Interactive Decision Support System for Planned Student Scheduling and Incremental Changes in Large Scale Timetabling

A Thesis

submitted in partial fulfillment of the requirements for the award of the degree of

Doctor of Philosophy

in

Department of Computer Science and Engineering

by

Sanjeev Kumar Guleria
(Reg no: 951203001)

Under the Guidance of

Dr. Arvind Kumar Lal

Thapar Institute of Engineering and Technology
Patiala-147004, Punjab, India
February 2022

Candidate Declaration

I hereby certify that the work, which is being presented in the thesis, entitled **Interactive Decision Support System For Planned Student Scheduling And Incremental Changes In Large Scale Timetabling**, in partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** and submitted to the institution is an authentic record of my own work carried out during the period **July 2012** to **February 2022** under the supervision of **Dr. Arvind Kumar Lal**. I have also cited the reference about the text(s)/figure(s)/table(s) from where they have been taken.

The matter presented in this thesis has not been submitted elsewhere for the award of any other degree or diploma from any institution.

Date: 28-2-2022

Sanjeev Kumar Guleria
Candidate

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

Date: 28.02.2022

Dr. Arvind Kumar Lal
Supervisor

The Ph.D. Viva-Voice examination of **Sanjeev Kumar Guleria**, Research Scholar has been held on _____ .

(Supervisor)	(External Examiner)	(Chairperson of Research Committee)

.....dedicated to my parents

Abstract

A student joins an institution to earn some qualification. For this purpose, he/she has to successfully complete a set of courses in a span of few terms. Some of the students fail in some of the courses studied in a term. Such courses are called backlog courses. Backlog courses have to be taken up in any of the successive terms possible only if these are offered to junior class students in one or more sections. The planning needed to help students in deciding which courses can be taken with which of the section of junior class students, forms the core of present thesis. Besides this, the thesis also covers incremental changes to be made in a working time table once the whole process has stabilized. This thesis consists of six chapters.

Chapter One is introductory in nature. In this chapter, we briefly discuss the issues of backlog courses while preparing timetable. Two approaches generally used to handle this problem and some important terminologies are also discussed in this chapter. A brief survey of the literature available on the subject, and summary of the work presented in the succeeding chapters of the thesis, also appears in this chapter.

The second chapter describes the model for planning of student scheduling. Planning of student scheduling involves the linking of courses that the students will be attending with students of junior classes while they are free. This scenario is solved by a method developed to find out backlog courses which can be pooled together in same time slot along with keeping free the sections of students where regular courses are held. This process is termed as planning for student scheduling whose output can be considered as a set of constraints to be used in building the time table. The Hungarian method of assignment model has been repeatedly used to form links further required as an input for scheduling of the timetable. A core model has been designed using Dynamic Programming approach spread across five stages where each stage is referring to one year of study. Fifth stage and year refers to students who have studied all four year of programmes but are still having backlog courses at the end of it. In this chapter, the links are termed across years where each year has only one group of students studying together in single section.

Third chapter describes two models for real life implementation of planning of student scheduling. The assumptions and methodology used for arriving at data-sets is emphasized. Planning of student scheduling is done using repeated use of assignment problem solution and heuristics. The implementation details have been provided for testing the models developed for decision support system. The algorithms for implementing two variants of this model have been described in detail. Results of implementations are eval-

uated for their efficiency and achievement of objective. Planning of student scheduling model is judged against the current system in use. Results have been demonstrated in this regard.The above generated inputs can be used to build the time table.

Fourth chapter describes the model for interactive student scheduling. Students can easily get registered in courses planned to be backlog courses. However, the courses which are not planned this way can still be registered as backlog courses but this becomes a tedious work to scroll time table of whole institute to complete this search. One tool has been made which searches and confirms the course section combinations which are clash free to search in an already built time table. The students can get registered in combinations searched in this way for combination of back log and regular courses. This is termed as registration process. Interactive student scheduling provides a user interface that helps in identifying clash-free set of course sections to be registered based on the constructed time-table. Two models have been designed for searching clash free combinations of course-sections. These two models provide a working process that helps in decision making process at the time of registration by a student. As this is at present done manually and no other methods are available so it is not possible to measure the new models against anything. So both are tested against each other as they solve the same problem differently.

Fifth chapter describes the model for Incremental Changes in Large Scale Time-Tabling. Students start studying as per their student registrations, and there is a need to make change in time table, one search tool has been created which can approve or disapprove a time table change request while protecting the combinations of student registrations already made. This system will ensure the time tables of individual students are still clash free. This is termed as incremental changes in large scale timetabling. Incremental Changes in Large-Scale Timetabling decides whether the suggested change can be made without affecting the clash-free timetables of students who have earlier registered in any of the courses in which change is sought. Next, the implementation details are provided for testing the models developed for decision support system. The algorithms for implementing various scenarios for this model are described. The user interface for implementation is explained along with the explanation of terminology used. Incremental Changes in Large Scale Timetabling uses heuristics to find the feasibility of changes requested. Results of implementations are evaluated for their efficiency. These models are judged by the variety of options provided by the decision support system to achieve the goals. These models provide a working process that helps in decision making process at the time of changes to be made in time table post registration. As this is at present done manually so it is not possible to measure the new models against anything. These

models add to Quality of Service provided to the users for making changes possible in final time table.

The last and sixth chapter summarizes the contents of thesis and discusses the contributions as well as limitations of the present research. With summary of contributions, the chapter also envisions the future research work.

Acknowledgements

I praise and thank Almighty God for showering blessings throughout the research work for its fruitful completion. I owe a lot to my parents for their love and support throughout my life. I thank both of them for encouraging me at every stage of my personal and academic life. They longed to see this achievement come true. Alas, both my parents left this mortal world before completion of my work. I wish their souls rest in peace.

I owe a deep sense of gratitude to Dr. Arvind Kumar Lal, my research supervisor for accepting me as his student. I deeply thank him for his patient listening, reassuring encouragement, thoughtful critiques regarding research work and his constant support for giving me the freedom to explore my intellectual curiosity without objection. His advice and encouragement were always important guiding lights for when I lost my focus. His guidance helped me in all the time of research and writing of this thesis.

I would love to thank Dr. D.S. Bawa for providing me key ideas to experiment so that they reach the level of fruition. He had taught a lot to me and provided support whenever in need.

I am grateful to the members of my doctoral committee: Dr. Maneek Kumar, Dr. Shalini Batra and Dr. Anil Kumar Verma for happily giving their time to offer insightful comments towards improving my work. I had great pleasure of studying under teaching of Dr. Vikas Sharma and Madam Monika Saini for teaching me operations research and research methodology during the course work.

Numerous faculty members and staff members have always been very kind with their words of encouragement. Crossing paths with them while walking on campus has always brought a smile to my face.

I wish to thank my family for always supporting me. Irrespective of what they were dealing with on an individual level, they were always available for help and support. My son Harshit Singh Guleria needs a special mention due to his help in proofreading the thesis. Without his help I might have overlooked many points.

My heart overflows with gratitude for those who have been with me throughout the ordeal.

Sanjeev Kumar Guleria

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AYM	All Year Model
CGPA	Cumulative Grade Point Average
EAA	Expand All Algorithm
ICLST	Incremental Changes in Large Scale Time-Tabling
MCA	Merge and Consolidate Algorithm
PYM	Prior Year Model
QoS	Quality of Service

Chapter 1

Introduction

One of the problems encountered by most of the academic institutions in preparing time table, especially in handling those students who fail in one or more courses or are detained in some courses due to inadequate attendance, medical and other personal reasons. Many of these students who meet the minimum cumulative Grade Point Average (CGPA) norms or have pass percentage of marks in aggregate are promoted to higher classes with the provision that they shall have to clear their pending courses in a prescribed duration to become eligible for the award of the degree. The courses which could not be cleared by a student are called backlog course. A backlog arises when Two options are normally used to handle this problem.

Option I: Institutions may allow the students to appear in supplementary examinations / end semester examinations for clearing their backlog courses provided they have fulfilled minimum attendance criteria. In such a case, the students are required neither to attend the classes again nor go through the entire assessment process. The internal assessment is carried over and a student is required to appear in the end semester examination of the pending course only whenever it is next conducted. In case, student has low internal assessment, he may be required to submit additional assignments for evaluation. In fact, the student is left on his own to do self-study and appear in the examination. The student is required to pass all his courses within the prescribed maximum duration of the programme.

Option II: Institutions require these students to attend all the classes of pending courses all over again with their junior batches and pass their courses by going through the whole assessment process. These students have to register for additional courses along with their prescribed courses of their current semester. These courses are offered if and only if all the components of the proposed pending course are available in the time table in that semester. Some institutions may also offer summer terms during vacations for students to attend and pass their courses. However, such offer of summer terms is subject to the availability of faculty and adequate number of student's registration in that particular course in summer. The students have to pass all their courses within the maximum duration allowed for an academic programme. The students may also have to

exit the programme at certain prescribed stages, if they do not meet the CGPA requirements. Students who do not meet the desired pass percentage or CGPA requirement for promotion are detained in the same class and are required to attend classes with their junior batches.

A student can take a backlog course(s) only if all of its components (that is Lecture, Tutorial and practical) are available in the timetable with junior students and there is no clash in time table with their own assigned courses. In case there is a clash, the student is not registered for the backlog course. For example, in a forty hour week if the regular courses in a class require 32 hours, then for a student seeking registration for a backlog course requiring less than 8 hours per week, only 8 specific vacant hours are available in which all the components of his backlog course must lie in a junior class. Chances of finding all components of the backlog course, within his free periods, is just a coincidence. A student might have to leave some other course of his regular semester to register for his backlog course. This non availability of backlogs result in a large number of students remaining in the system even after normal duration of the programme. These students are required to pass their backlog courses in the extended period allowed by the institution to complete the degree.

The investigator is closely associated with making of the timetable at Thapar Institute of Engineering and Technology (TIET), which is deemed to be a University located in Patiala, Punjab India. It offers four year bachelors of engineering in nine branches namely Chemical engineering, Civil Engineering, Computer Engineering, Computer Science and engineering, Computer Science and Business Systems, Bio technology, Biomedical Engineering, Electrical Engineering, Electronics and Communication Engineering, Electronics and Computer Engineering, Electronics and Instrumentation Engineering and Mechanical Engineering, Mechanical Engineering (Production), Mechatronics. It also offers two year Master in Engineering programmes in Structural Engineering, Infrastructure Engineering, Computer Science and Engineering, Software Engineering, Electronics and Communication Engineering, Electronics Instrumentation and Control Engineering, Power Systems, CAD/CAM Engineering, Production Engineering, Thermal Engineering and Master in Technology programmes in Biotechnology, Chemical Engineering, Environmental Science & Technology, Energy Technology & Management, and VLSI Design. Besides these we have two-year Master in Science programmes in Biotechnology, Chemistry, Bio-chemistry, Mathematics, Mathematics and Computing, Physics and Environmental Sciences. Single branch programmes like two-year Master of Arts programme in Psychology and a three-year master of computer applications programme in computer applications are also offered. Apart from the above programmes it offers a

two years MBA and Programme, Ph.D. programmes in various branches of Engineering, Business Management and Bio Sciences.

The institute makes one master timetable for all undergraduate, postgraduate classes of engineering and master of computer applications students. The curriculum of programmes consists of core, specialization and elective courses with each one having lecture, tutorial and laboratory components as required. Prior to 1996 timetable was made manually by first scheduling the core courses and then each department will fill up the time slots on the chart with teacher subject combinations and finally allocation of rooms would be done. It required many corrections, rework to remove clashes of teachers, laboratories and rooms. Little effort was made to adjust the backlogs and therefore a large number of students remained in the system to complete their degrees.

The timetabling at TIET is a large scale timetabling problem. It involves dividing the classes into sections of 20 to 30 students depending upon the strength of various classes. The lecture classes may have 30 to 120 students. Tutorials are held section wise and practical classes have student strength varying from 20 to 60 students. The class rooms of varying capacities are scheduled as per preference of departments, teaching aid facilities and strength of students. The duration of the periods may vary from 1 hour to 4 hours. The institute developed its timetabling software in house using a network heuristic that maximizes teachers' preference for time slots and efficient utilization of rooms. The institute deploys Option II for handling backlog of students.

This study focuses on institutions using the second option for handling backlogs of students and becomes a very complex timetabling problem. Then the question arises: Can enough opportunities be created for students to register for additional courses within the limitations of time available to them during a semester? This thesis explores to find answer of such question. The objective is to let students pass out with their own batch of students or let them be in the system for a minimum of extended time after normal duration of the programme.

The researchers in the past have done considerable work on the various aspects of large scale timetabling but the problem of handling of backlog courses has not been explored to the satisfaction of students. The literature survey on the design of support system for student scheduling is given in the following section.

1.1 Review of literature

Timetabling involving day, time, faculty, subject and room allocations has been of interest to researchers. Earlier studies suggested different approaches in solving large scale time tabling scheduling problems. Andrew and Collins [1] discussed linear programming model of timetable. Dyer and Mulvey [2] and Mulvey [3] worked on a network model; Breslaw [4], Mcclure and wells [5] discussed integer programming model of timetabling. Study by Dinkel, Mote and Venkatramanan [6] developed a comprehensive network based Decision Support System to schedule all graduate and undergraduate courses in the college of Business Administration at Texas A & M University involving 175 faculty over 300 sections 20 rooms and 16 time slots for each semester with improved instructor satisfaction levels and efficient room utilization. Feldman and Golumbic [7] & [8] implemented and compared various algorithms to arrive at one of the algorithm to solve student scheduling problem as a variant of constraint satisfaction problem. Tripathy [9] formulated a student scheduling problem as an integer linear programming problem and Lagrangian relaxation was used to solve it. Monfroglio [10] solved a case of school timetabling problem using hybrid genetic algorithms comprising of constrained heuristic search and genetic techniques. Burke et al [11] presented an overall overview as state of the art about timetabling. Saleh Elmohamed et.al [12] compared many simulated annealing techniques for making course timetables based on student preferences of courses. Muller and Bartak in [13] and [14] worked on interactive timetabling whereas Christian Blum et al [15] used Genetic Algorithm to solve timetabling problem. Head and Shaban [16] made a strategy to simultaneously handle the course and student scheduling by making a timetable using heuristic functions, student preferences and well defined constraints. The system does not have electives and single day schedule is reflected on all days for time slot range. Datta et. al. [17] worked on multi-objective evolutionary algorithm for university class timetabling. Badoni et. al [18] made a new hybrid algorithm for university course timetabling. Nogareda and Camacho [19] created two algorithms to handle multi-courses allocation and timetabling together. Babaei et. al [20] worked on timetabling for multi-departments common lecturers using hybrid fuzzy and clustering algorithms.

Several methods of scheduling exist in literature. Kaur and Bala [21] did a survey on prediction based resource scheduling techniques. Zhang et al [22] and Chen et al [23] concentrated on different applications of flexible job shop scheduling, while Wang et al [24] looked at multi-objective part of same problem. Manish et al [25] made a surwey on information retrieval techniques using web crawlers. Zhou et al [26] worked on multi-objective optimization using cuckoo search algorithm. Some methods of generating a timetable for academic organizations are available in the literature. Samir

et al [27] created REDOSPLAT language for defining the requirements of timetabling. Schaerf [28] discussed some of the most common variants of the basic formulation of course timetabling. Burke and Petrovic [29] summarized recent research directions used in automated timetabling. This area has been discussed as an activity by Carter [30], but no detail has been provided about the planning of identiying the courses which have to be scheduled together. In addition, there is no reference of the term 'backlog planning' in literature. To start with, Sheard and Hagan [31] made an effort to create a special learning environment for repeat students. Abidin [32] studied on planning the course work for students who were offered a backlog in one course of Universiti Teknologi Petronas, and also mentioned the hardships faced by students studying backlog courses mentioned as repeat course. Kristiansen [33] concentrated on the planning of elective courses for high schools.

The literature available on student scheduling problem considers preference list of the students for all courses after the timetable has been created. Laporte and Desrochers [34] mathematically formulated the student scheduling problem. Both hard and soft constraints are inbuilt in this formulation of optimization problem. The hard constraint, in this model, respects the student course selections. The soft constraints encompass the time conflicts for students. In case of any time conflict, the students are advised for a different course selection. To solve similar problems, a multi dimensional approach was initially considered by Pierskalla [35] and later followed by Frieze and Yadegar [36] to solve 3-dimensional assignment problems. As surveyed by Pentico [37], several references of different approaches of assignment problems are available in literature. Loiola et al [38] did a survey on quadratic assignment problems. Zhang et al [39] worked on an auxiliary optimization method for link prediction.

The student scheduling problem considers preference list of the students for all courses after the timetable has been created. This work comes under the category of student scheduling, student sectioning, assignment to teaching groups, student registration, add/drop registration etc. A simple heuristic approach was used by Almond [40] to create a timetable for one department having fixed courses and another for whole faculty having courses offered by different departments. Busam [41] talked about class scheduling based on section preferences and also employs means of balancing sections of same courses regarding the utilized strength.

Winters [42] had demonstrated a scheduling algorithm which is used for student registration using the course requests of students by processing them in a batch after sequencing them according to various groups of students. This algorithm also accepts the fact that there are incomplete schedules in a system which may be handled

by some manual process later. Miyaji et. al. [43] worked on partitioning students into groups.

Özer and Özturan [44] targeted only the add/drop process in their work and implemented it as a direct barter method to allow swap of courses amongst students and also restricted students to drop course only once as the same course can not be taken back as it might have been picked up by another student. Heitmann and Bruggermann [45] used mixed integer programme to assign students to multiple teaching groups. Dostert et. al. [46] demonstrated polynomial time algorithmic approach to student sectioning in existing timetables while assigning students to multi section courses.

Several methods of generating a timetable for academic organizations are available in the literature. However, we are suggesting an approach where assignment problem is solved repeatedly to arrive at the solution needed by us.

Problem of timetabling has been handled in various combinations by researchers in the past but problems related to backlog students have been kept out of the purview of these studies. This problem of backlogs handling is an underlying motivation for this study. The present thesis deals with the problem of "Interactive Decision Support System for Planned Student Scheduling and Incremental Changes in Large Scale Timetabling".

1.2 Objectives of the study

Keeping in view the gaps in previous study related to timetabling of students having backlogs, the following objectives have been proposed for the study:
(i) To design and develop a model for timetabling which generates constraints for maximizing the number of students whose backlogs can be adjusted in the constructed timetable.
(ii) To create an algorithm with a capability of searching a clash free timetable for a student in constructed time table leading to his registration.
(iii) To develop a model to maintain the constructed timetable amidst incremental change requests without affecting the students registered in various cross-section of courses.

The study presumes that the institute time table is feasible for all students who do not have any backlogs and has been made in a manner that maximizes the adjustment of backlogs of maximum students, still leaving a choice to students to select their own set of courses for registration as per the rules and regulations of the institution.

1.3 Concepts and Definitions

1. **Regular Courses**: Regular courses are a set of courses offered in different classes of an academic programme. The delivery of a course content may be made through its one or more components, that is Lectures, Tutorials and Practical. The number of lectures (L), tutorials(T) and practical(P) per week for each course are prescribed in the academic curriculum.

2. **Backlog Courses**: A backlog course arises for a student when he is unable to pass a course when it is offered to him for the first time in a programme, what so ever be the reasons. A student may not obtain a pass grade; may be detained due to inadequate attendance; may not be able to attend the final exam due to sickness or any other reason. A backlog can arise only in the courses of previous classes for which a student is enrolled. Thus a student of class I does not have any backlog because it is his initial class, a student of class II can have backlog(s) in courses of class I but not in courses offered in classes 2,3,4,... and so on.

3. **Available Courses**: The students of a class have all regular courses offered in previous classes as "available courses for backlogs", e.g. students of class IV have all regular courses offered in previous classes I, II, III as available courses for backlogs.

4. **Section**: A section represents a set of students studying together all the components of courses in a term for whole week. All the students of a particular programme and branch are distributed in a set of sections, each represented by a unique name.

5. **Course-section**: Course-section is a pair of course and section meaning that a particular course running in a particular section. This can be used as an instance of course running in a section.

6. **Dummy Course-section**: A dummy course section means a slot when no course is scheduled in that section or in other terms the section is purposely left unscheduled at that time.

7. **Link**: a set of course-sections is called a link, which are to be scheduled in same time-slots.

8. **Cancellation**: Any of the course-section combination in a link can be assigned a cancellation of a set of sections (also called null course section list such that students studying in their own section of regular courses which is offering a null course section can attend any of the course-section combinations). This means that the sections in cancellation list can not be assigned any class in those time slots when the course-sections

of link are scheduled in same time.

9. **Timetabling of Links and cancellations**: All the courses in the list of course-sections will be held in the 'same time-slots' in parallel, while all the courses offered in sections attached in null course section list will be held at 'different time-slots' so that they are clash free.

10. **The Registration Process**: Before the start of a semester the students are required to register for their courses. The students who do not have any backlogs register for the courses being offered for the ensuing semester The students, who have backlogs have following options available to them.

a. They can add a course(s) to the normal set of courses of their current semester within the available hours and without clash in the timetable.

b. They can drop a course from their current semester and add backlog courses within the available hours and without clash in the timetable.

c. The student may decide to have own mix of courses within the available hours and without clash in the timetable.

1.4 Present Work

To overcome the backlog problem, a system of links was developed which resulted in making available backlog courses to a large number of students of a section during the free slots of their section timetables. A link is a set of course components of various classes and vacant slots that are scheduled at the same time in the master timetable. The vacant slots in which ever section they appear allow the students of that section to attend to their backlogs in other classes. An effort has been made in the present work to give opportunity to maximum number of students to clear their backlogs during the normal duration of the programme and pass out with their batch mates. The objective was to optimize the formation of links and incorporate these as constraints in making of the timetable. The work had been demonstrated with real data at Thapar Institute of Engineering and Technology, Patiala from 1996 to 2006.

1.5 Thesis Organization

The thesis is organized into six chapters. A brief summary of work presented in succeeding chapters is as follows:

- **Chapter 2**: This chapter describes the model for Planning of Student Scheduling. Planning of Student Scheduling involves the linking of courses that the students will be attending with students of junior classes while they are free. The Hungarian method of assignment model has been repeatedly used to form links further required as an input for scheduling of the timetable. A core model has been made designed using Dynamic Programming approach spread across five stages where each stage is referring to one year of study. Fifth stage and year refers to students who have studied all four year of programmes but are still having backlog courses at the end of it. A walk through for this model has been shown in the appendix by taking a small sample data for understanding.

- **Chapter 3**: This chapter describes two models for Real life implementation of Planning of Student Scheduling. The implementation details are provided for testing the models developed for decision support system. The algorithms for implementing two variants of this model are described. The assumptions and methodology used for arriving at data-sets is emphasized. Planning of Student Scheduling is done using repeated use of assignment problem solution and heuristics. Results of implementations are evaluated for their efficiency and achievement of objective. Planning of Student Scheduling model is judged against the current system in use. Results have been demonstrated in this regard.

- **Chapter 4**: Fourth chapter describes the model for interactive student scheduling. Students can easily get registered in courses planned to be backlog courses. However, the courses which are not planned this way can still be registered as backlog courses but this becomes a tedious work to scroll time table of whole institute to complete this search. One tool has been made which searches and confirms the course section combinations which are clash free to search in an already built time table. The students can get registered in combinations searched in this way for combination of back log and regular courses. This is termed as registration process. Interactive student scheduling provides a user interface that helps in identifying clash-free set of course sections to be registered based on the constructed timetable. Two models have been designed for searching clash free combinations of course-sections. These two models provide a working process that helps in decision making process at the time of registration by a student. As this is at present done manually and no other methods are available so it is not possible to measure the new models against anything. So both are tested against each other as they solve the same problem differently. These models add to Quality of Service (QoS) provided to the users.

- **Chapter 5**: This chapter describes the model for Incremental Changes in Large Scale Time-Tabling(ICLST). Students start studying as per their student registrations, and there is a need to make change in time table, one search tool has been created which can approve or disapprove a time table change request while protecting the combinations of student registrations already made. This system will ensure the time tables of individual students are still clash free. This is termed as incremental changes in large scale timetabling. ICLST decides whether the suggested change can be made without affecting the clash-free timetables of students who have earlier registered in any of the courses in which change is sought. Next, the implementation details are provided for testing the models developed for decision support system. The algorithms for implementing various scenarios for this model are described. The user interface for implementation is explained along with the explanation of terminology used. ICLST uses heuristics to find the feasibility of changes requested. Results of implementations are evaluated for their efficiency. These models are judged by the variety of options provided by the decision support system to achieve the goals. These models provide a working process that helps in decision making process at the time of changes to be made in time table post registration. As this is at present done manually so it is not possible to measure the new models against anything. These models add to Quality of Service(QoS) provided to the users for making changes possible in final time table.
- **Chapter 6**: This chapter summarizes the contents of thesis and discusses the contributions as well as limitations of the present research. With summary of contributions, the chapter also envisions the future research work.

Chapter 2

Planning of Student Scheduling Across Years

Student Scheduling refers to the process of assigning courses to the student for an academic session which corresponds to a clash-free schedule or timetable. In simple terms, this can be termed as registration of students in particular courses. All the students in institute registered in any program are already segregated into several sections and master time table is created as per those sections. The creation of time table itself serves the purpose by just assigning one of those sections to students and intimating them. The question arises: Where does this planning part comes into picture? Planning refers to a process of making arrangements so that students are able to attend classes across these sections where students mostly belong to different years of their study. This is required for those students who have some backlog courses pending which are to be studied with students of junior classes along with the regular courses in their own section.

The first objective is to design and develop a model for timetabling which generates constraints for maximizing the number of students whose backlogs can be adjusted in the constructed timetable. In other words, to develop an algorithm that create links of course components that optimize adjustment of maximum number of students for their backlogs. These links act as additional constraints in timetabling.

Significant work has been done on the problem of timetabling, yet there is no appropriate model, available in the literature, to address the backlog planning and subsequent student scheduling. We are proposing an approach for creation of links by solving assignment problem repeatedly. These links can further be used to form timetable by following same-time scheduling. This chapter explains the process of finding courses in such a manner that maximum number of the students can be allocated at least one backlog course as first priority. If possible then one more course may be adjusted for studying in addition to one course identified for this purpose. This planning will generate inputs or constraints for the timetable so that most of the students' backlogs might be adjusted in created time table.

