The multiphase course timetabling problem

Rasul Esmaeilbeigi*[1], Vicky Mak-Hau[1], John Yearwood[1], and Vivian Nguyen[2]

[1]*School of Information Technology, Deakin University, Geelong, VIC 3218, Australia*
[2]*Defence Science and Technology Group, Australian Defence, Fishermans Bend, VIC 3207, Australia*

Abstract

This paper introduces the multiphase course timetabling problem and presents mathematical formulations and effective solution algorithms to solve it in a real case study. Consider a pool of lessons and a number of students who are required to take a subset of these lessons to graduate. Each lesson consists of a predetermined and consecutive sequence of phases with possibly different resource requirements and duration. Students who take a lesson must be present in certain phases of the lesson. Similarly, resources that are allocated to the lesson must be present in certain phases of the lesson depending on their roles. The objective is to maximize the weighted sum of student-lesson allocations subject to some capacity, availability and prerequisite constraints. The problem is formulated as an integer linear program called the session-index formulation. The formulation is then extended by introducing various side constraints that capture business requirements of a pilot training program. An enhanced branch-and-check algorithm is proposed to solve the problem more efficiently. Large instances of the problem are solved by utilizing an effective fix-and-optimize matheuristic. Results of a computational study on a set of real-world instances of the problem demonstrate the efficacy of the proposed exact and matheuristic algorithms. Due to the novelty and complexity of the problem, the efficacy of the proposed methodology to solve its real-world instances, and the implementation and realization of these contributions within a decision support system, this work was recognized as a semi-finalist of INFORMS Franz Edelman Award (2021).

Keywords— Timetabling, Multiphase; Course Scheduling; Matheuristic; Logic-Based Benders Decomposition

1 Introduction

Timetabling problems constitute a large class of combinatorial optimization problems (Wren, 1995; Burke and Petrovic, 2002; Ernst et al., 2004b,a). The educational timetabling problems such as university course timetabling, high school timetabling, examination timetabling and student sectioning are widely studied in the literature as reviewed in the recent surveys by Kristiansen and Stidsen (2013); Pillay (2014); Babaei et al. (2015); Bettinelli et al. (2015); Pillay (2016); Bashab et al. (2020) and Tan et al. (2021). These problems are typically challenging and theoretically NP-hard in almost all variants (Lewis, 2008).

Student sectioning can be seen as the simplest form of educational timetabling problems. In this problem, a course has been split into one or more copies of the course called sections or classes. Each section/class has its own time and resources (room, teacher, etc). The problem is to assign students to sections in order to respect student's preferences while satisfying some hard and soft constraints. Dostert et al. (2016) provide a complexity analysis and an algorithmic approach to student sectioning problems. The authors present a polynomial time algorithm for a special case of the problem in which student-to-section allocations are performed while ensuring that individual capacities of all sections are not exceeded and no student has more than one appointment per time slot. The authors also prove that the problem becomes NP-complete under any single of the following constraints: 1) Students can select their courses from a larger set of courses offered, and this selection must be respected; 2) Students have individual time slot restrictions due to their unavailability; and 3) Sections may have multiple events, that is, the same section is assigned to more than one time slot in the input timetable, and the assignment must be conflict

*E-mail address: *r.esmaeilbeigi@deakin.edu.au*

free with respect to time slot clashes between different pairs of section events. This complexity analysis demonstrates the high combinatorial nature of the educational timetabling problems in practice.

In the high school timetabling problem, students are grouped into classes prior to the timetabling problem. In other words, time and resource allocations are performed for each class (group of students) during the optimization as opposed to the student sectioning problem. The high school timetabling problem seeks to construct a feasible schedule subject to some hard and soft constraints. A list of 22 (hard and soft) constraints for the problem can be found in the recent survey of the problem by Tan et al. (2021). The university course timetabling and the examination timetabling problems are similar to the high school timetabling problem in the sense that they generally have similar constraints and objectives. A possible difference is that the student clustering may be considered as part of the university course timetabling while it is a separate step for the high school timetabling problem. A detailed comparison of the educational timetabling problems can be found in Kristiansen and Stidsen (2013).

