Resource Optimization in Examination Timetabling using Graph Coloring

Thesis

submitted in partial fulfillment of the requirements

for the award of degree of

Master of Engineering

in

Computer Science and Engineering

Submitted By

Sandeep Saharan

Roll No: 801232022

Under the supervision of

Mr. Ravinder Kumar

Assistant Professor (CSED)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2014

Dedicated to my parents

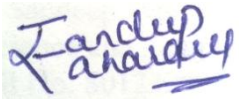
Smt. Rajpati Devi

Sh. Randhir Singh Saharan

CERTIFICATE

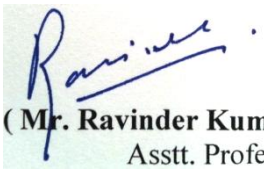
I hereby certify that the work which is being presented in this thesis entitled, "*Resource Optimization in Examination Timetabling using Graph Coloring*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mr. Ravinder Kumar and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.



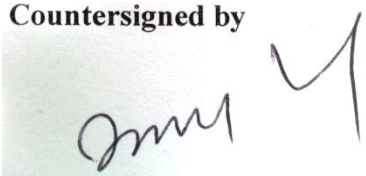
(Sandeep Saharan)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.



(Mr. Ravinder Kumar)
Asstt. Professor
CSED, Thapar University
Patiala, Punjab.

Countersigned by



(Dr. Deepak Garg)
Head & Associate Professor
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgments

I express my sincere gratitude towards my guide **Mr. Ravinder Kumar** for his constant help, encouragement and inspiration throughout the project and thesis work. Without his invaluable guidance, this work would never have been a successful one. I would like to thank **Dr. Deepak Garg** Head, CSED for his efforts on making positive, encouraging and research oriented environment among students in the department.

Last but not the least, I would like to thank **Mr. Karun Verma** (Asstt. Prof.), CSED, my friends Gurmeet Singh, Bhavishya Bansal for their valuable suggestions and helpful discussions.

Sandeep Saharan
M.E.(Computer Science and Engg.)
Roll No: 801232022
Thapar University, Patiala.

Abstract

The examination timetabling problem is a classical, old and famous problem in the field of optimization problems. Examination timetabling is an NP-Complete problem. Time table as a schedule requires to do work accurately and efficiently in an organized manner without any conflicts. Here, a novice approach for resource optimization in examination timetabling using graph coloring, is shown which specifically shows how examination can be scheduled efficiently considering *No Room Splitting* not even a soft constraint. This approach specifically describes two methods. First, in which examinations are removed from the independent set of examinations we got after graph coloring if the total students in examinations are more than the available seats in the institution. The second method describes if possible, how the removed examinations from all independent sets will be adjusted into independent sets other than it removed from, to minimize the total time slots and hence to utilize resources in better way. In the work, few constraints and assumptions, closely related to the general examination timetabling are considered.

CONTENTS

List of Figures

List of Tables

List of Abbreviations

1. 3DM..... Three Dimensional Matching
2. CBR.....Case Base Reasoning
3. IP.....Integer Programming
4. SAT.....Satisfiability
5. 3SAT.....Three Satisfiability
6. NAE3SAT…..Not All Equal-3SAT

Chapter 1: Introduction

1.1 Time Tabling

A timetable provide information about the time for specific events to happen, and relates to resources allocation. Timetabling problems can be called as a type of scheduling problem. The task of making compromised time table of good quality is challenging due to the large number of events needed to be scheduled with large number of constraints and given preferences must be satisfied. It is a NP hard combinatorial optimization type of problem. According to [1]:

"A timetabling problem is a problem having four parameters: T, a finite set of times; R, a finite set of resources; M, a finite set of meetings; and C, a finite set of constraints. The problem is to assign times and resources to the meetings so as to satisfy constraints as far as possible."

It have various practical aspects like educational timetabling which includes course and examination timetabling, sports timetabling, vehicle timetabling, employee timetabling. Specific to educational timetabling it have categories as follows.

Table 1: Types of Education Timetabling

Category	**Descriptions**
School Timetabling	Scheduling school classes over time slots in one week.
Course Timetabling	Scheduling courses over time slots and rooms in one week.
Examination Timetabling	Scheduling examinations over time slots and rooms.

1.1.1 Examination Timetabling

As a part of educational timetabling following is the definition of examination timetabling.

"Examination timetabling is organizing various examinations, students and resources (like time slots, faculty, rooms...) according to the time at which they take place and usage respectively, along with satisfying much of the given constraints."

In examination timetabling, broadly there are three types of resources associated with students and their examination i.e. invigilator, rooms/seats, time slots. Let they are entities then all entities and relations among them will be as follows.

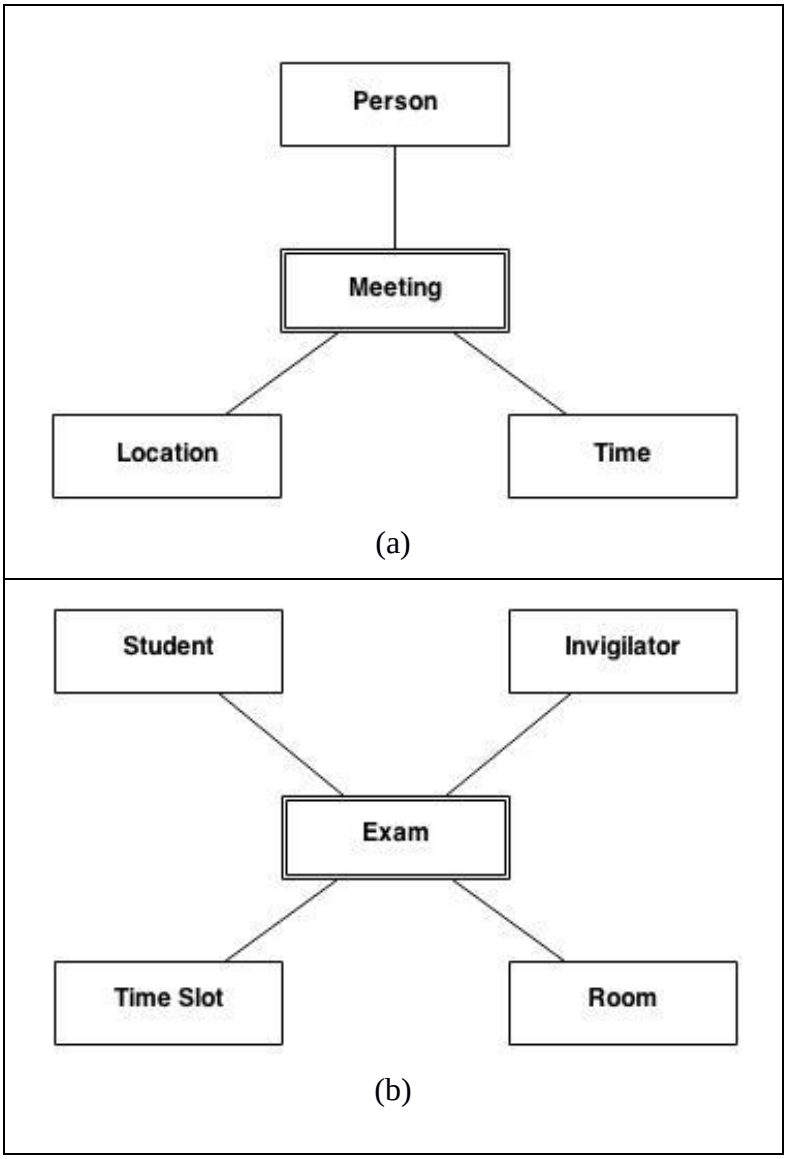


Figure 1: Entities and relation between entities for time tabling (a) in general (b) for examination in specific

As stated above scheduling problem are combinatorial problems, there are lots of constraints in examination timetabling. These constraints broadly can be categorized as follows

Table 2: Constraint Types

Hard Constraints	Constraints which must not be violated at any cost.
Soft Constraints	Constraints which while making a compromised solution of good quality, can be violated

Some examples of hard and soft type of constraints in examination timetabling are as follows.

Table 3: Examples of constraint types

Type of constraint	Examples
Hard	Every examination must not be happen in multiple sittings. Examinations of two subjects having common students should not happen in one time slot. There must be at least one invigilator in the room. There must be enough seats in each time slot for all examinations scheduled in that time slot. Certain examinations must be scheduled at specific time slots or into specific rooms. Examination for each subject should not split over time slots.
Soft	Not more than predefined examinations taking place simultaneously if resources are surplus. 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. Teacher or student preferences. Distance between rooms having same examination should be minimized (when room splitting is allowed.) Examinations should be completed in predefined time slots etc.

As solely depends on one's requirement, in nature these hard and soft constraints are subjective. Finding feasible solution purely depends upon the number and nature of given constraints. Sometimes it becomes impossible to find solution whereas sometimes many 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 students can sit for all the examinations that they are required to do while scheduling all examinations.