This chapter is organized as follows: Section 2.1 explains the model and problem formulation for planning of student scheduling across years while section 2.2 shows procedure for generation of links using dynamic programming approach considering stu-

dents across different years of study.

2.1 Modelling and problem formulation for planning of student scheduling across years by generation of links

Let m and n denote maximum number of courses and classes, respectively.

B_i : Number of courses offered, $i = 1, \ldots, m$

L_j : Number of classes in the programme, $j = 1, \ldots, n$

S_{ij} : Set of students of class j having backlogs in course i

C_{ij} : $|S_{ij}|$ Number of students of class j having backlogs in course i

K_{ij} : $\begin{cases} 1, \text{ if course i is offered as backlog in class j} \\ 0, \text{ otherwise} \end{cases}$

t_{ij} : Number of time-slots(periods) required by a regular course i of class j during a week

T_j : Total number of time slots required for regular courses of class j and is given by

$$T_j = \sum_{i=1}^{m} t_{ij}, \qquad \forall j = 1, \ldots, n \tag{2.1}$$

H_j : Maximum time slots available to a class j during a week (same for all classes).

A_j : Time Available for scheduling of backlogs which is defined as

$$A_j = H_j - T_j = H_j - \sum_{i=1}^{m} t_{ij}, \quad \forall j = 1, \ldots, n \tag{2.2}$$

D_{ij} : ith dummy course for class j

L : Number of hours used in a week by lecture component of course

T : Number of hours used in a week by tutorial component of course

P : Number of hours used in a week by practical component of course

Considering the notations and their description, the problem of allocating backlogs can be formulated as (0,1) linear programming as follows:

$$\begin{aligned} \text{maximize} \quad & Z = \sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{m} t_{ij} x_{ij} \leq (H_j - T_j),\ j = 1, \ldots, n \end{aligned} \tag{2.3}$$

where

$$x_{ij} = \begin{cases} 1, \text{ if course i is assigned to class j} \\ 0, \text{ otherwise} \end{cases} \tag{2.4}$$

The solution to this linear programming problem will give the maximum number of backlogs that will be adjusted in course i of class j, where $x_{ij} = 1$, that is,

$$Z = \sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} x_{ij} \tag{2.5}$$

The number of students adjusted will be the union of all sets of students, that is, $\cup S_{ij}$, where $x_{ij} = 1$.

The above problem can be solved by Dynamic Programming approach. The approach helps in developing a structure for making of academic timetable that adjusts maximum number of students having backlogs within the same time frame for which the time table is made. The approach will comprise of defining (i) stages, (ii) states and (iii) the policy of generating alternatives and criteria for taking optimal decision and (iv) The optimal policy.

i) Stages: The classes form the stages that is $j = 1, \ldots, n$.

ii) States: The course available for backlogs

$$i = \begin{cases} 0, & \text{for } j = 1 \\ 1, \ldots, m & \text{for } 2 \leq j \leq n. \end{cases} \tag{2.6}$$

iii) Policy: Generate alternatives of course combinations within the limits of time available, and select the combination of courses for which $n(\cup S_{ij})$ is maximum. The number of students adjusted in this combination of courses becomes the optimal policy at a stage given by $= \cup S_{ij}$.

iv) The optimum policy is the sum of total number of students adjusted at each of the stages, $j = 1, \ldots, n$.

Definitions

1. **Super Class:** Super class is a terminal class of any academic programme. In dynamic programming it is the last stage of the problem. Super class consists of those students in a system who need to pass out their backlog courses although their batch mates have already passed out. There are no regular courses scheduled in a superclass. All the time slots in a week are available to students for attending to their backlog courses offered in

various regular classes of the programme. The following students of superclass in a given timetable are adjusted for their backlogs by default:
(a) Students having only one backlog.
(b) Students having one or more backlogs in any one of the classes of the programme.
For analysis, the names of such students are removed from the backlog list.
2. **Dummy Course:** A dummy course consumes time and no other resource. It does not require any faculty or room. The appearance of a dummy course in a class merely indicates a free period for regular students.
3. **Link:** A set of courses scheduled in the same time slot. The courses are of different classes and all their lecture, tutorial and practical components are linked and are scheduled in the same time-slots links for adjustment of backlogs will always have at least one dummy course in the set. The class that has a dummy course allows its students to attend their backlog courses in the link. A link has the following characteristics:
(i.) A link can have a maximum of n-1 courses and a minimum of 1 dummy course.
(ii.) A link can have a minimum of 1 course and maximum of n-1 dummy courses.
(iii.) A student can register only for one backlog in a link, as all the courses in a link are scheduled at the same time.
(iv.) The time consumed by a dummy course is the maximum (time taken by courses in the link in previous classes) That is

$$max\{t_{ij}\},\ where\ t_{ij} = L_{ij} + T_{ij} + P_{ij} \tag{2.7}$$

(v.) The time taken by a link at any stage is the maximum of (time taken by courses in the link till that stage) That is $max\{t_{ij}\}$ $where$ $t_{ij} = L_{ij} + T_{ij} + P_{ij}$
(vi.) Students adjusted in a link refers to Union of all student sets having backlogs in the courses in the link that is $\{\cup S_{ij}\}$
(vii.) A link is treated as one of the available courses with time period of $max\{t_{ij}\}$ and student set of $\{\cup S_{ij}\}$
4. **Combinations:** It is the set of courses considered for offering backlogs to a class. All possible combinations of courses available for backlogs satisfying the time availability constraint are listed and one combination that adjusts maximum students is offered as backlogs to that class. If 'n 'is the number of courses available then combinations of 'r' courses (without replacement) is given by

$$^{n}C_{r} = \left(\frac{n!}{r!(n-r)!}\right) \tag{2.8}$$

All available courses (including links) in the combination are linked to the given class by providing dummy courses corresponding to the courses in the combination.
The time consumed by the dummy courses in single section classes

$$= \sum time\ periods\ of\ courses\ in\ the\ combination.$$

$$= (L_1 + T_1 + P_1) + (L_2 + T_2 + T_2) + \ldots \tag{2.9}$$

2.2 Procedure for generating links

In this chapter, Stage-I, Stage II, Stage-III and Stage -IV refer Class-I, Class-II, Class-III and Class-IV, respectively.

(a) **Stage – I (j=1)**

- Course available : Zero
- Backlog of students : Zero
- Backlog adjusted : Zero
- Time required for regular courses:

$$T_1 = \sum_{i=1}^{m} t_{i1} \tag{2.10}$$

(b) **Stage – II (j=2)**

- Course available : $i = 1, \ldots, m$, for $j = 1$
- Time required for regular courses:

$$T_2 = \sum_{i=1}^{m} t_{i2} \tag{2.11}$$

- Time available for backlog course:

$$A_2 = H_2 - T_2 \tag{2.12}$$

Generating alternatives and decision making: This module will generate alternatives and do the decision making by choosing the best alternative for creation

of links using Algorithm 2.1.

Algorithm 2.1 (Module-I)

FirstModule(i,j) **for all** course available for backlogs to class j **do**
for all students of class j have backlogs **do**
  for all combinations of courses having $\sum t_{ij-1} \leq A_j$ **do**
    if $|\cup S_{ij}|$ is Maximum **then**
      Select this combination.
      if Multiple items in $|\cup S_{ij}|$ **then**
        Select one requiring $min \sum t_{ij}$.
      end if
      Add a dummy course to each of the course / link in class j such that the course(s) and the dummy course(s) are in a link.
    end if
  end for
end for
end for
Students adjusted for backlogs in selected courses $\leftarrow \bigcup S_{ij}$.
Number of students adjusted $\leftarrow |S_{ij}|$.
Time consumed by a dummy course $\leftarrow$ Max $\{t_{ij}\}$ of courses in the link
Total time utilized $\leftarrow \sum t_{ij}$.
Time available for scheduling of regular courses $\leftarrow H - \sum t_{ij}$, where t_{ij} is the time consumed by the links offered as backlogs, that is, the time consumed by the dummy courses offered in class j.

(c) **Stage – III (j=3)**

- Courses available for backlogs: i=1, . . . ,m for j=1,2
- Time required for regular courses of class III:

$$T_3 = \sum_{i=1}^{m} t_{i3} \tag{2.13}$$

- Time available for backlog courses:

$$A_3 = H_3 - T_3 \tag{2.14}$$

Generation of alternatives and decision making: Generation of alternatives and decision making procedures are carried out in the following module.In order to maximize the number of students adjusted for backlogs, Hungarian method of assignment problem has been formulated in Module-II (that is, algorithm 2.2) for allocating courses of class I with that of class II.

The above problem have to be balanced for solving the assignment problem. If there exists a dummy course in class j-1, then the courses in that link may not be included for assignment problem. Courses in which students of class j do not have any backlogs may not be considered for assignment, as it does not contribute to any additional student adjustment in the links.

Algorithm 2.2 (Module-II)

1:

$$\begin{aligned} \text{maximize} \quad & Z = \sum_{l=1}^{m} \sum_{k=1}^{m} C_{lk} x_{lk} \\ \text{subject to} \quad & \sum_{i=1}^{m} x_{lk} = 1, \quad k = 1, \ldots, m., \\ & \sum_{j=1}^{m} x_{lk} = 1, \quad l = 1, \ldots, m, \\ & x_{lk} = 0 \text{ or } 1, \quad \textit{for all } l \textit{ and } k., \\ & C_{lk} = \{S_{l1} \cup S_{l2} \cup \ldots \cup S_{lj-1} \cup S_{kj}\} \end{aligned} \tag{2.15}$$

2: Where l and k denote the all of courses/ links upto class j-1 (Array 1) and class j (array-II), respectively.

3: The solution to the problem will specify the maximum number of backlogs that will be adjusted in courses l and courses k (in a link) for $x_{lk} = 1$.

4: The number of students adjusted will be the union of all sets of courses, where $x_{ij} = 1$. Specifically, $l = 1, \ldots, m$ and $k = 1, \ldots, n$ refer to courses of links made upto class j-1 and class j, respectively.

Outcomes of the Module-II

- The solution thus obtained from the assignment problem gives rise to a set of links.
- Number of students adjusted in a link= $|S_{ij-2} \cup S_{ij-1}|$ for $j = 3$.
- Time slots required by linked courses $t_{ij} = max\{t_{ij-2}, t_{ij-1}\}$ for $j = 3$.
- Treat each of these links as a course with set of students=$\{S_{ij-2} \cup S_{ij-1}\}$ and $t_{ij} = max\{t_{ij-2}, t_{ij-1}\}$.
- All links formed by assignment algorithm cannot be offered for backlog in class j due to the limited time available $A_3 = T_3 - \sum t_{i3}$.
- Consider for combinations: all links formed, all previous links (if any) not considered due to dummy course in class j-1 and all other courses not considered due to imbalance in the assignment problem.

- Make all possible combinations and decisions as given in Module I by repeating again.

(d) **Stage – IV (j=4)**

Courses available for backlogs in Stage-IV

- Links formed upto class III.
- Remaining courses of class 1 and class 2.
- Courses of class 3, $\quad i = 1, \ldots, m \quad for\ j = 3.$
- Time required for regular courses of class IV:

$$T_4 = \sum_{i=1}^{m} t_{i4} \tag{2.16}$$

- Time available for backlogs:

$$A_4 = H_4 - T_4 \tag{2.17}$$

- Form some new links by assignment algorithm. Use array of courses and links available upto stage $j-2$ and courses of $j-1$.
- Use module II and module I as discussed in previous subsections (b) and (c) for deciding the set of students and number of students adjusted for their backlogs.

(e) **Stage - V** (for $j = 5$)

In stage V, we adjust students of the superclass for their backlogs.

- Courses available for backlogs:
  All courses offered in classes 1, 2, 3 and 4.
- Time required for regular courses:

$$T_5 = \sum_{i=1}^{m} t_{i5} = 0 \tag{2.18}$$

- Time available for backlogs:

$$A_5 = H_5 - T_5 = H_5 \tag{2.19}$$

.

- All the time-slots during a week are available for scheduling of backlogs.
- Repeat module II & I as in algorithm 2.2 and algorithm 2.1, respectively to make optimal allocation.

(f) **Optimal Policy of Stage V:**
Set of students adjusted for backlogs

$$= \sum No\ of\ students\ adjusted\ at\ each\ stage\ of\ the\ programme$$

$$= \sum S_{ij}\ at\ each\ stage \tag{2.20}$$

(The set of students at various stages are disjoint sets, as these students belong to a specific class)

2.3 Conclusion

In this chapter, a dynamic programming based model has been discussed by making each stage corresponding to one year of study. Maximum years for completing a degree in a programme is four years. Students who are not able to complete their programme in maximum of four years, are considered to be in fifth year who will not study with any of the regular courses.The links are formed across years where each year has only one section of students studying together. Thus, the links have been formed across the years. A sample data has been used to explain the above procedure step by step and is presented in Appendix A.5.

Chapter 3

Planning of Student Scheduling Across Sections

In the previous chapter, a dynamic programming based model has been discussed to explain the concept of making each stage corresponding to one year of study. Maximum years in a programme is four years. Students who are not able to complete their programme in maximum of four years are considered to be in fifth year who do not have to study any of the regular courses. The links have been formed across years where we have only one section of students, studying in each year. Thus, the links have been formed year wise. At the time of real implementation, it was noticed that students studying in same years can be studying in different set of subjects. This may be due to difference in scheme of subjects to be studied by students of different branches. Sometimes, students of same branch may also be studying different subjects due to the institute policy of distribution of students across different set of subjects in one semester.

This observation led to defining the stage at the level of section. In this way more students can be adjusted in same year as students in different sections can become part of the same link. Link in itself is of no use if students in higher classes are not free at the time of scheduling of a link. The provision made in time table for keeping higher classes free in the time slot of link is called cancellation. This setup gave rise to two more variations:

(i). The purpose of creation of links is to maximize the set of unique students using one of the following ways:
(a) Students having backlogs in all years higher than the year in which this subject is running, is treated as regular subject and
(b) Students having backlogs in just one year higher than the year in which this subject is running, is considered as regular subject.

(ii). After a cancellation has been made in a higher class against a link, the following two conditions exist:

[0]Guleria,Sanjeev Kumar, Lal, Arvind Kumar *"Interactive Decision Support System for planned student scheduling using same-time scheduling"*, Modern Physics Letters-B, May 2019.

(a) all the students in this class have not been adjusted and
(b) their is a scope in available hours for scheduling of this higher class section, so that we get an opportunity to create another cancellation which increases the number of students adjusted as a whole.

Both of these concepts are going to be used in the real time implementation. This is carried out by using dynamic programming approach considering section based staging. First concept is used to make two models namely: "All Year Model" and "Prior Year Model". The second concept of providing double cancellation makes these models multi-level. The links thus created from both of these models can be used to form timetable by following same-time scheduling.

This chapter explains the process of finding courses in such a manner that maximum number of the students can be allocated at least one backlog course as first priority. One more course may be adjusted for studying in addition to one course already identified for this purpose, if possible. This planning will generate inputs or constraints for the timetable so that most of the students' backlogs might be adjusted in created time table.

This chapter is organized as follows:- Section 3.1 explains the model and problem formulation of planning of student scheduling across sections while section 3.2 elaborates the applicability of algorithm for both the models using multi level dynamic programming approach. The proposed methods have been applied to real data of four semesters from Thapar Institute of Engineering and Technology and discussed in section 3.3 by explaining implementation method for the same. The results thus obtained are finally discussed in section 3.4.

3.1 Methodology

3.1.1 Description of student scheduling problem during offering of backlog courses

In the present problem, the students having a backlog course(s) need to be adjusted in the timetable in such a way that there is no clash between their regular courses and backlog courses while attending the lectures, tutorials and practicals of a particular course. The following assumptions are taken into account:

(i) A student is studying in one of the sections.

(ii) A normal student is one who is having no backlog course that can be offered in the

current semester.

(iii) A backlog student is having one or more backlog courses that are offered in the current semester.

(iv) Each course is being run in one or more sections.

(v) Each course is specified by three components, namely lecture(L), tutorial(T) and practical(P) and is associated with the number of hours required in a week. Each tutorial is held for one hour at a time (If two hours have been devoted for tutorials then it will be held at two different time slots). Generally, practicals are held for two or three hours in one session. These components serve the following two purposes:
(i) these add up to total number of hours consumed by this course; and
(ii) the variability in the values of L,T and P gives us another dimension of scheduling in parallel.

If two courses are running in parallel, then it is obvious that their lecture components are scheduled in the 'same time-slots', whereas the tutorials and practicals can be held in parallel provided one of the courses has no tutorial and the other course also has no practical.

3.1.2 Approach

As discussed in previous section, the main goal, to be achieved of the proposed problem is as follows:

(i) To accommodate as many backlog course requests as possible while using a minimum number of links per regular section and

(ii) assign all course-sections combinations to one of the possible links while maintaining feasibility.

(iii) utilize the existing infrastructure in terms of ongoing sections where students are studying.

These goals will help us in framing the constraints of the problem which can help in creating the timetable with the adjustment of backlogs. This process can simply be called creation of links. It is not possible to grant backlog course requests to all the students due to resource limitation of already created sections and number of hours for which timetable has to be made in the existing practice.

Other option for the student is to wait for next semester in which he has lesser load of his regular courses. If a student wants to study backlog in current semester in the case of a clash, then this can be achieved only by dropping a course in his regular section which is clashing. By following this approach, student will be creating one more backlog for next semester. Moreover, no extra faculty is required in this proposed method.

In another approach, the organization can provide additional faculty and class rooms to conduct extra classes for backlog courses.

The proposed work, thus, consists of the following objectives:
(a) Maximize the number of backlog courses requests granted,
(b) Maximize the utilization of links per regular section where the total number of hours spent are within limits and
(c) Look for any student missing this utilization and give them a chance to utilize another link.

3.1.3 Assumptions, Notations and basic terminology

Firstly, a multi level optimization model is formulated, aiming to maximize the number of students whose backlog course requests are granted while maintaining the necessary conditions. In order to formulate the problem, following notations have been used throughout the paper. For a particular institution there are a set of students, faculty, sections, courses. Let these sets be denoted by T, F, S and C, respectively. The set of links is denoted by L. The symbols t, f, s and c represent a particular element of the sets T, F, S and C, respectively. Similarly, l is one of the members of set L.

Besides this, there is a set of LTP. If $A_{ltp} \in$ LTP then one of the components of Lecture, Tutorial or Practical, hold. That is,

$$A_{ltp} = \begin{cases} L, & Lecture \\ T, & Tutorial \\ P, & Practical \end{cases}$$

L_c : Lecture hours for course c.

T_c : Tutorial hours for course c.

P_c : Practical hours for course c.

H_w : Maximum hours in a week for which time table is to be constructed.

H_c : Number of hours taken for course c.

H_s : Number of hours taken for section s.

H_l : Number of hours taken for a link l.

H_{sl} : Number of hours taken for a link l in parallel to which no course is to be scheduled.

H_{sl2} : Number of hours taken for a link l in parallel to which no course is to be scheduled in excess to H_{sl}.

B_{cl}^{as} : Number of unique backlog students if a course c is assigned to a link l in section s.

B_{cl}^{ays} : Number of unique backlog students in prior year if a course c is assigned to a link l in section s.

B_{sl}^{e} : Number of unique backlog students if section s is running Dummy course in parallel to courses assigned to link l.

B_{sl}^{e2} : Number of unique backlog students in excess to B_{sl}^{e} if section s is also kept unscheduled second time in parallel to courses assigned to link l when $Y_{sl}^{e} = 0$.

$$U_{ts} : \begin{cases} 1, & \text{if student t is studying in section s} \\ 0, & \text{otherwise} \end{cases}$$

$$V_{tc} : \begin{cases} 1, & \text{if student t is having backlog in course c} \\ 0, & \text{otherwise} \end{cases}$$

$$W_{cs} : \begin{cases} 1, & \text{if course c is associated with section s} \\ 0, & \text{otherwise} \end{cases}$$

$$X_{cl}^{as}: \begin{cases} 1, & \text{if course c is assigned with link l in section s} \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{sl}^{e}: \begin{cases} 1, & \text{if section s is running Dummy course in parallel to courses assigned to link l} \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{sl}^{e2}: \begin{cases} 1, & \text{if section s is running Dummy course in parallel to courses assigned to link l and } Y_{sl}^{e}=0 \\ 0, & \text{otherwise} \end{cases}$$

$$Z_{csf}^{ltp}: \begin{cases} 1, & \text{if faculty f is teaching course c in section s for a component of } A_{ltp} \\ 0, & \text{otherwise} \end{cases}$$

$$TL: \begin{cases} 1, & \text{if teaching load is ready} \\ 0, & \text{otherwise} \end{cases}$$

Adj	:	Adjustment = maximum number of extra courses that can be adjusted in a section
N	:	Number of links = Max Number of courses in a section + Adj
ST_c	:	Set of students having backlogs in course c
$ST_l(s)$	:	Set of students having backlogs in all courses attached to link l at stage of section s.
$ST_{cl}(s)$	:	Set of students having backlogs in course c and all other courses already attached to link l at the stage of section s.
$ST_{cl}^{e}(s)$	:	Set of students studying in section s having backlogs in course c and all other courses already attached to link l at the stage of section s.
$ST_{cl}^{e2}(s)$	:	Set of students studying in section s having backlogs in course c and all other courses not attached to link l at the stage of section s and $Y_{sl}^{e} = 0$.

ST_{yc}	:	Set of students who are just one year ahead in study having backlogs in course c.
$ST_l(ys)$	:	Set of students who are just one year ahead in study and having backlogs in all courses attached to link l at stage of section s.
$ST_{cl}(ys)$	:	Set of students who are just one year ahead in study and having backlogs in course c and all other courses already attached to link l at the stage of section s.
$ST^e_{cl}(ys)$	:	Set of students studying in section s who are just one year ahead in study and having backlogs in course c and all other courses already attached to link l at the stage of section s.
$ST^{e2}_{cl}(ys)$	:	Set of students studying in section s who are just one year ahead in study and having backlogs in course c and all other courses not attached to link l at the stage of section s where $\mathrm{Y}^e_{sl} = 0$.

Using the above notations and their descriptions, we have the following equations:

$$H_c = L_c + T_c + P_c \tag{3.1}$$

$$H_s = \sum_{c,s} H_c \tag{3.2}$$

$$H_l = Max(L_c) + Max(T_c, P_c) + Min(T_c, P_c) \tag{3.3}$$

3.1.4 Modelling and formulation of the problem of student scheduling across sections

This problem formulation can be best understood by the explanation of this multi-level optimization method. The objective function of link creation process can be stated as follows:

$$\begin{aligned}
\text{maximize} \quad & Z = \sum_{s=0}^{S} \left[\sum_{c \in C} \sum_{l \in L} B^{as}_{cl} X^{as}_{cl} \right] \\
\text{subject to} \quad & \sum_{s \in S} \sum_{c \in C} X^{as}_{cl} = 1, \quad \forall l \in L, \\
& \sum_{s \in S} \sum_{l \in L} X^{as}_{cl} = 1, \quad \forall c \in C, \\
& \sum_{s \in S} \sum_{l \in L} W_{cs} X^{as}_{cl} Z^{ltp}_{csf} = 1, \quad \forall\ f \in F,\ \forall\ c \in C,\ A_{ltp} = L,\ TL = 1, \\
& X^{as}_{cl} \geqq 0, \quad \forall\ s \in S,\ c \in C, \forall\ l \in L
\end{aligned} \tag{3.4}$$

In the **first level** shown in equation (3.4), links are created which determine the maximum number of students having backlogs fall in each link. This process, of course, is a simple assignment model with Null courses in sections where there are lesser number of courses than the links to be created. The objective function (3.4) maximizes the number of students whose backlogs could be adjusted. The first two constraints in (3.4) indicate that there will be only one course assigned to one link and vice-versa. The third constraint in (3.4) indicates that only one faculty will be allowed in one lecture link, that is, the same faculty may not be put accidentally in two courses which have to be held parallel in one link.

In **level 2**, we identify for each section teaching regular courses to backloggers, one link is singled out which adjusts maximum number of individual backloggers of this section. The maximization problem to obtain Y_{sl}^{e} has been formulated as:

$$\begin{aligned} \text{maximize} \quad & \left[\sum_{s=0}^{S} B_{sl}^{e} Y_{sl}^{e}\right] \\ \text{subject to} \quad & : \ B_{sl}^{e} \subset B_{cl}^{as}, \ \ \forall s \in S, \\ & Y_{sl}^{e} = 1, H_{sl} = Max(H_l), \ \ \forall l \in L, \ H_w - H_s \geqq H_l, \\ & Y_{sl}^{e} \geqq 0, \ \ \forall s \in S, \ \forall \ c \in C, \ \forall \ l \in L \end{aligned} \tag{3.5}$$

In **level 3**, we identify whether there are still any backloggers left in this section who could not be adjusted in the level 2 from (3.5). Out of the pending backloggers, we find the link that adjusts the maximum number of pending backlog students. In order to determine Y_{cl}^{e2}, we solve the following maximization problem:

$$\begin{aligned} \text{maximize} \quad & \left[\sum_{s=0}^{S} B_{sl}^{e2} Y_{sl}^{e2}\right] \\ \text{subject to} \quad & : \ B_{sl}^{e2} \subset B_{cl}^{as}, \quad \forall s \in S, \\ & Y_{sl}^{e2} = 1, H_{sl2} = Max(H_l), \quad \forall l \in L, \\ & Y_{sl}^{e} = 0, \quad H_w - H_s - H_{sl} \geqq H_l, \\ & Y_{sl}^{e2} \geqq 0, \quad \forall \ s \in S, \ \forall \ c \in C, \ \forall \ l \in L \end{aligned} \tag{3.6}$$

Thus, each level corresponds to all objectives mentioned at (a-c) in subsection 3.1.2. The main challenge, in solving this model, is the calculation of these three coefficients: B_{cl}^{as}, B_{sl}^{e} *and* B_{sl}^{e2} each corresponding to a different level. The model is deterministic but could be better understood and solved in terms of dynamic programming as these coefficients have to be calculated at each stage which is identified by a different section.

