha	C2	1	2
5	IT 101 B	Hietham	C5	1	3
6	IT 101 B	Hietham	C4	1	3
….					
329	CS 254	Elsharri	D3	4	165
330	CS 254	Elsharri	C2	4	165

III. TIME TABLE

The weekly timetable is generated using the inputs table given above, which contains data on sixteen classrooms, seven time-periods and 330 courses (Fig. 1). As can be seen, there are six teaching days, each consisting of seven time-periods during which courses can be held in sixteen classrooms.

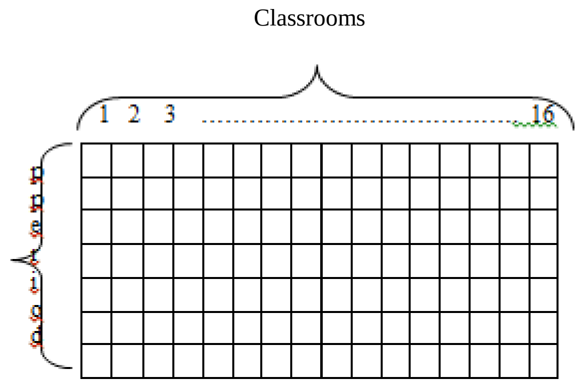

Fig.1 One day time table with 16 classrooms and 7 time periods

IV. MAPPING THE TIMETABLING PROBLEM TO THE UNDIRECTED GRAPH

The mapping table is created using the data contained in the input table. This table will be the key point for the mapping process and is similar to the main input table, with the following changes: the second column is removed; the third column for the lecturer is replaced with the key value for each lecturer; in the fourth column, the classroom name is replaced by the corresponding key value; the fifth column shows the course corresponding to course level; and the last column is used to indicate the similarity between the courses.

TABLE VI
MAIN INPUTS

Course Key no.	Lecturer	Classroom	Year	Similarity
1	17	8	3	1
2	17	9	3	1
3	22	4	1	2
4	22	4	1	2
5	6	2	1	3
6	6	2	1	3
….				
330	4	3	2	165

A. Undirected Graph

A graph G = (V; E), is a mathematical structure consisting of a set of vertices V and a set of edges E connecting the vertices.

Formally: G = (V; E), where V is a set of vertices and $E \subseteq V \times V$

G = (V; E) is undirected if for all $v; w \in V$:

$$(v, w) \in E \leftrightarrow (w, v) \in E$$

Vertices: Each time-period (cell) in the timetable can be denoted as vertex, whereby the number of vertices can be calculated as follows:

$$no.of\ vertices = no.of days \times no.of time\ pierods \times no.of\ classrooms$$

According to the above example, the number of vertices will be $6 \times 7 \times 16 = 672$, indicating that the time table can contain 672 time periods. As during the semester only 330 subjects will be taught, there will be $672 - 330 = 342$ free time periods.

Edges: The weights of edges are representing the clash type between each two time periods (cells) in the time table, as given below.

$$W_{i,j} = the\ weight\ of\ the\ edge\ between\ the\ two\ cells\ i\ and\ j$$

Based on the mapping table, we can create the weight matrix depending on the following four hard constraints for the time table:

- The lecturer cannot teach more than one course in the same time period on the same day.
- The classroom should not be booked for more than one class in the same time period on the same day.
- The second lecture of each course should not be scheduled on the same day.
- The first and third year courses should be held on the same day, and the same applies to the second and fourth year courses.

The weight value of each edge consists of four-digit binary number, whereby each digit indicates one of the four constraints, i.e., if none of the constraints exist between cell i and cell j then, $W_{i,j} = 0000$, if only the second and forth constraint exist between cell i and cell j then $W_{i,j} = 0101$ and so on.