1.1.2 No Room Splitting: *a constraint*

According to this constraint one examination should not split over two or more rooms which mean all students of one examination should present in one room only while giving examination. According to this constraint any number of examinations can be scheduled in a room depending on its seating capacity. This type of constraint is included for purpose of ease the work. All work relating to any examination will be in one room only.

1.2 Complexity Analysis of Standard Problems Useful in Timetabling

Here main objective is to find feasible solution, i.e., an assignment of values/resources to all decision variables/examinations such that all constraints satisfied.

We introduce the notions of *input size* and *time complexity*, for computational complexity of combinatorial problems. The count of symbols required to represent the problem determine size of the problem. The time complexity of an algorithm for each input size is the maximum amount of time needed to solve it. We say that an algorithm will solve input of the feasible problem if it returns valid answer. It means, algorithm which solve the problem have to return feasible solution if, and only if, solution of such type exists for it. An algorithm will solve problem if it solves all inputs of the problem.

If, and only if, there exists a function that upper bound the time complexity of an algorithm then it will be called as *polynomial-time algorithm*. Otherwise, algorithm will be called as *super polynomial time algorithm*. The problem may be called as an *easy* problem if it is polynomial algorithm to solve it and *hard* if no polynomial-time algorithm exists. The class P consists of all decision problems which are polynomially solvable. It has been already proven that $P \subseteq NP$. To relate the complexity of two problems we often use the concept of polynomial reducibility.

1.2.1 Polynomial Reducibility

Let A and B are two decision problems of NP category. Then A will be *polynomially reducible* to B if, and only if there exists an method or algorithm which can maps instances *a* of A into instances *b* of B,

- The method/algorithm should be polynomial-time.
- *a* will be the yes instance for A if, and only if *b* is a yes instance for B.

If A is polynomially reducible to B, then we can write A $\propto$ B, and informal way we can say that A is a *special case* of B.

1.2.2 NP-Complete Problems

A problem P will come under this category if, and only if

- It belongs to NP problems category.
- Each problem belongs to NP problems category must be polynomially reducible to it.

1.2.3 NP Hard Problems

A Problem P will come under this category if, and only if each problem belongs to NP problems category must be reducible to it polynomially.

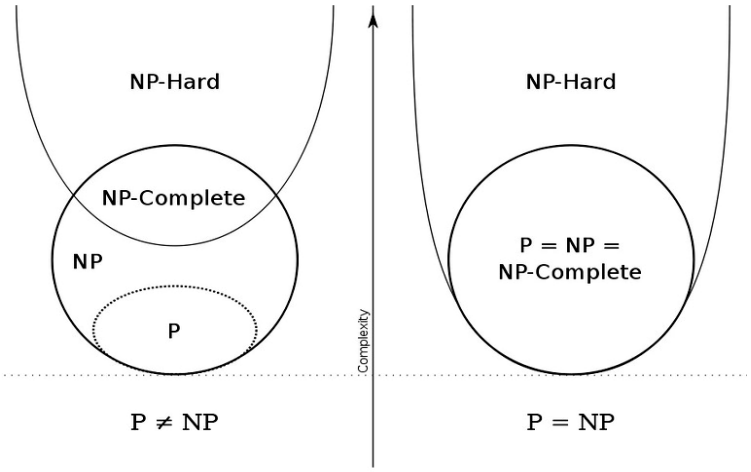


Figure 2: Set notation for P, NPC and NPH

It must be noted here that if any one out of all NP-complete problems is polynomially solvable, then hence due to the reducible property all NP problems will become polynomially solvable. This imply that NP $\subseteq$ P.

1.2.4 P-Class Problems

1.2.4.1 Bipartite Matching Problem

In timetabling problems much of the purposefulness can be seen as assignments of resources to the events. The restrictions like resource can be assigned to at most one meeting/event and every resource must not assign to every meeting/event, are common type of constraints in time tabling. These types of problems very often formulated as bipartite matching problem. In the formulated problem, events and resources represent two vertex sets respectively and edges set corresponds to assignments of resources to the events.

Definition: An instance of bipartite matching problem is a three tuple (A, B, C), where

- A and B are two finite sets of vertices, and
- $C \subseteq A \times B$ is a finite edges set between vertices of A and B.

A matching M will be the subset of edges C i.e. $M \subseteq C$ such that none of any two edges in M incident on same vertex. The objective of bipartite problem is to locate matching of maximum cardinality.

1.2.4.2 Interval Graph Coloring Problem

When we partitioned items belonging to one set into subsets of items such that they are mutually independent then we can see this activity as finding a feasible coloring of all items which may be treated as vertices of the graph and if independent then any two vertices are not connected. In coloring with K colors, out of K colors one will be assigned to each vertex such that adjacent vertices do not get same color. *Chromatic number* χ of the graph is the smallest count of colors by which proper coloring of the graph can be done. Coloring problem for interval graphs can be solved in polynomial time. A graph will be an *interval graph* if each vertex can be thought as a time interval and if two time intervals overlap then an edge will exists between corresponding vertices.

Definition: Let instance of an interval graph is G = (V, E) and objective to find out here is the chromatic number of G.

1.2.5 NP Complete Problems

To prove problem A belongs to NP-complete category, there must exists a problem A' belongs to NPC which is polynomial time reducible to problem A. This is possible if for one problem according to definition already NP-completeness is proven.

1.2.5.1 Satisfiability

Satisfiability problem is the basic NP-complete problem. In this problem, a set consisting of boolean variables within boolean formula and set of clauses. Disjunction of literals is there in each clause. A literal is either boolean or its negation variable. If for each boolean variable having either true or false assignment such that it evaluate every clause to true then the boolean formula will be called as satisfiable.

Definition: Its instance is a pair {A, B}, where

- A is a set of variables
- B is a consisting of clauses on A.

The objective of satisfiable problem is to check that whether there exists any assignment for B which evaluates to true. Many following special cases of SAT are also NP-complete.

1.2.5.2 Three Satisfiability

The instance of 3SAT is a pair, say {A, C} where A is variables set and on A, C is a set of clauses. The condition on each clause $c \in C$ is that they have three literals only. Here the objective is to find whether there exists a truth assignment that evaluates C as true.

1.2.5.3 Not All Equal-3SAT

The instance of NAE3SAT is a pair; say {A, B} where A, B are set of variables and clauses respectively. Here the objective is to find whether there exist assignments to the literals such that not all literals in each clause have same value and all clauses evaluate to true.

1.2.5.4 Graph K-Colorability

The instance of K-colorability is a pair, say {G, K} where

- G is a graph and
- K is a positive integer.

Here the objective is to find G is K-colorable or not i.e., whether vertices after getting one out of K colors, no two adjacent vertex having same color.

1.2.5.5 Bin Packing

Graph coloring can sought as partitioning sets of items into independent subsets of items. Sometimes items are independent of each other, but each may have own size. The problem is known as Bin Packing Problem if we have maximum subsets and bound on total size of each subset.

Definition: The instance of Bin Packing Problem is having four tuple, say {A, *b*, C, D} where

- A is a finite set of items,
- *b* is a vector of size |A|, in which for item a, *b*(*a*) is denoting its size,
- C is denoting capacity of the bin,
- D is a positive integer.

Here the objective is to find whether A can be partitioned into sets A_1,..., A_D such that they must be disjoint and in each bin A_i, the sum of the sizes of items must be equal or less than C.

1.2.5.6 Three-Dimensional Matching

Bipartite matching problem can be extended to matching in three dimensions.

Definition: The instance of 3DM is a four tuple, say {V, X, Y, Z} where

- V, X & Y are having the equal number of elements and they are disjoint sets.
- $Z \subseteq V \times X \times Y$ is a set of triples.

Here objective is to check whether Z is having size |V| matching Z', and two elements in Z' should not agree in any coordinate.

1.3 Graph Coloring

Here in the graph, node represents examination and edge between two nodes exists if both the examinations having common student(s) because of which we can't schedule those two examinations in the same time slot. After minimum graph coloring[2][3] 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.

Chapter 2: Motivation and Related Work

2.1 Exact Methodology

Exact methodology represents a classical search approach that uses direct mathematical procedures. This type of approach aims to find an optimal solution for specific issue. This approach is not always successful, especially when lot of constraints are there.

Integer programming method[4] was used in solving the difficult sub problems in examination time tabling. Firstly problem was divided into *easy* and *difficult* sets as divided in [5]. The difficult sub-problem set then in order to find optimal solution it was solved using an integer programming. As difficult set even having small problem size but add large amount of computational cost. The IP model used was adapted with clique inequalities. Even problem of specific cutting planes was used to reduce the gap to the optimal solution. This IP approach on hybridization with a decomposition approach, gave good results.