3.1.5 Model Computation

Keeping these in view, the following two models have been proposed for level one computation using dynamic programming. The criteria using which students count is considered for calculating the variable values of B_{cl}^{as} is the main difference in both the models. The values of B_{sl}^{e} and B_{sl}^{e2} are calculated using level 2 and 3 depend upon the data of B_{cl}^{as}. In both of these models, let s be stages of processing, where $s \in S$ and each link l has one of the following states:

(i) First Dummy Course of section s	–	First provision of backlog for students of section s
(ii) Second Dummy Course of section s	–	Second provision of backlog for students of section s
(iii) Course c of section s	–	All courses running in section s
(iv) Null Course	–	Assigned when we have a section having lesser number of courses than maximum number of courses in any section

3.1.5.1 All Year Model(AYM)

This model computes the coefficients: $B_{cl}^{as}, B_{sl}^{e}, B_{sl}^{e2}$ without making any distinction between the year in which a section falls. This model works on the presumption that the students may be having backlogs in any of the previous years. As defined above, each stage is represented by s, whereas each course in a link has one of the four defined states from (i) to (iv) discussed in subsection 3.1.5.

Assume that, $ST_{cl}(s) = ST_c$ for $s = 0$. We define successively as follows for $s = 1, \ldots, S$:

$$ST_{cl}(s) = ST_l(s-1) \bigcup ST_c, \quad \forall c \in C, W_{cs} = 1, \ \forall l \in L \tag{3.7}$$

$$B_{cl}^{as} = \mid ST_{cl}(s) \mid, \forall c \in C, U_{ts} = 1, V_{tc} = 1, W_{cs} = 1 \ \forall l \in L \tag{3.8}$$

$$B_{sl}^{e} = \mid ST_{cl}^{e}(s) \mid, \ where \ ST_{cl}^{e}(s) \subset ST_{cl}(s), X_{cl}^{as} = 1, \ \forall l \in L \tag{3.9}$$

$$
\begin{aligned}
\text{maximize} \quad & \sum_{c\in C}\sum_{l\in L} B^{as}_{cl} X^{as}_{cl} \\
\text{subject to} \quad & Y^{e}_{sl}, \quad Y^{e2}_{sl} = 0, \\
& \sum_{l\in L} Y^{e}_{sl} = 1, \quad \sum_{l\in L} Y^{e2}_{sl} = 1, \\
& \sum_{l\in L} W_{cs} X^{as}_{cl} Z^{ltp}_{csf} = 1, \quad \forall f \in F,\ \forall\ c \in C,\ A_{ltp} = L, \\
& X^{as}_{cl}, Y^{e}_{sl}, Y^{e2}_{sl} \geqq 0, \quad \forall\ c \in C,\ l \in L
\end{aligned}
\tag{3.10}
$$

$$
\begin{aligned}
\text{maximize} \quad & \sum_{l\in L} B^{e}_{sl} Y^{e}_{sl} \\
\text{subject to} \quad & Y^{e}_{sl} = 1, \quad H_{sl} = Max(H_l),\ H_w - H_s \geqq H_l, \\
& \forall l \in L, \quad , \\
& B^{e2}_{sl} = \mid ST^{e2}_{cl}(s) \mid , where \quad ST^{e2}_{cl}(s) \subset ST_{cl}(s), \\
& ST^{e2}_{cl}(s) \not\subset ST^{e}_{cl}(s)
\end{aligned}
\tag{3.11}
$$

$$
\begin{aligned}
\text{maximize} \quad & \sum_{l\in L} B^{e2}_{sl} Y^{e2}_{sl} \\
\text{subject to} \quad & Y^{e2}_{sl} = 1, \quad H_{sl2} = Max(H_l), \forall l \in L, \\
& Y^{e}_{sl} = 0, \quad H_w - H_s - H_{sl} \geqq H_l
\end{aligned}
\tag{3.12}
$$

3.1.5.2 Prior Year Model(PYM)

This model differs from the previous model with a presumption that most of the students may be having backlogs only in the last previous year. This model computes the coefficients: $B^{as}_{cl}, B^{e}_{sl}, B^{e2}_{sl}$ by making distinction between the year in which a section falls. So, the sections are ordered year wise and the coefficients are reset at the time of year change. All the sections are divided year wise and hence each year contains a small set of sections. Let y be different years of study where y∈Y. As defined above in the start of section, each stage is represented by s, whereas each course in a link has one of the four defined states from (i) to (iv) discussed in subsection 3.1.5.
Assuming $ST_{cl}(ys) = ST_{yc}$ for $s = 0$, we define successively the following for $y = 1, \ldots, Y$, and $s = 1, \ldots, S$:

$$
ST_{cl}(ys) = ST_l(ys-1) \bigcup ST_c, W_{cs} = 1,\ \forall l \in L\ \forall c \in C, \tag{3.13}
$$

$$
B^{as}_{cl} = \mid ST_{cl}(ys) \mid, U_{ts} = 1, V_{tc} = 1, W_{cs} = 1,\ \forall l \in L,\ \forall c \in C \tag{3.14}
$$

$$B^e_{sl} =\mid ST^e_{cl}(ys) \mid,\ where\ ST^e_{cl}(ys) \subset ST_{cl}(ys), X^{as}_{cl} = 1,\ \forall l \in L \tag{3.15}$$

$$\begin{aligned}
&\text{maximize} \quad \sum_{c\in C}\sum_{l\in L} B^{ays}_{cl} X^{as}_{cl} \\
&\text{subject to} \quad Y^e_{sl},\ \ Y^{e2}_{sl} = 0, \\
&\qquad \sum_{l\in L} Y^e_{sl} = 1,\ \ \sum_{l\in L} Y^{e2}_{sl} = 1, \\
&\qquad \sum_{l\in L} W_{cs} X^{as}_{cl} Z^{ltp}_{csf} = 1,\ A_{ltp} = L\ \ \forall f \in F,\ \forall\ c \in C, \\
&\qquad X^{as}_{cl}, Y^e_{sl}, Y^{e2}_{sl} \geqq 0,\ \ \forall\ c \in C,\ l \in L
\end{aligned} \tag{3.16}$$

$$\begin{aligned}
&\text{maximize} \quad \sum_{l\in L} B^e_{sl} Y^e_{sl} \\
&\text{subject to} \quad H_{sl} = Max(H_l),\ \ \forall l \in L, \\
&\qquad H_w - H_s \geqq H_l,\quad Y^e_{sl} = 1, \\
&\qquad B^{e2}_{sl} =\mid ST^{e2}_{cl}(ys) \mid, where \quad ST^{e2}_{cl}(ys) \subset ST_{cl}(ys), \\
&\qquad ST^{e2}_{cl}(ys) \not\subset ST^e_{cl}(ys)
\end{aligned} \tag{3.17}$$

$$\begin{aligned}
&\text{maximize} \quad \sum_{l\in L} B^{e2}_{sl} Y^{e2}_{sl} \\
&\text{subject to} \quad Y^{e2}_{sl} = 1,\ \ H_{sl2} = Max(H_l), \\
&\qquad \forall l \in L,\ Y^e_{sl} = 0,\ \ H_w - H_s - H_{sl} \geqq H_l
\end{aligned} \tag{3.18}$$

All the courses running in semester are mapped with set of students having backlogs in each of them. The Hungarian method of assignment model will be used repeatedly to assign courses of one section with the another one. Whichever assignments have been made, they continue to include unique students in the course-section combinations. As the Hungarian method only works for minimization, while our aim is to maximize the number of students in set. So, we need to subtract the unique number of students from a large number, so that solution of minimization problem leads to maximization. This idea will be used while writing the algorithm.

3.2 Algorithm

1. Step Inititalize

   Require:

   1: *backlogs*, {List of backlog course pending of each backlog student}

2: $ListOfCourses[Section]$, {List Of Courses running in session $foreach$ Section for which timetable is to be made}
3: $SectionTime[Section]$, {number of hours $foreach$ Section consumed by all the courses running in this Section adding all the lecture, tutorial and practical hours}
4: $MaxTime$, {number of hours for which timetable is to be constructed}
5: $LinkBuffer$ {Extra slots to be kept for making Links}
6: $MaxValue \leftarrow 999$ {Assuming that number of backlog students in a course is less than this number}
7: $NumberOfLinks \leftarrow MaxCount[Count[ListOfCourses[Section]]]+LinkBuffer$

8: $JointSection \leftarrow 1stSection$
9: **for all** $Link = 1$ **to** $NumberOfLinks$ **do**
10: $SetOfCourses[Link] \leftarrow ListOfCourses[JointSection][Link]$
11: $Set[Link] \leftarrow SetofBacklogStudentshavingbackloginSetOfCourses[Link]$
12: **end for**

2. Step Main Loop

1: **for all** $Section = 2$ **to** n **do**
2: $PivotSection \leftarrow Section$
3: **for all** $PivotLink = 1$ **to** $NumberOfLinks$ **do**
4: $SetOfCourses[PivotLink] \leftarrow ListOfCourses[PivotSection][PivotLink]$
5: **end for**
6: **end for**
7: **for all** $JointLink = 1$ **to** $NumberOfLinks$ **do**
8: **for all** $PivotLink = 1$ **to** $NumberOfLinks$ **do**
9: $ArraySet[JointLink][PivotLink] \leftarrow UniqueSetofBacklogStudentshavingbackloginSet$

10: UNION $SetofBacklogStudentshavingbackloginSetOfCourses[PivotLink]$

11: $WorkArraySet[JointLink][PivotLink] \leftarrow MaxValue-Count[ArraySet[JointLink][Pi$
12: **end for**
13: $ResultSet[][] \leftarrow MunkresAssigmentAlgorithm[WorkArraySet[][]]$
14: **for all** $Link = 1$ **to** $NumberOfLinks$ **do**
15: $SetOfCourses[Link] \leftarrow ResultSet[][]$
16: $Set[Link] \leftarrow UniqueSetofBacklogStudentshavingbackloginSetOfCourses[Link]$
17: **end for**

```
18:  for all Link = 1 to NumberOfLinks do
19:    LinkHours[Link] ← Minimum[NumberofHours]bySetOfCourses[Link]
20:  end for
21:  for all Link = 1 to NumberOfLinks do
22:    if (LinkHours[Link] < (MaxTime − SectionTime[PivotSection]]] then
23:      ValidLinks[Link] ← 1
24:    else
25:      ValidLinks[Link] ← 0
26:    end if
27:  end for
28:  MaxLink ← MaxLinkHours ← StudentsAdjusted ← 0
29:  for all Link = 1 to NumberOfLinks do
30:    StudentsAdjustedIn[Link] {contains the number of students in Section
       who have backlogs in courses of Link}
31:  end for
32:  if Anystudentispresentlystudyinginsection then
33:    for all Link = 1 to NumberOfLinks do
34:      if ValidLink[Link] == 1 then
35:        if (StudentsAdjustedIn[Link] > StudentsAdjusted] then
36:          StudentsAdjusted ← StudentsAdjustedIn[Link]
37:          MaxLink ← Link
38:          MaxLinkHours ← LinkHours[Link]
39:        end if
40:      end if
41:    end for
42:    for all Link = 1 to NumberOfLinks do
43:      if (LinkHours[Link] < (MaxTime−(MaxLinkHours+SectionTime[PivotSection]]]
         then
44:        StillValidLinks[Link] ← 1
45:      else
46:        StillValidLinks[Link] ← 0
47:      end if
48:    end for
49:    PendingSetOfStudents ← SetOfStudentsinSectionwhicharenotcoveredbysubjectsinti

50:    for all Link in StillValidLinks do
51:      PendingCount[Link]isidentifiedfromthePendingSetOfStudentsinthecurrentSectio
```

```
52:       end for
53:       SecondMaxLink ← SecondMaxLinkHours ← PendingStudentsAdjusted ←
          0
54:       for all Link = 1 to NumberOfLinks do
55:         if StillValidLinks[Link] == 1 then
56:           if (PendingCount[Link] > PendingStudentsAdjusted] then
57:             PendingStudentsAdjusted ← PendingCount[Link]
58:             SecondMaxLink ← Link
59:             SecondMaxLinkHours ← LinkHours[Link]
60:           end if
61:         end if
62:       end for
63:     end if
64: end for
```

3. Step Remove Extra

```
1: for all Link = 1 to NumberOfLinks do
2:   for all Section = 2 to n do
3:     for all student do
4:       {getting adjusted in each course are identified which do adjust any back-
         logs using MaxLink or SecondMaxLink}
5:       NumberOfBacklogs[Course] ← NumberOfBacklogs[Course] + 1
6:     end for
7:   end for
8: end for
9: for all Link do
10:   if NumberOfBacklogs[Course] = 0 then
11:     ThisCourseisremovedfromthisLink
12:   end if
13: end for
14: return Links of course sections and vacant course sections which can be sched-
    uled at same time to maximize backlog allocation
```

3.3 Implementation of AYM and PYM

The purpose of this process is to find course-sections which can be scheduled together such that highest number of students can get adjusted in one of these courses, whose

own section is kept free of any classes at this time. In order to implement the proposed algorithm, following inputs are required:

- List of students and their pending backlog courses which are running in the semester under consideration
- Sections for which timetable is to be made
- Scheme of courses which includes Courses running in each of those sections and their LTP
- Teaching load of the semester. If this is not ready, there should be provision to work without this also.

After successfully implementing the algorithm, we get the following outputs:

- Sets of list of course-section combinations which should be scheduled together and termed as Links.
- Each of these lists may contain list of sections which have to be kept free when these course-section combinations are scheduled for timetabling. This list is termed as cancellations.
- List of students whose backlogs will be adjusted if time table is made following these constraints.

3.4 Tests and Results

The proposed system was tested with the data provided by the Thapar Institute of Engineering and Technology for four consecutive semesters. A program was made in Perl language and implemented as a web interface. The assignment problem solves the assignment at each section level using Perl module of Munkres Algorithm [47, 48].

The data collected (shown in Table 3.1) corresponds to the list of actual students having backlogs and the actual courses running in these semesters. The results obtained were compared with the actual number of backlog students adjusted in the registration of these semesters (shown in Table 3.2). This table shows that in semeter 1, all years model(AYM) adjusts 723% of students in comparison to actual students adjusted in this semester; while this number rises to 745% of students adjusted in prior year model(PYM). Conversely, in semester 4, the rise is minimum in all data sets giving a 431% of students adjusted using AYM in comparison to actual data and 421% stu-

dents adjusted using the PYM. However, these differences are minimum in all data sets, but this is more than four times those adjusted actually and more than seven times as highest in semester 1. Other results are between these two extremes. This shows a lot of improvement in adjusting backlog courses in those semesters in comparison to actual number of students adjusted. The percentage of students having backlogs in just the previous years is shown in Table 3.3.

This comparison clearly shows the advantage of the proposed system of backlog adjustment vis a vis the actually adjusted backlogs. The increase in number of students adjusted is substantially high. As regards the two models are concerned, both have their benefits in terms of percentage of students adjusted in the created timetable for different datasets. The percentage of students whose backlogs are adjusted depends upon how the backlogs are spread over different years. If they are spread over years, the AYM will give better results. If the backlogs are more concentrated in the last previous years only the PYM will be adjusting more students. The results shown in Table 3.3 exhibit that the percentage of students having backlogs in just the previous years is more in case of first two semesters. There is direct correlation of percentage of students adjusted in the PYM. The AYM outshines where half of students having backlogs in just the previous years.

The comparison graph shown in Figure (3.1) clearly shows the advantage of both algorithms.

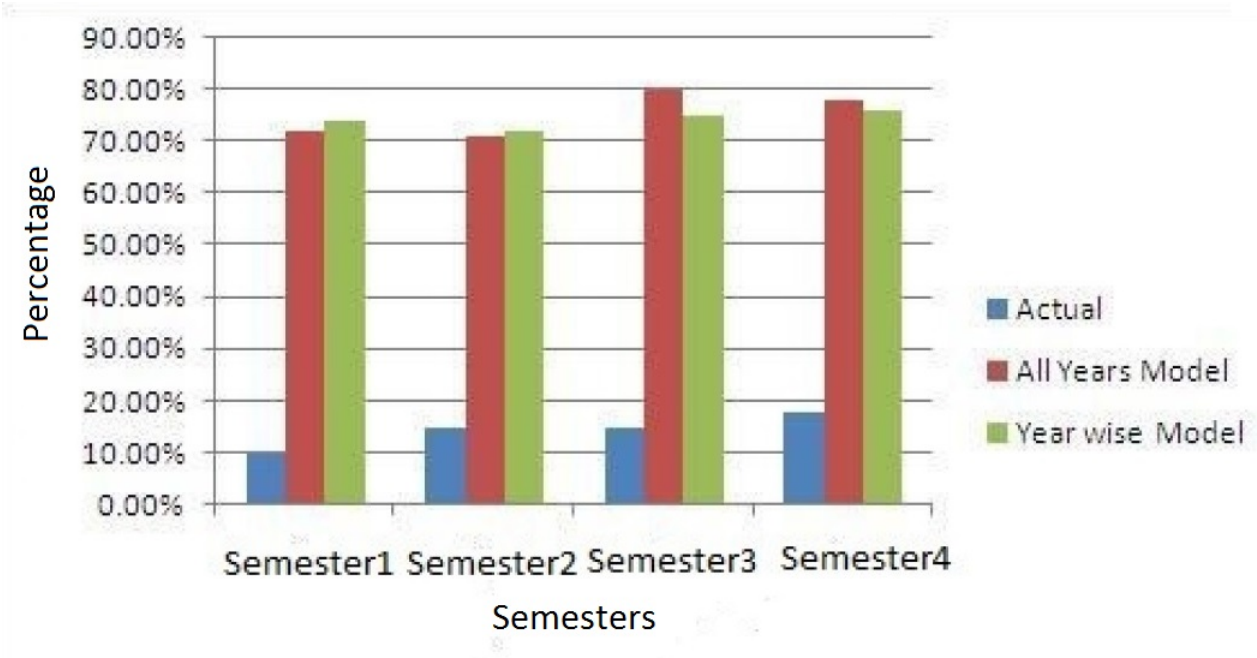


Figure 3.1: Comparison of the AYM and PYM with actual figures of backlog adjustment

Table 3.1: All Four Semester Data of TIET

SEMESTER	SEMESTER CODE	SEMESTER Description	TOTAL backlog Students	TOTAL Running Courses
Semester1	0708ODDSEM	Odd Semester 2007-08 Session	221	375
Semester2	0708EVESEM	Even Semester 2007-08 Session	190	374
Semester3	0809ODDSEM	Odd Semester 2008-09 Session	354	754
Semester4	0809EVESEM	Even Semester 2008-09 Session	315	740

Table 3.2: The adjustment of backlog courses in percentage

SEMESTER	Actual adjusted backlogs	All Years Model	Prior Year Model	Best
Semester1	9.95	71.94	74.21	PYM
Semester2	14.74	71.05	72.11	PYM
Semester3	14.97	79.94	74.86	AYM
Semester4	18.09	78.09	76.19	AYM

Table 3.3: The percentage of students having backlog courses in just the previous years

SEMESTER	Percentage	Approximately
Semester1	74.87	three-fourths
Semester2	66.75	two-thirds
Semester3	52.95	half
Semester4	47.59	half

Chapter 4

Supporting Student Registration Across Sections

This chapter describes the model for interactive student scheduling. Interactive student scheduling provides a user interface that helps in identifying clash-free set of course-sections to be registered based on the constructed time-table. Interactive Student Scheduling uses the heuristics along with generation of Structured Query Language to find clash-free timetables as well as verifying the suggested timetable decided by student based on already constructed time-table. The algorithms for implementing this model are described. The implementation details for supporting student registration across sections are provided for testing the models developed for this decision support system. Results of implementations are further analysed for their efficiency. These models are judged on the basis of various options provided by the decision support system to achieve the goals. Also, these provide a working process that helps in decision making process at the time of registration of student. At present, this is being done manually and no other methods are available. So, it is not possible to measure the efficiency of new models against anything. These models, if successfully implemented, will provide a quality of service (QoS) to the users.

The second objective is to create an algorithm with a capability of searching a clash free timetable for a student in constructed time table leading to his registration. In other words, to develop an algorithm that can verify, the feasibility of a proposed set of courses for registration of a student. If all the components of courses are available to the student without any clash, then the proposal is considered to be feasible. After that, the students are allowed to register for the proposed courses.

The models, thus developed, have been applied to engineering students at TIET, Patiala (India) for registration as a case study. Presently, there are 8500 students in its undergraduate and post graduate programmes. TIET offers courses in Biotech, Chemical, Civil, Computer Science, Electrical, Electronics & Communication, Electronics Instrumentation and control, Mechanical, Mechatronics and Mechanical Production

[0]Sanjeev Kumar Guleria, Arvind Kumar Lal, *"Supporting Student registration across sections using a curriculum based existing time-table"*, International Journal of Advanced Science and Technology Volume 29, No. 3 pp.5521-5531.

Engineering streams. It also offers post graduate programmes in pure sciences, computer applications and various streams in engineering.

At undergraduate level, many of the courses in the first four semesters are common to engineering students of all branches. Electives and optional courses, in later semesters, are available to more than one branch of students. The institute shares faculty and infrastructure wherever possible for an economical and efficient operation of its programmes. The students are divided into sections on the basis of programmes and branch. Each section has around 20-30 students. A timetable is prepared by a time-table committee and displayed on the date of registration. The students get themselves registered for the courses they have to study during the semester and note their timetable as per alloted section.

The registration process becomes tedious for students having one or more backlogs. A backlog arises when a student is not able to pass a course when it was first offered to him during his studies. Thus, students may have (0 - n) backlogs, where n is number of maximum backlogs a student can carry through the course. After clearing all backlog courses, students will be eligible to get their degree within permissible time limit as per the institute rules. Students with zero backlogs get their registration done for all the courses offered in their section during the semester and these course-teacher-room combinations appear in their weekly time-table. However, the students with one or more backlogs are offered an "Add or drop facility", through which they can register for courses they wish to study during the semester. They can add courses or drop some and add other courses to the normal courses offered in the timetable.

The students are required to ensure that the set of courses, chosen by them, is clash-free and all its components in terms of lectures, tutorials and laboratory hours are available to him for study. The time-table proposed by a student, that is, drawn from the master time-table is to be verified and approved by the designated faculty incharge, to enable the students to take up his proposed courses for study during the semester. The verification and registration of such students is highly time consuming and each student's set of courses need to be checked for validity as per the given time-table.

Earlier, we [49] worked on a problem in a pre timetabling effort targeted at making adjustments in the time table to be constructed so that clash free environment can be made across sections in courses having more backlogs. In the previous chapter 2 and 3, we have discussed on planning of student scheduling across years and sections. These approaches create a scenario for a large number of backlog students in which their backlog subjects can be adjusted in time-table.

This is dependent upon the data of backlog students available at the time of running the algorithms. The present algorithm will work whether such an approach is used or not. Though, the results will be better if it is used. Further, the proposed system can search for combinations that are not scheduled purposely. To handle the problem of identification of clash free sets, case of one student is taken at a time. In the proposed algorithm, our approach will be to get first of all the data relevant to a particular student. The study involved methodology of the problem, and explanation of the algorithm; followed by testing of the algorithm and the outcome.

This chapter is organised as follows: The basic terminologies and assumptions related to the topic are discussed in section 4.1. Section 4.2 deals with both the algorithms. The results, thus obtained, are finally analysed in section 4.3 with the help of tables and figures.

4.1 Basic Terminology and assumptions

The following are some of the basic terminologies used in algorithms:-

4.1.1 Student data

There will be a set of student data having his/ her roll number and already finalized section. The sectioning is made with a purpose of distributing students so that they can be accommodated in rooms. In case this chosen section has clashes with the intended backlog course, student can search for another combination by changing his/her own section.

4.1.2 Student backlog data

The details of backlog courses earned by the student are available in one data set. Student will be shown choices of only those backlog courses which are running in the current semester as per scheme of courses and constructed timetable. All the sections, where each course is running, are also shown to the student.

4.1.3 Combination set

Combination set is a unique timetable string used for identifying each of the generated timetables based on different sections for our concerned courses. This combination set in a form of a string, attached to each course-section combination, participating to form different timetables to identify them at any later stage. For example, this string is of

the form $(x_1, x_2, \ldots, x_n) = (00000123)$, where n is the number of courses in the search criteria chosen by student. In this example, a string is used to depict a unique string for 8 courses and

$$x_i = \begin{cases} 0 & first\ section \\ 1 & second\ section \\ 2 & third\ section \\ \vdots & \\ 8 & ninth\ section \\ 9 & tenth\ section \\ A & eleventh\ section \\ B & twelfth\ section \\ \vdots & \\ Z & thirty\ sixth\ section \\ a & thirty\ seventh\ section \\ \vdots & \\ z & sixty\ second\ section \end{cases} \tag{4.1}$$

In this way we can handle courses where we have upto sixty two sections for each course.

4.1.4 Timetable slots

As we have five day time table for 9 slots per day, we have made forty five slots. Due to this reason, we create a string of zeroes and ones for around forty five places. This string is of the form $s_{45}, s_{44}, s_{43}, ..s_1) = (000000001000000001000000001000000001000000001)$. This string values are one of these two values

$$s_i = \begin{cases} 0, & free\ time\ slot \\ 1, & used\ time\ slot \end{cases} \tag{4.2}$$

Each slot of string signifies a slot corresponding to one particular day and period combination and shows whether this slot is free or used in timetable. Their mapping is shown in Table 4.1. Using this notation we can define any of the following timetables:
(i) Main Time table: This is a set which consists of different set of timetable slots corresponding to each section for regular courses of each student.
(ii) Temporary Time table: This set will be a set of timetable slots of backlog courses of student for each of the sections in which this course is running.

(iii) Valid Time table: If there is no clash between $MainTimetable$ and $TemporaryTimetable$ then a union of both of these time tables becomes a $ValidTimetable$.

$$ValidTimetable = MainTimetable \cup TemporaryTimetable \tag{4.3}$$

4.1.5 Assumptions

The following assumptions have been made for the model of supporting student registration across sections:

1. The timetable has been constructed for whole institute. Time table of each section is clash-free.
2. We assume that one feasible timetable will be ready as per curriculum. In the timetable a course might be running in one or multiple sections.
3. The slots are identified by two variables day(d) and period(p).
4. There are a set of students, faculty, sections, courses. Let these sets be denoted by T, F, S *and* C respectively. The symbols t, f, s and c, represent a particular element of the sets T, F, S and C, respectively.
5. The type of contact in each slot is represented as one of the values of ltp. So,

$$ltp = \begin{cases} L, & Lecture \\ T, & Tutorial \\ P, & Practical \end{cases}$$

6. Each course, section, ltp, day, period combination is assigned to one faculty and room.
7. In this way, the set TT will be timetable consisting of repeated sets of course, section, ltp, day, period, faculty, room, which will become the input data needed for search algorithm to work.
8. This timetable will be stored in the database for the algorithm to work effectively.

