UNIVERSITY COURSE TIMETABLING: A GENERAL MODEL

Muhammad Rozi Malim[1], Ahamad Tajudin Khader[2], Adli Mustafa[3]
[1]School of Computer Science, University of Science Malaysia,
11800 Minden, Penang, Malaysia
malimmr@celcom.net.my (Tel: 019-2124374)
[2]School of Computer Science, University of Science Malaysia,
11800 Minden, Penang, Malaysia
tajudin@cs.usm.my (Tel: 04-6533646)
[3]School of Mathematical Science, University of Science Malaysia,
11800 Minden, Penang, Malaysia
adli@cs.usm.my (Tel: 04-6533968)

Abstract: The university course timetabling is known to be a highly constrained combinatorial optimization problem. Various course timetabling models have been described in the literature but most of the models were developed for isolated problems. This paper discusses the university course timetabling in general. The works on course timetabling available in the literature are studied, and all constraints are gathered. The ultimate goal is to develop a general model of university course timetabling problem. The general model is presented; each of the constraints (hard and soft) is mathematically formulated as a 0-1 integer programming. For future work, this model will be used to develop a unified model for university timetabling problems.

Keywords: Course timetabling; timetabling constraints; general model.

1. Introduction

The university timetabling problems (UTPs) has attracted the attention of scientists from a number of differing disciplines, including operations research and artificial intelligence, since the 1950s. UTPs can be divided into two main categories: *course timetabling* - involves the scheduling of courses within a given number of timeslots and their allocation into available classrooms (weekly basis) while satisfying certain constraints [Burke et al. (2003)]; and *examination timetabling* - involves the scheduling exams (one for each course) within a given amount of time in a set of rooms. The main difference is, in course timetabling there cannot be more than one course or event per classroom, while in examination timetabling there can be more than one exam.

University course timetabling problem (UCTP) is known to be a highly constrained combinatorial optimization problem (NP-hard). The UCTP can be viewed as a multi-dimensional assignment problem [Carter and Laporte (1998)]. Given a set of courses, a set of lecturers, a set of timeslots, a set of classrooms, and a set of student enrollments to courses, the problem is to assign lecturers to courses, courses to timeslots, and courses to classrooms subject to a set of *hard* and *soft* constraints. The main difficulty is to obtain a *conflict-free* within a limited number of timeslots and classrooms. Conflicting objectives and the changing set of constraints in different institutions makes the course timetabling problem (CTP) very challenging. Various course timetabling models have been described in the literature, but most of the models were developed for isolated problems, i.e. problem-based models.

This paper discusses the UCTP in general. The works on course timetabling available in the literature are studied and all constraints (hard and soft) are gathered. The ultimate goal is to develop a general model of the UCTP. This general model is presented in Section 4; all constraints are mathematically formulated as *0-1 integer programming*.

2. Course Timetabling Constraints

The sets of constraints differ significantly from institution to institution. Different categories of people have different priorities in the course timetabling process. There are three main categories of people who are affected by the results of the process; administration, lecturers, and students [Burke et al. (2001)]. Consequently, the quality of a timetable can be assessed from various points of view and the importance of a particular constraint can very much depend upon the priorities of the three categories.

Loo et al. [1986] set out four categories of constraints: *room requirements* (1-6), *student-group requirements* (7-14), *staff requirements* (15-22), and *class requirements* (23-29).

(1) The utilization of classroom capacity should be maximized.

(2) Preferred rooms for each student-group should be used first; minimizes the need to change rooms.

(3) Certain classrooms should be used as much as possible.

(4) Some rooms (e.g. conference rooms) should not be used unless there is no other room available.

(5) Certain rooms (e.g. drawing rooms) should be used only for particular subjects.

(6) It should be possible to group rooms according to service packages (air-conditioning, lighting, etc).

(7) Study workload for students should be evenly distributed over the week.

(8) Certain student-groups may require a common free timeslot once a week for their project works.

(9) The consecutive hours for lectures or tutorials should not exceed a predefined length of time.

(10) There should be limits on the number of lectures and tutorials to be attended by students each day.

(11) Lunch and dinner breaks should be provided for students.

(12) Each student group should have adequate chances to attend most of the optional subjects.

(13) Students should be able to be divided into groups for tutorials or combined for common lectures.

(14) There should be a limit on the total number of first-period events attended by the students each week.

(15) Teaching workload for staff should be evenly distributed over the week.

(16) Time constraints for lecturers from other faculties or part-time lecturers should be considered.

(17) All full time lecturers may require a common free timeslots once a week for the purpose of meetings.

(18) The consecutive hours for lecturing/tutoring by a lecturer should not exceed a predefined length.

(19) There should be limits on the number of lectures and tutorials to be conducted by a lecturer each day.

(20) Lunch and dinner breaks should be provided for lecturers; these breaks may be staggered.

(21) Allowance should be made for several lecturers teaching different portions of a given subject.

(22) There should be a limit on the total number of first-period classes conducted by a lecturer each week.

(23) The morning should be free for lecturer or student having evening events later in the day.

(24) Repeated lectures (large enrollments) should not be conducted by the same lecturer in the same day.

(25) Students from different departments may be grouped together for common lectures if necessary.

(26) It is preferred that lectures and tutorials be conducted in the morning if possible.

(27) If possible, Saturday is to be scheduled as a free day for professor as well as students.

(28) Sometimes it is desirable to have at least a 15-minute break between events.

(29) Some events (e.g. ad hoc or make-up sessions) are to be arranged manually.

Corne et al. [1994] described the five kinds of constraints of general CTP: *Unary Constraints* - involve just one event, fall into two classes: exclusions - an event must not take place in a given room, must not start at a given time, or cannot be assigned to a certain agent; specifications - an event must take place at a given time, in a given place, or must be assigned to a given agent. *Binary Constraints* - involve restrictions on the assignments to a pair of events, fall into two classes: edge constraints - arising because of the simple fact that people cannot be in two places at once; juxtaposition constraints - the ordering and/or time gap between two events is restricted in some way. *Capacity Constraints* - specify that some function of the given set of events occurring simultaneously at a certain place must not exceed a given maximum, e.g. room capacity. *Event-Spread Constraints* - the way that events are spread out in time. It may require that multiple lectures on the same topic should be spread out as evenly as possible during the week. *Agent Constraints* - involve restrictions on the total time assigned for an agent in the timetable, and restrictions and specifications on the events that each individual agent can be involved in.

In Alkan & Ozcan [2003], six different types of constraints can be identified for UCTP: *Exclusions* represent the excluded members of resources for the variables; e.g. 'Data Structures should not be scheduled on Tuesdays'. *Presets* represent the predetermined assignments for some variables; e.g. 'Digital Electronics is scheduled on Fridays at 14:00-17:00'. *Edge constraints* represent a pair of course meetings that should be scheduled without a clash. *Ordering constraints*, also known as juxtaposition, represent an ordering between course meetings. *Event-spread constraints* deal with the way how the course meetings are spread out in time. *Attribute constraints* represent restrictions that apply between the attributes of a course meeting and/or the attributes of its assignment; e.g., 'total number of students taking a course should not exceed the capacity of the classroom'.

The *common hard constraints* for UCTP that agreed by most researchers in the literature are as follows:

(ordered by their importance):

(1) All events/courses must be assigned to lecturers.

(2) All lecturers must be assigned to events/courses.

(3) No lecturer must be assigned to more than one event at the same time.

(4) A lecturer must not be assigned to events when he/she is unavailable.

(5) Each lecturer's minimum teaching requirement must be accomplished.

(6) The lecturer's number of overtime hours should not exceed a specified maximum.

(7) One afternoon (e.g. Monday) is reserved for lecturer' meetings.

(8) No student or student-group must be assigned to more than one event at the same time.

(9) Each event is assigned to at least one timeslot.

(10) No two or more events of required courses of a student-program can be scheduled at the same time.

(11) No events of required and optional courses of a student-program can be scheduled at the same time.

(12) Events of day courses must be scheduled during the day.

(13) Events of evening courses must be scheduled during the evening.

(14) Preassigned events must be scheduled at their preassigned timeslots.

(15) Some events are restricted to some specific timeslots and must not be assigned.

(16) Some events need to be held consecutively; consider a six-hour practical experiment.

(17) Each room must not host more than one event at any timeslot.

(18) The classrooms capacities assigned to events must equal to or larger than the student enrollments.

(19) An event can only be assigned to a classroom if and only if the room is available.

(20) Preassigned events must be scheduled into their preassigned classrooms.

(21) Some events are restricted to some specific classrooms and must not be assigned.

(22) Certain events must be taught in the same classroom at the same time.

(23) Certain events should have exactly the same teaching time but different classrooms.

The *common soft constraints* for UCTP that agreed by most researchers in the literature are as follows:

(ordered by their importance):

(1) A lecturer should not teach more than a specified number of the maximum weekly teaching load.

