Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/100338

Graph Coloring based Optimized Algorithm for Resource Utilization in Examination Scheduling

*Sandeep Saharan and Ravinder Kumar**

Department of Computer Science & Engineering, Thapar University, Patiala, Punjab - 147004, India.

Received: 4 Nov. 2015, Revised: 4 Feb. 2016, Accepted: 5 Feb. 2016
Published online: 1 May 2016

Abstract: Time table scheduling in any organization requires efficient allocation of constrained resources in an organized manner so that no conflict arises. This paper presents a novice method for optimized utilization of resources (for *e.g.* rooms, teachers, and time slots etc.) in an examination timetable using graph coloring, which clearly shows how examination can be scheduled efficiently. The approach designed in this paper has been discussed in two parts. Under first approach, examinations are removed from the independent set obtained by graph coloring if the total students in examinations are more than the available seats, in all rooms of the institution. The second approach describes adjustments of removed examinations into other independent sets to minimize the total time slots used. The designed scheme works successfully under some assumptions and constrained which are duly mentioned in the proposed approach.

Keywords: Examination Timetable, Scheduling, Graph Coloring, Constraints, Independent Set.

1 Introduction

In an educational system, time table problem has mainly two variants.

(a) Course time table of universities / schools consisting of weekly time table of various courses taught at universities / schools.
(b) Organizing and scheduling of examination time table.

Examination scheduling is organizing various examinations resources *viz.* time slots, teachers/invigilators and rooms, according to the time at which they take place and usage respectively, along with satisfying some given constraints. In this work, we have focused on the problem of organizing final examination of the university / educational institution. Solution of this problem is to schedule all examinations in a timetable by considering common students that will attend to these exams of limited room capacity constraints. To resolve this problem we have categorized all constraints broadly into two categories *Hard* constraints and *Soft* constraints. *Hard constraints* are those that should not be violated at all. For e.g. Each and every examination must be scheduled in exactly one time slot; There must be at least one invigilator in the room; Examinations of two subjects having common student(s) should not happen in one time slot; There must be enough seats in each time slot for all examinations scheduled; Moreover certain examinations must be scheduled at specific time slots or into specific rooms. Secondly, *Soft constraints* are those that can be tolerated or ignored some time. For e.g. Not more than predefined students scheduled to sit for examinations at any particular time slot if resources are surplus; Examination for each subject should not split across rooms; Not more than one examination in a room at a time; Distance between rooms having same examination should be minimized (when room splitting is allowed); Examinations should be completed in predefined time slots etc.

Hard and soft constraints are subjective in nature and it solely depends on ones requirement. Finding feasible solution purely depends upon the number and nature of given constraints. Sometimes it becomes impossible to find solution whereas sometimes a large number of feasible solutions exist and then focus shifts to extracting the solution which violates minimum soft constraints. The main goal of an examination timetable is to guarantee that all examinations are scheduled and students can sit for all the examinations in which they are required to appear. It is a Non Polynomial (NP) hard combinatorial optimization problem.