In the present paper, we introduce the Multiphase Course Timetabling Problem (MCTP). This is an educational timetabling problem which is motivated by real-world scheduling challenges encountered in a pilot training program. The educational timetabling literature employs a variety of terminology that could mean different concepts in different publications. For clarity of exposition, we begin by defining the terminology used to describe the MCTP at the outset. We shall use the term 'cohort' to refer to the set of students who are enrolled in the program at the same time. We also use the terms 'course' or 'lesson' interchangeably to refer to a specific subject or skill that needs to be learned by a student. The term 'syllabus' is used to denote a set of lessons that must be taken by the relevant students to graduate. The alternative term 'curriculum' is not used in the pilot training program under study. A training session, or simply a 'session', is a scheduled lesson. A lesson can have multiple sessions, implying that the same lesson is scheduled multiple times in a given planning horizon. Furthermore, multiple sessions of a lesson can be scheduled at the same time. Note that the literature sometimes employs the term 'section' or 'class' to denote scheduled or unscheduled groups of students and/or resources. These terms are not used in the present paper.

The MCTP can be summarized as follows. Consider a pool of lessons and a number of students who need to take a subset of these lessons to graduate. Each lesson consists of a predetermined and consecutive sequence of phases with possibly different resource requirements and duration. Lesson phases are consecutive since they naturally form a lesson together. A flying lesson, for instance, has preparation, brief, flying event and debrief phases that must immediately follow each other. Students who take a lesson must be present in certain phases of the lesson. In other words, these students may be absent from some phases of the lesson. Similarly, resources that are allocated to the lesson must be present in certain phases of the lesson depending on their roles. Scheduling of lessons for different cohorts of students is performed over a rolling horizon. Each time the planning begins, only lessons that have not previously been scheduled for a student are considered for that student (including lessons that the student has taken, but not passed). The MCTP seeks to maximize the weighted sum of student-lesson allocations in a given planning horizon subject to some capacity, availability and prerequisite constraints. This objective function ensures graduation of all students in time by considering their priorities. Without this consideration, some students may never graduate since new cohorts of students are constantly enrolled in the program. Students are expected to graduate in nearly one year. The priorities of students are updated based on their expected progress, which depends on how long they have been training in the program.

A defining characteristic of the MCTP is the existence of predefined and consecutive phases for a lesson with possibly different resource requirements for different phases of the same lesson. Furthermore, the MCTP explicitly considers the precedence relations between the different lessons taken by individual students as part of the optimization problem. To our knowledge, the MCTP is a new educational timetabling problem which is first introduced in the present paper. The educational timetabling literature also includes the notion of subevents for an event/lesson which is considered in the so-called Generalized High School Timetabling Problem (GHSTP). The GHSTP which is studied by Kristiansen et al. (2015) and Fonseca et al. (2017) is fundamentally different from the MCTP.

In the MCTP, the sequence and duration of the phases for a lesson are predetermined and these phases must be scheduled consecutively. However, in the GHSTP, an event is partitioned into a number of subevents as part of the optimization problem. Therefore, the sequence and duration of these subevents are not predetermined. Furthermore, the duration of all active (or selected) subevents in a feasible solution of the GHSTP must be equal to the length of the event. In the MCTP, the duration of all active phases of a lesson in a feasible solution depends on the number of sessions of that lesson in the solution which is a decision variable. This is due to the fact that different sessions of a lesson in the MCTP must have exactly the same sequence of phases scheduled consecutively.

Another difference between the GHSTP and the MCTP is in the way students and resources are

allocated to an event. In the MCTP, each phase of a lesson has its own resource requirements while in the GHSTP, each subevent has exactly the same resource requirements as the event. Furthermore, the GHSTP does not consider constraints related to individual students. For instance, in the MCTP students who take a lesson are not required to attend every phase of the lesson. The existence of the precedence relations between the different lessons taken by individual students is another aspect of the MCTP which is not considered in the GHSTP.