(2) A lecturer should not teach more than a specified number of the maximum daily teaching load.

(3) A lecturer should have at least one timeslot free between any two teaching timeslots.

(4) Some lecturers, by contrast, wish to have all their events scheduled to consecutive periods.

(5) Lecturers should be assigned to their preferred events/courses.

(6) An event should not be assigned to a certain lecturer.

(7) The morning should be free for lecturers having afternoon events on the same day.

(8) No student/student-group can attend more than x events in a day.

(9) A student should not have only one event in a day.

(10) Students or student-groups don't like to have many free timeslots between two events.

(11) Events of optional courses of a student-program should be scheduled in fewest time conflicts.

(12) Events of the same course and student-group should not be scheduled on the same day.

(13) Student-groups with the same event and student-program should be scheduled at the same timeslot.

(14)Student-groups with the same event but different student-programs should not be scheduled at the same time.

(15) All tutorials or lab sessions of any course should occur later in the week than the first lecture.

(16) If an event is held more than once a week, there will always be a least one day in between.

(17) Friday afternoon shall be used only after all other scheduling possibilities are exhausted.

(18) Events should not be assigned during lunch time.

(19) Certain student-groups may require a common free timeslot once a week for their project works.

(20) Events should be assigned to rooms in such a way that the room utilization can be maximized

(21) Events should be scheduled into classrooms as close as possible to their lecturer.

(22) Events should be scheduled into the classrooms at their departments.

(23) Events should be scheduled into classrooms of the type specified by the department or professor

(24) All events of the same student-group should be scheduled in the same classroom.

(25) There must be a limit on the total number of classrooms assigned to the first-timeslot events (early morning).

(26) Classrooms should not be scheduled during those times they have been reserved.

(27) Students or student-groups should be assigned to their preferred classrooms.

(28) Sufficient time should be provided for students to move from one room to another..

3. Course Timetabling Models

Some of the course timetabling models, described in the literature, are presented as follows:

In Ross et al. [1994], the basic element of a UCTP is a set of events $E=\{e_1,\ldots,e_v\}$. Each member of E is a unique event requiring assignment of a time and a place. It may be a lecture, a tutorial, a lab session or some other event which plays a part in the term timetable. Each event e_i has an associated length l_i and an associated size s_i which is either known or an estimate of the number of students expected to attend that event. There is also a set of agents $A=\{a_1,\ldots,a_t\}$; these are lecturers, tutors, technicians, etc. people with some kind of distinguished role to play in an event. Finally, there is a set of places $P=\{p_1,\ldots,p_q\}$, and a set

of times $T=\{t_1,\ldots,t_s\}$. An *assignment* is a four-tuple (a, b, c, d) in which $a\in E$, $b\in T$, $c\in P$, $d\in A$, with the interpretation 'event a start at time b in place c and is taught by agent d'. A lecture timetable is simply a collection of n assignments, one for each event.

In the weekly UCTP defined by Erben and Keppler [1995], each day of the week is divided into 6 periods (of 90 minutes' duration). There are 5 working days per week. Hence, the set P of *periods* consists of 30 elements. These elements are denoted by *mon1,...,mon6,tue1,tue2,...,fri6.* A number of *courses* are offered in different departments, and each course lasts eight semesters. A *class* consists of all students studying a given course in the same semester. C denotes the set of all classes. For each class $c\in C$ the (estimated) maximum number of students must be given as part of the input. T is the set of all *teachers*. Each teacher requires a number of free-timeslots (or even one or two free days) where he or she is unavailable. R is the set of all *rooms* available in the university. A room can be a laboratory or a lecture theatre. Some rooms can only be used by the department to which they belong. Each room has a maximum capacity which must be given. Each department offers a number of *course modules.* A course module may be taken by students of one or more classes. It may either be compulsory or optional. A course module consists of one or more *lessons.* A lesson may be a lecture, a laboratory or a group exercise class, and it has duration of 90 minutes. The lessons belonging to one course module may be taught by different teachers. The set of all lessons is denoted by L. For each $l\in L$, its unique teacher and the coordinate list of classes must be present.

In Alkan and Ozcan [2003], CTPs are defined as constraint optimization problems that can be represented by a 3-tuple (V,D,C). V is a finite set of course meetings in a department, faculty or university, $V=\{v_1,\ldots,v_N\}$, $D=\{d_1,\ldots,d_N\}$, is a finite set of domains of variables. For example, let $G=\{t_1,\ldots,t_M\}$ represents a set of start times for a course meetings, then a possible domain of each variable can be $d_i\subseteq G$ and C is a set of constraints to be satisfied, $C=\{c_1,\ldots,c_L\}$. Domain of a variable can be a product of sets, each representing a different resource. For example, $d_i\subseteq G\times S$ can be a domain of a variable, where S represents the set of classrooms. Resources other than time were ignored. UCTP can be described as a search for finding the best *assignment* (v_i, t_j) for each variable $v_i\in V$ such that all the constraints are satisfied. The assignment implies that the course meeting of v_i starts at t_j.

Avella and IVasil'ev [2003] introduced some notations and definitions for CTP as follows. Let $C=\{1,\ldots,\bar{c}\}$ be a set of courses. For any $c\in C$, let n_c be the number of hours to be scheduled per week and let $n^c_{\min}$ and $n^c_{\max}$ be the minimum and maximum daily number of teaching hours. Let $R=\{1,\ldots,\bar{r}\}$ be a set of rooms. Let $T=\{1,\ldots,\bar{t}\}$ be a set of one-hour time periods. Let $D=\{1,\ldots,\bar{d}\}$ be the days of the week when teaching is allowed. For any $d\in D$, let τ_d and ι_d be the first timeslots of the morning and afternoon session in day d. Let $G=\{1,\ldots,\bar{g}\}$ be a set of classes. For any class $g\in G$, let $C_g\subset C$ denote courses that the class g should attend. Let $S=\{1,\ldots,\bar{s}\}$ be a set of teachers and, for any $s\in S$, let $C_s\subset C$ be a subset of courses taught by teacher s. Let k_s denote the maximum weekly number of teaching days for the teacher s. Let $l_{\max}$ be the maximum daily number of teaching hours for any class $g\in G$. Let p_{ct} be a penalty if the course c is scheduled at the time t. Usually p_{ct} measures the 'undesirability' of the teacher s to teach the course c at time t. A *timetable* is an assignment of courses C to rooms R and to time periods T.

The course timetabling model by Ozdemir and Gasimov [2003] involves assigning instructors to courses. The instructors are grouped as tenured and recent faculty. The model parameters, and decision variables are: $I=\{1,\ldots,m\}$ courses; $J=\{1,\ldots,n\}$ instructors; $J_0=\{1,\ldots,k\}$ tenured instructors; $J_n=\{k+1,\ldots,n\}$ recent instructors; $J=J_0\cup J_n$ and $k<n$; h_i: total number of lecture

hours for the ith course in a week; l_j and u_j: lower and upper bounds for the jth instructor's weekly load, respectively; t_{ij}: preference level of the ith course by the jth instructor ($t_{ij} \geq 1$, 1 indicates the most desired course); a_{ij}: administrative preference level for the assignment of the ith course to jth instructor. In this model the *decision variable* x_{ij} represents the assignment of a course to an instructor and is defined $x_{ij} = 1$ if course i is assigned to instructor j, 0 otherwise.

4. General Model of University Course Timetabling

Here, a *general university course timetabling model* (GUCTM) is presented. This model is based on the models, and constraints, discussed in the literature. Each of the constraints (hard and soft) is mathematically formulated as a *0-1 integer programming*. UCTP is known to be a highly constrained and complex optimization problem. To reduce the complexity of the problem, it is normally divided into *three subproblems*; 'course-professor assignment', 'event-timeslot assignment', and 'event-room assignment'. These three subproblems can be modeled separately. In 'course-professor assignment', the professors are scheduled to courses and events; in 'event-timeslot assignment', the events are scheduled into timeslots; and in 'event-room assignment', events (in each timeslot) are assigned to rooms. Hence, in a UCTP, an *assignment* is an ordered *4-tuple* (a, b, c, d), where $a \in E$, $b \in T$, $c \in R$, $d \in P$, and is interpreted as 'event a starts at timeslot b in room c, and taught by professor d'. The main components of a university course timetabling model are *problem definition and initialization*, *hard constraints*, and *soft constraints*.