4.2 Algorithm for Searching clash free schedule

Both of Expand All Algorithm (EAA) and Merge and Consolidate Algorithm(MCA) require these steps as initial part

Require:

1: *timetable*: {timetable constructed for institute}
2: *student*:{student having backlogs whose case is being processed}
3: *backlogs*: {List of backlog courses pending for this *student*}
4: *section*: {section of this *student*}
5: $ListOfCourses[section]$: {List Of Courses running in session foreach Section for which timetable had been constructed}
6: Get a set of courses that a student is going to study
7: With each course student tags one section in which it will be studied. Student tags section 'ANY' with the courses whose section need to be searched.

4.2.1 Expand All Algorithm(EAA)

In this algorithm, we have to identify a set of course sections which are clash-free in an existing timetable for a particular student who has to study a set of courses. The algorithm has been explained in flowchart as shown in Figure (4.1). For this purpose, all possible timetables are created using notation for combination sets to identify each of them uniquely.

1. Generate all possible combination of timetables for each of the sections where these courses are running.
2. Identify those time tables which clash with one another course.
3. Remove these identified timetables which have a clash within them.
4. The remaining sets of timetables are clash free.
5. One of them is assigned to student after using randomization.

4.2.2 Merge and Consolidate Algorithm (MCA)

In this algorithm, a set of timetable slots as defined earlier have been created as intermediate data. The algorithm has been explained in a flowchart as shown in Figure (4.2).

4.3 Results and Analysis

The implementation has been made in MS Windows based environment using Perl as a scripting language. As this problem handles data related to timetable, Sql Server has been used for this purpose. The input screen that takes options from student as an Student Registration Form, shown in appendix (A.7). In this chapter, dataset from semester 2, of 2005-06 session is used for comparison. Both approaches for verification of the availability of proposed set of courses by a student in the given time-table were programmed and

1: First extract a set of timeslot from timetable as per the course sections selected by student. Let this be called 'MainTimetable'. While compiling this MainTimetable do check if the newly added time slot is not already in the set , otherwise we have found clash at this stage and course-sections set supplied by student is having clashes.
2: Take one of the courses from ANY set and identify those sections whose timeslots do not have any clash with timetable already selected
3: Merge the time slots of non-clashing course section combinations with the Main-Timetable and make new timetable sets as ValidTimetables
4: Take next course from ANY set . Identify its course section combinations that do not have any clash with the MainTimetable. Next check with each of the ValidTimetables to find clash free sets. Merge the clash free sets found with the ValidTimetables .
5: Generate all possible combination of timetables for these courses.
6: One of them is assigned to student.

were used during registration of students at TIET in semester 2, 2005-06 session. This generated timetable of institute had a faculty strength of about 135, 39 lecture and tutorial rooms with capacities varying from 25-125 students, 47 laboratories common and specific to various branches of engineering. This was generated for 88 sections in all for around 2522 students. The results are discussed and given in conclusions.

EAA algorithm works by following a data base centered approach as some of the code runs as a stored procedure to carry out the steps. Both the algorithms were run with actual data for around 73 cases where this was a mix of two types of cases. First case deals with the candidate who knew beforehand by looking at the timetable about the section in which backlog course was to be studied which needed the verification of schedule combination. Second case dealt with the candidate to whom the section of backlog course was not known who needed the searching of appropriate combination. The times taken in verification set of 59 cases is shown in figure (4.3) and the times taken to search schedules in 14 cases is depicted in figure (4.4).

After running the EAA algorithm successfully for 59 cases as per data from figure (4.3), in verification mode, the average CPU time has come out to be 0.0252 seconds whereas average CPU time comes out to be 0.0024 seconds in case of MCA algorithm. This may be of interest that in 50 cases this time is zero seconds. In the shortlisting mode of 14 cases as per data from Figure (4.4), average CPU time has come out to be 0.0313 seconds using EAA algorithm. in comparison to this average CPU time in case of MCA algorithm comes out to be 0.0104 seconds. This clearly shows the advantage of MCA algorithm over the EAA algorithm.

Table 4.1: The equivalence of slot with corresponding day and period combination in MCA and EAA algorithm

Slot	1	2	3	...	9	10	11	12	...	29	...	44	45
Day	1	1	1		1	2	2	2		4		5	5
Period	1	2	3		9	1	2	3		2		8	9

Table 4.2: The detail of shortlisting in MCA algorithm

Case	Total	Shortlisted	Number of courses in any set	Percentage of shortlisting
1	5	4	3	80
2	8	2	1	25
3	8	2	1	25
4	11	4	1	36.36
5	11	4	1	36.36
6	11	1	1	9.09
7	18	2	1	11.11
8	117	27	3	23.07
9	143	15	2	10.48
10	242	72	2	29.75
11	242	76	2	31.4
12	1936	82	3	4.23
13	2178	296	3	13.59
14	10648	303	4	2.84

Table 4.3: The detail of shortlisting in EAA algorithm

Case	Total	Shortlisted	Number of courses in any set	Percentage of shortlisting
1	5	3	3	60
2	1	1	1	100
3	8	2	2	25
4	1	1	0	100
5	11	1	1	9.09
6	11	1	1	9.09
7	18	5	1	27.77
8	3	1	1	33.33
9	13	9	1	69.23
10	1	1	0	100
11	4	1	1	25
12	22	8	1	36.36
13	11	9	1	81.81
14	22	1	1	4.54

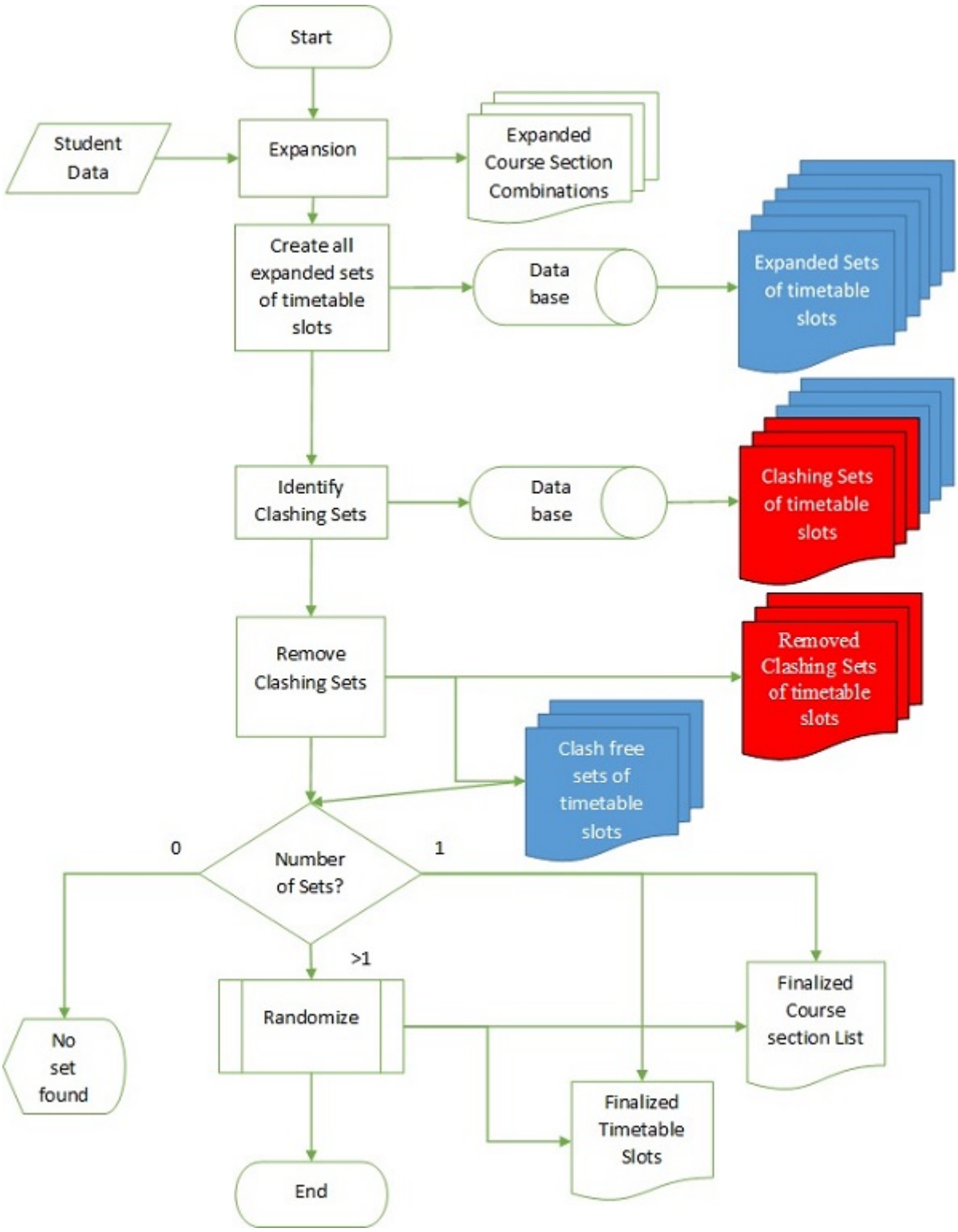


Figure 4.1: Flowchart for EAA Algorithm

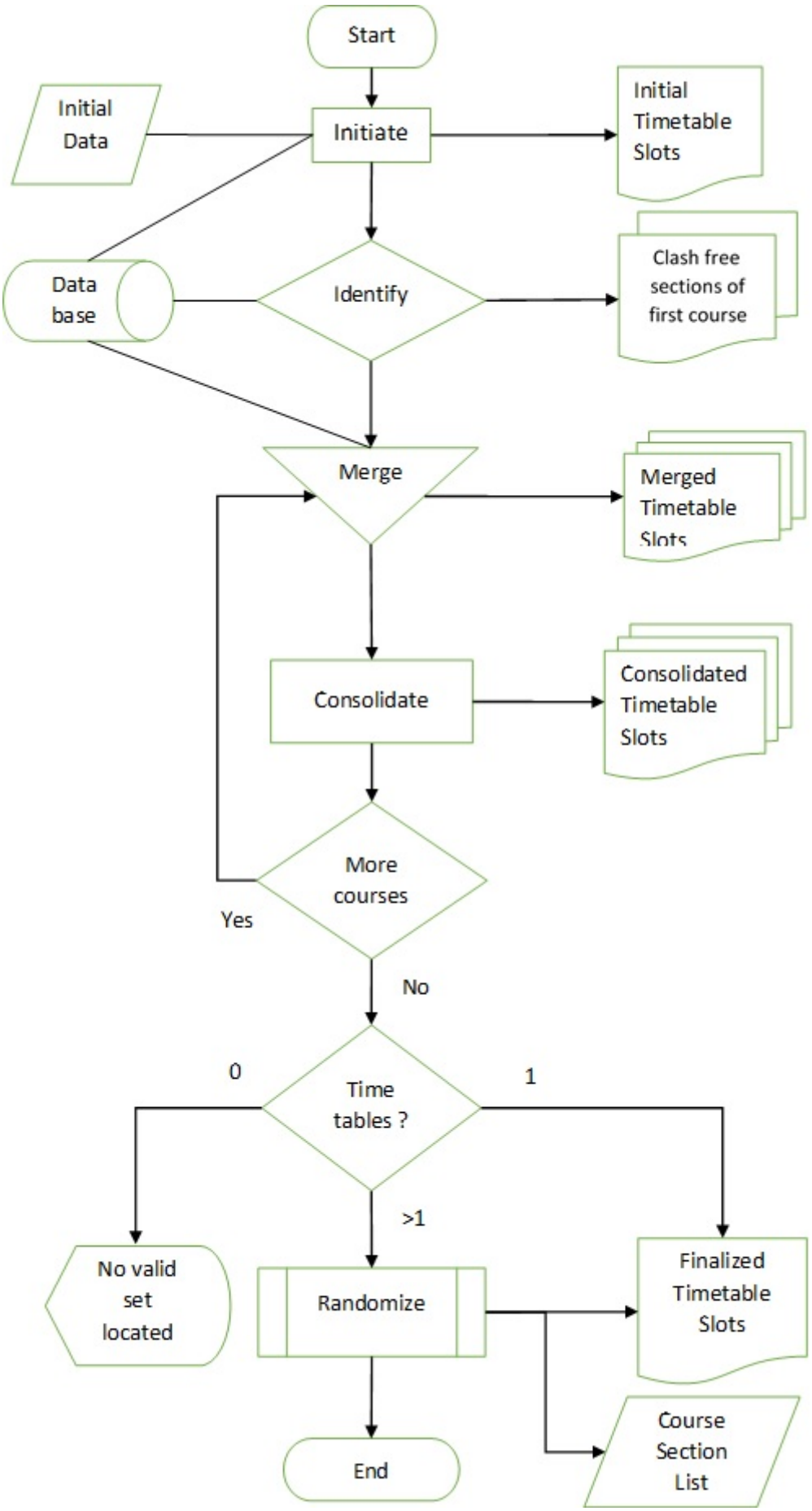


Figure 4.2: Flowchart for MCA Algorithm

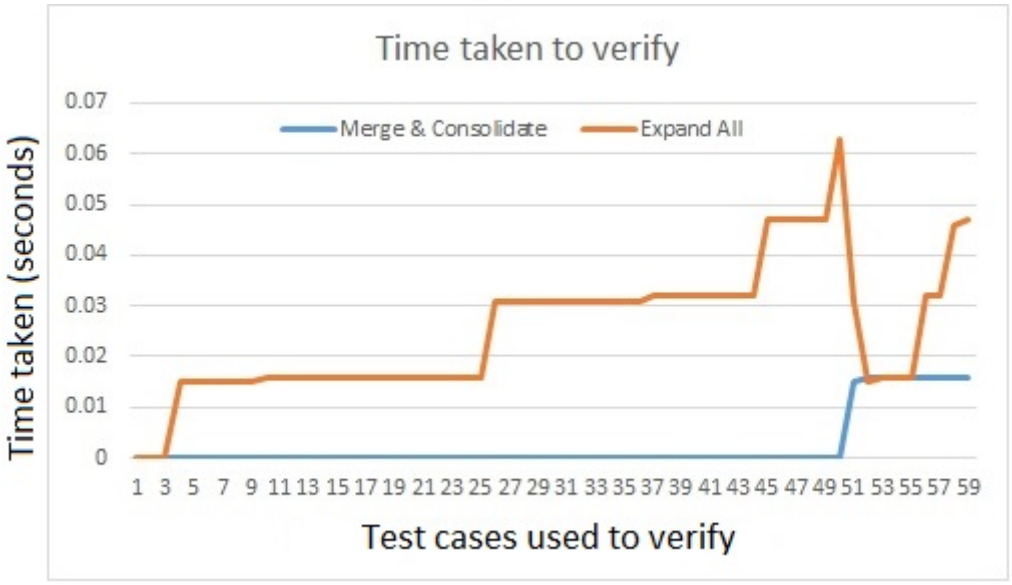


Figure 4.3: CPU Time taken to verify schedules

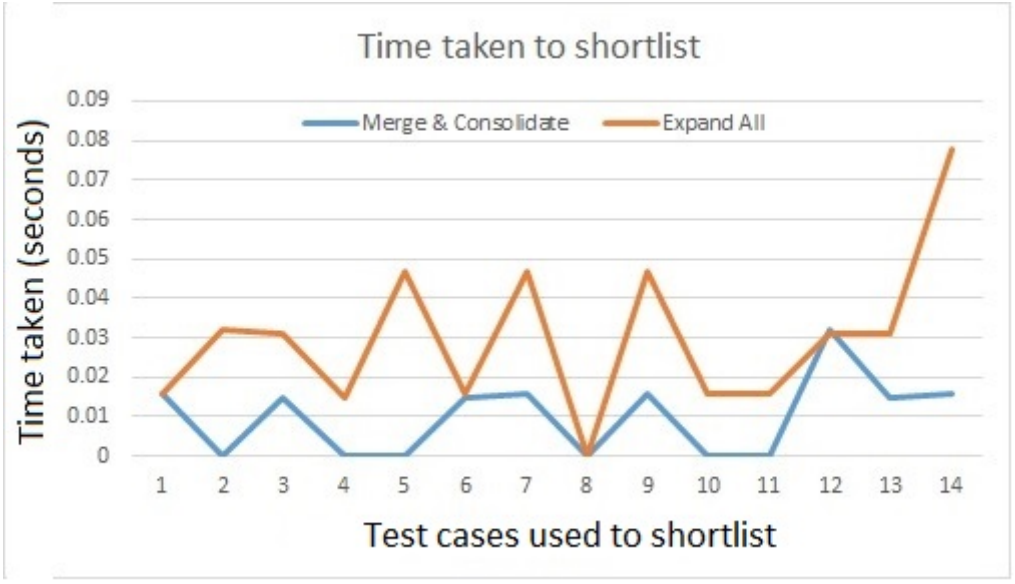


Figure 4.4: Time taken by CPU to search schedules

Chapter 5

Incremental Changes in Large Scale Timetabling

This chapter caters to third objective of my thesis which is "To develop a model to maintain the constructed timetable amidst incremental change requests without affecting the students registered in various cross-section of courses". The term incremental is only used to signify that all changes can not be made in one go but will be executed in steps. The changes that can be made in a constructed timetable is the need of institution who have made a working timetable. After the publishing of timetable, some exigencies do arise due to which change requests to be made in existing timetable start pouring in. As the session has already started and using this timetable as a base many registrations do have taken place across sections. If we consider a single section, the changes that can be made in its timetable seem obvious but taking the individual student timetables of this section into consideration one can be aware of some constraints which should not be violated. These constraints can be of following types:-

1. The course in question is a multi-section course so the change has to be made in all the sections which need this change to be implemented,
2. Some time the electives that are held at the same time are also to be shifted so that students are free to attend the other one also.
3. The slot where class is to be moved is mostly for use of a backlog student. Every backlog student might have to attend a class in some other section when his own section is free. This free section might attract time table changes by his own classmates, which may be cause of concern to all students registered in courses registered keeping in mind this free section. This aspect makes a search of applicability of new timetable to all such students.

In this chapter we present a model that describes incremental changes in large scale time-tabling (ICLST), which decides whether the suggested change can be made without affecting the clash-free timetables of students, who have earlier registered in any of the courses in which change is sought. Next, the implementation details are provided for testing the models developed for decision support system. The algorithms for implementing

various scenarios for this model are described. The user interface for implementation is discussed along with the basic terminologies used. ICLST is based on heuristics approach to find the feasibility of accepting request for changes. The results obtained from implementation of the proposed method have been analysed and also evaluated the efficiency of this method.

These models are judged by the variety of options provided by the decision support system to achieve the goals. These two models provide a working process that helps in decision making process at the time of registration by a student and further changes in time table post registration. As this is at present done manually and no other methods are available so it is not possible to measure the new models against anything. These models add to Quality of Service provided to the users.

The present chapter suggests a method that maximizes the number of requests for change in constructed timetable by considering the already made registrations. Any institute allowing a change in constructed timetable can become role model for such a problem. The work had been demonstrated with real data taken from Thapar Institute of Engineering and Technology, Patiala.

The study involved explanation of the problem, and explanation of the algorithm; followed by testing of the algorithm and the outcome. This chapter is organised as follows: In section 5.1 we discuss the methodology and explanation of the problem of ICLST. The explanation of algorithm is further discussed in section 5.2. In section 5.3, the testing of algorithm with certain inputs and outcomes are finally discussed.

5.1 Methodology

5.1.1 Description of student registration for backlog courses

The method of registration involves finalising course section combinations using which if a student attends classes the resultant timetable will be clash-free. This finalised set of course section combinations forms the core data of student registration.

5.1.2 Basic Changes

Changes in timetabling can be made either by one or combination of the following:

(i) Change in faculty

(ii) Change in room

(iii) change in slot

The changes in timetable can be one of these:

(a) Course section is part of a link

(b) Course is an elective

(c) Course has a spread of sections scheduled together

(d) Normal course not having any of the above three features

5.1.2.1 Change in faculty

We have to search for clash in faculty, whether that faculty is already engaged in another class at that time slot. Arrangements can be made for those faculty who are visible and are free in this slot. In case of (a) and (b) mentioned above, the new faculty should not be attached to any of the courses in link or other elective scheduled in same time. Also the new faculty should not have been already scheduled at same time anywhere else

5.1.2.2 Change in room

We have to ensure that the room is not already booked at this time slot or we can make arrangement in a manner that only those rooms are visible which are free in this time slot. The new room should not be attached already in link as in (a) or another elective in the same set. Also the room should not be used earlier in any of other part. So earlier checking is redundant

5.1.2.3 Change in slot

The slots should be available in each of sections of (a), (b) and (c) and same section in (d). If a slot has to be changed then we have to identify timetable slots of all students studying in this time slot so that this change does not create any clash for any of them. This step is most important of all the steps. The process involves identification of all course-sections of all the students studying in both of these slots. All students means identifying the different course-section combination sets of all the students and checking for each of them.
In case this course section combination is one of the links created at the pre timetabling level, then the full link should move in new slot to make an effective change. However, if this course-section combination is not opted by any of the students, this can be moved. If such a movement is made, then no student will be able to register this course in future.

So, ideally a link should move together along with movement of any one of the course section combinations.

5.1.3 Combination changes

Basic changes can be applied simultaneously in combinations like

1. change in room and faculty together
2. change in room and slot together
3. change in faculty and slot together
4. change in room, faculty and slot together

5.1.4 Assumptions and Notations

Exiting time table is clash-free. This is first priority as if there is problem in original time table, it will be difficult to ascertain whether the clash was created by this change or already there in the timetable.
Add-drop registrations should have been completed. As we rely on the already made registrations to deny any change, some of the changes may be allowed which have no opponents for this change. Changes should be made one at a time.

5.1.4.1 Timetable slots

We create a string of zeroes and ones for around forty five places. We use forty five slots as we have five day time table for 9 slots per day. This string is of the form $s_{45}, s_{44}, s_{43}, ..s_1) = (000000001000000001000000001000000001000000001)$. This string values are one of these two values $s_i = \begin{cases} 0 & free\ time\ slot \\ 1 & used\ time\ slot \end{cases}$

Each slot of string signifies a slot corresponding to one particular day and period combination and shows whether this slot is free or used in timetable. Their mapping is shown in table 4.1. Using this notation we can define any of the following timetables.

5.1.4.2 Main time table

This set will be a set of timetable slots of courses section combination registered by one student.

5.1.4.3 Drop time table

This set will be a set of timetable slot that we have decided to leave or the slot we are going to change.

5.1.4.4 Add time table

This set will be a set of timetable slots that we have decided to add or the one which we are arriving at.

5.1.4.5 New time table

First the Drop time table slot will be removed from the Main time table. If there is no clash between Main time table and Add time table then a union of both of these becomes a new time table meaning that

$$IF\ (Main - timetable\ \cap\ Add - time - table\ = \phi)$$

$$THEN\ New - Time - table = Main - timetable\ \cup\ Add - time - table \tag{5.1}$$

5.2 Algorithm

5.3 Tests and Results

The implementation has been made in MS Windows based environment using Perl as a scripting language. As this problem handles data related to timetable, Sql Server has been used for this purpose. To do the testing of algorithm, a constructed time table was taken along with the registrations made using this timetable. The test cases are formed to make the changes one by one. For each change, system first does some basic tests using which feasibility of this change is checked. If this stage is cleared, the changes are assumed to be made in a new time table. The registrations of each student registered in changed course are considered as a sample. For each sample the new timetable is used for presence of any conflicts. These conflicts are recorded. In the case of any conflicts, the change is rejected. Only in cases where the conflicts are zero, the change is allowed. A sample set of change requests and outcomes are shown in appendix A.8.