In study[6], three-phase hybrid algorithm for capacitated and un-capacitated problems of examination timetabling was presented. Three phases make use of constraint programming, hill climbing and simulated annealing. The constraint programming applied here was similar to that used in [7]. Constraint programming is used for making feasible solutions. Then simulated annealing with the Kempe-chain neighbourhood[8] was used to improve solutions. Then hill climbing method will further improve the solutions.

2.2 Constructive Heuristic Techniques

2.2.1 Graph-based Heuristics

When examination timetabling problem thought as a coloring problem in graphs then examinations denoted by vertices, conflict between two examinations is denoted by edges and each color represent one time-slot in the timetable.

The method in [1] uses graph coloring to create timetable using sequential strategy. In its ordering strategy *difficult* examination was chosen first, while placing examination into time slots. Here commonly used graph coloring heuristics for examination

timetabling are also listed the, i.e. largest weighted degree, largest degree, saturation degree and color degree.

The relation between graph coloring and examination timetabling was first studied in [9] in which it is identified that the count of time-slots needed for scheduling is equal to chromatic number of its graph.

A dynamic ordering strategy, *saturation degree* is effective as compared with other heuristics because it colours the vertex dynamically. It has been applied successfully to examination timetabling in [10][11]. It gives priority to the vertex having least color available to be colored first. Practically, the vertex having the least available color is the most difficult to schedule because it is having smallest saturation degree.

Examination timetables have been constructed[12] on the basis of the *difficulty factor* obtained from heuristics of graph coloring. During its first phase examinations are assigned to time slots by ordering strategy which uses the combination of both largest degree as well as largest enrolment. Then simulated annealing was used in the next phase to improve quality of the obtained timetable solution.

2.2.2 Fuzzy-based Techniques

A fuzzy technique in examination timetabling was implemented[13], which on the basis of examination's difficulties ordered them while combining many sets of two graph coloring heuristics. Out of three graph coloring heuristics (largest enrolment, largest degree and saturation degree) three combinations were used of two heuristics. In the work, this approach was used to represent heuristics knowledge, then their evaluation took place and as an input variable examination weight was constructed. Then based on these input weight values, they were placed in a decreasing order and scheduled in time slots. *Bumped back* is used if infeasible solution occurred. The work showed that good quality solutions were obtained.

In the extended work [14] focus was on the construction phase. The performance was evaluated based on three criteria; penalty cost compared with other constructive methods, count of bump-back strategies required and the processing time of static and dynamic heuristic for each combination.

2.2.3 Decomposition Techniques

Large and complex problems are difficult to solve because of their search space size. Therefore decomposition technique will be an alternative to solve this kind of a problem. Problem in this technique is divided into small-small size sub-problems. These in turn are easy to handle and give high quality solutions.

A memetic algorithm along with decomposition strategy in examination timetabling was implemented[15]. Here also large problem was divided into smaller sub problems. If infeasible timetable produced in early assignment then backtracking and forward checking strategies were used.

An adaptive decomposition approach[5] was studied to construct examination timetables. The problem was grouped into two sets, i.e. *easy* and *difficult.* The difficult set was having difficult examinations that were picked according to the examination's feasibility in a previous iteration. After it on the other side, easy set was reordered to improve the solution quality. Examination comes in overlapping region of *difficult* and *easy* set introduced in set of *boundary examinations*. The study found that the *difficult set* examinations contribute a significant amount of penalty cost irrespective to its size even small. The approach found to be simple and effective when tested on the Toronto benchmark datasets.

2.2.4 Neural Networks

In 2006 Kohonen self-organising neural network approach with graph coloring heuristics was applied to train the input data known as *feature vectors*[16]. At first, difficulty of each examination was calculated using the neural network prior to going for ordering and scheduling the most difficult examination first to a time-slot. The whole network divided all available examinations into three categories as position of scheduling *early*, *middle* and *late* during construction of the timetable. Examinations were assigned to the time slots adaptively as they were ordered. As compared to the single graph colouring heuristic this approach has created feasibility in the solution significantly.

2.3 Meta-heuristic and Improvement Heuristic Techniques

Meta-heuristic approaches has been utilised with great success in the area of timetabling problems. In these techniques search strategies have been employed to improve solution quality after begin with one or more initial solutions.

2.3.1 Local Search based Methods

Such approaches search extensively in space and their aim is to avoid the search from being stuck in local optima. Following are the various methods so far used in examination timetabling.

2.3.1.1 Hill Climbing

The hill climbing method illustrates the incremental changes in quality of solution iteratively. It is straightforward to implement, which makes it popular in implementation of optimization problems.

Local search based algorithms was proposed[17] to solve examination timetabling issues. In the work there were two main objectives: minimization of number of time slots and to maximize timetable quality for a fixed time slots. Started with greedy scheduler examinations were assigned to the time slots w.r.t. the conflict-free requirements. An examination having higher priority, i.e. more conflicts, was selected first for the assignment. At early stage, total time slots were increased because of ensuring that all examinations were scheduled. Then, this approach i.e. hill climbing was applied to refine the quality of timetable as a penalty-decreaser while keeping current number of time-slots constant or with decreasing it until there was no further improvement.

Then *late acceptance strategy* was introduced in hill climbing to solve the timetabling problem[18]. To decide the acceptance of current solution, it considered many objective functions of previous steps as reference. As having many previous values, current solution had many choices for its acceptance or rejection. Accepted current solution was added while last element of the list will be discarded.

Afterwards *late acceptance strategy* was implemented with a hyper-heuristic techniques[19] with a combination of different heuristic selection methods. It was

observed that late acceptance strategy with simple random heuristic selection performed better than the other combinations.

2.3.1.2 Tabu Search

It uses adaptive memories known as *tabu list* for restricting some moves while improving current solutions. Therefore it avoids cycling of previous non improving solutions.

Using four phase system *OTTABU* investigation of short as well as long term memory of tabu search into examination timetabling was done[20]. When both type of memories were used then algorithm worked effectively while long term memory improves the quality of the solution by 34% as compared to when short term memory used alone. Further based on the frequency of moves, long term memory detected active examinations and to prevent cycling of these examinations, their movement was avoided. Active examinations helped in estimation of size of tabu list. Relaxation of tabu list was done to discover new neighbourhood to find better solution during the search.

Extension to the work in [21] has been given after hybridising tabu search with exponential Monte-Carlo procedure as specified in [22]. As its characteristic tabu search is applied to stop non improving moves in solution search. To accept or not the worse solution successive non improving iterations were counted. Results of the implementation demonstrate its success.

2.3.1.3 Simulated Annealing

It works on the concept of annealing material in metallurgy. This algorithm reaches to global optimum by selecting neighbouring solution of current solution and then moving towards it with certain probability which is controlled by monotonically decreasing temperature with successive iteration. The three parameters which direct the search to improve quality of the solution are initial temperature, cooling schedule and ending temperature.

In examination timetabling using time-predefined methods, the application of great deluge algorithm and simulated annealing was done in [23]. User defined parameters like desired solution quality and computational time were incorporated in the study. Quality of the solution was improved by longer searches if exploration of the search space by the

method is more. An additional time predefined algorithm is employed which make sure that it did not converge too quickly.

2.3.2 Population-based Search Methodologies

A collection of starting/initial solutions is called *population* and is used to generate many new distinct solutions. Many a technique used in examination timetabling are as follows.

2.3.2.1 Genetic Algorithms

It works on the principle called *survival of fittest*. Solutions are represented by chromosomes where fitness function determines quality of the solution and accordingly new generations of solutions will evolve using genetic operators.

Grouping strategies of genetic algorithms was applied to graph coloring and examination timetabling in [24]. It uses tournament selection and set of examinations came under same time slot was taken as group. Mutation operator used in the work swap two groups of examination chosen randomly. It was found in the algorithm that representing individuals is very important.

Real world examination timetabling problems were also solved using genetic algorithm[25][26]. In [25] highly constrained problem mapped into constraint satisfaction problem which was solved using genetic algorithm while in [26] examination timetabling problem in a training center was solved by genetic algorithm. The evaluation of relationship between the examinations was done by using an activity-on-arrow network. Both of the implemented approaches were successful in the institutions.

A hybrid bi-objective evolutionary algorithm[27] was investigated for un-capacitated examination timetabling problem with tabu search as a local search operator. Two objectives were minimization of timetable length and getting feasible timetable of good quality. At starting random initialization was done in which even infeasible timetables are also generated. Tabu search was used to minimize violations in the constraints. Simple variable neighbourhood descent was used to increase quality of the solution. In the approach non-dominated timetables were produced using different lengths of the timetable.

For capacitated examination timetabling a multi-objective evolutionary algorithm was demonstrated in [28]. Without knowing timetable length, timetable could be generated easily. Introducing variable-length chromosome, length of the examination session can be changed flexibly, while to diversify examinations, day-exchange crossover was performed. The results of this approach showed that feasible solution could easily produce and which was better than many well known approaches on benchmark datasets.