4.1 Problem Definition and Initialization

There are *nine* sets of variables that should be taken into account in a university course timetabling:

Course: represents the course to be scheduled. The domain of this variable, C, is the set of all courses in all student-programs. Each course c_i, $i \in \{1,...,n_1\}$ has a list of events, a list of student enrollments, preassigned professor(s), the number of weekly contact hours, and type. There are two types of courses; required courses, c_i, $i \in \{1,...,\alpha_2\}$, and optional courses, c_i, $i \in \{\alpha_2+1,...,n_1\}$. Required courses can be subdivided into two subgroups, day courses, c_i, $i \in \{1,...,\alpha_1\}$, and evening courses, c_i, $i \in \{\alpha_1+1,...,\alpha_2\}$. Similarly, optional courses can be subdivided into two subgroups, day courses, c_i, $i \in \{\alpha_2+1,...,\alpha_3\}$, and evening courses, c_i, $i \in \{\alpha_3+1,...,n_1\}$; where $\alpha_1 < \alpha_2 < \alpha_3 < n_1$.

Event: represent the event (lecture, tutorial, lab, or seminar) to be timetabled. The domain of this variable, E, is the set of all events in all courses. Each event, e_j, $j \in \{1,...,n_2\}$, has type, length (in hours), weekly frequency, preassigned professor(s), preassigned timeslot(s), restricted timeslot(s), preassigned room, and restricted room(s). There are two groups of events; events of required courses, e_j, $j \in \{1,...,\beta_2\}$, and events of optional courses, e_j, $j \in \{\beta_2+1,...,n_2\}$. Events of required courses can be subdivided into two subgroups, day event, e_j, $j \in \{1,...,\beta_1\}$, and evening events, e_j, $j \in \{\beta_1+1,...,\beta_2\}$. Similarly, event of optional courses can be subdivided into two subgroups, day events, e_j, $j \in \{\beta_2+1,...,\beta_3\}$, and evening events, e_j, $j \in \{\beta_3+1,...,n_2\}$; where $\beta_1 < \beta_2 < \beta_3 < n_2$.

Department: represents the department or faculty. The domain of this variable, D, is the set of all departments or faculties at an institution. Each department d_j, $j \in \{1,...,n_3\}$, has a list of professors, a list of student-programs, and a list of rooms.

Professor: represents the lecturer or tutor. The domain of this variable, P, is the set of all teaching professors. Each professor p_i, $i \in \{1,...,n_4\}$, has a list of courses which he/she can teach (each with a preference level), a list of timeslots during which he/she is unavailable, lower and upper limits of weekly load, and type (junior or senior).

Student-program: represents the student academic program. The domain of this variable, M, is the set of all student programs in all the departments or faculties at an institution. Each student-program m_j, $j \in \{1,...,n_5\}$, has a list required courses, a list of optional courses, student-groups, and type. There are two types of student-programs; day student-programs, $m_j, j \in \{1,...,\delta\}$, and evening student-programs, $m_j, j \in \{\delta+1,...,n_5\}$, $\delta < n_5$.

Student-group: represents the student-class group. For each student-program with a large number of students, the students are allocated into a number of different student-groups or classes. The domain of this variable, G, is the set of all student-groups in all student-programs. Each student-group g_i, $i \in \{1,...,n_6\}$, has a list of students, a list of events, and a list of preferred rooms.

Student: represents the student enrolled for the courses in a student-program. The domain of this variable, S, is the set of all students enrolled at an institution. Each student s_i, $i \in \{1,...,n_7\}$, has a list of required courses, a list of optional courses, and the total number of weekly load (in hours), and type (day or evening).

Timeslot: represents the time occupied by an event. The domain of this variable, T, is the set of all (one-hour) timeslots (in a week) permitted in the timeslot system. Each timeslot t_i, $i \in \{1,...,n_8\}$, has start time, finish time, a list of available rooms, and type. The timeslots are divided into five groups (days) of equal number of timeslots, and for each day the timeslots are subdivided into four types (morning, lunch, afternoon, evening). If each day starts at 8am and finish at 10pm, then there are 14 timeslots per day: Monday: $\{t_1,..,t_5\}$, $\{t_6\}$, $\{t_7,..,t_{10}\}$, $\{t_{11},..,t_{14}\}$; Tuesday: $\{t_{15},..,t_{19}\}$, $\{t_{20}\}$, $\{t_{21},..,t_{24}\}$, $\{t_{25},..,t_{28}\}$; Wednesday: $\{t_{29},..,t_{33}\}$, $\{t_{34}\}$, $\{t_{35},..,t_{38}\}$, $\{t_{39},..,t_{42}\}$; Thursday: $\{t_{43},..,t_{47}\}$, $\{t_{48}\}$, $\{t_{49},..,t_{52}\}$, $\{t_{53},..,t_{56}\}$; Friday: $\{t_{57},..,t_{61}\}$, $\{t_{62}\}$, $\{t_{63},..,t_{66}\}$, $\{t_{67},..,t_{70}\}$; hence $n_8 = 70$.

Room: represents the room where an event to be held. The domain of this variable, R, is the set of all rooms available to the system. Each room r_j, $j \in \{1,...,n_9\}$, has a room-type (lecture hall, tutorial room, laboratories, seminar/conference room), seating-type, capacity, and a list of reserved rooms.

The matrices are required to show the interrelationships between these sets of variables. These matrices would assist the formulation of the hard and soft constraints as 0-1 integer programming. There are two types of matrices, *input matrices* and *output matrices*. The input matrices are the matrices where the values are known earlier and have been allocated or preassigned, i.e. the timetabling data and preassignments. The output matrices are the *assignment* matrices where the values need to be determined by solving the CTP.

4.1.1 *Input Matrices*

Course–event allocation matrix: each course consists of events; $\mathbf{A}=C\times E$, $a_{ij}=1$ if course c_i consists of event e_j, 0 otherwise; $n_A(c_i)$ is the number of events in course c_i, and $n_A(e_j)=1$ since each event is belong to only one course.

Course–professor preference matrix: $\mathbf{B}=C\times P$, b_{ij} represents the preference level of professor p_j to teach course c_i; '1' indicates the highest level of teaching preference, '2' indicates the second highest level, and so on.

Course–professor preassignment matrix: $\mathbf{D}=C\times P$, $d_{ij}=1$ if course c_i is preassigned to professor p_j, 0 otherwise; $n_{\mathbf{D}}(c_i)$ is the number of preassigned professors for course c_i, and $n_{\mathbf{D}}(p_j)$ is the number of courses preassigned to professor p_j.

Course–student enrollment matrix: $\mathbf{E}=C\times S$, $e_{ij}=1$ if student s_j is enrolled in course c_i, 0 otherwise; $n_{\mathbf{E}}(c_i)$ is the number of students enrolled in course c_i, and $n_{\mathbf{E}}(s_j)$ is the number of courses enrolled by student s_j.

Course–student-program matrix: $\mathbf{F}=C\times M$, $f_{ij}=1$ if course c_i is offered in student-program m_j, 0 otherwise; $n_{\mathbf{F}}(c_i)$ is the number of student-programs that offer course c_i, and $n_{\mathbf{F}}(m_j)$ is the number of courses in student-program m_j.

Event–professor preassignment matrix: $\mathbf{G}=E\times P$, $g_{ij}=1$ if event e_i is preassigned to professor p_j, 0 otherwise; $n_{\mathbf{G}}(e_i)$ is the number of professors that preassigned to event e_i, and $n_{\mathbf{G}}(p_j)$ is the number of events preassigned to professor p_j.

Event–timeslot preassignment matrix: $\mathbf{H}=E\times T$, $h_{ij}=1$ if event e_i is preassigned to timeslot t_j, 0 otherwise; $n_{\mathbf{H}}(e_i)$ is the number of timeslot that preassigned to event e_i, and $n_{\mathbf{H}}(t_j)$ represents the number of preassigned events in timeslot t_j.

Event–timeslot restriction matrix: $\mathbf{I}=E\times T$, $i_{ij}=1$ if event e_i is restricted to timeslot t_j, 0 otherwise; $n_{\mathbf{I}}(e_i)$ is the number of restricted timeslots for event e_i, and $n_{\mathbf{I}}(t_j)$ represents the number of restricted events for timeslot t_j.

Event–room preassignment matrix: $\mathbf{J}=E\times R$, $j_{ij}=1$ if event e_i is preassigned to room r_j, 0 otherwise; $n_{\mathbf{J}}(e_i)$ is the number of rooms that preassigned to event e_i, and $n_{\mathbf{J}}(r_j)$ represents the number of preassigned events in room r_j.

Event–room restriction matrix: $\mathbf{K}=E\times R$, $k_{ij}=1$ if event e_i is restricted to room r_j, 0 otherwise; $n_{\mathbf{K}}(e_i)$ is the number of restricted rooms for event e_i, and $n_{\mathbf{K}}(r_j)$ represents the number of restricted events for room r_j.

Department–professor matrix: $\mathbf{L}=D\times P$, $l_{ij}=1$ if professor p_j belongs to department d_i, 0 otherwise; $n_{\mathbf{L}}(p_j)=1$ since each professor belongs to only one department, and $n_{\mathbf{L}}(d_i)$ is the number of professors in department d_i.