Require:

1: *timetable*, {timetable constructed for institute}
2: *registration*,{Data of registration of all the students}
3: Request for Change, {This can be any of the following}
{a}) a faculty member allotted to an event of course-section is to be changed
{b}) a room allotted to an event of course section is to be changed
{c}) a slot of course section combination is to be changed
4: Combination change patterns may call any combination of basic steps shown above as one of these
(i) A change in room and faculty together using a combination of basic set (a) and (b)
(ii) A change in room and slot together using a combination of basic set (b) and (c)
(iii) A change in faculty and slot together using a combination of basic set (a) and (c)
(iv) A change in room, faculty and slot together using a combination of basic set (a), (b) an (c)
5: After the changes have been made, temporary timetables are created for all the students having registrations in the sections which have been affected by change
6: These timetables are checked for clashes within timetables of each student
7: All discrepancies are taken into consideration to allow or disallow the requested changes
8: The allowed changes are incorporated in main time table

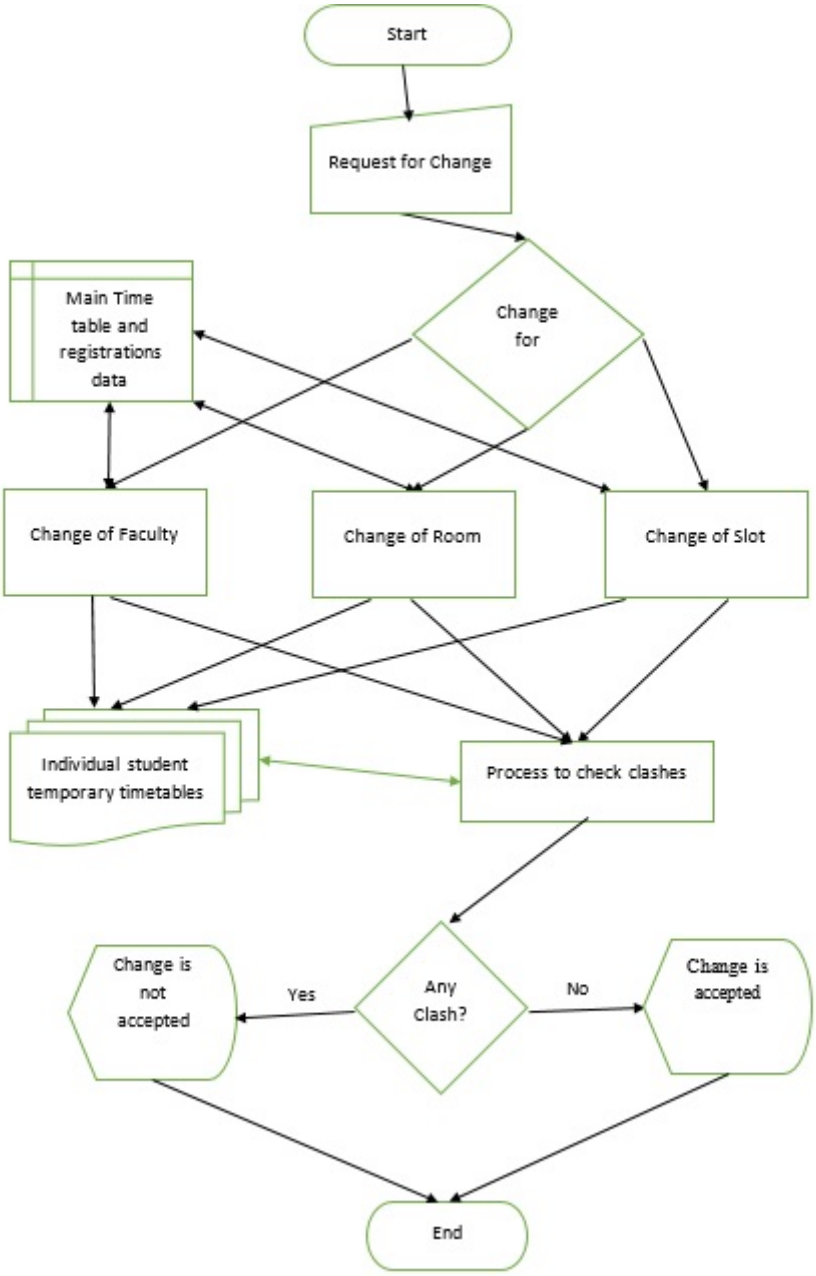


Figure 5.1: Flowchart for ICLST

Chapter 6

Conclusions and Summary

In this chapter, we critically review in brief the work presented in earlier chapters of this thesis. Besides discussing the limitations and scope of the techniques developed in earlier chapters to study interactive decision support system for planned student scheduling and incremental changes in large scale timetabling; we also present possible directions along which the work in this field can further be proceeded. In section 6.1 we critically review the techniques (or methods) discussed in earlier chapters. The limitations and scope of previous chapters has been discussed in section 6.2. In section 6.3 we finally pointed out the possible directions along which the present work can further be processed.

6.1 Review of various techniques used in previous chapters

Major contributions of this research can be summarized as follows:

- This research proposes a new model for planning of student scheduling
- This research proposes a student scheduling process
- This research contributes a model for preservation of student scheduling process at the time of incremental changes of large scale time table.

6.1.1 Planning of Student Scheduling across years

The second chapter basically makes a simple model for understanding the concept considering a programme of four years where all the students in a year study together. One year is made as superclass which has students who have completed the four years of study and have pending subjects in one or more of years of study in junior classes. An example with a sample data has been explained in appendix for this model.

6.1.2 Planning of Student Scheduling across sections

The work discussed in third chapter shows a methodology to suggest same-time scheduling by creation of links for the purpose of backlog adjustment in constructed time table. For this purpose two approaches had been created to solve this problem. First one is All year model which considers students of all years to select the courses to become part of links. The second approach is Prior Year model which considers just the students of prior year in which course is run to select the course for the formation of links. Both approaches were run on four data sets for comparison.

6.1.3 Registration requests based on pre-constructed timetable

In the fourth chapter, two models were created and compared with each other. These models have been used as a part of registration process to identify a combination of courses to be registered or validate a set of course section combinations identified by student himself/herself.

6.1.4 Incremental Changes in Large scale timetabling

The fifth chapter analyzes the approaches used for making incremental changes in constructed timetable. As the change can come in any part of timetable, all the possible approaches are discussed along the type of effect such change can make on system. Some of the changes will need a detailed search of the timetable for effect of change on already made registrations, which can help in deciding whether to allow such a change or not.

6.2 Essence of Thesis

The complete work pertaining to thesis has been explained in figure 6.1. Work done in three objectives has been shown as three processes as

- Creation of Links,
- Student Scheduling and
- Incremental Changes

The data has been segregated as these six components

- Backlog Data
- Links and Cancellations

- Course-section/ course List
- Timetable
- Student Registration
- Change Requests
- Change Effects Status

The Creation of Links process uses as input the backlog data for creation of links and cancellations. The time table is created using links and cancellations as one of the inputs. The student scheduling process uses list of course-section / courses list provided by student to create student registration data. This student registration data and timetable is used by incremental changes process to process the change requests given by students to give change effects status as response.

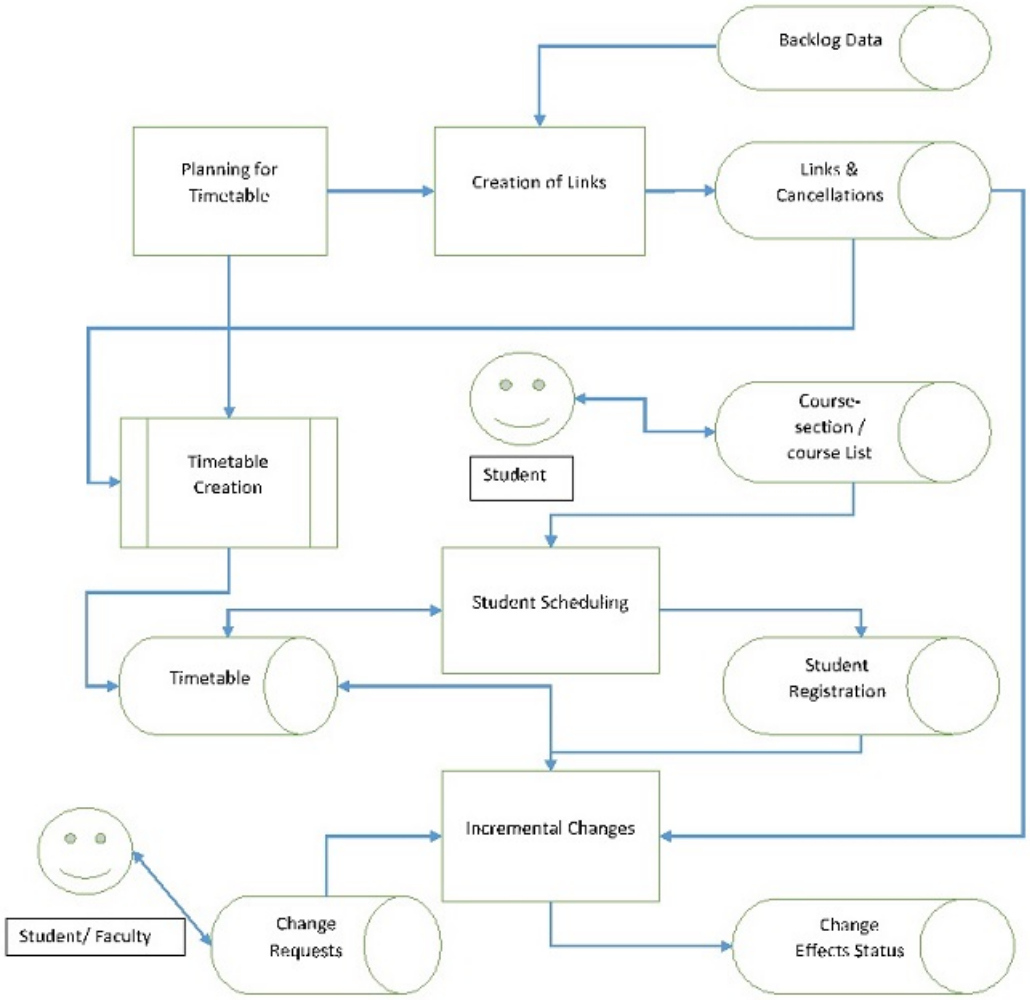


Figure 6.1: Essence of Thesis

6.3 Limitations and Scope

First limitation in the scope of this thesis is that creation of timetable has been dropped out of its scope. This has been done despite the fact that creation of time table is an activity whose existence cannot be negated in executing any of the three objectives of this thesis. The first objective works by creating some inputs for creation of time table. The second and third objective work by using a working time table created already. The creation of time table was kept out of the thesis only due to the reason that creation of time table had been in use in our institute from 1997 till 2006 and the author was part of creating time tables for institute for most of this time. So the approach was to work on areas not touched in the routine working of institute. Leaving the creation of timetable out of the scope makes it possible to work with any of the scenarios whether the timetable is created using an automated process or manually.

6.3.1 Planning of Student Scheduling across years

This chapter is limited to single section class in each year for making a simple model for understanding. Due to this limitation the effect of multiple sections can not be understood.

6.3.2 Planning of Student Scheduling across sections

In this activity only two level of cancellations are created. This can be said in its scope that the third or fourth level has not been added. Though, these can be added any time in the same way as the second level has been created. We can say that the scope of this activity has been limited up to two levels only.

6.3.3 Registration requests based on pre-constructed timetable

While searching for fulfilment of a registration request, the answer only comes in the form that whether there is a solution available for given set of subjects with paired sections or not. The solution will not suggest that it will be workable if one or two of these subjects may be dropped to get a solution. This part is out of scope of current thesis.

6.3.4 Incremental Changes in Large scale timetabling

The approach used to make changes in large scale timetabling is incremental in nature, the changes are made one at a time. The changes handled are simple ones. The complex ones are not yet handled like swap two faculty members in two subjects at once. Though,

this can be handled by splitting the change into two changes made one at a time. When we have made one change their might be conflicts as when one faculty is made faculty of one subject, the load of same faculty is doubled and their might be clashes, which otherwise, would not be there if the change is understood holistically. When two slots are swapped, this will be two steps taken together, followed by the checking. The checking can not take place simultaneously as one step will always create a clash in first hand. Mostly this type of arrangement will not create a problem as the students are already engaged in these time slots and only their courses are shifting. But, in case one of these courses are attended by even one of the students from some other section then that section should be free at the swapped time.

In this case both the slots are checked first for clash of faculty and room in the other slot. If there is no clash of faculty and room to be interchanged, change can be made after running the all student timetable check for a clash after the change for students registered in course section combination of both the slots. In case of swap, no need to check slots as no new slot is coming in picture

6.4 Scope for future work

This method of providing guidelines for making timetable using same time scheduling can be used in two more applications like elective time tabling and handling of pre-requisites which have to be scheduled at same time. This will be helpful as a student can attend any of the electives scheduled at same time. The pre-requisite scheduled at same time will help students who have not cleared their pre-requisite earlier by dropping their other course and studying the pending course. In future, system should be able to suggest a workable set of courses to be taken which is feasible by considering fewer of subjects also. In the future, a mechanism has to be made to handle complex changes simultaneously or checking the validity after few changes to make realistic changes in timetable. Complex changes like swapping of two slots, and those needing multiple changes before checking for clashes will be targeted in future. The insights gained in incremental changes in timetable can be used to make a method to check for feasibility of teaching load for making a timetable.

The purpose of making the three objectives was to fill the gap in automated timetabling activities already handled in institute. As the gaps do get filled in using the approaches proposed in this thesis, we may work upon setting a fully developed automated timetabling system in the institute which is integrated with all components.

References

[1] G.M. Andrew and R. Collins. Matching Faculty to Courses. *College and University*, 46(2):83–89, 1971.

[2] James S. Dyer and John M. Mulvey. An integrated optimization/information system for academic departmental planning. *Management Science*, 22(12):1332–1341, 1976.

[3] John M. Mulvey. A classroom/time assignment model. *European Journal of Operational Research*, 9(1):64–70, 1982.

[4] Jon Breslaw. A linear programming solution to the faculty assignment problem. *Socio-Economic Planning Sciences*, 10(6):227–230, 1976.

[5] Richard H. McClure and Charles E. Wells. A mathematical programming model for faculty course assignments. *Decision Sciences*, 15(3):409–420, 1984.

[6] John J. Dinkel, John Mote, and M. A. Venkataramanan. Or practice—an efficient decision support system for academic course scheduling. *Operations Research*, 37(6):853–864, 1989.

[7] R Feldman and M C Golumbic. Constraint satisfiability algorithms for interactive student scheduling. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pages 1010–1016, 1989.

[8] R Feldman and M C Golumbic. Optimization Algorithms for Student Scheduling via Constraint Satisfiability. *The Computer Journal*, 33:356–364, 1990.

[9] Arabinda Tripathy. Computerised decision aid for timetabling - a case analysis. *Discrete Applied Mathematics*, 35(3):313–323, 1992.

[10] A Monfroglio. Timetabling through Constrained Heuristic Search and Genetic Algorithms. *Software-Practice and Experience*, 26(3):251–279, 1996.

[11] Edmund Burke, Kirk Jackson, Jeffrey H Kingston, and Rupert Weare. Automated university timetabling: The state of the art. *The computer journal*, 40(9):565–571, 1997.

[12] M. A. Saleh Elmohamed, Paul Coddington, and Geoffrey Fox. A comparison of annealing techniques for academic course scheduling. In Edmund Burke and Michael Carter, editors, *Practice and Theory of Automated Timetabling II*, pages 92–112, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[13] Tomas Muller and Roman Barták. Interactive timetabling. *arXiv preprint cs/0109022*, 2001.

[14] Tomás Müller. Interactive heuristic search algorithm. In *CP*, 2002.

[15] Christian Blum, Sebastiao Correia, Marco Dorigo, Ben Paechter, Olivia Rossi-Doria,

and Marko Snoek. A ga evolving instructions for a timetable builder. In *4th International Conference on Practice and Theory of Automated Timetabling, PATAT 2002*, pages 120–123. KaHo Sint-Lieven, 2002.

[16] Christopher Head and Sami Shaban. A heuristic approach to simultaneous course/student timetabling. *Computers & Operations Research*, 34(4):919–933, 2007.

[17] Datta D., Deb K., and Fonseca C.M. Multi-objective evolutionary algorithm for university class timetabling problem. In Dahal K.P., Tan K.C., and Cowling P.I., editors, *Evolutionary Scheduling*, pages 197–236. Springer Berlin Heidelberg, 2007.

[18] Rakesh P Badoni, D K Gupta, and Pallavi Mishra. A new hybrid algorithm for university course timetabling problem using events based on groupings of students. *Computers industrial engineering*, 78:12–25, 2014.

[19] Ana Maria Nogareda and David Camacho. Optimizing satisfaction in a multi-courses allocation problem combined with a timetabling problem. *Soft Comput.*, 21(17):4873–4882, 2017.

[20] Hamed Babaei, Jaber Karimpour, and Amin Hadidi. Generating an optimal timetabling for multi-departments common lecturers using hybrid fuzzy and clustering algorithms. *Soft Computing*, 23, 07 2019.

[21] Gurleen Kaur and Anju Bala. A survey of prediction-based resource scheduling techniques for physics-based scientific applications. *Modern Physics Letters B*, 32(25), 2018.

[22] Xiaoxing Zhang, Zhicheng Ji, and Yan Wang. An improved SFLA for flexible job shop scheduling problem considering energy consumption. *Modern Physics Letters B*, 32, 2018.

[23] Chao Chen, Zhicheng Ji, and Yan Wang. NSGA-II applied to dynamic flexible job shop scheduling problems with machine breakdown. *Modern Physics Letters B*, 32, 2018.

[24] Chun Wang, Zhicheng Ji, and Yan Wang. Multi-objective flexible job shop scheduling problem using variable neighborhood evolutionary algorithm. *Modern Physics Letters B*, 31(19), 2017.

[25] Manish Kumar, Rajesh Bhatia, and Dhavleesh Rattan. A survey of Web crawlers for information retrieval. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(6):e1218, 2017.

[26] Xu Zhou, Yanheng Liu, and Bin Li. A multi-objective discrete cuckoo search algorithm with local search for community detection in complex networks. *Modern Physics Letters B*, 30(07), 2016.

[27] Samir Rebic, Razija Turcinhodzic, Amela Muratovic-Ribic, and Tomaz Kosar. REDOSPLAT: A readable domain-specific language for timetabling requirements defi-

nition. *Computer Languages, Systems & Structures*, 54(07):252–272, 2018.

[28] Andrea Schaerf. A survey of automated timetabling. *Artif. Intell. Rev.*, 13(2):87–127, 1999.

[29] Edmund K Burke and S Petrovic. Recent research directions in automated timetabling. *European Journal of Operational Research*, 140:266–280, 2002.

[30] Michael W Carter. A Comprehensive Course Timetabling and Student Scheduling System at the University of Waterloo. In *Lecture Notes in Computer Science*, volume 2079, pages 64–82, Konstanz, Germany, 2000. Third International Conference on the Practice and Theory of Automated Timetabling, Springer.

[31] Judy Sheard and Dianne Hagan. A Special Learning Environment for Repeat Students. In Bill Manaris, editor, *ITiCSE '99 Proceedings of the 4th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education*, pages 56–59, Univ. of Southwestern Louisiana, 1999.

[32] Azizan Zaimal Abidin. Remedial Tutorials for Differential Equations. *Journal of Applied Sciences*, 11(7):1231–1236, 2011.

[33] Simon Kristiansen, Matias Sorensen, and Thomas R Stidsen. Elective Course Planning. *European Journal of Operational Research*, 215:713–720, 2011.

[34] Gilbert Laporte and Sylvain Desrochers. The problem of assigning students to course sections in a large engineering school. *Computational Operations Research*, 13:387–394, 1986.

[35] William P Pierskalla. The Multi Assignment Problem. *Operations Research*, 16(2):422–431, 1968.

[36] A M Frieze and J Yadegar. An Algorithm for Solving 3-Dimensional Assignment Problems with Application to Scheduling a Teaching Practice. *Journal of Operational Research Society*, 32:989–995, 1981.

[37] David W Pentico. Assignment Problems: A golden anniversary survey. *European Journal of Operational Research*, 176:774–793, 2007.

[38] Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. A survey for the quadratic assignment problem. *European journal of operational research*, 176(2):657–690, 2007.

[39] Lin Zhang, Jian Lu, Xianfei Yue, Jialin Zhou, Yunxuan Li, and Qian Wan. An auxiliary optimization method for complex public transit route network based on link prediction. *Modern Physics Letters B*, 32(05), 2018.

[40] M Almond. An algorithm for constructing University timetables. *The Computer Journal*, 8(4):331–340, 1966.

[41] V A Busam. An algorithm for class scheduling with section preference. *Communications of the ACM*, 10(9):567–569, 1967.

[42] W K Winters. A scheduling algorithm for Computer Assisted Registration System. *Communications of the ACM*, 14(3):166–171, 1971.

[43] I Miyaji, K Ohno, and H Mine. Solution method for partitioning students into groups. *European Journal of Operational Research*, 33:82–90, 1981.

[44] A. H. Özer and C. Özturan. A direct barter model for course add/drop process. *Discrete Applied Mathematics*, 159:812–825, 2011.

[45] H. Heitmann and W. Bruggermann. Preference based assignment of university students to multiple groups. *OR Spectrum*, 36:607–629, 2014.

[46] M. Dostert, A. Politz, and H. Schmitz. A complexity analysis and and algorithmic approach to student sectioning in exiting timetables. *Journal of Scheduling*, 19(9):285–293, 2016.

[47] J Munkres. Algorithms for the assignment and transportation Problems. *J. Siam*, 5:32–38, 1957.

[48] FranÄ§ois Bourgeois and Jean-Claude Lassalle. An extension of the Munkres algorithm for the assignment problem to rectangular matrices. *Communication of the ACM*, 14(12):802–804, 1971.

[49] Sanjeev K Guleria and Arvind K Lal. Interactive Decision Support System for planned student scheduling using same time scheduling in time-table. *Modern Physics Letters B*, 33(13):1950161, 2019.

Appendix

A.5 A step by step proposed method using Dynamic Programming Approach on a sample data

To understand the proposed model, a sample data has been generated in table 6.1 using which the model will be explained. Here a scenario is created for a four year program where subjects are taught in all the four years. Each subject S_i is having a unique LTP which is a combination of three numbers depicting the hours taken for Lecture, tutorial and practical in a week. The first year students of class L_1 are assumed to be not having any backlogs. But a superclass L_5 has been created which may consist of students who have completed their regular study but they are having some backlog courses to be cleared.

A.5.1 Stage – I (Class- I)

1. Regular Courses: B_1, B_2, B_3, B_4, B_5

2. Courses available for backlogs: Nil

3. Total time required for scheduling regular classes $T_1 = \sum_{i=1}^{5} t_{i1} = 6+4+7+7+6 =$ 30 Hrs.

4. Backlogs adjusted= Nil

A.5.2 Stage-II (Class- II)

1. Regular Courses: $B_6, B_7, B_8, B_9, B_{10}$

2. Courses available for backlogs: B_1, B_2, B_3, B_4, B_5

3. Total time required for scheduling regular classes

Table 6.1: Sample set of data of backlog distribution

	LTP	$t_{ij} = L + T + P$	i	$B_i = Subject$	S_{i1}	S_{i2}	S_{i3}	S_{i4}	S_{i5}
First	3 1 2	6	1	$B_1 = A$	{0}	{1,3,5,9}	{11,12}	{21,23,24}	{31}
year	3 1 0	4	2	$B_2 = B$	{0}	{2,6,8,9}	{13,16,17}	{22,28}	{32,33,34,35}
courses	3 2 2	7	3	$B_3 = C$	{0}	{1,5,7}	{15,18,20}	{27}	{35,37,39}
Class-I	3 2 2	7	4	$B_4 = D$	{0}	{1,2,3}	{14,19}	{25,30}	{38}
L_1	3 1 2	6	5	$B_5 = E$	{0}	{1,3,4,6,10}	{0}	{26}	{39,40}
	T_1	30							
Second	2 2 4	8	6	$B_6 = F$		{0}	{11}	{22,27}	{32}
year	3 1 2	6	7	$B_7 = G$		{0}	{12,13}	{21,29}	{34,35}
courses	3 1 2	6	8	$B_8 = H$		{0}	{15}	{21}	{33}
Class-II	1 0 3	4	9	$B_9 = I$		{0}	{14}	{23,30}	{39}
L_2	3 1 2	6	10	$B_{10} = J$		{0}	{16,17}	{24,25,26}	{40}
	T_2	30							
Third	3 1 2	6	11	$B_{11} = K$			{0}	{21}	{32}
year	3 1 0	4	12	$B_{12} = L$			{0}	{0}	{31}
courses	3 1 0	4	13	$B_{13} = M$			{0}	{23,24}	{0}
Class-III	3 1 0	4	14	$B_{14} = N$			{0}	{25}	{39}
L_3	3 1 2	6	15	$B_{15} = O$			{0}	{39}	{36}
	3 1 0	4	16	$B_{16} = P$			{0}	{22,25}	{35}
	T_3	28							
Fourth	3 1 2	6	17	$B_{17} = Q$				{0}	{31,32}
year	3 1 2	6	18	$B_{18} = R$				{0}	{34}
courses	3 1 2	6	19	$B_{19} = S$				{0}	{35}
Class-IV	3 1 0	4	20	$B_{20} = T$				{0}	{37}
L_4	3 1 0	4	21	$B_{21} = V$				{0}	{37,38}
	3 1 0	4	22	$B_{22} = W$				{0}	{0}
	T_4	30							

Table 6.2: Sample set analysis

L_j	L_1	L_2	L_3	L_4	L_5
Time for regular courses T_j	30	30	28	30	0
Periods in a week H_j	40	40	40	40	40
Periods available for backlog A_j	10	10	12	10	40
Total students $N = \cup\|S_{ij}\|$	10	10	10	10	10

Table 6.3: Course combinations with time $\leq A_2$ i.e. $\leq$ 10 Hrs.

	T	Sij	Cij
B_1	6	S_{11} ={1, 3, 5, 9}	4
B_2	4	S_{21} ={2, 6, 8, 9}	4
B_3	7	S_{31} ={1, 5, 7}	3
B_4	7	S_{41} ={1, 2, 3}	3
B_5	6	S_{51} ={1, 3, 4, 6, 10}	5
B_1B_2	10	{1,2,3,5,6,8,9}	7
B_2B_5	10	{1,2,3,4,6,8,9,10}	8

4. $T_2 = \sum_{i=1}^{5} t_{i2} = 8 + 6 + 6 + 4 + 6 = 30$ Hrs.

5. Time available for scheduling of backlogs $A_2 = H_2 - T_2 = 40 - 30 = 10$ Hrs.

6. Generating alternatives

All other combinations will require more than 10 hours. Therefore, they are rejected. The courses B_2 and B_5 are selected for backlogs.
Introduce a dummy courses say D_2^1 and D_2^2 in link with B_2 and B_5 respectively. Time consumed by dummy courses D_2^1 and D_2^2 are 4 and 6 hrs. respectively

- Students adjusted for backlogs $B_2 \cup B_5 = S_1^2 \cup S_1^5 = \{1, 2, 3, 4, 6, 8, 9, 10\}$
- Number of students adjusted = $n[S_1^2 \cup S_1^5] = 8$
- Time utilized for backlogs = 10 hrs.
- Time remaining for scheduling of regular courses = 40-10=30 hrs.
- Links formed: $B_2D_2^1$ and $B_5D_2^2$

A.5.3 Stage-III (Class-III)

1. Regular Courses: $B_{11}, B_{12}, B_{13}, B_{14}, B_{15}, B_{16}$.
2. Courses available for backlogs: $B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8, B_9, B_{10}$
3. Total time required for scheduling regular classes $T_3 = \sum_{i=1}^{6} t_{i3} = 6 + 4 + 4 + 4 + 6 + 4 = 28$ Hrs.
4. Time available for scheduling of backlogs $A_3 = H_3 - T_3 = 40 - 28 = 12$ Hrs.