The MCTP may also be compared to the course timetabling problem in ITC2019 (`www.itc2019.org`). That problem also schedules lessons with complex structures while considering assignment of times, rooms, and students to each event of a lesson. Lessons can have one or more configurations, subparts for each configuration, and events for each subpart. A student must attend one event from each subpart of a single configuration (*i.e.,* the configuration that he/she is allocated to). There is also a parent-child relation between events of different subparts of a configuration that are used to link students of particular pairs of a laboratory and a seminar together, so that the same instructor is teaching them. The existence of different configurations for a lesson means that a student can be optionally allocated to one of the configurations of the lesson, but he/she must then attend one event of every subpart of the allocated configuration.

The ITC2019 problem does not consider multiple consecutive phases with different resource requirements as in the MCTP. The problem also has many soft constraints, as opposed to the MCTP which does not employ any soft constraints. In fact, maximizing the total throughput of students while considering their priorities is the most important objective as prescribed by the pilot training experts. This is also considered by the existing greedy algorithm (called the auto planner), which is designed and trusted by the technical experts to solve the problem in practice. In the ITC2019 problem, each student must be allocated to all lessons he/she has requested, but there may be a conflict when the student is allocated to two overlapping events. Minimizing such conflicts is one of the soft constraints in the problem. In the MCTP, however, student-lesson allocations are decision variables whose weighted sum is maximized (no conflict is allowed). Another soft constraint of the ITC2019 problem is satisfying the precedence constraints. For lessons that have multiple events in a week or in different weeks, such constraints only consider the first event of the lesson. The MCTP considers precedence constraints for every student and every phase of the sessions that the student attends. The reader is referred to Müller et al. (2018) for more details about the ITC2019 problem.

In this paper, we formally define the MCTP and formulate the problem as an integer linear program. This formulation is called the session-index formulation since it employs a collection of session-index decision variables. The session-index formulation is then extended to capture various business requirements of the pilot training program under study. These requirements include turnaround time for some physical resources such as aircraft and simulators as well as the incorporation of three types of lessons referred to as one-time-prepared lessons, shared lessons and double bubbles.

The session-index formulation suffers from the symmetry in the numbering of the sessions for a lesson. Our computational study using a commercial linear solver indicates that real instances of this formulation cannot be solved in a reasonable time even in the presence of symmetry breaking constraints. Therefore, we present a relaxed formulation which is solved by using a logic-based Benders decomposition approach (Hooker and Ottosson, 2003). In our implementation, Benders cuts are generated within a Branch and Bound (B&B) algorithm, that is, by using a single search tree. This strategy is often referred to as Branch and Check (B&CH) in the literature (Thorsteinsson, 2001; Beck, 2010; Rahmaniani et al., 2017).

The review of the literature indicates that B&CH is the state-of-the-art exact method for solving a variety of combinatorial optimization problems. In fact, even the basic version of the algorithm (*i.e.,* the sequential variant that solves Benders subproblems at an optimal solution of the master problem rather than its feasible solutions) is the leading exact method in some cases. For instance, Zohali et al. (2021) make significant improvements in solving large instances of a class of assembly line balancing and scheduling problems by employing this approach. Their methodology outperforms a commercial solver that solves a superior formulation of the problem from the literature (Esmaeilbeigi et al., 2016b). There are also cases where B&CH outperforms state-of-the-art branch and price algorithms. An example is the integrated operating room planning and scheduling problem studied by Roshanaei and Naderi (2021). The authors show that their B&CH algorithm achieves an average optimality gap that is nearly 3 times smaller than the gap obtained from an existing Branch-Price-and-Cut (BP&C) algorithm in the literature.

The leading class of exact algorithms for solving many vehicle routing problems has long known to be BP&C algorithms (Costa et al., 2019). We found that, in fact, B&CH is the cornerstone of the state-of-the-art exact algorithms for solving vehicle routing problems with split deliveries, as proposed in Archetti et al. (2014) and Bianchessi and Irnich (2019). Although the authors do not refer to their algorithms as B&CH (or logic-based Benders decomposition in general), their methodology is indeed B&CH. Both studies rely on a master problem that utilizes the general integer variable ϑ_e for edge e to count the

number of times that the edge is visited by any vehicles (vehicle-index variables are not used to avoid symmetry). The Benders feasibility cut used in both studies has the form $\sum_{e\in\mathcal{E}} \vartheta_e \geq 1$, in which $\mathcal{E}$ corresponds to the set of edges that setting their ϑ variables to zero (all at the same time) will certainly result in an infeasible solution. Results of both studies prove optimality for several previously unsolved instances from the literature. The reader is referred to the papers for more details.