* Corresponding author e-mail: ravinder@thapar.edu

2 Motivation and Related Work on Examination Scheduling

An algorithm [1] for graph coloring was proposed in which non adjacent pair of vertices were merged to give the same color and done for all vertices in descending order of number of common adjacent vertices. That algorithm was used to propose a reasonable graph coloring algorithm [2]. In that algorithm, initially the maximum degree vertex was selected and all its triples are found out. In finding triple, first and third vertex should not be adjacent to each other. Then first and third vertex is merged to make both having the same color. In this way, coloring of whole graph continues. A new method [2] was given in which exam scheduling and room allocation algorithm were combined. In the work no room splitting was taken as a hard constraint. Various directions [3] of research in automated time tabling were provided in which sequential, cluster, constraint based and meta-heuristic methods were explained. Traditionally room assignment was done after coloring the course conflicting graph while in alternative approach [4] rooms were assigned to courses during graph coloring phase. In new model for automated time tabling given in [5], various required hard and soft constraints have been discussed with various penalty functions applied which gave more correctness or feasibility of the solution. Multi criteria decision problems have also been discussed [6]. A criterion was meant by any hard or soft constraint. There were two phases, in the first phase various timetables have been generated satisfying all criteria individually, while in the second phase a compromised solution was found out satisfying all criteria at parallel level. In the deterministic algorithm [7], hyper-heuristics were applied which is capable of solving a range of non-trivial exam and class timetable problems. Varied problem description was considered and many different methods for measuring the fitness of the algorithm were provided. A hyper heuristic algorithm [8] used in which Tabu search approach was used on various low level graph heuristics in specific environment and time. In evolutionary approach based method [9] for constructing time table, solution was given in two stages. In the first stage, solution was finding out according to the given set of constraints and in the second stage those constraints were established. After following the genetic approach, Tabu search approach was applied to stop blind search when given deviation in the population was left. Guided adaptive length chromosome hyper-GA method [10] includes for scheduling problems. Used guided strategy refers to removal of poorly performing heuristics and the injection of promising heuristics. A method [11] defined in illustration graph colouring, with heuristic approach that uses cluster heuristic and sequential heuristic to solve examination time table problem for Multimedia University, Malaysia. In hybrid algorithm [12] for time tabling three methods, constraint programming to obtain a feasible timetable, simulated annealing to improve the quality of the timetable, hill climbing for further refinement of the timetable were used. In new exam scheduling algorithm [13], subjects were picked according to their degree and subject with highest degree was scheduled first. When graph coloring priority was given to the nodes which are connected to highest weight edges with weight of an edge is defined as the number of common students between two subjects. In [14] users were allowed to view and easily edit a cognitively manageable representation of each timetable, using the STARK (Semantically Transparent Approach to Representing Knowledge) approach and directly control the heuristics which were used to generate solutions automatically, using the HuSSH (Human Selection of Scheduling Heuristics) approach. An adaptive decomposition and ordering approach [15] brings solutions for exam scheduling problems in which problems are decomposed adaptively into two subsets, the difficult set and the easy set. These set were used to construct solutions by adjusting the ordering of the exams in one set while fixing the other. Graph heuristics such as largest weighted degree and saturation degree respectively were adaptively hybridised at different stages of the solution construction for both exam timetable and graph colouring problems in automated heuristic construction approach [16]. In IP model [17], solving sub-problems of exam timetable were identified by an adaptive decomposition approach. Difficult sets which are adaptively decomposed from the original large problems were solved optimally by the model. For better formulation, clique inequalities derived from set packing problems and problem specific cutting planes were included and introduced respectively. The method given in [18] stores a case-base of examination timetable problems with the heuristic or meta-heuristic technique which gave optimum results. It suggested so far best technique based on large base of past knowledge. The method in [19] improves time taken to find the solution and also quality of the solution. The method was a hybrid of heuristic sequencing and evolutionary methods which orders the events according to expected difficulty instead of taking the single most difficult event and scheduling it into the best available period, after that evolutionary algorithm is applied to find the best placements with respect to each other and with already scheduled events. The method [20] is a hybrid genetic algorithm in which initial population is stored into red-black tree data structure. Then the algorithm generates new offspring from previous individuals by its reproduction operators. Hill climbing is also used to improve local exploitation. In [21] a RIFD system provides a solution for scheduling different exams in same halls simultaneously. So, halls capacity will be used more efficiently and cheating attempts will be decreased considerably. A genetic-based approach for scheduling examination timetable with one-point and two-point crossover operators and propagates distinctive timetable features to generate better solutions even for complex cases was discussed in [22].

3 Existing Methodology

In existing methodologies [2], *No Room Splitting* is taken as a hard constraint and examination scheduling is combined with room allocation algorithm. Because of the course timetable, it was assumed that largest subject can fit into the largest room. Starting with the smallest examination and then continuing through the list, and putting each examination in the smallest room, it will fit in, while assuming that no other examinations are scheduled. If there are now more students scheduled in the room than the room can cope with, they displaces the examination having minimum number of students to the next bigger room so that the combined size of the examinations remaining is less than the size of the room, repeating with the next bigger room and so on. If there are no more rooms left then the displaced examinations are put into list of unscheduled examinations in the current iteration. This will produce an assignment of examinations to the rooms and a list of examinations for which rooms cannot be found. In single iteration, examinations scheduled comes under one time slot. Then whole process along with graph coloring is carried out on remaining examinations and so on for scheduling all examinations.

3.1 Graph Coloring

The purpose of using graph coloring is to divide all examinations into different groups so that no clashes happen. In graph, node represents examination and edge between two examinations exists if both the examinations have common student(s) because of which we cannot schedule those two examinations in the same time slot. After minimum graph coloring [1][2] done, we get set of independent sets of examinations where one independent set corresponds to one color and only one independent set can be schedule in one time slot.