Table 6.4: The assignment problem class I & II

	Class II courses				
	B_6	B_7	B_8	B_9	B_{10}
	$\{11\}$	$\{12,13\}$	$\{15\}$	$\{14\}$	$\{16,17\}$
$B_1\{11,12\}$	2	3	3	3	4
$B_3\{15,18,20\}$	4	5	3	4	5
$B_4\{14,19\}$	3	4	3	2	4

Table 6.5: Students adjusted for backlogs after assignment of class I and II courses

	Time required	S_{ij}		$\lvert S_{ij}\rvert$
B_1B_{10}	$max\{6,6\}=6$	$B_1 \cup B_{10} = \{11,12,16,17\}$	:	4
B_3B_7	$max\{7,6\}=7$	$B_3 \cup B_7 = \{12,13,15,18,20\}$	:	5
B_4B_8	$max\{7,6\}=7$	$B_4 \cup B_8 = \{14,15,19\}$	:	3

5. Generating alternatives

List all course combinations with time $\leq A_3$ i.e. $\leq$ 12 Hrs. Courses B_2&B_3 are not considered for assignment allocation as there exists dummy courses corresponding to these in class j-1 i.e. class 2. Course B_5 is also not considered as none of the students of class III have backlog in course B_5.

The cell values are C_{ij} values $= |\cup S_{ij}|$

The above table shows the final allocation, and the links formed. The time required by the linked courses in class III, and students adjusted for backlogs are as follows:-

Add all courses not considered in assignment allocation due to imbalanced problem. Add all courses having dummies in class j-1

Course B_5 is not included as no student has backlog in B_5. Select the combination max (Cardinal number. In case of a tie, select the combination that takes lesser number of hours. The links to be offered for backlogs in class III are B_3B_7 and B_2 Add

Table 6.6: Courses considered for combinations after making link of class I and II

Link/Courses	Time	S_{ij}
B_1B_{10}	6	$\{11,12,16,17\}$
B_3B_7	7	$\{12,13,15,18,20\}$
B_4B_8	7	$\{14,15,19\}$
B_2	4	$\{13,16,17\}$
B_6	8	$\{11\}$
B_9	4	$\{14\}$

Table 6.7: Combinations of links made upto class II with other courses having $t_{ij} \leq 12$ hours

			Students Adjusted S_{ij}	$CN = \|S_{ij}\|$
$B_1B_{10}B_2$	AJB	10 hrs	$\{11, 12, 13, 16, 17\}$	5
$B_3B_7B_2$	CGB	11 hrs	$\{12, 13, 15, 16, 17, 18, 20\}$	7
$B_4B_8B_2$	DHB	11 hrs	$\{13, 14, 15, 16, 17, 19\}$	6
B_6B_2	FB	12 hrs	$\{11, 13, 16, 17\}$	4
B_9B_2	IB	8 hrs	$\{13, 14, 16, 17\}$	4
$B_1B_{10}B_9$	AJI	10 hrs	{11,12,14,16,17}	5
$B_3B_7B_9$	CGI	11 hrs	$\{12, 13, 14, 15, 18, 20\}$	6
$B_4B_8B_9$	DHI	11 hrs	$\{14, 15, 19\}$	3
B_6B_9	FI	12 hrs	$\{11, 14\}$	2

dummy courses D_3^1 and D_3^2 in class III corresponding to link B_3B_7 and B_2

- Students adjusted for backlogs = $B_3 \cup B_7 \cup B_2 = \{12, 13, 15, 16, 17, 18, 20\}$
- Number of students adjusted = 7
- Links formed : $B_3B_7D_3^1$ and $B_2D_2^1D_3^2$
- Time utilized for backlogs = 11hrs.
- Time remaining for scheduling of regular courses = 40-11 = 29 hrs

A.5.4 Stage – IV (Class-IV)

1. Regular Courses: $B_{17}, B_{18}, B_{19}, B_{20}, B_{21}, B_{22}$.
2. Classes available for backlogs: $B_1, B_2^*, B_3B_7^*, B_4, B_5^*, B_6, B_8, B_9, B_{10}, B_{11}, B_{12}, B_{13}, B_{14}, B_{15}, B_{16}$ $^*B_2, B_3B_7, B_5$ are links, dummy courses are not shown as they do not affect the computation
3. Total time required for scheduling regular classes $T_4 = \sum_{i=1}^{m} t_{i4} = 6 + 6 + 6 + 4 + 4 + 4 = 30$ Hrs.
4. Time available for scheduling of backlogs $A_4 = H_4 - T - 4 = 40 - 30 = 10$ Hrs.
5. Generating alternatives:
   - Link B_2 is not considered for allocation as it has a dummy D_3^2 in class j-1 i.e. class III
   - Link B_3B_7 is not considered for allocation as it has a dummy D_3^1 in class III

Table 6.8: The assignment problem between links upto class II and class III courses

	Class III courses				
	B_{11} $\{21\}$	B_{13} $\{23,24\}$	B_{14} $\{25\}$	B_{15} $\{30\}$	B_{16} $\{22,25\}$
$B_1\{21,23,24\}$	3	3	4	4	5
$B_4\{25,30\}$	3	4	2	2	3
$B_5\{26\}$	2	3	2	2	3
$B_6\{22,27\}$	3	4	3	3	3
$B_8\{21\}$	1	3	2	2	3
$B_9\{23,30\}$	3	3	3	3	4
$B_{10}\{24,25,26\}$	4	4	3	3	4

Table 6.9: students adjusted for backlogs in links created upto class III

	Time required	S_{ij}		$\lvert S_{ij}\rvert$
B_1B_{16}	$max\{6,4\}=6$	$B_1 \cup B_{16} = \{21,22,23,24,25\}$	:	5
B_4B_{13}	$max\{7,4\}=7$	$B_4 \cup B_{13} = \{23,24,25,30\}$	:	4
B_6B_{15}	$max\{8,6\}=8$	$B_6 \cup B_{15} = \{22,27,30\}$	:	3
B_9B_{14}	$max\{4,4\}=4$	$B_9 \cup B_{14} = \{23,25,30\}$	:	3
$B_{10}B_{11}$	$max\{6,6\}=6$	$B_{10} \cup B_{11} = \{21,24,25,26\}$	:	4

- Course B_{12} is not considered as student of class IV do not have any backlog in B_{12}

The table 6.12 shows the final allocation and the links formed. The time required by the linked courses in class IV and the students adjusted for backlogs are as shown in table 6.13

Add all courses not considered in assignment allocation due to unbalanced problem and all courses having dummies in class III

Table 6.10: Courses to be considered for combinations using links upto class III

Link/Courses	Time	S_{ij}
B_1B_{16}AP	6	$\{21,22,23,24,25\}$
B_4B_{13}DM	7	$\{23,24,25,30\}$
B_6B_{15}FO	8	$\{22,27,30\}$
B_9B_{14}IN	4	$\{23,25,30\}$
$B_{10}B_{11}$JK	6	$\{21,24,25,26\}$
B_2B	6	$\{22,28\}$
B_3B_7CG	7	$\{21,27,29\}$
B_5	6	$\{28\}$
B_8H	6	$\{21\}$

Table 6.11: Combinations of ccourses and links crated upto class III having $t_{ij} \leq 10$ hours

			Students Adjusted S_{ij}	$CN = \|S_{ij}\|$
B_1B_{16}, B_9B_{14}	AP,IN	10 hrs	$\{21, 22, 23, 24, 25, 30\}$	6
$B_{10}B_{11}, B_9B_{14}$	JK,IN	10 hrs	$\{21, 23, 24, 25, 26, 30\}$	6
B_2, B_9B_{14}	B, IN	10 hrs	$\{22, 23, 25, 28, 30\}$	5
B_5, B_9B_{14}	E,IN	10 hrs	$\{23, 25, 26, 30\}$	4
B_8, B_9B_{14}	H, IN	10 hrs	$\{24, 27, 29\}$	3

Course B_{12} is not included as no student has backlog in B_{12}

All other combinations require more than 10 hrs

Select the combination with max cardinal number. In case of a tie select the combination requiring less time. In case of a tie of cardinal number and time required, select any of the combinations.

Thus, we may select the combination B_1B_{16}, B_9B_{14} AP, IN. then the links to be offered are B_1B_{16} and B_9B_{14} [AP & IN]

Add dummy courses D_4^1, D_4^2 in class IV corresponding to the links B_1B_{16} and B_9B_{14} respectively.

- Students adjusted for backlogs = $B_1 \cup B_{16} \cup B_9 \cup B_{14} = \{21, 22, 23, 24, 25, 30\}$
- Number of students adjusted : 6
- Links formed : $B_1B_{16}D_4^1 and B_9B_{14}D_4^2$
- Time utilized for backlogs : 10 hrs
- Time remaining for scheduling of regular courses = 40 – 10=30 hrs.

A.5.5 Stage – V (Supercourse)

1. Regular Courses: Nil
2. Classes available for backlogs:
   (All courses offered in classes I, II, III, IV (either in links or otherwise)
3. $B_1B_{16}, B_2, B_3B_7, B_5, B_9B_{14}, B_4, B_6, B_8, B_{10}, B_{11}, B_{12}, B_{13}, B_{15}, B_{17}, B_{18}, B_{19}, B_{20}, B_{21}, B_{22}$
4. AP,B, CG, E,IN, D, F, H, J, K, L, M, O, Q, R, S, T, V,W
5. Total time required for scheduling regular classes $T_5 = \sum_{i=1}^{m} t_{i5} = 0$
6. Time available for scheduling of backlogs $A_5 = H_5 - T_5 = 40 - 0 = 40$ Hrs.

7. Generating alternatives

   - Links $B_1B_{16}\&B_9B_{14}$ are not considered as they have dummies $D_4^1\&D_4^2$ corresponding to them in class j-1 i.e. class IV
   - Courses M and W are not included for assignment as students of superclass j=5 do not have any backlog in M and W

The assignment problem is set as follows:-

Table 6.12: The assignment problem between links upto class III and class IV courses

	Class IV courses				
	$B_{17}-Q$ $\{31\}$	$B_{18}-R$ $\{34\}$	$B_{19}-S$ $\{35\}$	$B_{20}-T$ $\{37\}$	$B_{21}-V$ $\{37,38\}$
$B_2-B\{32,33,34,35\}$	5	4	4	5	6
$B_3B_7-CG\{34,35,37,39\}$	6	4	4	4	5
$B_5-E\{39,40\}$	4	3	3	3	4
$B_4-D\{38\}$	3	2	2	2	2
$B_6-F\{32\}$	2	2	2	2	3
$B_8-H\{33\}$	3	2	2	2	3
$B_10-J\{40\}$	3	2	2	2	3
$B_{11}\{32\}$	2	2	2	2	3
$B_{12}\{31\}$	2	2	2	2	3
$B_{15}\{36\}$	3	2	2	2	3

The above table shows one of the multiple optimal allocations and the links formed. The time required by the linked courses in class V and students adjusted for backlogs are as follows:-

- Add all courses not considered in assignment algorithm due to imbalanced problem
- Add all courses having dummy courses in class 4
- Do not include courses B_{13} and B_{22}, as there are no backlogs of students of class V in courses B_{13} and B_{22}.

Table 6.13: students adjusted for backlogs after creating links upto class IV

	Time required	S_{ij}		$\lvert S_{ij}\rvert$
B_2B_{21}	$max\{6,4\}=6$	$B_2\cup B_{21}=\{32,33,34,35,37,38\}$	:	6
$B_3B_7B_{17}$	$max\{7,6,6\}=7$	$B_3\cup B_7\cup B_{17}=\{31,32,34,35,37,38\}$	:	6
B_5B_{19}	$max\{6,6\}=6$	$B_5\cup B_{19}=\{35,39,40\}$	:	3
B_4B_{18}	$max\{7,6\}=7$	$B_4\cup B_{18}=\{34,38\}$	:	2
B_6B_{20}	$max\{8,4\}=8$	$B_6\cup B_{20}=\{32,37\}$	:	2

Table 6.14: Courses considered for combination upto links of class IV

	T	Students Adjusted S_{ij}	$CN = \|S_{ij}\|$
B_2B_{21}	6 hrs	$\{32, 33, 34, 35, 37, 38\}$	6
$B_3B_7B_{17}$	7 hrs	$\{31, 32, 34, 35, 37, 38\}$	6
B_5B_{19}	6 hrs	$\{35, 39, 40\}$	3
B_4B_{18}	7 hrs	$\{34, 38\}$	2
B_6B_{20}	8 hrs	$\{32, 37\}$	2
B_8	6 hrs	$\{33\}$	1
B_{10}	6 hrs	$\{40\}$	1
B_{11}	6 hrs	$\{32\}$	1
B_{12}	4 hrs	$\{31\}$	1
B_{15}	6 hrs	$\{36\}$	1
B_1B_{16}	6 hrs	$\{31, 35\}$	2
B_9B_{14}	4 hrs	$\{39, 40\}$	2

The objective at this stage i.e. supercourse is not to offer one backlog to each student, but to offer as many as possible within the given time limit. Students having only one backlog or backlogs of any one class are adjusted in a timetable by default, as they don't have any regular courses to attend.

The best combination of courses in this case is to offer courses in order of cardinal numbers till $\sum t \leq A_5$

In case of a tie course with a lesser t_{ij} may be offered.

The combination of courses will be $B_2B_{21}, B_3B_7B_{17}, B_5B_{19}, B_9B_{14}, B_1B_{16}, B_4B_{18}$. BV, CGQ, ES, IN, AP, DR.

This combination will require 6+7+6+4+6+7=36 hrs. The new links formed are $B_2B_{21}, B_3B_7B_{17}, B_5B_{19}$

Introduce dummy courses $D_5^1, D_5^2, D_5^3, D_5^4, D_5^5$ and D_5^6 corresponding to each of these links in class V.

- Students adjusted for backlogs = $B_2B_{21} \cup B_3B_7B_{17} \cup B_5B_{19} \cup B_9B_{14} \cup B_1B_{16} \cup B_4B_{18}$. = $\{31, 32, 33, 34, 35, 37, 38, 39, 40\}$
- Student with roll number 36 is adjusted by default.
- Number of students adjusted = 9+1 = 10
- Links formed $B_2B_{21}D_5^1, B_3B_7B_{17}D_5^2, B_5B_{19}D_5^3, B_9B_{14}D_5^4, B_1B_{16}D_5^5, B_4B_{18}D_5^6$
- Time utilized for backlogs = 36 hrs.
- Time remaining for regular courses : Not applicable
- Optimal Policy: Total number of students adjusted = $\sum$ students adjusted in stages

Table 6.15: Chart Cum Table for Links formed, Students Adjusted for Backlogs, Number of Students Adjusted, Time available and utilized for backlogs

		I	II	III	IV	V
Links formed	1	B_2	D_2^1	D_3^2	B_{21}	D_5^1
	2	B_5	D_2^2		B_{19}	D_5^3
	3	B_3	B_7	D_3^1	B_{17}	D_5^2
	4	B_1		B_{16}	D_4^1	D_5^5
	5		B_9	B_{14}	D_4^2	D_5^4
	S_{ij}	{1,2,3, 4,6,8 ,9,10}	{12,13,15, 16,17, 18,20}	{21,22,23, 24,25,30}	{31,32,33, 34,35,36, 37,38, 39,40}	
	N	0	8	7	6	10
	AJ		10	12	10	40
	Time Utilized for backlogs		10	11	10	36

I to Stage V $= 0 + 8 + 7 + 6 + 10 = 31$.

A.6 Cursory look at the working of creation of links

The input form used for creation of links is shown in figure 6.2 where user can select from the options shown in drop down and choose one of the models for creation of links. User presses Make Links for AYM and "Make Links Year Wise" for PYM. As the output for creation of links is long, it is shown in three parts. The first part shows top view of output where the links are shown while being created. Here, all the subjects are visible for each section, the number of hours taken for initial teaching load of each section, finalised subjects and cancellations in each sections, the total hours of each link after the cancellations along with time taken to process this section. The middle part is shown in figure 6.4 where the total time taken to create this set of links is shown along the percentage of students adjusted followed by pruned set of links. Here, the subjects which are not to be taken as backlog by students of higher classes are dropped, so that only subjects of concern are left in the output. The bottom part of report also shown the list of students adjusted in this creation of links in the drop down.

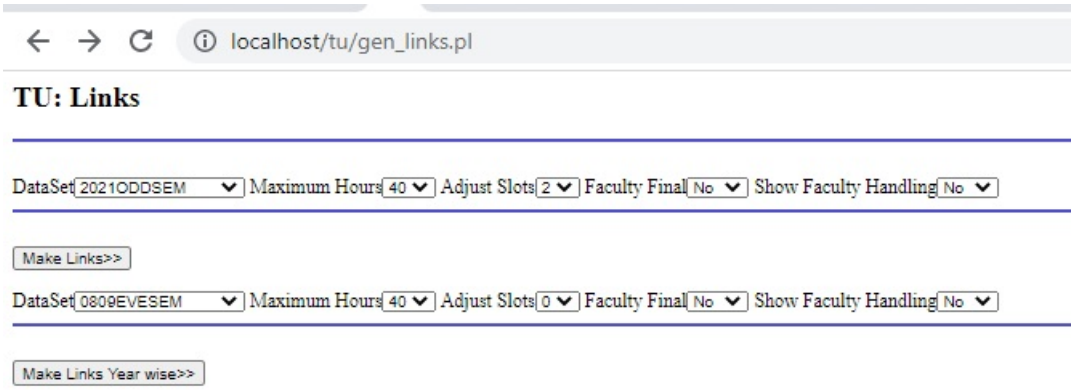
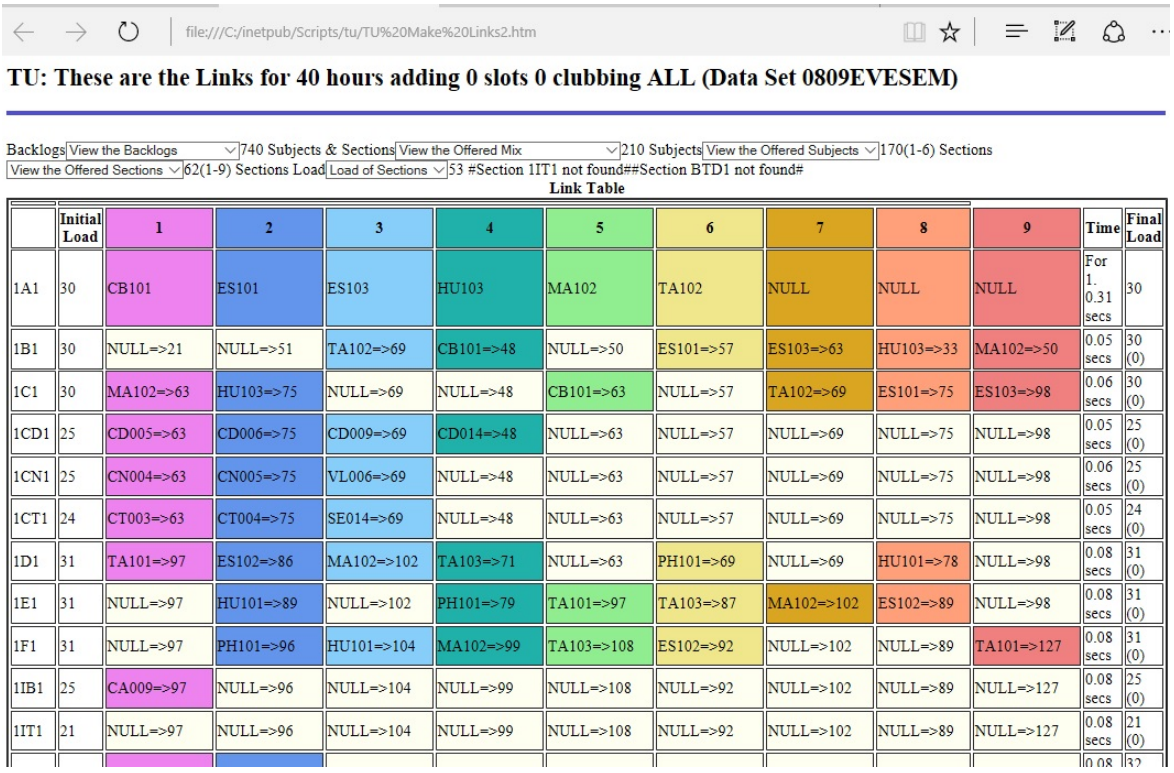


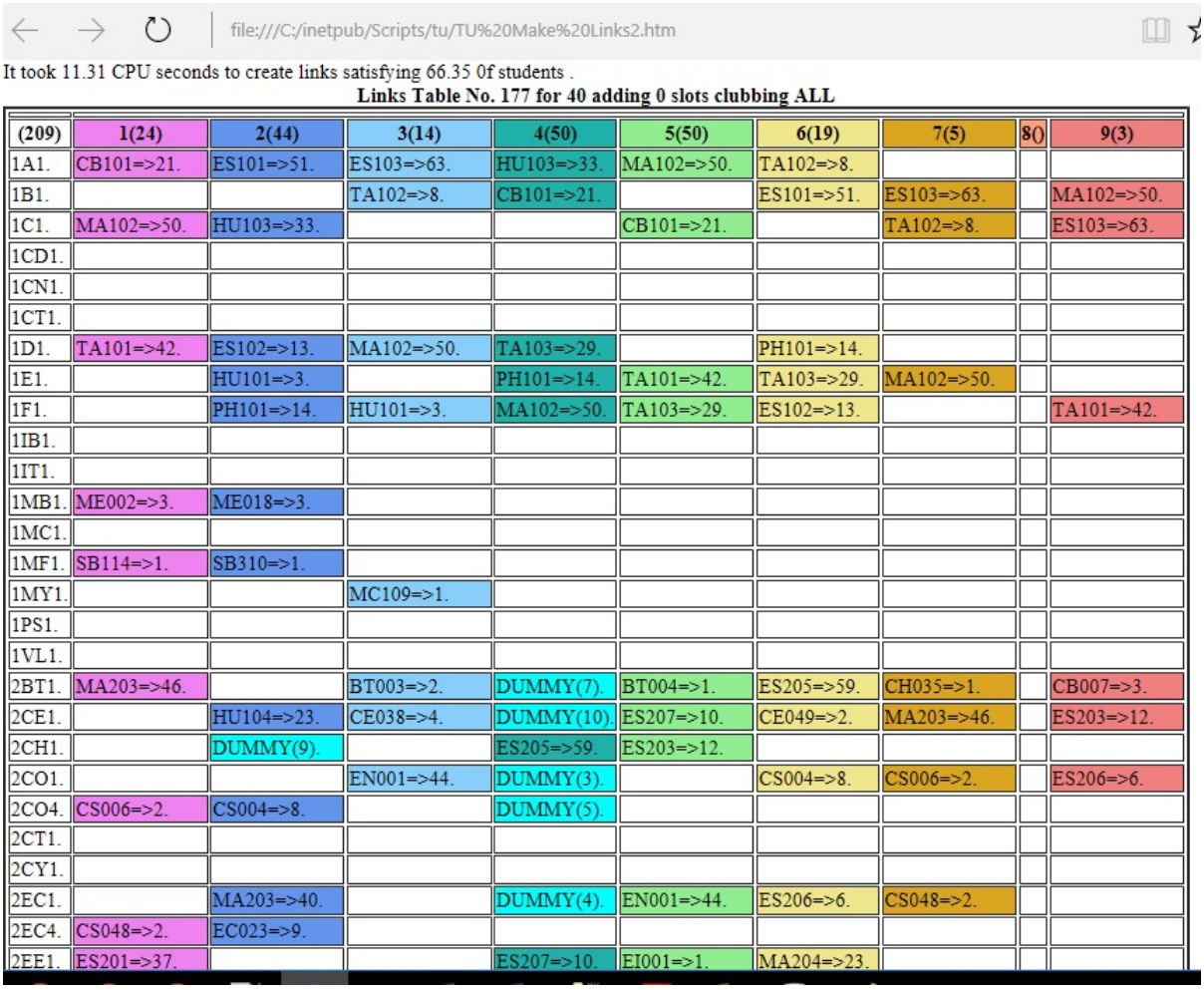
Figure 6.2: Input form to Create Links

A model of working of creation of links is shown in figure 6.6. This model shows that backlog data is used by "creation of links" module to create links. These links are used by "Creation of Timetable" process for creating a timetable in which the created links have a one to one linking with corresponding set of course-sections being scheduled at same time.