2.3.2.2 Memetic Algorithms

This category of algorithm is a hybridisation of evolutionary algorithms with local search methods which is used to improve quality of the solution.

Evolutionary algorithm combined with local search method to study examination timetabling in [29]. In timetable, to assign single and groups of examinations a combination of light and heavy mutation is used in the algorithm. After mutation to make better the solution quality, hill climbing method is applied in the next phase. The evaluation function penalised heavily the unscheduled examinations. The computational cost increased because of the hill climbing method but still this technique showed success.

In study[15] memetic algorithm used in [29] was implemented with decomposition strategy for examination scheduling problem. Large problem are split into number of small components. It works using multi stage approach and before approaching to next stage/component all examinations must scheduled at one go. Three graph coloring heuristics i.e color, largest and saturation degree were hybridised and among these three, saturation degree showed improvement. While testing on four benchmark datasets it was observed that processing time reduced by significant amount and quality of the solution was improved. To check if infeasible timetable generated in the early assignments, backtracking and forward checking strategy were used.

In examination timetabling problem to select best hill climber in memetic algorithm a hyper heuristic method called as *hyper hill climber* [30] was proposed. Here three types of hill climbers were introduced, i.e. adaptive, self adaptive and deterministic which were evaluated in different ordering within the memetic algorithm. After tested on

Toronto benchmark datasets approach got promising results. It is found that hyperhill climber with implementation of single hill climber performed best at any one time.

2.3.2.3 Ant Algorithms

Marco Dorigo introduced the idea how real ants foraging for food. After getting food when searching for the shortest way to their nest, ants produce pheromone trails. The path having higher pheromone level indicates the higher quantity and quality of food brought back. This also becomes guidance to other ants.

An ant algorithm was used to solve examination timetabling problem in which in order to find constructive heuristic and pheromone trail, max-min ant systems with two randomised strategies was included[31]. In the implementation various factors were set like weighting factors, evaporation rate and number of cycles to make algorithm work efficiently. When tested on the Toronto benchmark datasets approach obtained encouraging results.

This approach is also used in [32][33]. In earlier using tabu search, simulated annealing, ant colony system and genetic algorithms within similar framework, ten random generated datasets for examination timetabling problem were experimented. Initial solution for ant colony system was created considering heuristic figures while for others it was created randomly. Comparison showed ability of ant colony to make out good quality of solutions but for improving the solution it was found that tabu search was more effective than ant colony system. In the later study hybridisation of ant colony system was more examined and it was found that tabu search and sequential ant colony system comes out to be best for the tested problem. Solution converges in the search because of the effect of pheromones that is why tabu search was applied to disrupt the search.

Examination timetabling problem using ant colony algorithm and graph coloring has been solved in [34]. By considering clash free scheduling as a hard constraint, the motive was to get least number of time slots. Two variant based on two graph coloring heuristics i.e. recursive largest first and saturation degree was proposed. Also two statistical measurements were used to find performance of the combinations introduced.

2.4 Hyper-Heuristic

In timetabling problem recently hyper-heuristic which is a search methodology got much attention. It is a self regulating procedure to generate or select heuristics to solve hard computational search problems

2.4.1 Heuristic Selection Methodologies

A heuristic selection methodology includes a selection of one of the existing heuristics in the hyper-heuristic framework. The most common constructive heuristics used in the examination timetabling within hyper-heuristic approaches are graph coloring heuristics and moving strategies. Many heuristic selection methodologies used in the literature are simple random, greedy search, random descent, choice function, tabu search, reinforcement learning and the Monte-Carlo procedure.

In the studies[35][36] tabu search hyper-heuristic framework was implemented for examination timetabling problem with Toronto benchmark datasets at the University Technology MARA. As a low level heuristics many graph coloring heuristics with moving strategies have been considered in the approach. Due to low level heuristics executes as strategy exploration in solution space is allowed. Whenever low level heuristic could not improve the quality of the solution then they were stored in tabu list of fixed length. The solutions got from implementation were better in at least 80% of cases in quality of the solution when they were compared with timetables generated manually.

For constructing examination timetables with hyper heuristic framework hybrid graph coloring heuristics have been proposed in [37]. Saturation and largest degree were employed as a two low level graph coloring heuristics and as a perturbation heuristic tabu search is used within hyper-heuristic. Case based reasoning was hybridised with tabu search hyper-heuristic and based on knowledge of appropriate heuristic graph coloring heuristic was chosen. Comparable results have been produced when this approach was tested on Toronto benchmark as well as on random data sets.

In [38] a multi stage hyper heuristic in which graph coloring heuristic have been permutated into two stages, investigated. Based on learning mechanism after applied low level graph heuristics solutions were obtained and to modify them high level heuristic was applied indirectly. As a high level heuristic tabu search was then

applied. To construct solutions of this timetabling problem many permutations of graph heuristic have been employed. It is found that solution quality would increase significantly as the number of low level heuristics increases. However computational time also increases as the size of search space grows up. This approach produces good results when tested on benchmark timetabling problem.

Based on the performance of low level heuristics, a heuristic can be selected. To construct examination timetabling problem and solving graph coloring problem an adaptive heuristic hybridisation was introduced[39]. Random iterative graph based hyper heuristic was the name of the approach which using different heuristic sequences used to construct various qualities of the solution. Various heuristic sequences were used like largest weighted, saturation, largest and largest enrolment degree. Sequence of these heuristics was chosen randomly while constructing solution and then it was analysed in terms of feasibility. Discarding infeasible solution, only feasible solution carried further for analysis. To produce good quality of solution at early stage of solution construction, it was observed that saturation degree and largest weighted degree heuristics have been hybridise. How automatically graph coloring heuristics have been hybridised was also analysed in the study.

2.4.2 Heuristic Generation Methodologies

In 2007 using genetic programming low level heuristics have been generated for un-capacitated examination timetabling problem[40]. Using genetic operators number of low level heuristic sequences was evolved which was used to construct examination timetables. The approach was able to produce feasible timetable when tested on Toronto benchmark datasets for all tested problems.

Then extension to the above study was done[41] in which based on scheduling difficulty a function for generating combination of low level heuristics was evolved by genetic programming. Combination of both was used hierarchically to construct examination timetables.

2.5 Case-based Reasoning

It is a knowledge-based system which collects information from previous problems and stores their solutions in the case base. When solving current problems from the case base it retrieved the most similar case using a similarity measure.

CBR was investigated as heuristic selection where many known heuristics were stored in the case base that worked effectively on timetabling problems and used to solve current problems[42]. Largest degree with tournament selection, largest degree first, saturation degree, color degree and hill climbing heuristic were used as a low level heuristics. In the first of two stages approach used to construct CBR system, using specific heuristics problem learning was involved and for the current problem best heuristics found in the case base were used. While in the next stage source case has been revised by removing unwanted heuristics to avoid confusions during retrieval process. In the study it is found that by choosing right heuristic to be used CBR could work at general level in solving timetabling problems.

In the implemented CBR approach[43] before proceeding to the next phase which employed great deluge algorithm suitable graph coloring heuristic for initialization strategy have been decided. A similarity measure of two different graph representations has been proposed based on fuzzy set. When tested on Toronto benchmark datasets approach obtained many best results and hence proved to be successful.

2.6 Multi-criteria and Multi-objective Techniques

Real world timetabling problems comes under this category most of the times. To make decisions several criteria need to be considered and approach to solve it must be able to handle these criteria simultaneously. In many of the approaches simple linear combination of the multiple objectives with weighted aggregating functions has been done in order to obtain solutions but it is desirable to put many compromise solutions to the decision maker which can make decision by fulfilling various requirements. Pareto optimisation technique was taken to perform this strategy for multi-objective problems. In multi-objective timetabling problems for Pareto optimisation evolutionary algorithm was reported as a method.

Evolutionary algorithms showed high rate of success when used to solve multi objective of real world problems. To solve examination timetabling problems an application multi-objective evolutionary algorithm have been investigated at University of Algarve[44]. To evaluate each objective of the problem this approach used direct representation and Pareto ranking of the population. In another study[28], for capacitated examination timetabling problems a multi-objective evolutionary

algorithm have been demonstrated. This approach tackled the problem while distributing the students in the timetable and by minimising the timetable period.

In the study[45], a fuzzy evaluation function which involved multiple decision criteria have been proposed for examination timetabling problems. Here in addition to most common objective i.e. average penalty per student, to evaluate quality of the timetable solutions highest penalty imposed on any student also considered. This study shown that on multiple decision criteria problems fuzzy reasoning could be applied successfully.