One independent set = One time slot

*3.2 No Room Splitting:*a constraint

In most of the work done till now in the field of examination timetabling, *No Room Splitting* is taken as a *hard* constraint. In course timetable students having same subject should be in same room while having lecture or they may split across different rooms according to their sections. Each section may have same or different faculty according to when their lectures are scheduled. But only one examination should be there for each subject if multiple sittings are not allowed. If sections are there then examination will definitely split across rooms because generally sections are made due to shortage of seats even in the largest room. However, if sections are not there then although examinations can happen only in one room but why dont we use vacant seats if there in existing method and split examination across different rooms while examination etiquette remains same for all the students giving same or different examinations in a room within a same time slot. For this reason, we can omit it even as a soft constraint. By doing *room splitting* efficiency in terms of minimization of time slots and utilization of resources can be achieved.

4 Functional Model of Examination Timetabling Problem

In the functional model of examination timetabling problem, we introduce notation using sets, relations and functions for data representations. Set is denoted by upper case letter while its elements are denoted by same lower case letters for *e.g.* T for set of time slots which contains t as its element. Relation is also a set which specifically shows association between elements of two or more same or different sets. Relations can also be seen as functions. In the function name $f_r(d)$, f shows name of the relation, d denotes valid domain of the relation and r denotes the range for the given domain. In domain and range upper case letter denotes sets while lower case letter denotes single element accordingly.

Definition 1: If $T = \{t|t \text{ denotes time slot}\}$ and $D = \{d|d \text{ denotes day of week}\}$ are finite sets. A provides relation, between time slot t and day d *i.e.*, input time slot t and return day d of that time slot.

$$A = \{a = (t,d)|\ t \in T \wedge d \in D\} \subset T \times D \qquad (1)$$

Definition 2: If $S = \{s|s \text{ denotes student}\}$, $E = \{e|e \text{ denotes exam}\}$ are finite sets. B is relation, between student s and set of examinations V *i.e.*, input student s and return set V contains examinations in which particular student s is enrolled.

$$B = \{b = (s,V)|\ s \in S \wedge V \subseteq E\} \subset S \times 2^E \qquad (2)$$

F denotes relation, between examination e and set of students H *i.e.*, input examination e and return set H contains students enrolled in examination e.

$$F = \{f = (e,H)|\ e \in E \wedge H \subseteq S\} \subset E \times 2^S \qquad (3)$$

Definition 3: If $R = \{r|r \text{ denote room}\}$ and $C = \{c|c \text{ denotes seat}\}$ are finite sets. K denotes relation, between room r and set of seats M *i.e.*, input room r and return set M contains seats of room r.

$$K = \{k = (r,M)|\ r \in R \wedge M \subseteq C\} \subset R \times 2^C \qquad (4)$$

U denotes relation, between seat c and room r *i.e.*, input seat c and return room r containing particular seat c.

$$U = \{u = (c,r) | \, c \in C \wedge r \in R\} \subset C \times R \tag{5}$$

Definition 4: [Graph Input] A graph G is a set of ordered pair elements $g = (e,N)$ comprising examination e together with a set N of examinations if e is dependent on each examination of set N.

$$G = \{g = (e,N) \text{if edge exists between e and elements of N } | e \in E \wedge \ N \subseteq E\} \subset E \times 2^E, \tag{6}$$

Definition 5: [Graph Coloring] Let P is resultant set of independent sets of examination after graph coloring properties and p is an element of P, where P is an independent set that is a subset of E.

$$P = \{p \subseteq E | (|p_i \cap p_j| = 0 | i \neq j) \wedge (\cup_{i=1}^{totalcolors} p_i = E) \tag{7}$$

5 Modified Method

As intake of various courses may increase and more students can choose same common subject. However, seats in all rooms remain same until new rooms are constructed. This poses the problem that we need to create more than one section of a class or for the subject. But in most of the cases examination of all the sections will be same and must be scheduled in the same time slot. In the presence of these constrains, examination will not schedule according to the above mentioned existing method.

Because of *No Room Splitting* it may be possible that some seats are left vacant in case examinations scheduled in the room are having less total students than seats in that room. Adding those left over seats of all the rooms may allow scheduling one or more unscheduled examination. In the modified method after removing these limitations, we got better results in terms of less number of time slots used and hence better resource utilization. Let us assume total seats of all the rooms are equal or greater than the largest examination. This algorithmic approach is divided as follows:
Final Independent Sets: Algorithm (1) $\rightarrow$ *Remove Examinations:* Algorithm (2) $\rightarrow$ *Adjust Examinations:* Algorithm (3) $\rightarrow$ *Time Slots Assignment:* Algorithm (4) $\rightarrow$ *Rooms/Seats Allotment:* Algorithm (5)

5.1 Final Independent Sets

This process will run until each and every examination is not included under one of the final independent set or time slot. See Algorithm (1)