	Initial Load	1	2	3	4	5	6	7	8	9	Time	Final Load
1A1	30	CB101	ES101	ES103	HU103	MA102	TA102	NULL	NULL	NULL	For 1. 0.31 secs	30
1B1	30	NULL=>21	NULL=>51	TA102=>69	CB101=>48	NULL=>50	ES101=>57	ES103=>63	HU103=>33	MA102=>50	0.05 secs	30 (0)
1C1	30	MA102=>63	HU103=>75	NULL=>69	NULL=>48	CB101=>63	NULL=>57	TA102=>69	ES101=>75	ES103=>98	0.06 secs	30 (0)
1CD1	25	CD005=>63	CD006=>75	CD009=>69	CD014=>48	NULL=>63	NULL=>57	NULL=>69	NULL=>75	NULL=>98	0.05 secs	25 (0)
1CN1	25	CN004=>63	CN005=>75	VL006=>69	NULL=>48	NULL=>63	NULL=>57	NULL=>69	NULL=>75	NULL=>98	0.06 secs	25 (0)
1CT1	24	CT003=>63	CT004=>75	SE014=>69	NULL=>48	NULL=>63	NULL=>57	NULL=>69	NULL=>75	NULL=>98	0.05 secs	24 (0)
1D1	31	TA101=>97	ES102=>86	MA102=>102	TA103=>71	NULL=>63	PH101=>69	NULL=>69	HU101=>78	NULL=>98	0.08 secs	31 (0)
1E1	31	NULL=>97	HU101=>89	NULL=>102	PH101=>79	TA101=>97	TA103=>87	MA102=>102	ES102=>89	NULL=>98	0.08 secs	31 (0)
1F1	31	NULL=>97	PH101=>96	HU101=>104	MA102=>99	TA103=>108	ES102=>92	NULL=>102	NULL=>89	TA101=>127	0.08 secs	31 (0)
1IB1	25	CA009=>97	NULL=>96	NULL=>104	NULL=>99	NULL=>108	NULL=>92	NULL=>102	NULL=>89	NULL=>127	0.08 secs	25 (0)
1IT1	21	NULL=>97	NULL=>96	NULL=>104	NULL=>99	NULL=>108	NULL=>92	NULL=>102	NULL=>89	NULL=>127	0.08 secs	21 (0)

Figure 6.3: Created Links top view showing details of links getting created



(209)	1(24)	2(44)	3(14)	4(50)	5(50)	6(19)	7(5)	8()	9(3)
1A1.	CB101=>21.	ES101=>51.	ES103=>63.	HU103=>33.	MA102=>50.	TA102=>8.			
1B1.			TA102=>8.	CB101=>21.		ES101=>51.	ES103=>63.		MA102=>50.
1C1.	MA102=>50.	HU103=>33.			CB101=>21.		TA102=>8.		ES103=>63.
1CD1.									
1CN1.									
1CT1.									
1D1.	TA101=>42.	ES102=>13.	MA102=>50.	TA103=>29.		PH101=>14.			
1E1.		HU101=>3.		PH101=>14.	TA101=>42.	TA103=>29.	MA102=>50.		
1F1.		PH101=>14.	HU101=>3.	MA102=>50.	TA103=>29.	ES102=>13.			TA101=>42.
1IB1.									
1IT1.									
1MB1.	ME002=>3.	ME018=>3.							
1MC1.									
1MF1.	SB114=>1.	SB310=>1.							
1MY1.			MC109=>1.						
1PS1.									
1VL1.									
2BT1.	MA203=>46.		BT003=>2.	DUMMY(7).	BT004=>1.	ES205=>59.	CH035=>1.		CB007=>3.
2CE1.		HU104=>23.	CE038=>4.	DUMMY(10).	ES207=>10.	CE049=>2.	MA203=>46.		ES203=>12.
2CH1.		DUMMY(9).		ES205=>59.	ES203=>12.				
2CO1.			EN001=>44.	DUMMY(3).		CS004=>8.	CS006=>2.		ES206=>6.
2CO4.	CS006=>2.	CS004=>8.		DUMMY(5).					
2CT1.									
2CY1.									
2EC1.		MA203=>40.		DUMMY(4).	EN001=>44.	ES206=>6.	CS048=>2.		
2EC4.	CS048=>2.	EC023=>9.							
2EE1.	ES201=>37.			ES207=>10.	EI001=>1.	MA204=>23.			

Figure 6.4: Created Links middle view showing pruned links

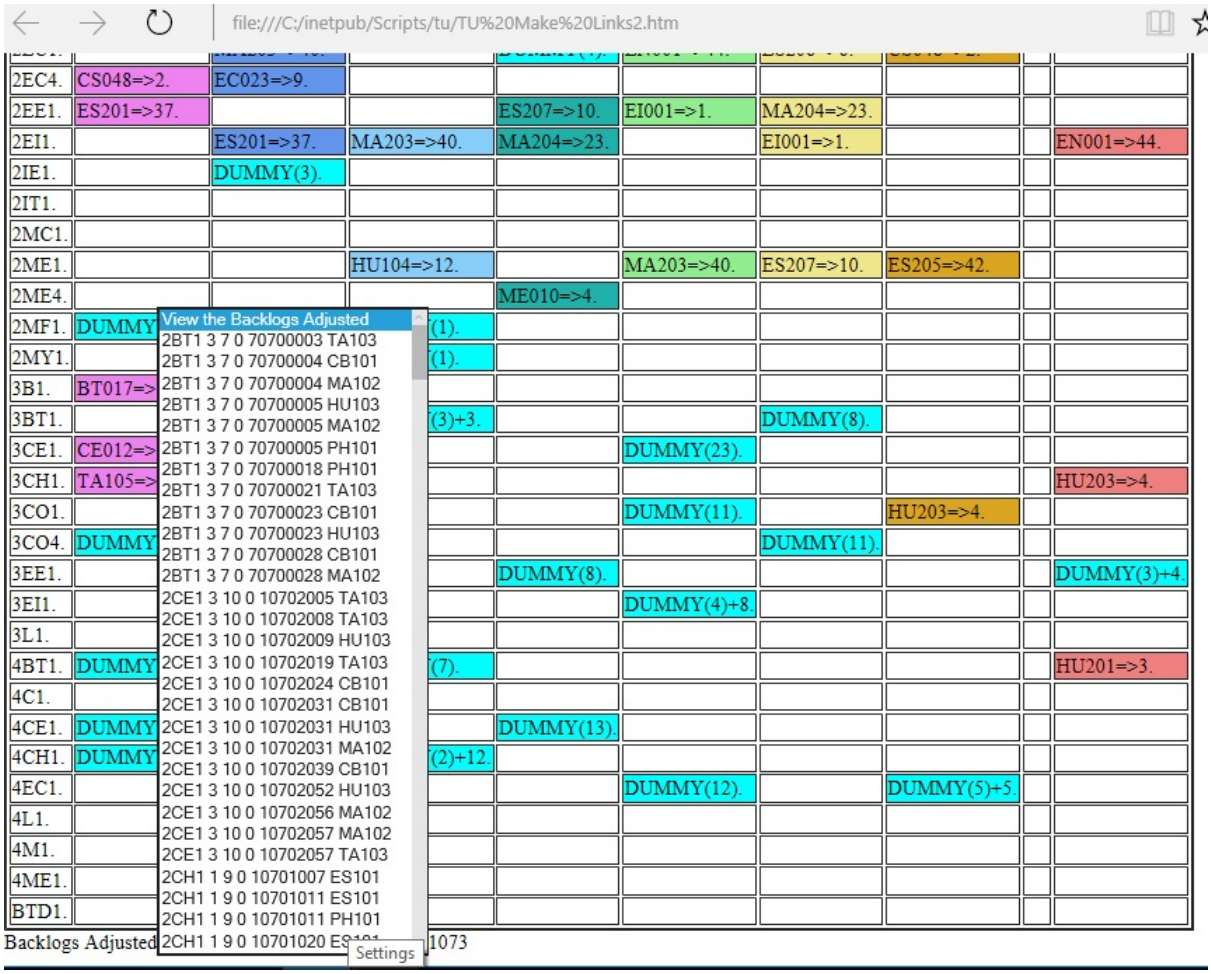


Figure 6.5: Created Links bottom view with benefiting students list

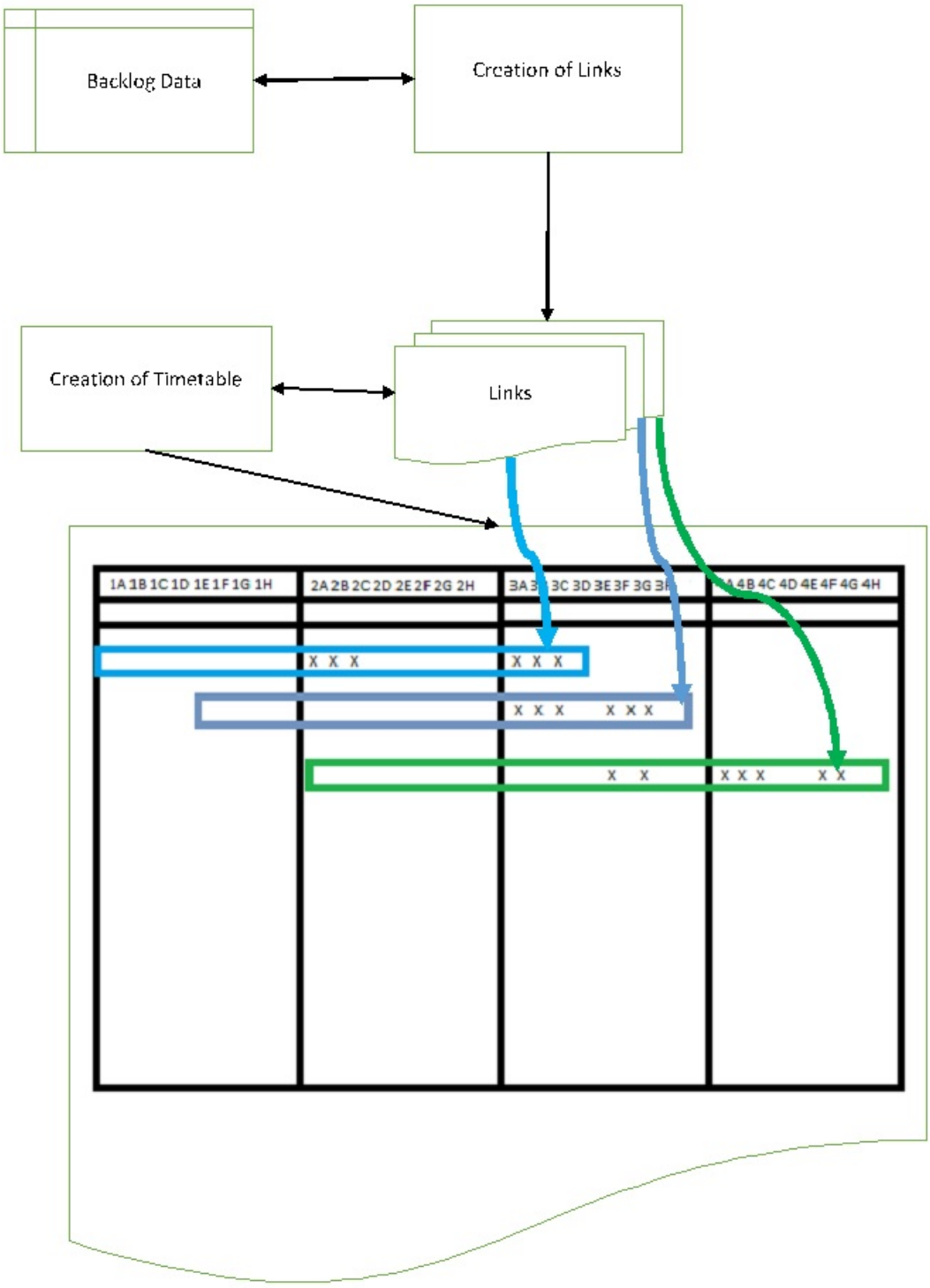


Figure 6.6: Creation of Links

A.7 Student Registration request format and process

When a student uses system to do registration of courses, he/she is shown a set of regular courses to choose from as shown in figure 6.7. He is also shown the list of courses which this student is having as backlogs and running in current semester. The subjects can be selected under Register column as per the choice of student. Regarding backlog courses, he can select which one he is interested to attend. If student has checked time table and knows the section in which no clashes are there, this section can be selected from drop down. In case the section is not known before hand, 'ANY' option may be selected sop that system searches this section for student. In the current case student has dropped one of his regular subjects and selected two backlog subjects using ANY option under section. The response from this request is shown in figure **??** where system shows the response as negative even after of searching for 242 combinations for various timetables related various sections. When this input is changed and only one backlog subject is tried as in figure 6.9, student gets a positive response as in figure 6.10 by searching through 22 combinations and student is registered in section 1B1 along with other subjects.

The process of registration has been shown using a block diagram where a student is shown interacting with process of registration by providing a list of course-sections/ Course List. The registration process starts a process to "Search Clash Free timetable" which interacts with time table and creates individual timetables to finalize a clash free time table to student if possible.

STUDENT REGISTRATION FORM

Courses

Roll Number :1040324
Name :GURPREET SINGH GILL
BRANCH/GROUP :CS
YEAR :2004

Regular Subjects :-

Teacher	COURSE	Register	Credits	NAME
SUB	CS-004	☑	3.500	Computer System Architecture
RIR	CS-006	☑	4.500	Operating System Concepts
DG	CS-010	☑	3.500	System Analysis & Design
KB	CS-055	☑	4.000	Principles of Programming Languages
MA	EN-001	☑	3.000	Environmental Studies
SKU	ES-206	☐	4.500	Electrical Engineering Materials
NEG	MA-204	☑	4.500	Numerical and Statistical Methods

Backlog Subjects Pending for you:-

Section	COURSE	Register	Credits	NAME
ANY	MA-102	☑	3.500	Mathematics-II
ANY	TA-101	☑	4.000	Engineering Graphics

2T1 | I want to register in these subjects

Start all over again

Figure 6.7: Student Registration Form Input sample for two backlogs

localhost/cgi-bin/reg/regidf.pl

STUDENT REGISTRATION FORM

Removed previous combinations
Removed previous combinations
Trying all combinations, around 242
MA-102: 1A1 1A2 1A3 1A4 1A5 1A6 **1B1** 1B2 **1B3 1B4** 1B5 1D1 1D2 1D3 1D4 1D5 1D6 1E1 1E2 1E3 1E4 1E5
TA-101: 1D1 1D2 1D3 1D4 1D5 1D6 1E1 1E2 1E3 1E4 1E5
No clash free time table iss available at course level 1 for TA-101 TA-101
No clash free time table is available at course level 1 /0 for TA-101=11TA-101
This Completed in 0.03100 CPU secondContent-type: text/html
Status: SORRY! Your Registration is Cancelled. You selected courses having clashes 9 242 0 0

Please contact Sanjeev K. Guleria (skguleria@thapar.edu) for more information.

Figure 6.8: Student Registration Form Output sample for two backlogs

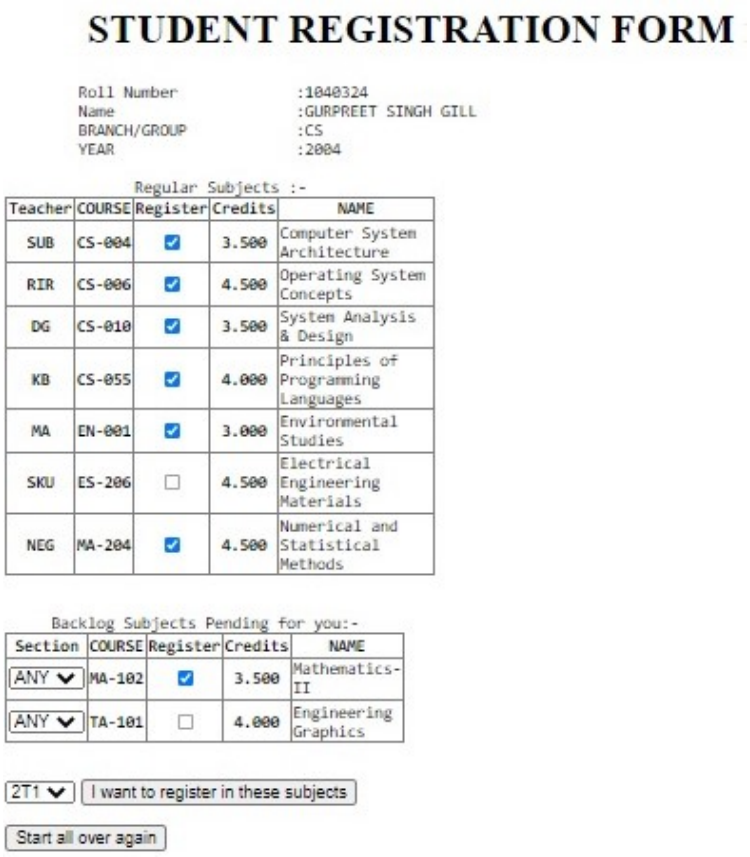


Figure 6.9: Student Registration Form Input sample for single backlog

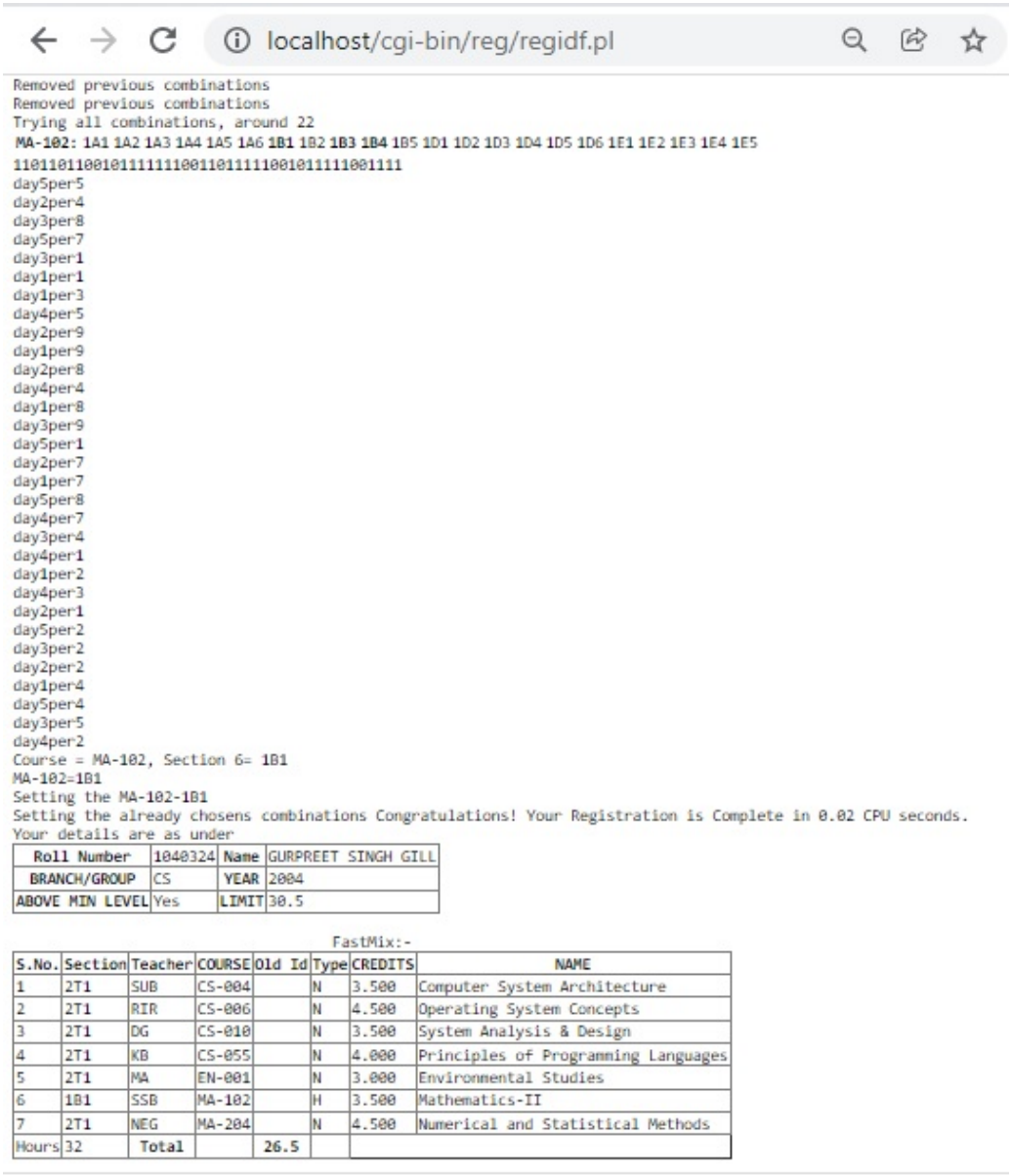


Figure 6.10: Student Registration Form Output sample for single backlog

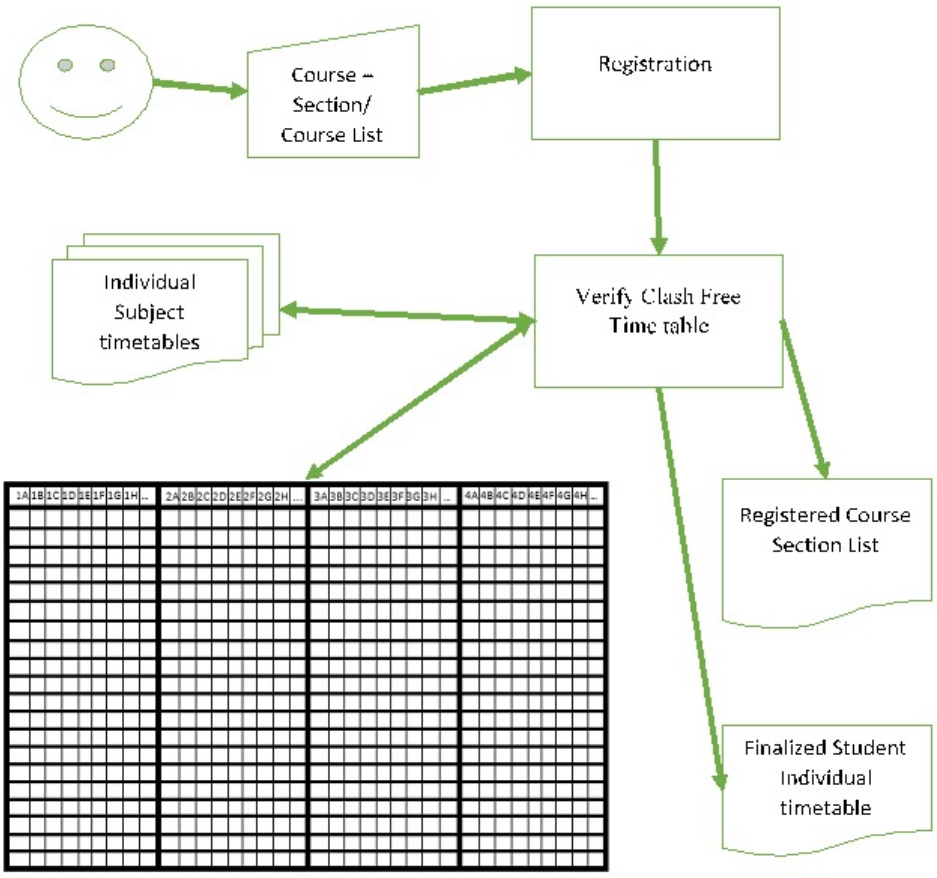


Figure 6.11: Registration Process

A.8 A peep into incremental changes in Large Scale timetabling

Figure 6.12 shows all timetable slots to select the one case where we want to work upon. After choosing one of these rows, we proceed further to select which part of this row has to be changed at figure 6.13. If we do not make any change the response is shown as in figure 6.14 as system identifies that no change is applied while showing the From and To data values.

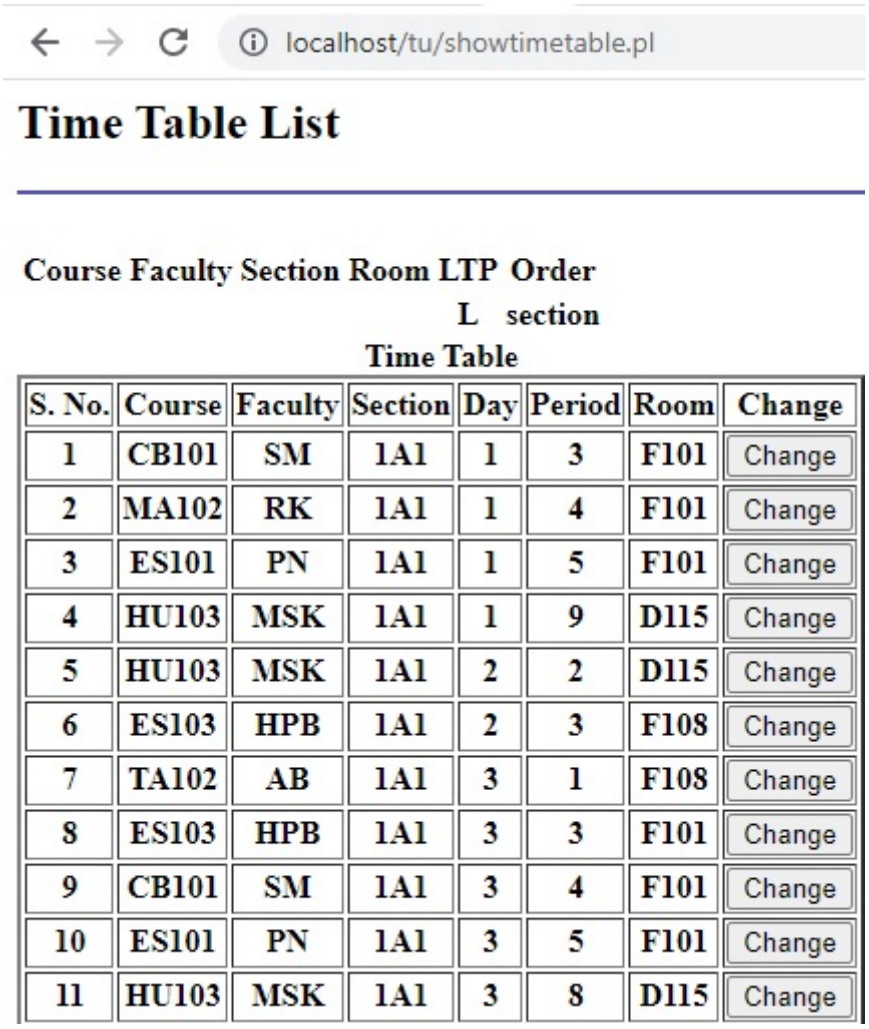


Figure 6.12: Showing timetable slots to choose one for making a change

To change the room, new room is selected from drop down in the row starting with # as shown in figure 6.15 and its negative response is shown in figure 6.16 as we have a clash.