Chapter 3: Problem Statement

3.1 Existing Method

In existing method[3], *No Room Splitting* is taken as a hard constraint and examination scheduling is combined with room allocation algorithm. Mainly because of the course timetabling it was assumed that largest examination can fit into the largest room. Started with examination which was having least count of students then continuing through list sorted in increasing order of count of students in each examination and assuming that no other examinations was scheduled, and putting examination in smallest room in which it will fit. If currently in the room, students scheduled are more than the available seats, then in the next bigger room the examination which is having least count of students was displaced so that for the current room the size of the remaining examinations will be more but less than its seating capacity. Then this entire process will be repeated for next bigger room and so on. Displaced examinations will be put into some extra list of examinations that cannot be scheduled in the current iteration, if there will be no more rooms left. At last output will be got in which examinations and allotted rooms and extra list of examinations which do not get fit into any room or left unscheduled will be there. 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.

3.2 No Room Splitting: *a constraint*

In most of the work done till now in field of examination timetabling, "*No Room Splitting"* is taken as a *hard* constraint. In course timetabling students having same subject should be in same room while having lecture or they may split across different rooms according to their sections if shortage of seats is there. 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 it is most probable that examination will split across multiple rooms. If sections are not there then although examinations can happen only in one

room but why don't we use vacant seats if there, as left in the case of existing method and split examination across different rooms while examination etiquette remain same for all the students giving same or different examinations in same room. For this reason it can be omitted even as a soft constraint. By doing *Room Splitting* efficiency in terms of minimization of time slots can be achieved and hence in resource utilization.

3.3 Problem Statement

This work aims at improving the utilization of various resources like number of time slots, number of invigilators, rooms etc used in examination process by decreasing total time slots used in which all examinations will be scheduled after taking *No Room Splitting* not even soft constraint. The whole work covered by the novice algorithmic proof.

3.4 Methodology Used for Solving the Problem

The following methodology and steps have been used in achieving the objective of this thesis. The scope of this work is very wide can go to various optimizations.

- Input sets of examinations, students, rooms, time slots and various relations duly stated in ***Section 4.1*.**
- Apply Final Independent Sets Algorithm.[***Section 4.2.1***]
- Apply Graph Coloring Algorithm.[***Section 4.2.2***]
- Apply Remove Examination Algorithm *if required*. [***Section 4.2.3***]
- Apply Adjust Examination Algorithm *if required*. [***Section 4.2.4***]
- Apply Time Slots Assignment Algorithm on final independent sets of examinations got from previous steps. [***Section 4.2.5***]
- Apply Rooms/Seats Allotment Algorithm. [***Section 4.2.6***]

Chapter 4: Proposed Method for Examination Timetabling

4.1 Functional Model of Examination Timetabling

In functional model of examination timetabling, notation for sets, relations and then, all inputs are as following.

Set is denoted by single 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 elements. Relation is also a set which specifically shows association between two or more entities which may be set or elements. Here 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 as well as range upper case letter denotes set while lower case letter denotes particular element but according to how they are defined in the relation. Let S_i denotes i^{th} element of the set S. For e.g. let R = $\{r_1, r_2, r_3\}$ then $R_1 = r_1$ where R_1 is a path/formula to access first element of R and r_1 is the actual first element of R.

Inputs:

- *Time Input:*
  - T = {t | t denotes time slot}
  - D = {d | d denotes day of week}

- *Student Input:*
  - S = {s | s denotes student}

- *Examination Input:*
  - E = {e | e denotes examination}

- *Room Input:*
  - R = {r | r denote room}
  - C = {c | c denotes seat}

Relations / Functions:

- $A = \{a = (t, d) \mid t \in T \wedge d \in D\} \subset T \times D$
- $B = \{b = (s, V) \mid s \in S \wedge V \subseteq E\} \subset S \times 2^E$
- $F = \{f = (e, H) \mid e \in E \wedge H \subseteq S\} \subset E \times 2^S$
- $K = \{k = (r, M) \mid r \in R \wedge M \subseteq C\} \subset R \times 2^C$

- $U = \{u = (c, r) \mid c \in C \wedge r \in R\} \subset C \times R$
- *Graph Input:*
  - $G = \{g = (e, N)$ if edge exists between e & elements of $N \mid e \in E \ \& \ N \subseteq E\} \subset E \times 2^E$
- *After minimum coloring*
  - $P = \{p \subseteq E \mid (\mid p_i \cap p_j \mid = 0 \mid i \neq j) \wedge (\bigcup_{i=1}^{totalcolors} p_i = E)$

4.2 Proposed Approach

As intake of various courses may increase or more students can choose same common subject but seats in all rooms remains same until new rooms are constructed. Because of which generally we do make more than one section of a class or for the subject. But very often examination of same subject of all the sections will be same and must come under same time slot. On these conditions according to the above mentioned existing approach[3] even examination will not schedule.

Due to *no room splitting* taken as a hard constraint it may possible that some seats may left vacant if examinations scheduled in the room are having less total strength of 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 proposed approach these limitations were removed. Let the total seats of all the rooms are equal or greater than the largest examination. Here its algorithmic approach is divided into five parts:

- Final Independent Sets
- Graph Coloring
- Remove Examinations
- Adjust Examinations
- Time slots Assignment
- Rooms/Seats Allotment

4.2.1 Final Independent Sets

This process will run until each and every examination does not come under one of the final independent set/time slot.

Algorithm 4.2.1: *Final Independent Sets*

1. Let L be an empty set.
2. Let set Ḛ is a copy of set E.
3. **while** (|Ḛ| ≠ 0) **do**
   Execute Graph Coloring Algorithm on examinations of Ḛ.
   Execute algorithm 4.2.3
   Execute algorithm 4.2.4
   $L = L \cup P$
   for i =1 **to** | P | **do**
   for j=1 **to** | P_i | **do**
   Ḛ = Ḛ \ $\{P_{i_j}\}$
   end for
   end for
4. **end while**

4.2.2 Graph Coloring

The algorithm used in work is based on Dutton and Brigham's [2] algorithm. In the algorithm we find out triples such that first and third vertex in the triple should not be same or adjacent to each other while both must adjacent to the middle or second vertex of the triple. Then first and third vertex will merge to give them same color. Then after merge edges also removed accordingly. This whole process repeat until no choice left. Hence in this way graph coloring will be performed in the proposed work.

4.2.3 Remove Examinations

If we have fewer seats in all 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 sorted independent set of examinations in increasing order of students in the examinations from starting until difference vanishes while examination having maximum degree in that independent set should not removed in the process until it becomes mandatory to do so. This is because in the next step if possible removed examinations will be adjust 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 will be difficult to adjust and hence it may increase total time slots resulting in more resource utilization.

Algorithm 4.2.3: *Remove Examinations*

1. TSeats = $\sum_{i=1}^{|R|} | K_C(R_i) |$
2. **for** i = 1 **to** | P | **do**
   ∀j sort all P_{i_j}'s of P_i's in increasing order of $F_S\left(P_{i_j}\right)$ where j ∈ [1, $|P_i|$]
   end for
3. Let X = NULL.
4. **for** i = 1 **to** | P | **do**
   TStudents = $\sum_{j=1}^{|P_i|} |F_S\left(P_{i_j}\right)|$
   diff = TStudents – TSeats
   max = $| G_E(P_{i_1}) |$
   for j = 2 **to** $| P_i |$ **do**
   if max < $| G_E(P_{i_j}) |$ **then**
   max = $| G_E(P_{i_j}) |$
   end if
   end for
   if diff > 0 **then**
   for j = 1 **to** $| P_i |$ **do**
   if $| G_E(P_{i_j}) | \neq$ max **then**
   diff = diff – $| F_S\left(P_{i_j}\right) |$
   X = X ∪ P_{i_j}
   $P_i = P_i \setminus \{P_{i_j}\}$
   end if
   end for
   for j = 1 **to** $| P_i |$ **do**
   diff = diff – $| F_S\left(P_{i_j}\right) |$
   X = X ∪ P_{i_j}
   $P_i = P_i \setminus \{P_{i_j}\}$
   end for
   end if
   end for
5. **for** i = 1 **to** | P | **do**
   if $| P_i |$ = 0 **then**
   P = P \ { P_i }
   end if
   end for

4.2.4 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 adjust the removed examinations from all independent sets contained in set X into existing independent sets, if possible to minimize total time slots.

Algorithm 4.2.4: *Adjust Examinations*

1. if | X | ≠ 0 **then**
TSeats = $\sum_{i=1}^{|R|} | K_C(R_i) |$
∀i sort P_i 's of P in decreasing order of $\sum_{j=1}^{|P_i|} | F_S\left(P_{i_j}\right) |$ where i ∈ [1, | P |]
for i = 1 **to** | P | **do**
TStudents = $\sum_{j=1}^{|P_i|} | F_S\left(P_{i_j}\right) |$
LSeats = TSeats – TStudents
if LSeats > 0 **then**
Let Y be an empty set.
for j = 1 **to** | P_i | **do**
Y = Y ∪ $G_E(P_{i_j})$
end for
Let Z be an empty set.
Z = X \ Y
∀j sort Z_j 's of Z in decreasing order of | $F_S(Z_j)$ | where j ∈ [1, | Z |]
for j=1 **to** | Z | **while** LSeats > 0 **do**
if | $F_S(Z_j)$ | ≤ LSeats **then**
$P_i = P_i$ U Z_j
LSeats = LSeats – | $F_S(Z_j)$ |
X ← X \ Z_j
end if
end for
end if
end loop
2. end if

4.2.5 Time Slots Assignment

Let T is a set of sorted time slots in chronological order of date and time as well. Until any specific conditions mentioned, time slots will be assigned to the independent sets and hence to the examinations as follows.