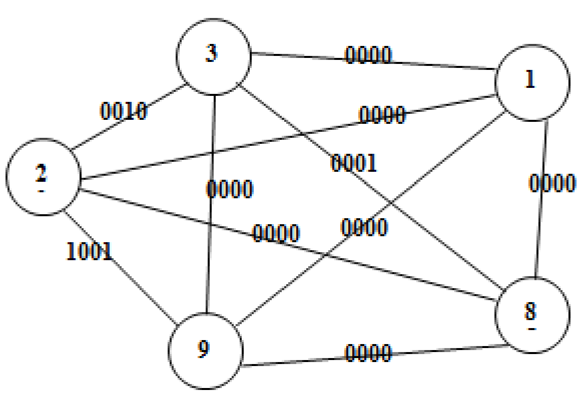

Fig. 2 shows the undirected graph for faculty time table and Figure 3. Shows its Weight matrix

To determine the entire weighting matrix w, the following steps will be followed:

$if\ i = j\ W_{i,j} = 0000$

If the cell i and j have the same lecturer, then $W_{i,j} = 1000$

If the cell i and j share the same classroom, then $W_{i,j} = 0100$

If the cell i and j share the same course, then $W_{i,j} = 0010$

If the sum of the year in the cell i and cell j is an odd number, then $W_{i,j} = 0001$

	1	2	3	4	…..	330
1	0000	0100	000	0000		0100
2	0100	0000	0010	0010		
3	0000	0010	0000	1100		
4	0000	0010	1100	0000		
….						
…..						
…						
330	0100					

Fig. 3 Weight matrix of the undirected graph of the faculty timetable

V. The Proposed Genetic Algorithm

Undirected graph simplifies the timetable problem, making the chromosome encoding and fitness function designing much easier.

A. Encoding

The faculty timetable shown in Figure 2 can be represented as two-dimensional array, with the number of rows equal to

$$Number\ of\ rows = no.of\ days \times no.of\ time\ pirods$$

and number of columns equal to $no.of\ classrooms$. Accordingly, this 2D array can be reshaped to one-dimensional array that contains all the integer numbers between

$one\ \ to\ no.of\ days\ no.of\ pirods\ mno.of\ classrooms$; i.e. (1, 2, …, 672). These numbers are the vertices by the conception of undirected graph and the genes that formed the chromosome of the GA. As each gene represents one key value of the mapping table, the GA chromosome for solving the timetable problem will be of the permutation encoding type Figure 4.

46	79	71	94	11	78	….	….	241	368

Fig. 4 The chromosome encoding

B. Fitness function

The fitness function is used to evaluate the quality of each chromosome in the GA population by calculating the number of clashes in the timetable. Our goal is to minimize the value of the fitness function until it reaches zero, as that indicates that the time table fits all the requirements and is free of any clashes. The fitness function can be formulated as following:

$$fitness = \sum_{i,j=1}^{n} w_{ij}$$

Where

$$w_{ij} = number\ of\ one's\ of\ the\ four\ digit\ binary\ number$$

C. Crossover

In this work, multi-crossover technique was used [1], as it guarantees best results in a short time if two individuals produce 100 children.

D. Mutation

The mutation operator used in this work is bitwise mutation.

E. Experimental results

For creating the timetable in this work, the actual data for the Faculty of Information Technology, University of Benghazi was used. The results were very promising, as we produced the faculty timetable that was free of clashes and fit the Faculty requirements. More specifically, 16 classrooms were booked in 3 - 5 minutes, and when the number of classrooms was reduced to 10, the time is increased to 7 - 12 minutes.

VI. Conclusion

In this work, new GA chromosome representation and fitness function was used to solve a timetabling problem. We implemented the algorithm using Matlab language and the actual data provided by Faculty of Information Technology, University of Benghazi, as a case study. The results we achieved were successful. This work can be applied to any scheduling problem and is thus valuable contribution both to the academic field and practice.

References

[1] Y. Elhaddad, A. Ganous, “New crossover technique for genetic algorithm” *International Conference on Computer and Software Modeling (ICCSM)* Manila, Philippines (2010)