Algorithm 1: Extraction of Independent Sets

```
Data: Input Sets C, E, F, G, K, P, R, S.
      Temporary Set E^c.
Result: Set L as final set of independent sets.
Let L be an empty set;
Let set E^c is a copy of set E;
while |E^c| ≠ 0 do
    Call Graph Coloring for E^c;
    Call Algorithm (2);
    Call Algorithm (3);
    L = L ∪ P
    for i ← 1 to |P| do
        for j ← 1 to |P_i| do
            E^c = E^c \ P_{i_j}
        end
    end
end
```

5.2 Remove Examinations

If we have fewer seats in all the rooms than the total students in any independent set, then in order to remove students we have to remove examinations from the set so that difference between the two vanishes.
Approach Used: Remove examinations from independent set of examinations sorted in increasing order of count of students until difference vanishes. While examination having maximum degree in that independent set should not be removed in the process until it becomes mandatory to do so. This is because in the next step if it is required and possible we will adjust removed examinations into other independent sets and if we remove examination having maximum degree then because of its dependency on examinations present in other independent sets, it may become difficult to adjust. Hence it may increase total time slots resulting in more resource utilization. See Algorithm (2)

5.3 Adjust Examinations

In a particular independent set after removing examinations if difference between students and seats becomes negative then seats will be vacant during the examination. So we can adjust the removed examinations from all independent sets contained in set X into existing independent sets if possible, in order to minimize total time slots. See Algorithm (3)

5.4 Time slots Assignment

Let T is a set of sorted time slots in chronological order. Until any specific conditions are mentioned, time slots will be assigned to the independent sets and hence to the examinations as follows. See Algorithm (4)

Algorithm 2: Remove Examinations

Data: Let X is temporary set that contain examinations which has been removed from set P. Variables contain, T_{seats} (count of total seats), $T_{students}$ (count of total students), $diff$ (difference between T_{seats} and $T_{students}$,) and max (maximum degree of vertices).

Result: Processed sets P, X for further references.

$T_{seats} = \sum_{i=1}^{|R|} |K_c(R_i)|$;
for $i \leftarrow 1$ **to** $|P|$ **do**
  $\forall_j$ sort all P_{i_j}'s of P_i's in increasing order of $F_S(P_{i_j})$ where $j \in [1, |P_i|]$;
end
Let X an empty set;
for $i \leftarrow 1$ **to** $|P|$ **do**
  $T_{students} = \sum_{j=1}^{|P_i|} |F_S(P_{i_j})|$;
  $diff = T_{students}$ - T_{seats};
  $max = |G_E(P_{i_1})|$;
  for $j \leftarrow 2$ **to** $|P_i|$ **do**
    if $max < |G_E(P_{i_j})|$ **then**
      $max = |G_E(P_{i_j})|$;
    end
  end
  for $j \leftarrow 1$ **to** $|P_i|$ **do**
    while $diff > 0$ **do**
      if $|G_E(P_{i_j})| \neq max$ **then**
        $diff = diff - |F_S(P_{i_j})|$;
        $X = X \cup P_{i_j}$;
        $P_i = P_i \setminus P_{i_j}$;
      end
    end
  end
  for $j \leftarrow 1$ **to** $|P_i|$ **do**
    while $diff > 0$ **do**
      $diff = diff - |F_S(P_{i_j})|$;
      $X = X \cup P_{i_j}$;
      $P_i = P_i \setminus P_{i_j}$;
    end
  end
end

Algorithm 3: Adjust Examinations

Data: Let Y is temporary set contains all adjacent examinations of given independent set. Z is a temporary set that contains all those examinations which are there in set X but not in set Y at a given time and L_{seats} is temporary variable that contains left over seats.

Result: Processed sets P, X for further references.

if $|X| \neq 0$ **then**
  $T_{seats} = \sum_{i=1}^{|R|} |K_c(R_i)|$;
  $\forall_i$ sort all P_i's of P in decreasing order of $\sum_{j=1}^{|P_i|} |F_S(P_{i_j})|$ where $i \in [1, |P|]$;
  for $i \leftarrow 1$ **to** $|P|$ **do**
    while $|X| \neq 0$ **do**
      $T_{students} = \sum_{j=1}^{|P_i|} |F_S(P_{i_j})|$;
      $L_{seats} = T_{seats} - T_{students}$;
      if $L_{seats} > 0$ **then**
        Let Y be an empty set;
        for $j \leftarrow 1$ **to** $|P_i|$ **do**
          $Y = Y \cup G_E(P_{i_j})$
        end
        Let Z be an empty set;
        $Z = X \setminus Y$;
        $\forall_j$ sort all Z_i of Z in decreasing order of $|F_S(Z_j)|$ where $j \in [1, |Z|]$;
        for $j \leftarrow 1$ **to** $|Z|$ **do**
          while $L_{seats} > 0$ **do**
            if $|F_S(Z_j)| \leq L_{seats}$ **then**
              $P_i = P_i \cup Z_j$;
              $L_{seats} = L_{seats} - |F_S(Z_j)|$;
              $X \leftarrow X \setminus Z_j$;
            end
          end
        end
      end
    end
  end
end

In algorithm(4) Q comes out as a set which contains all time slots and their corresponding independent set to which they have been assigned and set O contains all examinations and their respective time slot into which they have been scheduled.