Algorithm 4.2.5: *Time Slots Assignment*

1. if |T| ≥ |L| **then**
Let Q, O are 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 for
end for
else
Print "Insufficient time slots to schedule all independent sets/examinations."
2. end if

Here 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.

4.2.6 Rooms/Seats Allotment

Until any specific conditions mentioned, seats to the students or rooms to the examinations belonging to independent sets can be assigned as follows.

Algorithm 4.2.6: *Rooms/Seats Allotment*

---

1. Let initially Ă, Ď, Ĕ, Č, Ř are empty sets.
2. **for** i=1 **to** |R| **do**
  Ă= Ă ∪ $K_C(R_i)$
3. **end for**
4. **for** i=1 **to** | L | **do**
  Let m=1.
  for j=1 **to** | L_i | **do**
    Ď= $F_S(L_{i_j})$
    Ř= { $U_r(Ă_m)$ }
    for k=1 **to** | Ď | **do**
      if Ř ∩ $U_r(Ă_m)$ == ϕ **then**
        Ř=Ř ∪ { $U_r(Ă_m)$ }
      end if
      Ĕ=Ĕ∪{$Ď_k$,L_{i_j} ,$U_r(Ă_m)$, $Ă_m$}
      m← m+1
    end for
    Č= Č ∪ {L_{i_j}, Ř}
  end for
5. **end for**

Here Ĕ comes out as a set whose single element contains four values. First value is a particular student, second will tell his/her particular examination, third and fourth will tell in which room and particular seat examination of that student is scheduled. Set Č contains all examinations and their respective rooms into which they have been scheduled.

Chapter 5: Testing and Results

Let us illustrates proposed method by using following random dataset. Let set R = {r_1, r_2, r_3} and respective seating capacity i.e.| $K_C(r_i)$ | are 30, 60, 110. Let set E = {e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}, e_{12}} and respective strength of students | $F_S(e_i)$ | are 10, 20, 50, 100, 90, 100, 30, 70, 80, 60, 60, 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 students for examination e_4. Let the dependencies among the examinations shown through following graph.

Input Graph:

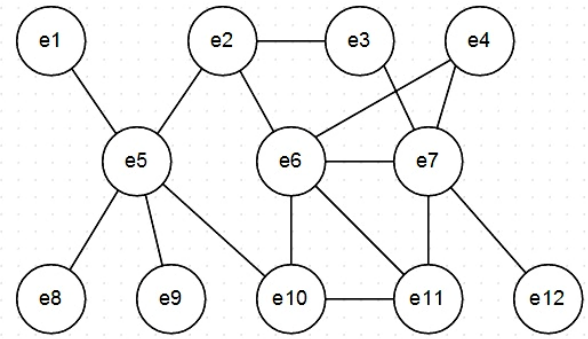


Figure 3: Initial graph

After applying Graph coloring algorithm [2] the structure will be as follows.

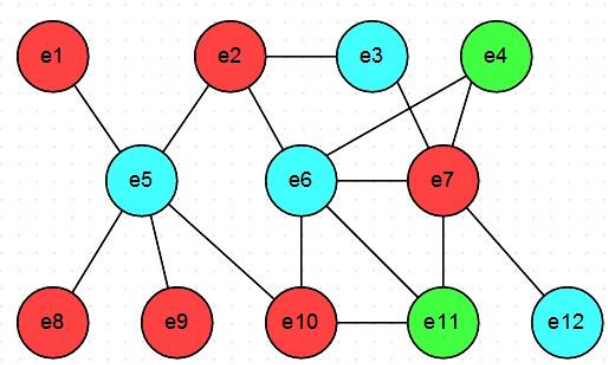


Figure 4: Colored graph

5.1 Using Existing Method[3]

In each iteration let say red color examinations will be inputted to existing method. Then in first iteration examinations {e_1, e_2, e_7, e_8, e_9, e_{10}} will be inputted to the existing method.

Table 4: Iteration 1 of existing method

Room Size	**Positioning of examination at each stage**					
Unscheduled Exams						e_8^{70}
110				$e_1^{10}\ e_2^{20}$	$e_1^{10}\ e_2^{20}\ e_8^{70}$	$e_1^{10}\ e_2^{20}\ e_9^{80}$
60			$e_1^{10}\ e_2^{20}$	e_{10}^{60}	e_{10}^{60}	e_{10}^{60}
30	e_1^{10}	$e_1^{10}\ e_2^{20}$	e_7^{30}	e_7^{30}	e_7^{30}	e_7^{30}

So $\{e_1, e_2, e_7, e_9, e_{10}\}$ will be scheduled in one time slot. Now again doing graph coloring on remaining examinations.

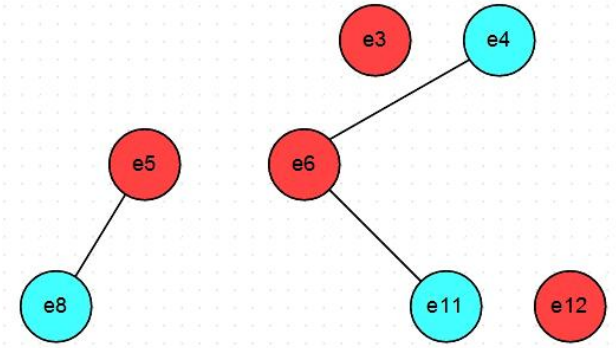


Figure 5: Colored graph of iteration 2 of existing method

In second iteration examinations $\{e_3, e_5, e_6, e_{12}\}$ will be inputted to the existing method.

Table 5: Iteration 2 of existing method

Room Size	**Positioning of examination at each stage**			
Unscheduled Exams			e_{12}^{80}	$e_{12}^{80}\ e_5^{90}$
110		e_{12}^{80}	e_5^{90}	e_6^{100}
60	e_3^{50}	e_3^{50}	e_3^{50}	e_3^{50}
30				

So $\{e_3, e_6\}$ will be scheduled in one time slot. Now again doing graph coloring on remaining examinations.

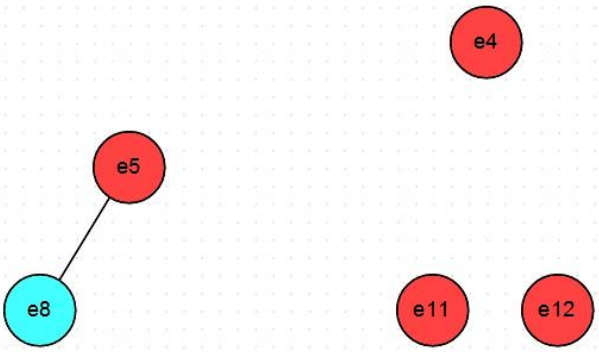


Figure 6: Colored graph of iteration 3 of existing method

In third iteration examinations $\{e_4, e_5, e_{11}, e_{12}\}$ will be inputted to the existing method.

Table 6: Iteration 3 of existing method

Room Size	**Positioning of examination at each stage**			
Unscheduled Exams			e_{12}^{80}	e_{12}^{80} e_5^{90}
110		e_{12}^{80}	e_5^{90}	e_4^{100}
60	e_{11}^{60}	e_{11}^{60}	e_{11}^{60}	e_{11}^{60}
30				

So $\{e_4, e_{11}\}$ will be scheduled in one time slot. Now again doing graph coloring on remaining examinations.

Figure 7: Colored graph of iteration 4 of existing method

In fourth iteration examinations $\{e_5, e_{12}\}$ will be inputted to the existing method.

Table 7: Iteration 4 of existing method

Room Size	**Positioning of examination at each stage**	
Unscheduled Exams		e_{12}^{80}
110	e_{12}^{80}	e_5^{90}
60		
30		

So $\{e_5\}$ will be scheduled in one time slot. Now again doing graph coloring on remaining examinations

Figure 8: Colored graph of iteration 5 of existing method

In fifth iteration examinations $\{e_8, e_{12}\}$ will be inputted to the existing method.

Table 8: Iteration 5 of existing method

Room Size	Positioning of examination at each stage	
Unscheduled Exams		e_8^{70}
110	e_8^{70}	e_{12}^{80}
60		
30		

So $\{e_{12}\}$ will be scheduled in one time slot. Now again doing graph coloring on remaining examinations.

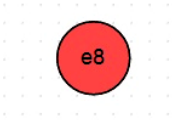


Figure 9: Colored graph of iteration 6 of existing method

In fifth iteration examination $\{e_8\}$ will be inputted to the existing method.