Department–student-program matrix: $\mathbf{M}=D\times M$, $m_{ij}=1$ if department d_i owns student-program m_j, 0 otherwise; $n_{\mathbf{M}}(d_i)$ is the number of student-programs in department d_i, and $n_{\mathbf{M}}(m_j)=1$, only one department for each student-program.

Department–room matrix: $\mathbf{N}=D\times R$, $n_{ij}=1$ if room r_j belongs to department d_i, 0 otherwise; $n_{\mathbf{N}}(d_i)$ is the number of rooms at department d_i, and $n_{\mathbf{N}}(r_j)$ is the number of departments that share the same room r_j.

Professor-timeslot availability matrix: $\mathbf{P}=P\times T$, $p_{ij}=1$ if professor p_i is available in timeslot t_j, 0 otherwise; $n_{\mathbf{P}}(p_i)$ is the number of timeslots during which professor p_i is available, and $n_{\mathbf{P}}(t_j)$ is the number of available professors in t_j.

Student-program–student-group matrix: $\mathbf{Q}=M\times G$, $q_{ij}=1$ if student-group g_j is registered under student-program m_i, 0 otherwise; $n_{\mathbf{Q}}(m_i)$ is the number of student-groups that registered under student-program m_i, and $n_{\mathbf{Q}}(g_j)=1$ since each student-group is registered under only one student-program.

Student-group–student matrix: $\mathbf{R}=G\times S$, $r_{ij}=1$ if student s_j is allocated to student-group g_i, 0 otherwise; $n_{\mathbf{R}}(g_i)$ is the number of students in student-group g_i, and $n_{\mathbf{R}}(s_j)=1$ since each student must be allocated to only one student-group.

Timeslot-room availability matrix: $\mathbf{S}=T\times R$, $s_{ij}=1$ if room r_j is available in timeslot t_i, 0 otherwise; $n_S(t_i)$ is the number of available rooms in timeslot t_i, and $n_S(r_j)$ is the number of timeslots during which room r_j is available.

Student-conflict matrix: $\mathbf{C} = E\times E$, c_{ij} is the number of students taking both event e_i and event e_j.

We also have the length (in hours) of each event e_i, $n_l(e_i)$, the weekly frequency of each event e_i, $n_f(e_i)$ (for most events $n_f(e_i)=1$), the maximum number of courses for each professor p_i, $n_{maxc}(p_i)$, the lower and upper limits (in hours) of weekly load for each professor p_i, $n_{min}(p_i)$ and $n_{max}(p_i)$, the maximum weekly excess load for all professors, $n_{maxl}(p)$, the maximum number of events per day for each professor, $n_{maxe}(p_i)$, the maximum daily load (in hours) for each professor p_i, $n_{maxd}(p_i)$, the maximum weekly load (in hours) for all students, $n_{maxw}(s)$, the maximum daily load (in hours) for all students, $n_{maxd}(s)$, and the room-capacity of students for each room, $n_c(r_i)$.

4.1.2 *Output Matrices*

The following (*six*) output matrices will form a complete university course timetable.

Course–professor assignment matrix: $\mathbf{T}=C\times P$, $t_{ij}=1$ if course c_i is assigned to professor p_j, 0 otherwise; $n_T(c_i)$ is the number of professors that are assigned to course c_i, and $n_T(p_j)$ is the number of courses assigned to professor p_j.

Event–professor assignment matrix: $\mathbf{U}=E\times P$, $u_{ij}=1$ if event e_i is assigned to professor p_j, 0 otherwise; $n_U(e_i)$ is the number of professors that are assigned to event e_i, and $n_U(p_j)$ is the number of events assigned to professor p_j.

Event–timeslot assignment matrix: $\mathbf{V}=E\times T$, $v_{ij}=1$ if event e_i is assigned to timeslot t_j, 0 otherwise; $n_V(e_i)$ is the number of timeslots that are assigned to event e_i, and $n_V(t_j)$ is the number of events that are assigned to timeslot t_j.

Event-room assignment matrix: $\mathbf{W}=E\times R$, $w_{ij}=1$ if event e_i is assigned to room r_j, 0 otherwise; $n_W(e_i)$ is the number of rooms that are assigned to event e_i, and $n_W(r_j)$ is the number of events that are assigned to room r_j.

Student-group–timeslot assignment matrix: $\mathbf{X}=G\times T$, $x_{ij}=1$ if student-group g_i is assigned to timeslot t_j, 0 otherwise; $n_X(g_i)$ is the number of timeslots assigned to g_i, and $n_X(t_j)$ is the number of student-groups assigned to timeslot t_j.

Student-group–room assignment matrix: $\mathbf{Y}=G\times R$, $y_{ij}=1$ if student-group g_i is assigned to room r_j, 0 otherwise; $n_Y(g_i)$ is the number of rooms assigned to student-group g_i, and $n_Y(r_j)$ is the number of student-groups assigned to room r_j.

4.2 Hard Constraints and Mathematical Formulation

Hard constraints must be satisfied in order to produce a *feasible* timetable. Any timetable which fails to satisfy all these constraints is deemed to be *infeasible*. The hard constraints for the three subproblems (*course-professor assignment*, *event-timeslot assignment* and *event-room assignment*) will be considered separately. Each institution will apply some or all of these hard constraints. However, each institution will have some unique combination of hard constraints, as policies differ from institution to institution.

4.2.1 *Hard Constraints for Course-Professor Assignment*

The common hard constraints for 'course-professor assignment' (ordered by their importance):

H1) *Course-professor*: Each course must be assigned to at least one professor (formulated using the matrix **T**);

$\sum_{i=1}^{n_1} x_{CP}(c_i, p_j) = 0$, where $x_{CP}(c_i, p_j) = 0$ if $\sum_{j=1}^{n_4} t_{ij} \geq 1$, and 1 otherwise.

H2) *Professor-course*: Each professor must be assigned to at least one course (formulated using the matrix **T**);

$\sum_{j=1}^{n_4} x_{PC}(c_i, p_j) = 0$, where $x_{PC}(c_i, p_j) = 0$ if $\sum_{i=1}^{n_1} t_{ij} \geq 1$, and 1 otherwise.

H3) *Professor-clashing*: A professor can only attend one event at a time (formulated using the matrices **U** and **V**);

$\sum_{j=1}^{n_4} \sum_{k=1}^{n_8} x(e_i, p_j, t_k) = 0$, where $x(e_i, p_j, t_k) = 0$ if $\sum_{i=1}^{n_2} u_{ij} \cdot v_{ik} \leq 1$, and 1 otherwise.

H4) *Professor-availability*: A professor must not be assigned to events in the timeslots during which he/she is unavailable (formulated using the matrices **P**, **U** and **V**); $\sum_{j=1}^{n_4} \sum_{i=1}^{n_2} \sum_{k=1}^{n_8} (1 - p_{jk}) \cdot u_{ij} \cdot v_{ik} = 0$.

H5) *Minimum-load*: The total weekly load (in hours) of all events assigned to each professor must not less than a specified number of the minimum weekly load of each professor (formulated using the matrix **U**);

$\sum_{j=1}^{n_4} x_{\min}(e_i, p_j) = 0$, where $x_{\min}(e_i, p_j) = 0$ if $\sum_{i=1}^{n_2} u_{ij} \cdot n_l(e_i) \cdot n_f(e_i) \geq n_{\min}(p_j)$, and 1 otherwise.

H6) *Excess-load*: The number of excess load assigned to each professor must not exceed the specified number of the maximum weekly excess load (formulated using the matrix **U**);

$\sum_{j=1}^{n_4} x_{\max l}(e_i, p_j) = 0$, where $x_{\max l}(e_i, p_j) = 0$ if $\sum_{i=1}^{n_2} u_{ij} \cdot n_l(e_i) \cdot n_f(e_i) - n_{\max}(p_j) \leq n_{\max l}(p)$, 1 otherwise.

H7) *Free-timeslot*: All professors at each of the departments must have a common free period (of timeslots) once a week for the purpose of meetings. This can easily be satisfied by assigning a value '1' to each of the corresponding event-timeslots in the *event-timeslot* restriction matrix (**I**).

4.2.2 *Hard Constraints for Event-Timeslot Assignment*

The common hard constraints for 'event-timeslot assignment' (ordered by their importance):

H8) *Student-clashing*: No student should be assigned to two different events at the same time, i.e. no student conflicts (formulated using the matrices **C** and **V**); $\sum_{k=1}^{n_8} \sum_{i=1}^{n_2 - 1} \sum_{j=i+1}^{n_2} c_{ij} \cdot v_{ik} \cdot v_{jk} = 0$.