5.5 Rooms/Seats Allotment

Until any specific conditions are mentioned, seats to the students or rooms to the examinations belonging to independent sets can be assigned as follows. See Algorithm (5)

In algorithm (5) $\hat{E}$ comes out as a set whose single element contains four values. First value is a particular student, second tell his/her particular examination, third and fourth tell in which room and particular seat examination of that student is scheduled. Set $\hat{C}$ contains all examinations and their respective rooms into which they have been scheduled.

6 Implementation and Experimental Results

Let all pairs of $\{r, |K_C(r)|\}$ are as follows $\{r_1, 30\}$, $\{r_2, 60\}$, $\{r_3, 110\}$. Let all pairs of $\{e, |F_S(e)|\}$ are as follows $\{e_1, 10\}$, $\{e_2, 20\}$, $\{e_3, 50\}$, $\{e_4, 100\}$, $\{e_5, 90\}$, $\{e_6, 100\}$, $\{e_7, 30\}$, $\{e_8, 70\}$, $\{e_9, 80\}$, $\{e_{10}, 60\}$, $\{e_{11}, 60\}$, $\{e_{12}, 80\}$. For simplicity, examination identity and number of students are representing as base and power of symbol e respectively. For *e.g.* $e_4^{100}$where e_4 is examination identity and 100 is number of registered

Algorithm 4: Time Slots Assignment

Data: Input Sets L, T.
Result: Set Q and O gives relation between particular time slot with list of examinations and particular examination respectively.

```
if |T| ≥ |L| then
    Let Q, O be the empty sets;
    for i ← 1 to |L| do
        Q = Q ∪ {T_i, L_i};
        for j ← 1 to |L_i| do
            O = O ∪ {L_{i_j}, T_i};
        end
    end
end
else
    Print: Insufficient time slots to schedule all
    independent sets/examinations;
end
```

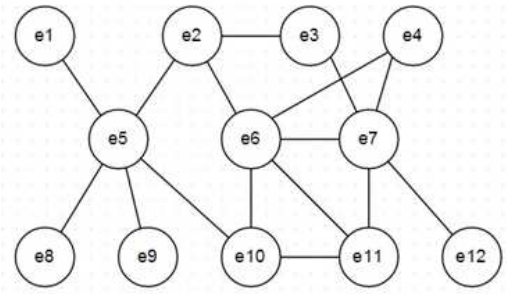

Fig. 1: Input Graph

students for examination e_4. The dependencies among the examinations are shown in Fig.1 which is a input graph.

After applying the graph coloring algorithm [2], the resultant graph will be as shown in Fig 2.

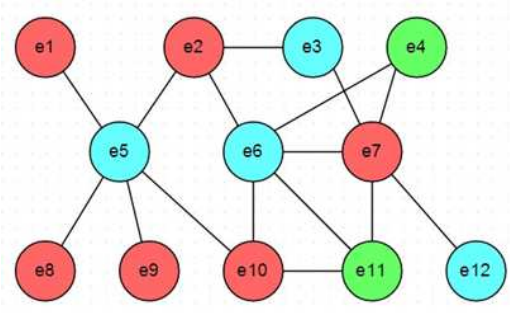

Fig. 2: Based on Graph Coloring Algorithm

Algorithm 5: Rooms/Seats Allotment

Data: Input Sets C, F, K, L, R, S, U.
Temporary Sets $\hat{A}$, $\hat{D}$, $\hat{R}$.
Result: Sets $\hat{E}$, $\hat{C}$ gives relation between students, examinations, rooms and seats.