To change the faculty, other faculty is selected from the drop down as in figure

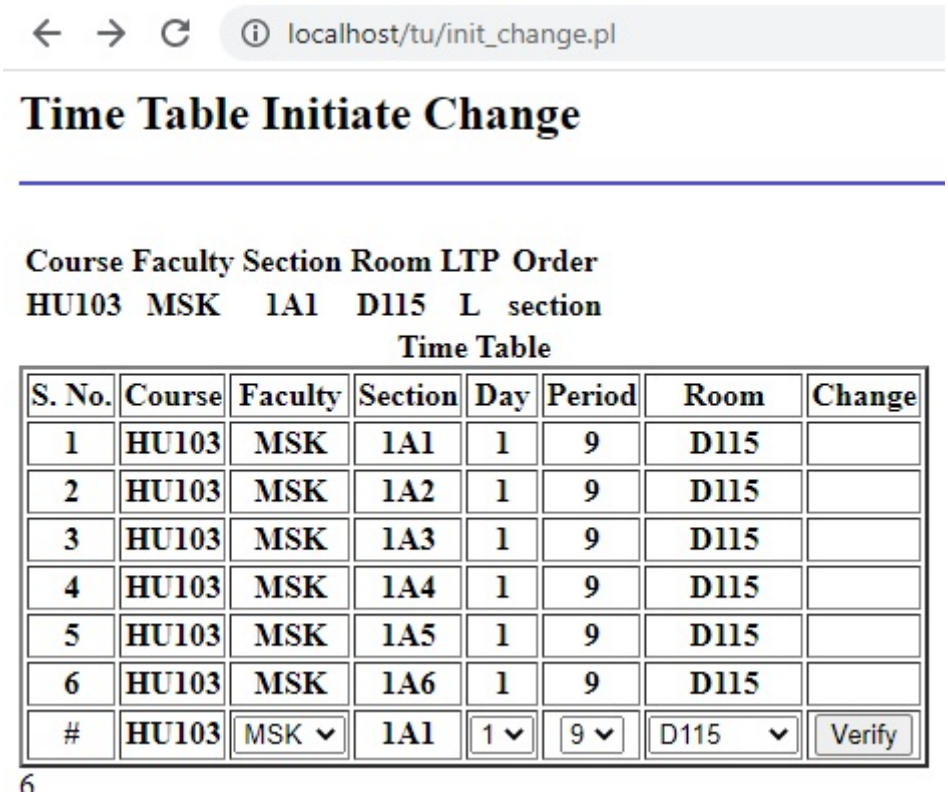


Figure 6.13: Sample input to make no change

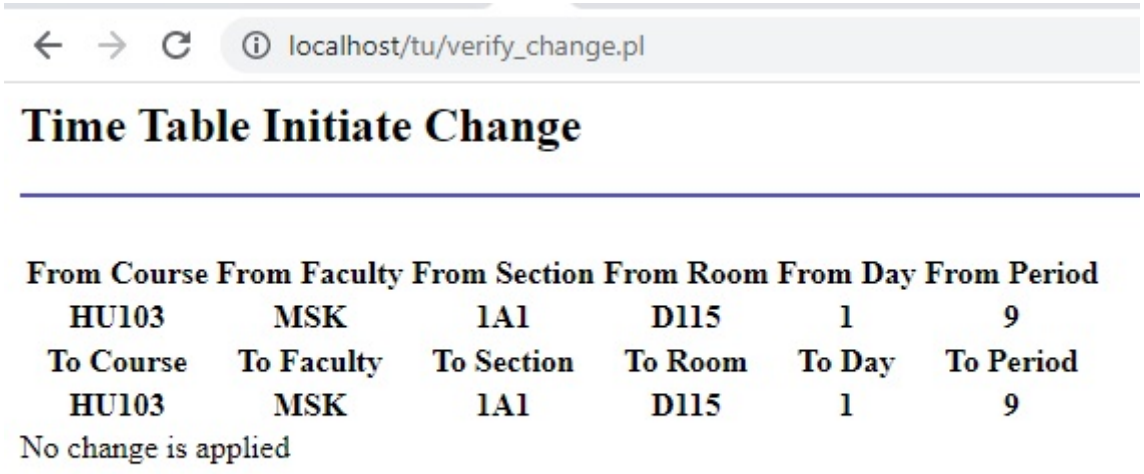


Figure 6.14: Sample output while making no change

6.17. Its response as shown in figure 6.18.

that change can not be made as we have a clash.

If slot has to be changed, the new slot can be selected as combination of two

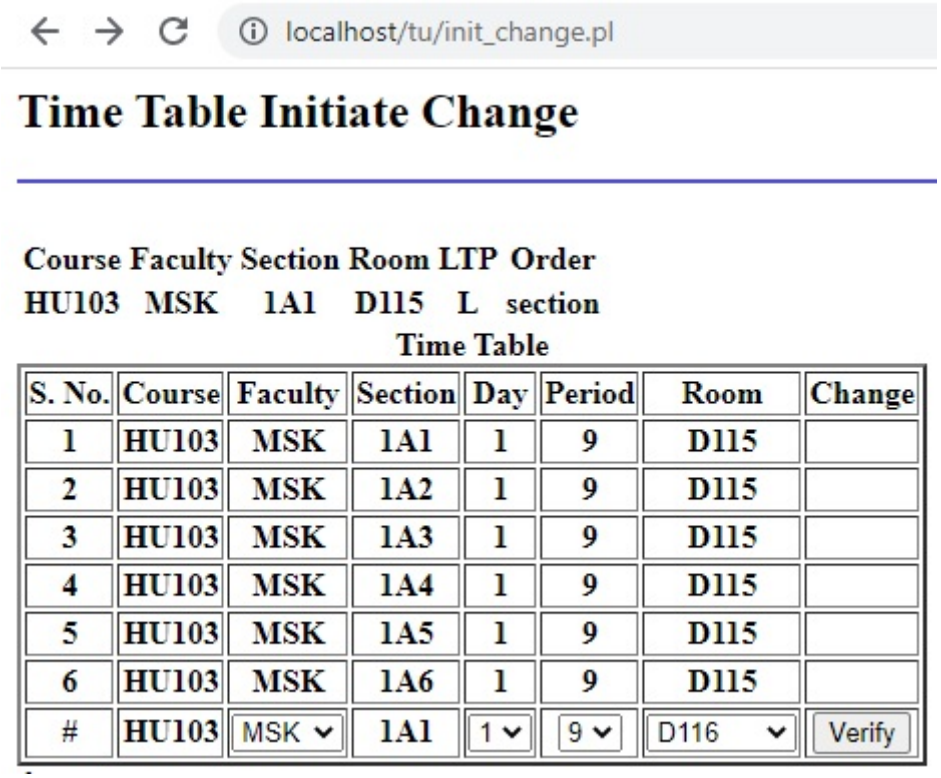


Figure 6.15: Sample input to make change of room

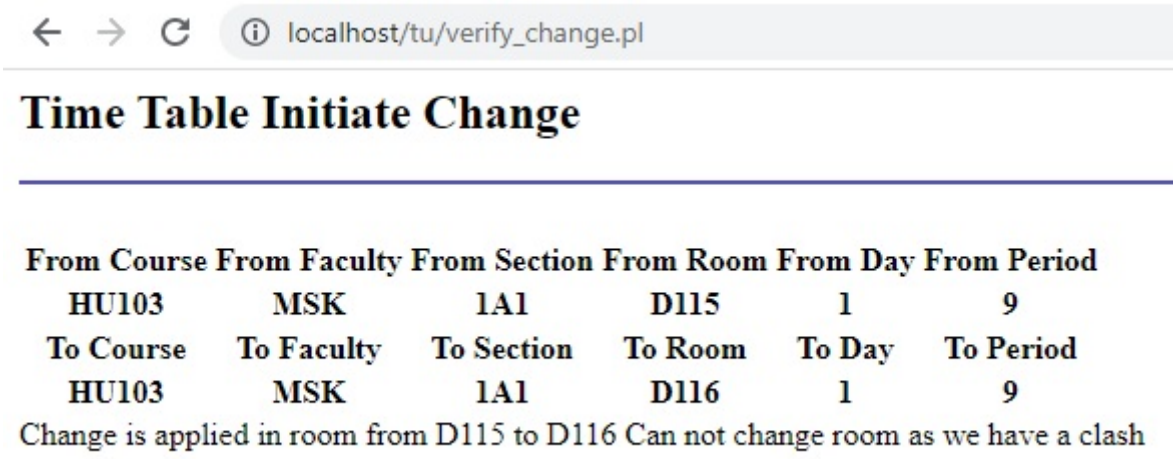


Figure 6.16: Sample output while making change in room

drop downs under Day and period in last row by giving input as in figure 6.19 and view its negative response is shown in figure 6.20 that we have a clash at section level.

Next, we try to change the faculty by giving input as per figure 6.21 and view its positive response as shown in figure 6.22 that we can change faculty.

We try to change the faculty and slot together by giving input as per figure 6.23

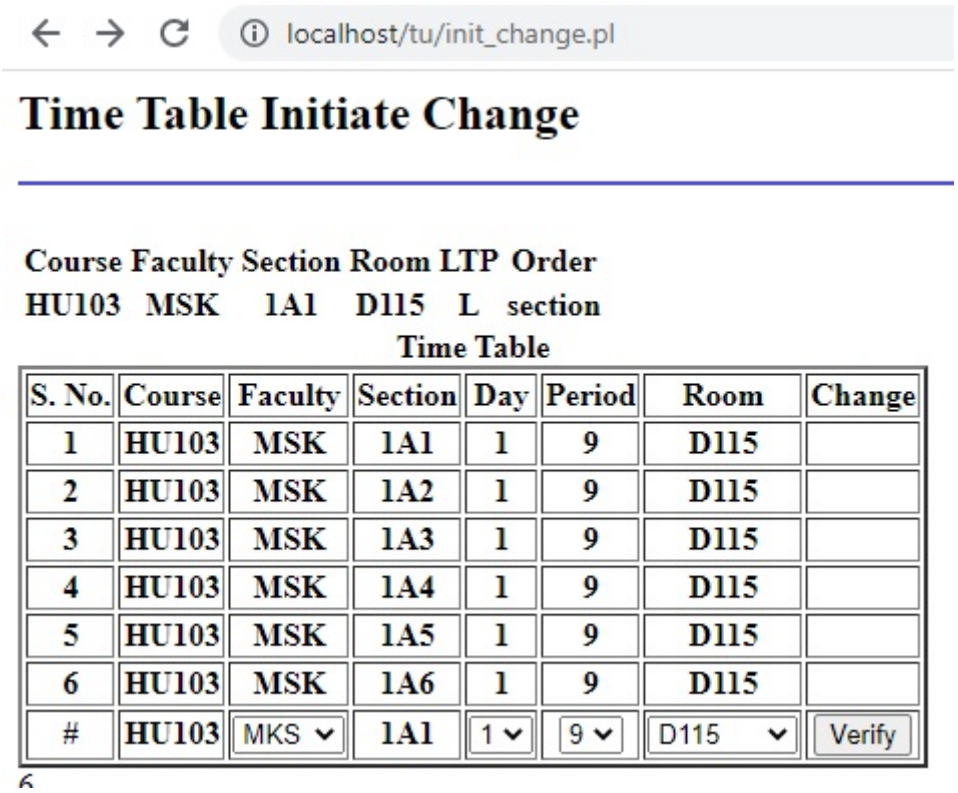


Figure 6.17: Sample input to make change of faculty

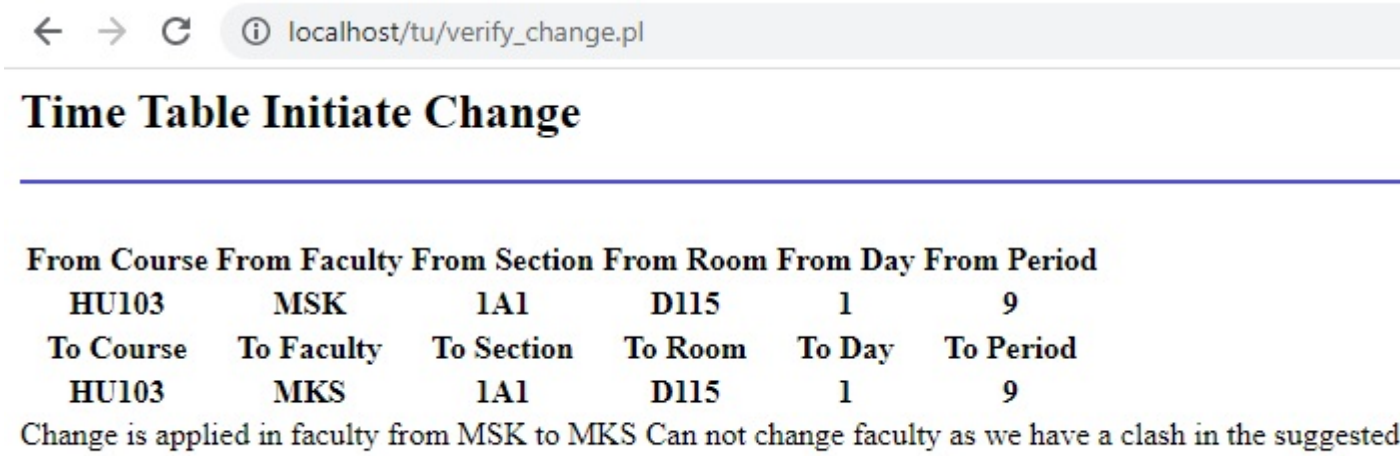


Figure 6.18: Sample output while making change of faculty where there is clash

and view detailed response as shown in figure 6.24 where system shows that individual timetables of students was cross-checked to find clashes in time tables. Ultimately, this change is not made.

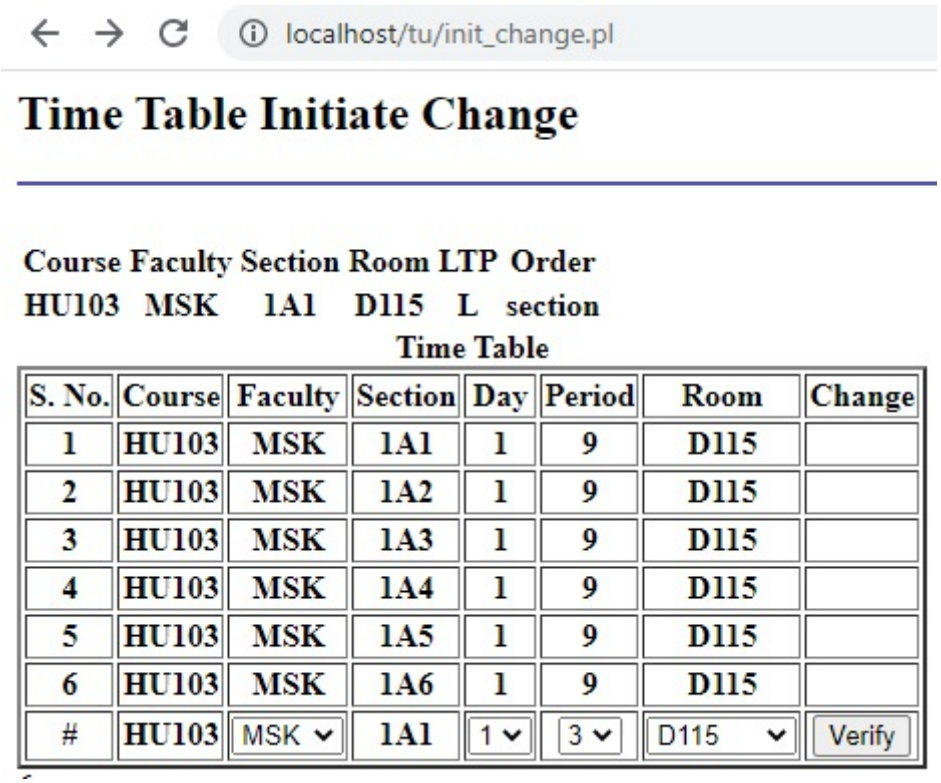


Figure 6.19: Sample input to make change of slot

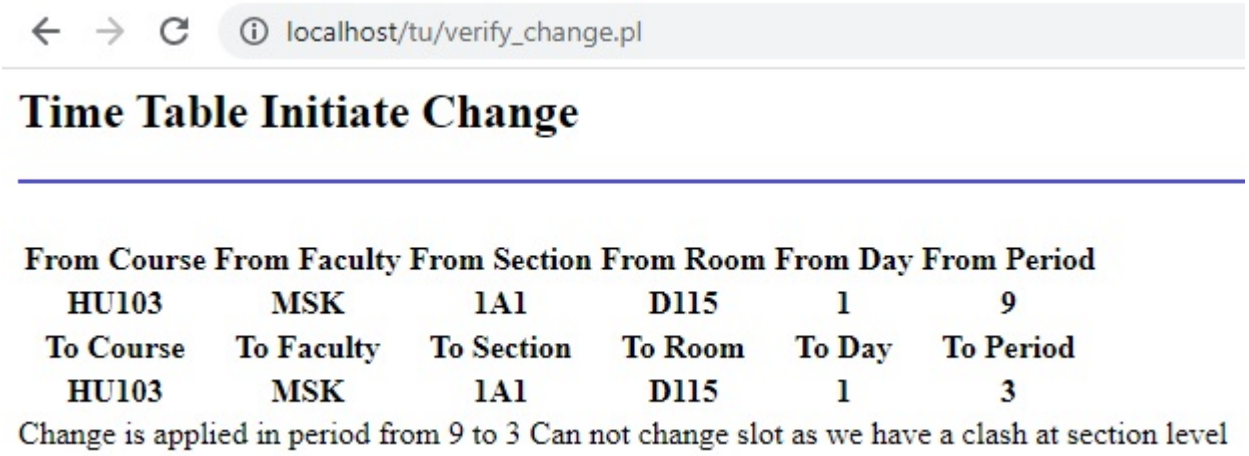


Figure 6.20: Sample output while making change in slot

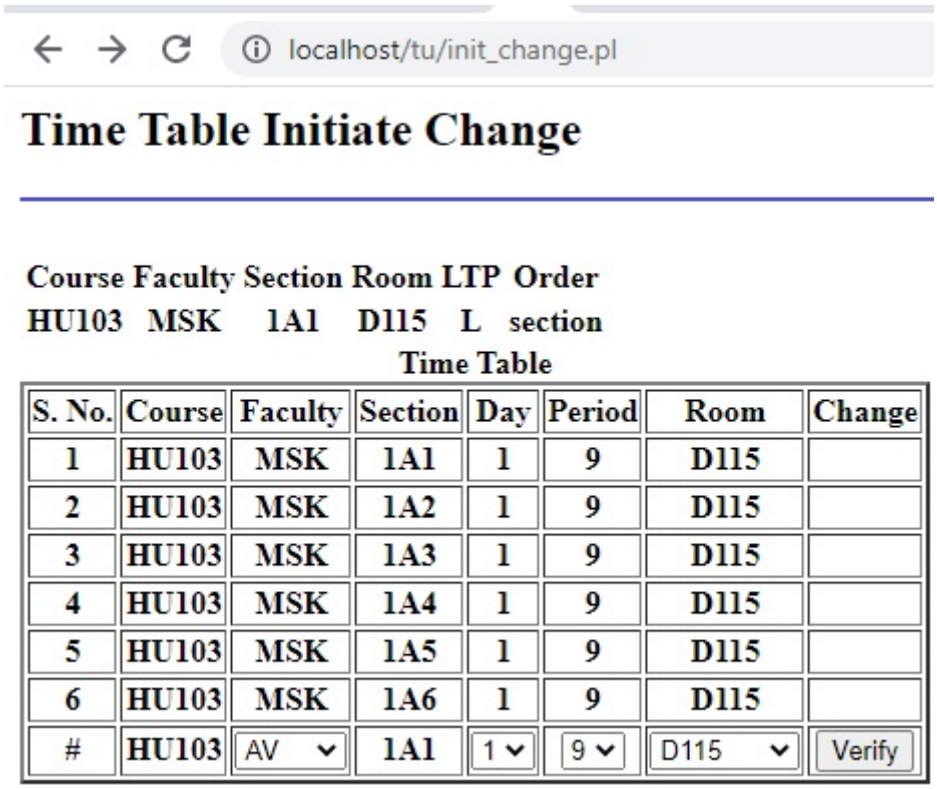


Figure 6.21: Sample input to make change of faculty

Figure 6.22: Sample output while making change of faculty where there is no clash

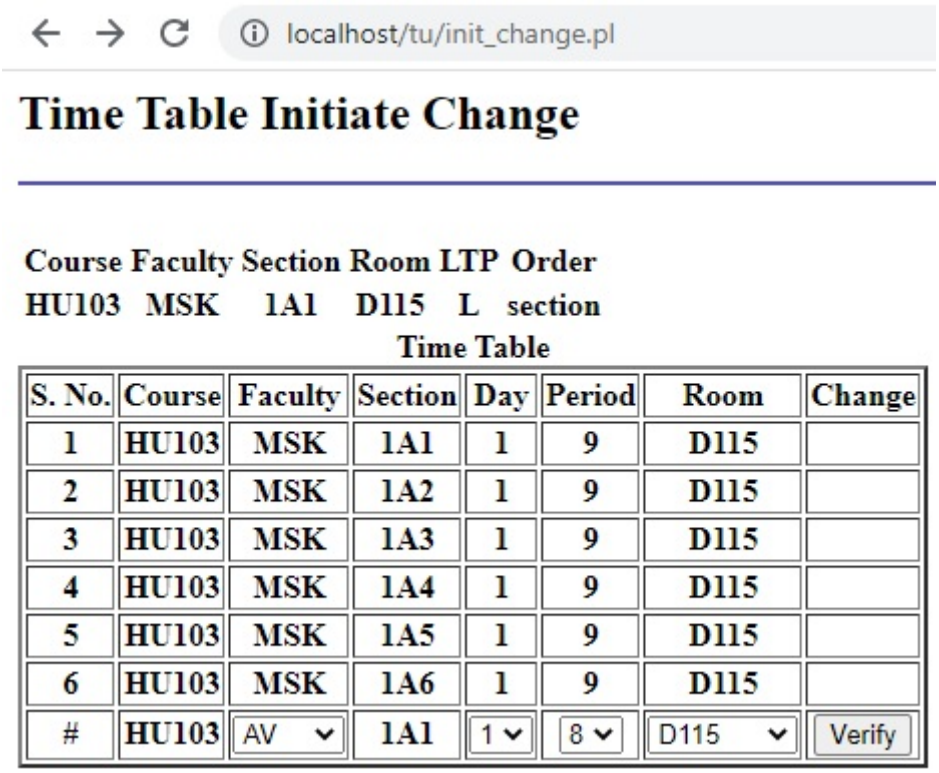


Figure 6.23: Sample input to make change of faculty and slot

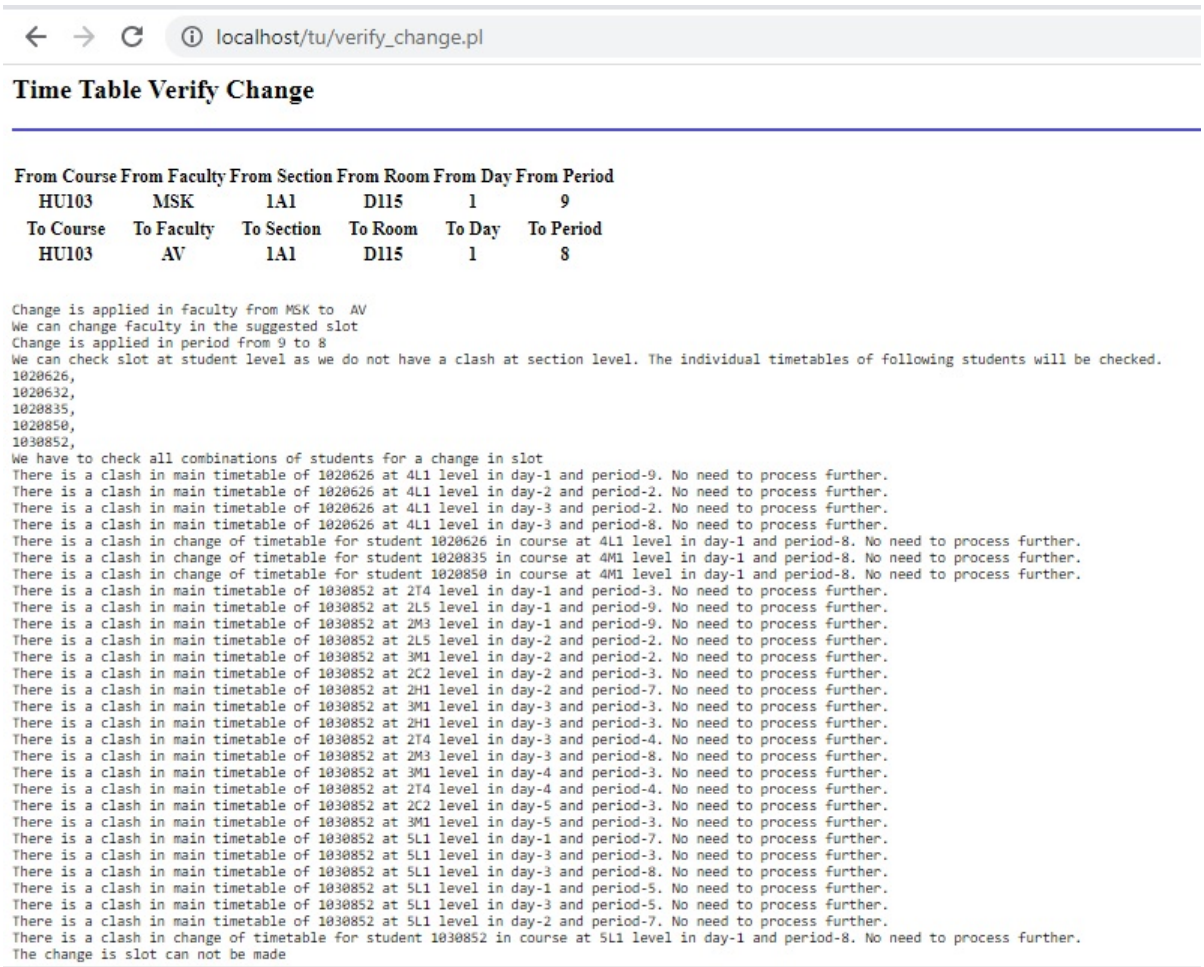


Figure 6.24: Sample output while making change of faculty and slot where there is no clash initially

List of Publications

International Journal

1. Sanjeev Kumar Guleria, A.K. Lal, "Interactive Decision Support System for planned student scheduling using same-time scheduling" in Modern Physics Letters-B Volume 33 No. 13 in May, 2019 which is SCI listed journal. Impact Factor = 1.668 http://dx.doi.org/10.1142/S0217984919501616

2. Sanjeev Kumar Guleria, A.K. Lal, "Supporting Student registration across sections using a curriculum based existing time-table" in International Journal of Advanced Science and Technology Volume 29, No. 3 pp. 5521-5531 in March, 2020 which is Scopus/ Google indexed journal having Impact Factor = 0.475 in 2022

International Conference

1. Sanjeev Kumar Guleria, "Timetabling of Exams and Courses at Thapar University: Challenges and Solutions" presented in PATAT Practice and Theory of Timetabling, 2012 August 28-31, 2012 at Son, Norway