H9) *Event-frequency*: The number of (one-hour) timeslots that assigned to each event must equal to the event's weekly frequency multiply by length (formulated using the matrix **V**);

$\sum_{i=1}^{n_2} x_{EF}(e_i, t_j) = 0$, where $x_{EF}(e_i, t_j) = 0$ if $\sum_{j=1}^{n_8} v_{ij} = n_f(e_i) \cdot n_l(e_i)$, and 1 otherwise.

H10) *Event-clashing 1*: Events of two required courses of the same student-program must not be scheduled at the same timeslot (formulated using the matrices **A**, **F** and **V**);

$\sum_{k=1}^{n_5} x_{C1}(c_i,e_j,m_k)=0$, where $x_{C1}(c_i,e_j,m_k)=0$ if $\sum_{i_1=1}^{\alpha_2-1}\sum_{i_2=i_1+1}^{\alpha_2} f_{i_1k}\cdot f_{i_2k}\cdot x_{C1}(e_j,m_k)=0$, 1 otherwise,

$x_{C1}(e_j,m_k)=0$ if $\sum_{j_1=1}^{\beta_2-1}\sum_{j_2=j_1+1}^{\beta_2} a_{i_1j_1}\cdot a_{i_2j_2}\cdot x_{C1}(e_{j_1},e_{j_2})=0$, 1 otherwise, and $x_{C1}(e_{j_1},e_{j_2})=0$ if $t_{(e_{j_1})}\neq t_{(e_{j_2})}$, 1 otherwise.

H11) *Event-clashing 2*: Two events of a required course and an optional course of the same student-program must not be scheduled at the same time (formulated using the matrices **A**, **F** and **V**);

$\sum_{k=1}^{n_5} x_{C2}(c_i,e_j,m_k)=0$, where $x_{C2}(c_i,e_j,m_k)=0$ if $\sum_{i_1=1}^{\alpha_2}\sum_{i_2=\alpha_2+1}^{n_1} f_{i_1k}\cdot f_{i_2k}\cdot x_{C2}(e_j,m_k)=0$, 1 otherwise, $x_{C2}(e_j,m_k)=0$

if $\sum_{j_1=1}^{\beta_2}\sum_{j_2=\beta_2+1}^{n_2} a_{i_1j_1}\cdot a_{i_2j_2}\cdot x_{C2}(e_{j_1},e_{j_2})=0$, 1 otherwise, $x_{C2}(e_{j_1},e_{j_2})=0$ if $t_{(e_{j_1})}\neq t_{(e_{j_2})}$, 1 otherwise.

H12) *Day-event*: Events of the day student-programs must be scheduled during the day (using matrices **A**, **F** & **V**); $\sum_{l=1}^{\delta}\sum_{i=1}^{n_1} f_{il}\cdot x_{Day}(c_i,e_j,t_k)=0$, where $x_{Day}(c_i,e_j,t_k)=\sum_{j=1}^{\beta_1} a_{ij}\cdot x_{Day}(e_j,t_k)+\sum_{j=\beta_2+1}^{\beta_3} a_{ij}\cdot x_{Day}(e_j,t_k)$,

$x_{Day}(e_j,t_k)=0$ if $\sum_{k=11}^{14} v_{jk}=\sum_{k=25}^{28} v_{jk}=\sum_{k=39}^{42} v_{jk}=\sum_{k=53}^{56} v_{jk}=\sum_{k=67}^{70} v_{jk}=0$, and 1 otherwise.

H13) *Evening-event*: Events of the evening student-programs must be scheduled during the evening (using the matrices **A**, **F** and **V**); $\sum_{l=\delta+1}^{n_5}\sum_{i=1}^{n_1} f_{il}\cdot x_{Eve}(c_i,e_j,t_k)=0$, where $x_{Eve}(c_i,e_j,t_k)=\sum_{j=\beta_1+1}^{\beta_2} a_{ij}\cdot x_{Eve}(e_j,t_k)+$

$\sum_{j=\beta_3+1}^{n_2} a_{ij}\cdot x_{Eve}(e_j,t_k)$, $x_{Eve}(e_j,t_k)=0$ if $\sum_{k=1}^{10} v_{jk}=\sum_{k=15}^{24} v_{jk}=\sum_{k=29}^{38} v_{jk}=\sum_{k=43}^{52} v_{jk}=\sum_{k=57}^{66} v_{jk}=0$, 1 otherwise.

H14) *Preassigned-timeslot*: Some events must be assigned to some specific timeslots as in event-timeslot preassignment matrix (formulated using the matrices **H** and **V**); $\sum_{i=1}^{n_2}\sum_{j=1}^{n_8} h_{ij}(1-v_{ij})=0$.

H15) *Restricted-timeslot*: Some events are restricted to some specific timeslots as in event-timeslot restriction matrix (formulated using the matrices **I** and **V**); $\sum_{i=1}^{n_2}\sum_{j=1}^{n_8} i_{ij}(1-v_{ij})=0$.

H16) *Consecutive-timeslot*: The events with consecutive timeslots (with length two, three or four hours) must not be interrupted (using the matrix **V**); $\sum_{i=1}^{n_2} x_{CT}(e_i,t_j)=0$, where $x_{CT}(e_i,t_j)=0$ if $n_l(e_i)=2$ & $\sum_{j=1}^{n_8-1} v_{ij}\cdot v_{ij+1}=n_f(e_i)$,

or if $n_l(e_i)=3$ & $\sum_{j=1}^{n_8-2} v_{ij}\cdot v_{ij+1}\cdot v_{ij+2}=n_f(e_i)$, or if $n_l(e_i)=4$ & $\sum_{j=1}^{n_8-3} v_{ij}\cdot v_{ij+1}\cdot v_{ij+2}\cdot v_{ij+3}=n_f(e_i)$, 1 otherwise.

4.2.3 *Hard Constraints for Event-Room Assignment*

The common hard constraints for 'event-room assignment' (ordered by their importance):

H17) *Room-clashing*: A room can only host one event at a time (formulated using the matrices **V** and **W**);

$\sum_{k=1}^{n_9}\sum_{j=1}^{n_8} x_{RC}(e_i,t_j,r_k)=0$, where $x_{RC}(e_i,t_j,r_k)=0$ if $\sum_{i=1}^{n_2} v_{ij}\cdot w_{ik}\leq 1$, and 1 otherwise.

H18) *Room-capacity*: There must be sufficient seats in each room to house all students present (using the matrices **A**, **E** and **W**); $\sum_{k=1}^{n_9}\sum_{j=1}^{n_2} x_C(c_i,e_j,r_k)=0$, where $x_C(c_i,e_j,r_k)=0$ if $\sum_{i=1}^{n_1} a_{ij}\cdot w_{jk}\cdot n_E(c_i)\le n_c(r_k)$, and 1 otherwise.

H19) *Room-availability*: An event can only be assigned to a room (in any timeslot) if and only if the room is available (formulated using the matrices **S**, **V** and **W**); $\sum_{k=1}^{n_9}\sum_{j=1}^{n_8}\sum_{i=1}^{n_2}(1-s_{jk})\cdot v_{ij}\cdot w_{ik}=0$.

H20) *Preassigned-room*: Some events must be assigned to special rooms as in event-room preassignment matrix (formulated using the matrices **J** and **W**); $\sum_{i=1}^{n_2}\sum_{j=1}^{n_9} j_{ij}(1-w_{ij})=0$.

H21) *Restricted-room*: Some events are restricted to some specific rooms as in event-room restriction matrix (formulated using the matrices **K** and **W**); $\sum_{i=1}^{n_2}\sum_{j=1}^{n_9} k_{ij}(1-w_{ij})=0$.

H22) *Same-room*: Certain events must be held at the same time in the same room; for instance, events e_i and e_j (formulated using the matrices **V** and **W**); $\sum_{k=1}^{n_8}\sum_{l=1}^{n_9} v_{ik}\cdot v_{jk}\cdot w_{il}\cdot w_{jl}\ge 1$.

H23) *Different-room*: Certain events must be held at the same time but different rooms; e.g. events e_i and e_j (using

the matrices **V** and **W**); $\sum_{k=1}^{n_8} v_{ik}\cdot v_{jk}\cdot x_{DR}(e_i,e_j,r_l)\ge 1$, where $x_{DR}(e_i,e_j,r_l)=0$ if $\sum_{l=1}^{n_9} w_{il}\cdot w_{jl}=1$, 1 otherwise.

4.3 Soft Constraints and Mathematical Formulation

Soft constraints are those which are desirable to be satisfied, but which in general cannot all be wholly met. Soft constraints are generally more numerous and varied and are far more dependent on the needs of the individual problem than the more obvious hard constraints. The *quality* of a resulting timetable can be measured according to an objective function which weights the violation of the soft constraints. The soft constraints for the three subproblems will be considered separately. Each institution will apply some or all of these soft constraints. The exact form (and penalty functions) will be dependent on the institution. However, each institution will have some unique combination of soft constraints, as policies differ from institution to institution. Furthermore, an institution may take different views on what constitutes the quality of a course timetable.