$\forall_i$ sort R_i's of R in decreasing order of $|K_C(R_i)|$ where $i \in [1,|R|]$;

```
for i ← 1 to |R| do
    Â = Â ∪ K_C(R_i);
end
for i ← 1 to |L| do
    Let m = 1;
    for j ← 1 to |L_i| do
        D̂ = F_S(L_{i_j});
        R̂ = U_r(Â_m);
        for k ← 1 to |D̂| do
            if R̂ ∩ U_r(Â_m) == φ then
                R̂ = R̂ ∪ U_r(Â_m);
            end
            Ê = Ê ∪ {D̂_k, L_{i_j}, U_r(Â_m), Â_m};
            m = m + 1;
        end
        Ĉ = Ĉ ∪ {L_{i_j}, R̂};
    end
end
```

6.1 Existing Methodology

See results by existing methodology in table(1)

Here, examination comes under each iteration can be assigned to one time slot in corresponding rooms. Hence, the final schedule will be as shown in table (2)

So, using existing methods total six time slots will be used to schedule all given examinations.

6.2 Modified Method

In each iteration contents of various sets (*i.e.* P, L, X) after execution of various nested algorithms (*i.e.* graph coloring, Algorithm (2), Algorithm (3)) in Algorithm (1) will be as shown in table (3).

Here, set L is set of independent sets which contains four independent sets. Examinations contained in each independent set can be assigned to one time slot. Hence the final schedule will be as shown in table (4). So using modified method total four time slots will be used to schedule all the given examinations.

The comparative results of resource utilization for given problem using existing and modified methods has been shown in Table (5).

$$TSI = TTU \times MIR - TRW$$

Efficiency in total sittings of invigilators (i.e. TSI) will be as follows

$$(TTU \times MIR - TRW) \,/\, (TTU \times MIR)$$

Table 1: Existing Room Allocation/Time Slots Assignment.

	Iteration 1	**Iteration 2**	**Iteration 3**	**Iteration 4**	**Iteration 5**	**Iteration 6**
Unscheduled Examinations	$\{e_8^{70}\}$	$\{e_{12}^{80},e_5^{90}\}$	$\{e_{12}^{80},e_5^{90}\}$	$\{e_{12}^{80}\}$	$\{e_8^{70}\}$	Null
(r_3,110)	$\{e_1^{10},e_2^{20},e_9^{80}\}$	$\{e_6^{100}\}$	$\{e_4^{100}\}$	$\{e_5^{90}\}$	$\{e_{12}^{80}\}$	$\{e_8^{70}\}$
(r_2,60)	$\{e_{10}^{60}\}$	$\{e_3^{50}\}$	$\{e_{11}^{60}\}$	Null	Null	Null
(r_1, 30)	$\{e_7^{30}\}$	Null	Null	Null	Null	Null
Graph Coloring	$\{e_1, e_2, e_7, e_8, e_9, e_{10}\}$	$\{e_3, e_5, e_6, e_{12}\}$	$\{e_4, e_5, e_{11}, e_{12}\}$	$\{e_5, e_{12}\}$	$\{e_8, e_{12}\}$	$\{e_8\}$

(In each iteration read data in upward direction)

Table 2: Existing Schedule.

Time Slot	**Examinations**	**Total Students**
t_1	$\{e_1, e_2, e_7, e_9, e_{10}\}$	200
t_2	$\{e_3, e_6\}$	150
t_3	$\{e_4, e_{11}\}$	160
t_4	$\{e_5\}$	90
t_5	$\{e_{12}\}$	80
t_6	$\{e_8\}$	70

Table 3: Modified Method Results

Algorithm (1)						
			Graph Coloring	Algorithm (2)	Algorithm (3)	L
Iteration 1	P	p_1	$\{ e_1^{10}, e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80}, e_{10}^{60} \}$	$\{ e_7^{30}, e_8^{70}, e_9^{80} \}$	$\{ e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80} \}$	$\{\{e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80}\}, \{e_5^{90}, e_6^{100}\}, \{e_1^{10}, e_4^{100}, e_{11}^{60}\} \}$
		p_2	$\{ e_3^{50}, e_5^{90}, e_6^{100}, e_{12}^{80} \}$	$\{ e_5^{90}, e_6^{100} \}$	$\{ e_5^{90}, e_6^{100} \}$	
		p_3	$\{ e_4^{100}, e_{11}^{60} \}$	$\{ e_4^{100}, e_{11}^{60} \}$	$\{ e_1^{10}, e_4^{100}, e_{11}^{60}\}$	
	X		Null	$\{ e_1^{10}, e_2^{20}, e_3^{50}, e_{10}^{60}, e_{12}^{80} \}$	$\{ e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$	
Iteration 2	P	p_1	$\{ e_3^{50}, e_{10}^{60}, e_{12}^{80} \}$	$\{ e_3^{50}, e_{10}^{60}, e_{12}^{80} \}$	$\{ e_3^{50}, e_{10}^{60}, e_{12}^{80} \}$	$\{ \{ e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80} \}, \{ e_5^{90}, e_6^{100} \}, \{ e_1^{10}, e_4^{100}, e_{11}^{60} \}, \{e_3^{50}, e_{10}^{60}, e_{12}^{80}\} \}$
	X		Null	Null	Null	