Table 9: Iteration 6 of existing method

Room Size	Positioning of examination at each stage
Unscheduled Exams	
110	e_8^{70}
60	
30	

So $\{e_8\}$ will be scheduled in one time slot. Let the final schedule by existing method is as follows.

Table 10: Schedule by existing method

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

So, using existing method total six time slots will be used for timetabling all given examinations.

5.2 Using Proposed Method

Initially L is an empty set. After execution of graph coloring algorithm in iteration 1 of algorithm 4.2.1, graph will look like as shown in figure 2. Contents of set P will be as follows.

Table 11: Set P initially in Iteration 1 of Algorithm 4.2.1

P	$\{p_1, p_2, p_3\}$	$\{\{e_1^{10}, e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80}, e_{10}^{60}\}, \{e_3^{50}, e_5^{90}, e_6^{100}, e_{12}^{80}\}, \{e_4^{100}, e_{11}^{60}\}\}$

After execution of algorithm 4.2.3 i.e. Remove Examinations, contents of Set P and Set X will be as follows.

Table 12: Set P & X after algorithm 4.2.3 in Iteration 1 of Algorithm 4.2.1

P	$\{p_1, p_2, p_3\}$	$\{\{e_7^{30}, e_8^{70}, e_9^{80}\}, \{e_5^{90}, e_6^{100}\}, \{e_4^{100}, e_{11}^{60}\}\}$
X		$\{e_1^{10}, e_2^{20}, e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$

Examinations e_7, e_5 and e_6 didn't remove from their respective sets because they have maximum degree in them. They are retained in their sets as if they are removed then they will be harder to adjust into other independent sets because of their dependencies present in them. Set P will be sorted on total students in each element of it as a step given in algorithm 4.2.4 which then become $\{p_2, p_1, p_3\}$. Then for independent set p_2, set Y and Z will be as follows.

Table 13: Set Y & Z for independent set p_2

Y	$\{e_1, e_2, e_4, e_7, e_8, e_9, e_{10}, e_{11}\}$
Z	$\{e_3, e_{12}\}$

After adjusting examinations from set Z into independent set p_2, independent set p_2 & set X will be as follows.

Table 14: Independent set p_2 & set X after adjustment

p_2	$\{e_5^{90}, e_6^{100}\}$
X	$\{e_1^{10}, e_2^{20}, e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$

For independent set p_1, set Y and Z will be as follows.

Table 15: Set Y & Z for independent set p_1

Y	$\{e_3, e_4, e_5, e_6, e_{11}, e_{12}\}$
Z	$\{e_1, e_2, e_{10}\}$

After adjusting examinations from set Z into independent set p_1, independent set p_1 & set X will be as follows.

Table 16: Independent set p_1 & set X after adjustment

p_1	$\{ e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80}\}$
X	$\{e_1^{10}, e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$

For independent set p_3, set Y and Z will be as follows.

Table 17: Set Y & Z for independent set p_3

Y	$\{e_6, e_7, e_{10}\}$
Z	$\{e_1, e_3, e_{12}\}$

After adjusting examinations from set Z into independent set p_3, independent set p_3 & set X will be as follows.

Table 18: Independent set p_3 & set X after adjustment

p_3	$\{ e_1^{10}, e_4^{100}, e_{11}^{60}\}$
X	$\{e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$

After execution of algorithm 4.2.4 i.e. Adjust Examinations, final contents of Set P & X at the end of iteration 1 will be as follows.

Table 19: Set P & X after algorithm 4.2.4 in Iteration 1 of algorithm 4.2.1

P	$\{p_1, p_2, p_3\}$	$\{\{ e_5^{90}, e_6^{100}\}, \{ e_2^{20}, e_7^{30}, e_8^{70}, e_9^{80}\}, \{ e_1^{10}, e_4^{100}, e_{11}^{60}\}\}$
X		$\{ e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$

At the end of iteration 1 of algorithm 4.2.1, contents of set L will be as follows.

Table 20: Set L after Iteration 1 of algorithm 4.2.1

L	$\{\{e_5, e_6\}, \{e_2, e_7, e_8, e_9\}, \{e_1, e_4, e_{11}\}\}$

In iteration 2 of algorithm 4.2.1 (i.e. Final independent sets) on Set Ḛ after graph coloring algorithm, colored graph will look like as follows.

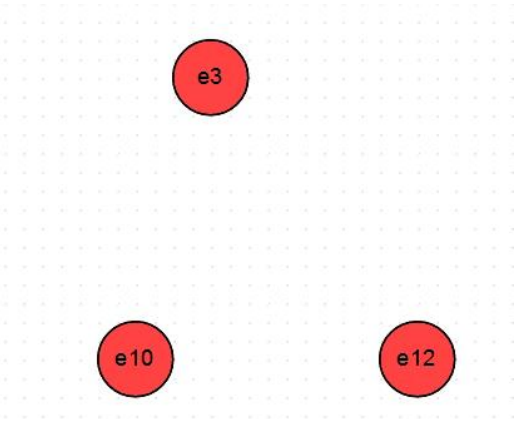


Figure 10: Colored graph in iteration 2 of algorithm 4.2.1

Contents of set P will be as follows.

Table 21: Set P initially in Iteration 2 of Algorithm 4.2.1

P	$\{p_1\}$	$\{e_3^{50}, e_{10}^{60}, e_{12}^{80}\}$

Final contents of set L after iteration 2 of algorithm 4.2.1 will be as follows.

Table 22: Set L after Iteration 2 of algorithm 4.2.1

L	$\{\{e_5, e_6\}, \{e_2, e_7, e_8, e_9\}, \{e_1, e_4, e_{11}\}, \{e_3, e_{10}, e_{12}\}\}$

So, final schedule by proposed method will be as follows.

Table 23: Schedule by proposed method

Time Slot	**Independent Set**	**Examinations**	**Total Students**
$\mathbf{t_1}$	l_1	$\{e_5, e_6\}$	190
$\mathbf{t_2}$	l_2	$\{e_2, e_7, e_8, e_9\}$	200
$\mathbf{t_3}$	l_3	$\{e_1, e_4, e_{11}\}$	170
$\mathbf{t_4}$	l_4	$\{e_3, e_{10}, e_{12}\}$	190

So using proposed method, total four time slots will be used for timetabling all the given examinations.

Chapter 6: Conclusion and Future Scope

It is concluded here that instead of doing graph coloring on left examinations in each iteration, only one time graph coloring then adjusting unscheduled examinations into different independent sets considering all dependencies among them will decrease space and time complexity significantly and will make things simple. In existing method[3], 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 to those as a limitation existing method is unable to schedule. Comparing results from table 10 and table 23 of tested problem, the proposed algorithm infers that the work which was completed in six time slots under existing methodology will now complete in only four time slots using proposed approach. So use of partially vacant seats in efficient manner and removing *no room splitting* even from soft constraints in the proposed approach results in the reduction of time slots and hence helps in better resource utilization while keeping all the examination etiquettes same. In the future various optimizations in the algorithms given here can be done. Reducing space and time complexity may be proposed.

References

[1] E.K. Burke, J. Kingston, and D. de Werra, *Handbook of graph theory, Chapter Application to timetabling*.: Chapman Hall/CRC Press, 2004.

[2] R.D. Dutton and R.C. Brigham, "A New Graph Coloring Algorithm," *The computer Journal*, vol. 24, pp. 85-86, 1981.

[3] E.K. Burke, D.G. Elliman, and R. Weare, "A university timetabling system based on graph colouring and constraint manipulation," *Journal of Research on Computing in Education*, vol. 27, pp. 1-1, 1994.

[4] R. Qu, F. He, and E. K. Burke, "Hybridizing integer programming models with an adaptive decomposition approach for exam timetabling problems," *The 4th Multidisciplinary International Scheduling: Theory and Applications*, pp. 435–446, 2009.

[5] R. Qu and E. K. Burke, "Adaptive decomposition and construction for examination timetabling problems," *Proceedings of the 3rd Multidisciplinary International Scheduling: Theory and Applications*, pp. 418–425, 2007.

[6] L. T. Merlot, N. Boland, B. D. Hughes, and P. J. Stuckey, "A hybrid algorithm for the examination timetabling problem," in *Practice and theory of automated timetabling IV*. Springer, 2003, pp. 207–231.

[7] P. Boizumault, Y. Delon, and L. Peridy, "Constraint logic programming for examination timetabling," *The Journal of Logic Programming*, vol. 26, no. 2, pp. 217–233, 1996.

[8] J. M. Thompson and K. A. Dowsland, "Variants of simulated annealing for the examination timetabling problem," *Annals of Operations research*, vol. 63, no. 1, pp. 105–128, 1996.

[9] D. Wood, "A technique for colouring a graph applicable to large scale timetabling problems," *The Computer Journal*, vol. 12, no. 4, pp. 317–319, 1969.