4.3.1 *Soft Constraints for Course-Professor Assignment*

The common soft constraints for 'course-professor assignment' (ordered by their importance):

S1) *Maximum-load*: The total weekly load (in hours) for all events assigned to each professor should not exceed a specified number of the maximum weekly load (formulated using the matrix **U**);

$f_1\left(\sum_{j=1}^{n_4} x_{\max}(e_i,p_j)\right)$, where $x_{\max}(e_i,p_j)=0$ if $\sum_{i=1}^{n_2} u_{ij}\cdot n_l(e_i)\cdot n_f(e_i)\le n_{\max}(p_j)$, and $[\sum_{i=1}^{n_2} u_{ij}\cdot n_l(e_i)\cdot n_f(e_i)-$

$n_{\max}(p_j)]$ otherwise, and f_1 is a penalty function based on the total number of excess load.

S2) *Daily-load*: The total daily load (in hours) for all events assigned to each professor per day should not exceed a specified maximum for each professor (formulated using the matrices **U** and **V**); $f_2\left(\sum_{j=1}^{n_4} x_{\max\, l}(e_i, p_j, t_k)\right)$,

where $x_{maxd}(e_i,p_j,t_k)=\sum_{D=1}^{5} x_{maxd,D}(e_i,p_j)$, $x_{maxd,D}(e_i,p_j)=0$ if $\sum_{k=T_1}^{T_2}\sum_{i=1}^{n_2} u_{ij}\cdot v_{ik} \le n_{maxd}(p_j)$, 1 otherwise, T_1 and T_2

are the first and last timeslots for day D, and f_2 is based on the number of professors with excess daily load.

S3) *Free-timeslot*: A professor should have at least one free timeslot between any two teaching events, i.e. lectures - equivalent to minimizing the number of pairs of lectures in adjacent timeslots (using the matrices **U** and **V**);

$f_3\left(\sum_{j=1}^{n_4}\sum_{i_1=1}^{n_2-1}\sum_{i_2=i_1+1}^{n_2} u_{i_1 j}\cdot u_{i_2 j}\cdot x_F(e_i,t_k)\right)$, where $x_F(e_i,t_k)=1$ if $|t_{(e_{i_1})}-t_{(e_{i_2})}|=0$ & both events are lectures, 0 otherwise,

and f_3 is a function based on the total number of pairs of lectures in adjacent timeslots for each of the professors.

S4) *Event-compactness*: Professors may prefer to have events in consecutive timeslots - equivalent to minimizing the total number of free timeslots (1 or 2) between events that assigned to each professor (using the matrices **U** and **V**); $f_4\left(\sum_{j=1}^{n_4}\sum_{i_1=1}^{n_2-1}\sum_{i_2=i_1+1}^{n_2} u_{i_1 j}\cdot u_{i_2 j}\cdot x_{EC}(e_i,t_k)\right)$ where $x_{EC}(e_i,t_k)=\omega_{t-1}$ if $|t_{(e_{i_1})}-t_{(e_{i_2})}|=t$, $t\in\{2,3\}$, 0 otherwise,

the weight ω_{t-1} reflects the penalty of having 1 or 2 free timeslots, and f_4 is a function based on the total weights.

S5) *Preferred-course*: Each professor should be assigned to the courses in such a way that the total preference level is minimized (formulated using the matrices **B** and **T**);

$f_5\left(\sum_{i=1}^{n_1}\sum_{j=1}^{n_4} b_{ij}\cdot t_{ij}\right)$, where f_5 is a penalty function based on total preference levels for all professors.

S6) *Event-professor*: An event should not be assigned to a certain professor; for instance, event e_i should not be assigned to professor p_j (formulated using the matrix **U**); $f_6\left(x_{EP}(e_i,p_j)\right)$, where $x_{EP}(e_i,p_j)=0$ if $u_{ij}=0$, and 1

otherwise, and f_6 is a penalty function that reflects the violation of this constraint.

S7) *Morning-free*: The morning should be free for professors having afternoon events on the same day (formulated using the matrices **U** and **V**); $f_7\left(\sum_{j=1}^{n_4} x_{MF}(e_i,p_j)\right)$, where $x_{MF}(e_i,p_j)=\sum_{D=1}^{5} x_{MF,D}(e_i,p_j)$, $x_{MF,D}(e_i,p_j)=1$ if

$\sum_{k=Ta_1}^{Ta_2}\sum_{i=1}^{n_2} u_{ij}\cdot v_{ik}>0$ and $\sum_{k=Tm_1}^{Tm_2}\sum_{i=1}^{n_2} u_{ij}\cdot v_{ik}>0$, 0 otherwise, for each day D, Ta_1 and Ta_2 are respectively the

first and last afternoon timeslots, Tm_1 and Tm_2 are respectively the first and last morning timeslots, and f_7 is a function based on the total number of professor-days having morning and afternoon events on the same day.

4.3.2 *Soft Constraints for Event-Timeslot Assignment*

The common soft constraints for 'event-timeslot assignment' (ordered by their importance):

S8) *Daily-event*: A student should not be expected to attend more than x events in a day (using matrices **A**, **E** & **V**);

$f_8\left(\sum_{k=1}^{n_7}\sum_{D=1}^{5} x_{DL}(c_i,e_j,s_k,t_l)\right)$, where $x_{DL}(c_i,e_j,s_k,t_l)=1$ if $\sum_{i=T_1}^{T_2}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} a_{ij}\cdot e_{ik}\cdot v_{jl}>x$, 0 otherwise, and

f_8 is a penalty function based on the number of student-days having more than x events in a day.

S9) *Student-event*: A student should not have only one event in a day (formulated using the matrices **A**, **E** and **V**);

$f_9\left(\sum_{k=1}^{n_7}\sum_{D=1}^{5} x_{SE}(c_i,e_j,s_k,t_l)\right)$, where $x_{SE}(c_i,e_j,s_k,t_l)=1$ if $\sum_{i=T_1}^{T_2}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} a_{ij}\cdot e_{ik}\cdot v_{jl}=1$, 0 otherwise, and f_9 is a

penalty function based on the number of student-days having only one event in a day.

S10) *Student-timeslot*: Students prefer to have their events in consecutive timeslots – equivalent to minimizing the number of free timeslots (1 or 2) between two timeslots assigned to each student-group (using the matrix **X**);

$f_{10}\left(\sum_{i=1}^{n_6} x_{ST}(g_i,t_j,t_k)\right)$, where $x_{ST}(g_i,t_j,t_k)=0$ if $\sum_{j=1}^{n_8-1}\sum_{k=j+1}^{n_8} x_{ij}\cdot x_{ik}\cdot x_{ST}(t_j,t_k)=0$, 1 otherwise, $x_{ST}(t_j,t_k)=\omega_{t-1}$

if $|t_j-t_k|=t$, $t\in\{2,3\}$, 0 otherwise, the weight ω_{t-1} reflects the penalty of having 1 or 2 free timeslots, and f_{10} is a penalty function based on the total weights.

S11) *Event-clashing*: Two events of two optional courses of the same student-program should not be scheduled at the same time (formulated using the matrices **A**, **F** and **V**); $f_{11}\left(\sum_{k=1}^{n_5} x_{EC2}(c_i,e_j,m_k)\right)$, where $x_{EC2}(c_i,e_j,m_k)=0$

if $\sum_{i_1=\alpha_2+1}^{n_1-1}\sum_{i_2=i_1+1}^{n_1} f_{i_1k}\cdot f_{i_2k}\cdot x_{EC2}(e_j,m_k)=0$, 1 otherwise, $x_{EC2}(e_j,m_k)=0$ if $\sum_{j_1=\beta_2+1}^{n_2-1}\sum_{j_2=j_1+1}^{n_2} a_{i_1j_1}\cdot a_{i_2j_2}\cdot x_{EC2}(e_{j_1},e_{j_2})=0$,

1 otherwise, $x_{2R}(e_{j_1},e_{j_2})=0$ if $t_{(e_{j_1})}\neq t_{(e_{j_2})}$, 1 otherwise, and f_{11} is a penalty function.

S12) *Same-course*: Events of the same course and student-group should not be scheduled on the same day (formulated using the matrices **A**, **F**, **Q** and **V**); $f_{12}\left(\sum_{l=1}^{n_5}\sum_{m=1}^{n_6}\sum_{i=1}^{n_1} f_{il}\cdot q_{lm}\cdot x_{SC}(c_i,e_j,t_k)\right)$, where $x_{SC}(c_i,e_j,t_k)=1$

if $\sum_{D=1}^{5} x_{SC}(e_j,t_k)>1$, 0 otherwise, $x_{SC}(e_j,t_k)=1$ if $\sum_{k=T_1}^{T_2}\sum_{j=1}^{n_2} a_{ij}\cdot v_{jk}>1$, 0 otherwise, and f_{12} is a penalty function.