Table 4: Modified Schedule

Time Slots (T)	**Independent Sets** (L)	**Examinations**	**Total Students**
t_1	l_1	$\{e_2, e_7, e_8, e_9\}$	200
t_2	l_2	$\{e_5, e_6\}$	190
t_3	l_3	$\{e_1, e_4, e_{11}\}$	170
t_4	l_4	$\{e_3, e_{10}, e_{12}\}$	190

Table 5: Resource utilization using both methods

Resource Type	Using Existing Method	Using Modified Method
Total time slots used (TTU)	6	4
Total rooms wasted (TRW)*If all rooms are available in each time slot used.*	8	1
Total seats wasted (TSW)	450	50
Min. invigilator required (MIR)*If one invigilator per room is used.*	3	3
Total sittings of invigilators (TSI)	10	11

Efficiency in TSI of existing method will be $10/18 = 0.55$.
Efficiency in TSI of modified method will be $11/12 = 0.92$.

Hence, it is clearly shown here that optimization in total sittings of invigilators (TSI), total seats wasted (TSW), total rooms wasted (TRW) can be achieved in modified method. The modified approach presented in this paper clearly provide on edge over the existing method in terms of better resource utilization.

7 Contributions of the work through various conditions

Two examinations must or must not schedule in adjacent time slots: We will draw an edge between those two examinations, so that they cannot schedule in the same time slot. For both must be in adjacent time slots, as soon as time slot assigned to independent set containing one of the given two examinations then the next time slot will be assigned to the independent set having other examination. While for another case we will look for independent set which dont contain any of the two given examinations and then we will assign it a time slot which lies between the two time slots assigned to the independent sets containing each given examination. In the worst case we can add in between a free time slot as a gap.

An examination must or must not schedule in a particular time slot: If an examination must occur in a particular time slot then firstly we will assign that particular time slot to the independent set containing that particular examination. Further, normal assignment of left time slots to the left independent sets will be done. While in the second case, we will assign a time slot (other than given time slot) to the independent set containing given examination and then normal assignments will take place.

One examination must schedule before or after another: This is a specific case of adjacent time slots. Again, firstly we will look for independent set containing given examination which should be schedule first. After assigning it a time slot, the immediate next time slot will be assigned to the independent set containing another given examination and then rest of the assignments will continue normally.

Larger examinations should schedule in the beginning as they take longer to evaluate: We can assign first time slot to the independent set which is having largest examination and then if required second time slot to the set which contain next unscheduled largest examination and so on.

An examination can only take place in specific rooms: At seats allotment stage if independent set contain given specific examination then at first we will directly allot seats of given specific room(s) to that given examination. Then examination and allotted seats will be removed from that independent set and set of seats respectively. Specific room(s) will be removed from set of rooms, if no one seat left vacant in the room(s). After it normal seat allotment procedure will be followed on the remaining set.

8 Conclusion

This approach provides that instead of doing graph coloring on remaining examinations each iteration, we should go for only one time graph coloring until it becomes mandatory to do so. In addition to this, adjusting unscheduled examinations into different independent sets considering all dependencies among them will decrease space and time complexity. In existing method, examination scheduling and room allocation algorithms were combined while taking *No Room Splitting* as a hard constraint. Examinations having students more than capacity of the largest room is very much possible in real life and can be scheduled using modified method. The results provided by modified approach clearly reveal that following modified approach we get better results that ensure optimized resource utilization. So the use of partially vacant seats in efficient manner and removing *No Room Splitting* even from soft constraints in the modified method results in the reduction of time slots and hence better resource utilization even with all the examination constraints.

Acknowledgments

The authors like to thanks Dr. Nidhi Walia, USAM, Punjabi University, Patiala for her helpful suggestions and discussions. The authors are grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References

[1] Dutton R.D. and Brigham R.C., A new graph coloring algorithm, The Computer Journal **24**, 85-86,(1981).

[2] Burke E.K., Elliman D.G., and Weare R.F., A university timetable system based on graph colouring and constraint manipulation, Journal of Research on Computing in Education **27**, 1-18 (1994).

[3] Burke E.K. and Petrovic S., Recent research directions in automated timetable, European Journal of Operational Research, **140**, 266-280,(2002).

[4] Redl T.A., University timetable via graph coloring: An alternative approach, Congressus Numerantium,**187**, 162-174, (2007).