[10] S. A. Rahman, A. Bargiela, E. K. Burke, E. Ozcan, and B. McCollum,"Construction of examination timetables based on ordering heuristics," in *Computer and Information Sciences, 2009. ISCIS 2009. 24th International Symposium on*. IEEE, 2009, pp. 680–685.

[11] E. K. Burke and J. P. Newall, "Solving examination timetabling problems through adaption of heuristic orderings," *Annals of operations Research*, vol. 129, no. 1-4, pp. 107–134, 2004.

[12] D. Johnson, "Timetabling university examinations," *Journal of the Operational Research Society*, pp. 39–47, 1990.

[13] H. Asmuni, E. K. Burke, J. M. Garibaldi, and B. McCollum, "Fuzzy multiple heuristic orderings for examination timetabling," in *Practice and Theory of Automated Timetabling V*. Springer, 2005, pp. 334–353.

[14] H. Asmuni, E. K. Burke, J. M. Garibaldi, B. McCollum, and A. J. Parkes, "An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables," *Computers & Operations Research*, vol. 36, no. 4, pp. 981–1001, 2009.

[15] E. K. Burke and J. P. Newall, "A multistage evolutionary algorithm for the timetable problem," *Evolutionary Computation, IEEE Transactions on*, vol. 3, no. 1, pp. 63–74, 1999.

[16] P. H. Corr, B. McCollum, M. McGreevy, and P. McMullan, "A new neural network based construction heuristic for the examination timetabling problem," in *Parallel Problem Solving from Nature-PPSN IX*. Springer, 2006, pp. 392–401.

[17] M. Caramia, P. Dell'Olmo, and G. F. Italiano, "Novel local-search-based approaches to university examination timetabling," *INFORMS Journal on Computing*, vol. 20, no. 1, pp. 86–99, 2008.

[18] E. K. Burke and Y. Bykov, "A late acceptance strategy in hill-climbing for exam timetabling problems," in *PATAT 2008 Conference, Montreal, Canada*, 2008.

[19] E. Ozcan, Y. Bykov, M. Birben, and E. K. Burke, "Examination timetabling using late acceptance hyper-heuristics," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on.* IEEE, 2009, pp.997–1004.

[20] G. M. White and B. S. Xie, "Examination timetables and tabu search with longer-term memory," in *Practice and Theory of Automated Timetabling III*. Springer, 2001, pp. 85–103.

[21] M. Ayob and G. Kendall, "A monte carlo hyper-heuristic to optimise component placement sequencing for multi head placement machine," in *Proceedings of the international conference on intelligent technologies, InTech*, vol. 3, 2003, pp. 132–141.

[22] N. R. Sabar, M. Ayob, and G. Kendall, "Tabu exponential monte-carlo with counter heuristic for examination timetabling," in *Computational Intelligence in Scheduling, 2009. CI-Sched'09. IEEE Symposium on.* IEEE, 2009, pp. 90–94.

[23] E. Burke, Y. Bykov, J. Newall, and S. Petrovic, "A time-predefined local search approach to exam timetabling problems," *IIE Transactions*, vol. 36, no. 6, pp. 509–528, 2004.

[24] W. Erben, "A grouping genetic algorithm for graph colouring and exam timetabling," in *Practice and Theory of Automated Timetabling III.* Springer, 2001, pp. 132–156.

[25] T. Wong, P. Cote, and P. Gely, "Final exam timetabling: a practical approach," in *Electrical and Computer Engineering, 2002. IEEE CCECE 2002. Canadian Conference on,* vol. 2. IEEE, 2002, pp. 726–731.

[26] K. Sheibani, "An evolutionary approach for the examination timetabling problems," in 2002). *Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling*. 21st-23rd August, 2002, pp. 387–396.

[27] P. Cˆot´e, T. Wong, and R. Sabourin, "A hybrid multi-objective evolutionary algorithm for the uncapacitated exam proximity problem," in *Practice and Theory of Automated Timetabling V.* Springer, 2005, pp.294–312.

[28] C. Cheong, K. Tan, and B. Veeravalli, "Solving the exam timetabling problem via a multi-objective evolutionary algorithm-a more general approach," in *Computational Intelligence in Scheduling, 2007. SCIS'07. IEEE Symposium on.* IEEE, 2007, pp. 165–172.

[29] E. K. Burke, J. P. Newall, and R. F. Weare, "A memetic algorithm for university exam timetabling," in *Practice and Theory of Automated Timetabling.* Springer, 1996, pp. 241–250.

[30] E. Ersoy, E. O¨zcan, and S¸ .Uyar, "Memetic algorithms and hyperhill-climbers," in *Proc. of the 3rd Multidisciplinary Int. conf. on scheduling: theory and applications, P. Baptiste, G. Kendall, AM Kordon and F. Sourd, Eds,* 2007, pp. 159–166.

[31] M. Eley, "Ant algorithms for the exam timetabling problem," in *Practice and Theory of Automated Timetabling VI.* Springer, 2007, pp. 364–382.

[32] Z. N. Azimi, "Comparison of metaheuristic algorithms for examination timetabling problem," *Journal of Applied Mathematics and computing,* vol. 16, no. 1-2, pp. 337–354, 2004.

[33] Z. Naji Azimi, "Hybrid heuristics for examination timetabling problem," *Applied Mathematics and Computation,* vol. 163, no. 2, pp. 705–733, 2005.

[34] K. Dowsland and J. Thompson, "Ant colony optimization for the examination scheduling problem," *Journal of the Operational Research Society,* vol. 56, no. 4, pp. 426–438, 2005.

[35] G. Kendall and N. M. Hussin, "An investigation of a tabu-searchbased hyper-heuristic for examination timetabling," in *Multidisciplinary Scheduling: Theory and Applications.* Springer, 2005, pp. 309–328.

[36] G. Kendall and N. M. Hussin, “A tabu search hyper-heuristic approach to the examination timetabling problem at the mara university of technology,” in *Practice and Theory of Automated Timetabling V.* Springer, 2005, pp. 270–293.

[37] E. K. Burke, M. Dror, S. Petrovic, and R. Qu., "Hybrid graph heuristics within a hyper-heuristic approach to exam timetabling problems," in *The next wave in computing, optimization, and decision technologies.:* Springer, 2005, pp. 79-91.

[38] E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, and R. Qu, “A graph-based hyper-heuristic for educational timetabling problems,” *European Journal of Operational Research,* vol. 176, no. 1, pp. 177–192, 2007.

[39] R. Qu, E. K. Burke, and B. McCollum, “Adaptive automated construction of hybrid heuristics for exam timetabling and graph colouring problems,” *European Journal of Operational Research,* vol. 198, no. 2, pp. 392–404, 2009.

[40] N. Pillay and W. Banzhaf, “A genetic programming approach to the generation of hyper-heuristics for the uncapacitated examination timetabling problem,” in *Progress in Artificial Intelligence.* Springer, 2007, pp. 223–234.

[41] N. Pillay, “Evolving hyper-heuristics for the uncapacitated examination timetabling problem,” *Journal of the Operational Research Society,* vol. 63, no. 1, pp. 47–58, 2012.

[42] E. K. Burke, S. Petrovic, and R. Qu, “Case-based heuristic selection for timetabling problems,” *Journal of Scheduling,* vol. 9, no. 2, pp.115–132, 2006.

[43] Y. Yang and S. Petrovic, “A novel similarity measure for heuristic selection in examination timetabling,” in *Practice and Theory of Automated Timetabling V.* Springer, 2005, pp. 247–269.

[44] L. F. Paquete and C. M. Fonseca, “A study of examination timetabling with multiobjective evolutionary algorithms,” in *Proceedings of the 4th Metaheuristics International Conference (MIC 2001),* 2001, pp. 149–154.

[45] H. Asmuni, E. K. Burke, J. M. Garibaldi, and B. McCollum, “A novel fuzzy approach to evaluate the quality of examination timetabling,” in *Practice and Theory of Automated Timetabling VI.* Springer, 2007, pp. 327–346.

Publications

1. Sandeep Saharan and Ravinder Kumar, "Resource Optimizing Algorithm for Examination Scheduling using Graph Coloring" communicated in Applied Mathematics and Information Sciences **[SCIE] Current status "*Under Review*".**

2. Sandeep Saharan and Karuna Kadian, "A Multi-Objective Genetic Room Allocation in Examination Scheduling Using Graph Coloring" accepted in International Conference on Signal Propagation and Computer Technology-ICSPCT 2014 proceedings will be published in **IEEE Xplore**.

3. Sandeep Saharan and Ravinder Kumar, "Multi Constrained Based Algorithm for Examination Scheduling Using Graph Coloring" accepted in Second International Conference on Emerging Research in Computing, Information, Communication and Application- ERCICA-2014 proceedings will be published in **Elsevier**.