S13) *Common-event 1*: Two or more student-groups with the same event and student-program should be scheduled at the same timeslot (using matrices **A**, **F**, **Q**, **V**, and **X**); $f_{13}\left(\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_8}\sum_{l=1}^{n_5} a_{ij}\cdot f_{il}\cdot v_{jk}\cdot x_{CE1}(m_l,g_m,t_k)\right)$,

where $x_{CE1}(m_l,g_m,t_k)=1$ if $n_Q(m_l)>1$ and $\sum_{m=1}^{n_6} q_{lm}\cdot x_{mk}<n_Q(m_l)$, 0 otherwise, and f_{13} is a penalty function.

S14) *Common-event 2*: Two student-groups with the same event but different student-programs should not be scheduled at the same timeslot (using matrices **A**, **F**, **Q**, **V** & **X**); $f_{14}\left(\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_8} a_{ij}\cdot v_{jk}\cdot x_{CE2}(c_i,m_l,g_m,t_k)\right)$,

where $x_{CE2}(c_l, m_l, g_m, t_k) = 0$ if $\sum_{l_1=1}^{n_5-1} \sum_{l_2=l_1+1}^{n_5} f_{il_1} \cdot f_{il_2} \cdot x_{CE2}(m_l, g_m, t_k) = 0$, 1 otherwise, $x_{CE2}(m_l, g_m, t_k) = 0$ if

$\sum_{m_1=1}^{n_6-1} \sum_{m_2=m_1+1}^{n_6} q_{l_1 m_1} \cdot q_{l_2 m_2} \cdot x_{m_1 k} \cdot x_{m_2 k} = 0$, 1 otherwise, and f_{14} is a penalty function.

S15) *Tutorial-lab*: All tutorials or lab sessions of any course should occur later in the week than the week's first lecture on that course (using the matrices **A** and **V**); $f_{15}\left(\sum_{i=1}^{n_1} \sum_{j_1=1}^{n_2-1} \sum_{j_2=j_1+1}^{n_2} a_{ij_1} \cdot a_{ij_2} \cdot x_{TL}(e_{j_1}, e_{j_2}, t_k)\right)$, where

$x_{TL}(e_{j_1}, e_{j_2}, t_k) = 0$ if $t_{(e_{j1})} < t_{(e_{j2})}$ and e_{j_1} is the first lecture, or $t_{(e_{j1})} > t_{(e_{j2})}$ and e_{j_2} is the first lecture, the events are of the same course, 0 otherwise, and f_{15} is a penalty function.

S16) *Frequent-event*: If an event is held more than once a week, there should be at least one day in between (formulated using the matrix **V**); $f_{16}\left(\sum_{i=1}^{n_2} x_{FE}(e_i, t_j, t_k)\right)$, where $x_{FE}(e_i, t_j, t_k) = 0$ if $n_f(e_i) < 2$, otherwise $x_{FE}(e_i, t_j, t_k)$

$= \sum_{j=1}^{n_8 - n_f(e_i)} \sum_{k=j+n_f(e_i)}^{n_8} v_{ij} \cdot v_{ik} \cdot x_{FE}(t_j, t_k)$, $x_{FE}(t_j, t_k) = 1/n_f(e_i)$ if $|t_j - t_k| < 28$, 0 otherwise, and f_{16} is a penalty function.

S17) *Friday-afternoon*: Friday afternoon shall be used only after all other scheduling possibilities are exhausted (formulated using the matrix **V**); $f_{17}\left(\sum_{i=1}^{n_2} \sum_{j=64}^{66} v_{ij}\right)$, where f_{17} is a penalty function based on the number of

events that are assigned to timeslots on Friday afternoon.

S18) *Lunch-time*: Events should not be assigned during lunch time (formulated using the matrix **V**); $f_{18}\left(\sum_{i=1}^{n_2} \sum_{j=lunch(D=1)}^{lunch(D=5)} v_{ij}\right)$, where f_{18} is based on the number of events that are assigned during the lunch time.

S19) *Free-timeslot*: Certain student-groups may require a common free timeslot once a week for their project works; for instance, the student-groups g_r and g_s require a common free timeslot t_l (formulated using the matrix **X**);

$f_{19}(x(g_r, g_s))$, where $x(g_r, g_s) = 0$ if $x_{rl} = x_{sl} = 0$, 1 otherwise, and f_{19} reflects the violation of this constraint.

4.3.3 *Soft Constraints for Event-Room Assignment*

The common soft constraints for 'event-room assignment' (ordered by their importance):

S20) *Room-utilization*: Events are assigned to rooms in such a way that the room utilization can be maximized, or *spare seats* in each room are minimized (using matrices **A**, **E** & **W**); $f_{20}\left(\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_9} a_{ij} \cdot w_{jk} \cdot x_{RU}(e_j, r_k)\right)$,

where $x_{RU}(e_j, r_k) = [n_c(r_k) - n_s(e_j)]$ if $n_c(r_k) > n_s(e_j)$, 0 otherwise, $n_s(e_j) = n_E(c_i)$, and f_{20} is based on the total spare seats.

S21) *Room-professor*: Events should be scheduled in the rooms as close as possible to their professors (using the matrices **L**, **N**, **U** and **W**); $f_{21}\left(\sum_{j=1}^{n_4} \sum_{i=1}^{n_2} \sum_{l=1}^{n_3} \sum_{k=1}^{n_9} u_{ij} \cdot l_{lj} \cdot w_{ik} \cdot (1 - n_{lk})\right)$, where f_{21} is a penalty function.

S22) *Room-department*: Events should be scheduled in the rooms at their departments (formulated using the matrices **A**, **F**, **M**, **N** and **W**); $f_{22}\left(\sum_{m=1}^{n_3}\sum_{l=1}^{n_5}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_9} m_{ml}\cdot f_{il}\cdot a_{ij}\cdot w_{jk}\cdot(1-n_{mk})\right)$, where f_{22} is a penalty function based on the number of events that are scheduled at different departments.

S23) *Room-type*: Events should be scheduled into rooms of the type specified by the department or professor (formulated using the matrix **W**); $f_{23}\left(\sum_{i=1}^{n_2}\sum_{j=1}^{n_9} w_{ij}\cdot x_{RT}(e_i,r_j)\right)$, where $x_{RT}(e_i,r_j)=0$ if the type of room assigned to event e_i is equal to its preferred room-type, 1 otherwise, and f_{23} is a penalty function.

S24) *Same-group*: All events of the same student-group should be scheduled in the same room, or a student-group should be scheduled to only one room (using the matrix **Y**); $f_{24}\left(\sum_{i=1}^{n_6} x_{SG}(g_i,r_j)\right)$, where $x_{SG}(g_i,r_j)=0$ if $\sum_{j=1}^{n_9} y_{ij}=1$, 1 otherwise, and f_{24} is based on the number student-groups that assigned to more than one room.

S25) *First-timeslot*: There must be a limit on the total number of rooms assigned to the first-timeslot events (early morning) on each day, e.g. to avoid traffic-jam; for instance, not more than x% of the available rooms must be assigned to the first-timeslot events each day (formulated using the matrices **V** and **W**); $f_{25}\left(\sum_{j\in T} x_{FT}(e_i,t_j,r_k)\right)$, where $x_{FT}(e_i,t_j,r_k)=0$ if $\sum_{i=1}^{n_2}\sum_{k=1}^{n_9} v_{ij}\cdot w_{ik}\le(n_9\cdot x/100)$, otherwise $x_{FT}(e_i,t_j,r_k)=\sum_{i=1}^{n_2}\sum_{k=1}^{n_9} v_{ij}\cdot w_{ik}-(n_9\cdot x/100)$, $T=\{t_1,t_{15},t_{29},t_{43},t_{57}\}$, and f_{25} is a penalty function based on the number of assigned rooms that exceed $(n_9\cdot x/100)$.

S26) *Reserved-room*: Reserved rooms should not be used unless there is no other room available (using the matrix **W**); $f_{26}\left(\sum_{i=1}^{n_2}\sum_{j=r}^{n_9} w_{ij}\right)$, where r_k, $k\in\{r,\ldots,n_9\}$, is the set of reserved rooms, and f_{26} is a penalty function.

S27) *Preferred-room*: Preferred rooms for each student-group should be used first, as this minimizes the need for students to change rooms between events (formulated using the matrices **A**, **F**, **Q** and **W**); $f_{27}\left(\sum_{k=1}^{n_5}\sum_{l=1}^{n_6}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{m=1}^{n_9} q_{kl}\cdot f_{ik}\cdot a_{ij}\cdot w_{jm}\cdot x_{PR}(e_j,r_m)\right)$, where $x_{PR}(e_j,r_m)=0$ if the room assigned to event e_j is one of the preferred rooms of student-group g_l, 1 otherwise, and f_{27} is a penalty function.