[5] McCollum B., McMullan P., Parkes A.J., Burke E.K., and Qu R., A new model for automated examination timetable, Annals of Operations Research, **194**, 291-315, (2012).

[6] Burke E., Bykov Y., and Petrovic S., A multicriteria approach to examination timetable, in Proc. of Third International conference Practice and Theory of Automated Timetable III, **2079**, 118-131, (2001).

[7] Ross P., Marin-Blazquez J.G., and Hart E., Hyper-heuristics applied to class and exam timetabling problems, in Proc. of Congress on Evolutionary Computation, **2**, 1691-1698, (2004).

[8] Burke E.K., Mccollum B., Meisels A., Petrovic S., and Qu R., A graph-based hyper-heuristic for educational timetable problems, European Journal of Operational Research, **176**, 177- 192, (2007).

[9] Norberciak M., Universal method for timetable construction based on evolutionary approach, International Scholarly and Scientific Research and Innovation, **2**, 693-698, (2006).

[10] Han L. and Kendall G., Guided operators for a hyper-heuristic genetic algorithm, in Proc. of 16th Australian Conference on Advances in Artificial Intelligence, **2903**, 807-820, (2003).

[11] Hussin B., Basari A.S.H., Shibghatullah A.S., Asmai S.A., and Othman N.S., Exam timetable using graph colouring approach, in Proc. IEEE Conference on Open Systems, 133-138, (2011).

[12] Merlot L.T.G., Boland N., Hughes B.D., and Stuckey P.J., A hybrid algorithm for the examination timetable problem, in Proc. of 4th International Conference on Practice and Theory of Automated Timetable IV, 207-231, (2003).

[13] Malkawi M., Hassan M.A-H., and Hassan O.A-H., A new exam scheduling algorithm using graph coloring, International Arab Journal of Information Technology, **5**, 80-87, (2008).

[14] Cowling P., Ahmadi S., Cheng P., and Barone R., Combining human and machine intelligence to produce effective examination timetables, in Proc. of the 4th Asia-Pacific conference on simulated evolution and learning, 662-666, (2002).

[15] Qu R. and Burke E.K., Adaptive decomposition and construction for examination timetable problems, in Proc. of the 3rd Multidisciplinary International Scheduling: Theory and Applications, 418-425, (2007).

[16] Qu R., Burke E.K., and McCollum B., Adaptive automated construction of hybrid heuristics for exam timetable and graph colouring problems, European Journal of Operational Research, **198**, 392-404, (2009).

[17] Qu R., He F., and Burke E.K., Hybridizing integer programming models with an adaptive decomposition approach for exam timetable problems, in Proc. of 4th Multidisciplinary International Scheduling: Theory and Applications, 435-446, (2009).

[18] Burke E., Eckersley A., McCollum B., Petrovic S., and Qu R., Analysing Similarity in Examination Timetable, in Proc. of 5th International Conference on the Practice and Theory of Automated Timetable, 89-106, (2004).

[19] Burke E.K. and Newall J.P., A Multi-stage Evolutionary Algorithm for the Timetable Problem, IEEE Transactions on Evolutionary Computation, **3**, 63-74, (1999).

[20] Karami A.H. and Hasanzadeh M., University course timetabling using a new hybrid genetic algorithm, in Proc. 2nd International Conference on Computer and Knowledge Engineering, 144-149, (2012).

[21] Akbulut A. and Yilmaz G., University Exam Scheduling System Using Graph Coloring Algorithm and RFID Technology, International Journal of Innovation, Management and Technology, **4**, 1-7, (2013).

[22] Obaid O.I., Ahmad M.S., Mostafa S.A., and Mohammed M.A., Comparing Performance of Genetic Algorithm with Varying Crossover in Solving Examination Timetabling Problem, Journal of Emerging Trends in Computing and Information Sciences, **3**, 1437-1434, (2012).

Sandeep Saharan is an research scholar at Department of Computer Science & Engineering, Thapar University Patiala, Punjab, India. He has received his B.Tech. degree in Computer Science from M.D.University, Rohtak, Haryana, India. He is an active member of various organizations like IEEE, ACM, CSI. His research interests are in the areas of genetic algorithms, graph theory, and scheduling problems. He has published research articles in reputed international journals as well as international conferences.

Ravinder Kumar is currently working as a Assistant Professor in Department of Computer Science & Engineering, Thapar University Patiala, Punjab, India. He has published many research articles in reputed international journals. He is an member of IEEE, ACM, CSI and many more organizations. He is referee of various international journals. His main research interests include theoretical and practical aspects of combinatorial optimization, approximation algorithm, pattern recognition and mathematical programming.