S28) *Travel-time*: Sufficient time should be provided for students to move from one room to another. The simplest solution is all events should finish 15 minutes earlier. The distance traveled will be minimized if the soft constraints (S22), (S24) and (S27) are minimized.

4.4 Course Timetabling Model

Quality measures of a course timetable are derived from the soft constraints, most frequently from the professor and student restrictions. If several different quality measures are used simultaneously, the objective function is a linear combination of these measures, with relative penalty functions that reflect their perceived importance. If all of these measures are used, soft constraints (S1) to (S27), the objective function becomes extremely long and impossible to evaluate. Similarly, if all of the hard constraints (H1) to (H23) are used simultaneously, a feasible timetable would never be produced. Since the UCTP has been considered as three separate subproblems, each will be separately modeled as a *0-1 integer programming*.

4.4.1 *'Course-Professor Assignment' Model*

The simplest model of 'course-professor assignment' may consist of the hard constraints *course-professor* (H1), *professor-clashing* (H3) and *minimum load* (H5), and the soft constraints *maximum-load* (S1) and *preferred-course* (S5). The (0-1 integer programming) 'course-professor assignment' model will be

$$\text{minimize} f_1\left(\sum_{j=1}^{n_4} x_{\max}(e_i, p_j)\right) + f_5\left(\sum_{i=1}^{n_1}\sum_{j=1}^{n_4} b_{ij} \cdot t_{ij}\right) \qquad [(S1)+(S5)]$$

$$\text{subject to} \quad \sum_{i=1}^{n_1} x_{CP}(c_i, p_j) = 0, \qquad (H1)$$

$$\sum_{j=1}^{n_4}\sum_{k=1}^{n_8} x(e_i, p_j, t_k) = 0, \qquad (H3)$$

$$\sum_{j=1}^{n_4} x_{\min}(e_i, p_j) = 0, \qquad (H5)$$

all variables are integers 0-1.

4.4.2 *'Event-Timeslot Assignment' Model*

The simplest model of 'event-timeslot assignment' may consist of the hard constraints *student-clashing* (H8), *event-frequency* (H9) and *event-clashing 1* (H10), and the soft constraints *daily-event* (S8) and *event-clashing* (S11). The (0-1 integer programming) 'event-timeslot assignment' model would be

$$\text{minimize} f_8\left(\sum_{k=1}^{n_7}\sum_{D=1}^{5} x_{DL}(c_i, e_j, s_k, t_l)\right) + f_{11}\left(\sum_{k=1}^{n_5} x_{EC2}(c_i, e_j, m_k)\right) \qquad [(S8)+(S11)]$$

$$\text{subject to} \quad \sum_{k=1}^{n_8}\sum_{i=1}^{n_2-1}\sum_{j=i+1}^{n_2} c_{ij} \cdot v_{ik} \cdot v_{jk} = 0, \qquad (H8)$$

$$\sum_{i=1}^{n_2} x_{EF}(e_i, t_j) = 0, \qquad (H9)$$

$$\sum_{k=1}^{n_5} x_{C1}(c_i, e_j, m_k) = 0, \qquad (H10)$$

all variables are integers 0-1.

4.4.3 *'Event-Room Assignment' Model*

The simplest model of 'event-room assignment' may consist of the hard constraints *room-clashing* (H17), *room-capacity* (H18), and *room-availability* (H19), and the soft constraints *room-utilization* (S21) and *first-timeslot* (S25) and. The (0-1 integer programming) 'event-room assignment' model would be

$$\text{minimize} f_{20}\left(\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_9} a_{ij} \cdot w_{jk} \cdot x_{RU}(e_j, r_k)\right) + f_{25}\left(\sum_{j \in T} x_{FT}(e_i, t_j, r_k)\right) \qquad [(S21)+(S25)]$$

$$\text{subject to} \quad \sum_{k=1}^{n_9}\sum_{j=1}^{n_8} x_{RC}(e_i, t_j, r_k) = 0, \qquad (H17)$$

$$\sum_{k=1}^{n_9}\sum_{j=1}^{n_2} x_C(c_i, e_j, r_k) = 0\,, \qquad \text{(H18)}$$

$$\sum_{k=1}^{n_9}\sum_{j=1}^{n_8}\sum_{i=1}^{n_2}(1 - s_{jk})\cdot v_{ij}\cdot w_{ik} = 0\,, \qquad \text{(H19)}$$

all variables are integers 0-1.

5. Discussion and Future Work

The GUCTM, presented in Section 4, would benefit those who are involved in course timetabling. Since different institutions have different needs, requirements and goals, this model shall be used as a guideline to develop an appropriate model for a given CTP; not some rules that every institution must follow. Many of the presented hard constraints and soft constraints will conflict either directly or indirectly. For many institutions, due to the difficulty of the problem, many are ignored altogether.

The general model has considered 23 hard constraints and 28 soft constraints. These are the common constraints used for solving CTPs in the literature. However, each of the hard constraints may be transformed into a soft constraint by adding an appropriate penalty function, or vice-versa, and there remain a huge number of other constraints depending on institutions. When solving a CTP, it is advisable to start with a few number of hard constraints (2 or 3), then other hard constraints may be added one at a time provided the timetable remains feasible. If any hard constraint caused an infeasibility, it may be transformed into a soft constraint. A similar approach should be used for the soft constraints; start with 2 or 3 soft constraints in the objective function, and then other soft constraints may be added one at a time until no further evaluation can be made. The more constraints used to solve a CTP, the more complex it would be. The number of hard constraints used in a CTP is usually more than the soft constraints. The formulation of the constraints as 0-1 integer programming would be much easier if more input matrices have been constructed, but more input matrices require more times and work. Formulating the constraints as 0-1 integer programming doesn't mean that we are going to use this approach. The objective is to show that the CTPs, however complex, are solvable.

The course timetabling model for a particular institution needs to be updated from time to time. As the number of students increased, the needs and requirements will also change and increase. Some hard constraints may need to be considered as soft constraints. As the problems changed, and new problems arise, more types of hard constraints and soft constraints should be added to the model so that it remains viable for solving the CTP for that institution.

For future work, the GUCTM will be used to develop a unified model for university timetabling problems. This unified model may be used to represent and solve all types of university timetabling problems, especially course and examination.

6. References

[1] Alkan, A., and Özcan, E. (2003) Memetic Algorithms for Timetabling. *Proceedings of the Congress on Evolutionary Computation 2003 (CEC'2003)*, Piscataway, New Jersey, Dec 2003, Volume 3, pp: 1796-1802.

[2] Avella, P., and Vasil'ev, I. (2003) Computational Study of a cutting plane algorithm for University Course Timetabling. *Technical Report 20-03*, DIS Universita di Roma "La Sapienza".

[3] Burke, E.K., MacCarthy, B., Petrovic, S., and Qu, R. (2001) Case-Based Reasoning in Course Timetabling: An Attribute Graph Approach. *Proceedings of the 4th. International Conference on Case-Based Reasoning* (ICCBR-2001), Vancouver, Canada, 30 July - 2 August 2001, pp: 90-104.

[4] Burke, E.K., Bykov, Y., Newall, J., and Petrovic, S. (2003) A Time-Predefined Approach to Course Timetabling. *Research paper*, Automated Scheduling, Optimization and Planning Group, School of Computer Science & IT, University of Nottingham, UK.

[5] Carter, M.W. and Laporte, G. (1998) Recent Developments in Practical Course Timetabling. *Proceedings of the International Conference on Practice and Theory of Automated Timetabling* (PATAT'97), pp: 3-19.

[6] Corne, D., Ross, P., and Fang, H.L. (1994) Fast Practical Evolutionary Timetabling. *Lecture Notes in Computer Science, Vol. 865, Evolutionary Computing AISB Workshop*; Leeds, UK, Springer-Verlag, pp: 251-263.

[7] Erben, W., and Keppler, J. (1995) A genetic algorithm solving a weekly course-timetabling problem. *Proc. of the 1st Inter. Conference on the Practice & Theory of Automated Timetabling* (ICPTAT '95), pp: 21-32.

[8] Loo, E.H., Goh, T.N., and Ong, H.L. (1986) A Heuristic Approach to Scheduling University Timetables. *Journal of Computers and Education*, Volume 10, pp: 379-388.

[9] Ozdemir, M.S., and Gasimov, R.N. (2003) The analytic hierarchy process and multiobjective 0-1 faculty course assignment. *European Journal of Operational Research 2003*, Elsevier B.V.

[10] Ross, P., Corne, D., and Fang, H.L. (1994) Successful Lecture Timetabling with Evolutionary Algorithms. *Applied Genetic and other Evolutionary Algorithms: Proceedings of the ECAI'94 Workshop*.