Superiority of logic-based Benders decomposition has also been demonstrated in stochastic settings. In a recent preprint, Elçi and Hooker (2020) show that B&CH can solve two-stage stochastic planning and scheduling problems with second-stage scheduling subproblems several orders of magnitude faster than the integer L-shaped method. Evidently, B&CH algorithms are receiving more and more attention due to their significant potential in solving various combinatorial optimization problems. Although we could not find examples of logic-based Benders decomposition in the educational timetabling literature, we believe that some interesting ideas already exist in this literature that could be utilized for designing such algorithms. We will elaborate on such ideas when concluding the paper. Each Benders subproblem in our study is the decision version of a bin packing problem (Coffman Jr et al., 1996). We accelerate the proposed B&CH algorithm by developing a fast matheuristic that provides an initial feasible solution, by presenting a pre-processing scheme which eliminates a significant number of decision variables, and by presenting a compact representation of the precedence constraints. The proposed matheuristic is a fix-and-optimize algorithm that solves a sequence of relaxed formulations.

We show that all improvements together halve the run time of the algorithm on a set of real instances. The matheuristic algorithm provides high quality solutions with only 4.5% optimality gap for these instances, on average. Overall, the enhanced B&CH algorithm achieves an average optimality gap of 0.01%, whereas the session-index formulation obtains an average optimality gap of 48.62%, while spending 21 times more computational time to solve these instances. A more than twofold increase is also achieved in the number of instances proved to optimality (*i.e.,* from 43 to 92). While the enhanced algorithm provides optimal solutions for 98.9% of the instances, the auto planner (the existing algorithm) could provide optimal solutions for only 3.2% of them. The aforementioned results are based on the daily instances of the problem for which a good upper bound can be obtained quickly. We also study a set of weekly instances which cannot be solved to optimality within a reasonable amount of time. We propose to solve such instances by decomposing them into a number of daily instances that are solved by the enhanced B&CH algorithm. The solutions provided by this approach have 22.3% better objective values, on average, compared to the solutions obtained from the auto planner. We show that this improvement is increased to 41.4% in a pessimistic scenario with restricted resources.

We highlight that the present paper encompasses numerous real-world assumptions, all of which have been prescribed and fully asserted by the relevant experts of the pilot training program under study. All mathematical formulations, algorithms, and improvements introduced in the paper have been implemented and validated in a decision support system called ARES. Specifically, the comparisons and the numerical results provided in the computational section of the paper have all been obtained by using ARES. These results are not based on a separate implementation of our algorithms. Solving the real-world problem involved many details such as processing and cleaning the data obtained from a database and post-processing of our solutions to be interpreted by a user interface. Such details are not discussed in the paper. Due to the novelty and complexity of the problem, the efficacy of the proposed methodology to solve its real-world instances, and the implementation and realization of these contributions within ARES, this work was recognized as a semi-finalist of INFORMS Franz Edelman Award (2021).

The rest of the paper is organized as follows. In Section 2, we introduce the pilot training program that motivated this study. We also provide details as to how resources are allocated to multiple phases of a lesson. In Section 3 we formally define the MCTP and introduce the notation used to formulate the problem. A mathematical formulation for the MCTP is presented in Section 4. The extended version of the MCTP is introduced in Section 5, where the formulation of the MCTP is updated to capture the additional requirements of the extended problem. In Section 6 we describe our proposed B&CH approach and the enhancement techniques that are employed for solving the problem more efficiently. Section 7 reports numerical results of our proposed methodology and improvements. The paper is concluded in Section 8.

2 Pilot Training

The Australian Defence Force (ADF) will spend $200 billion over the next decade acquiring new aircraft, vehicles, ships, submarines and other assets. Acquiring and training the people to operate these assets is complex and expensive. Pilot training, in particular, is both challenging and expensive. The ADF estimates that putting a military pilot through basic flight training costs a million dollars, with full training (including operational experiences) bringing that cost to more than $9 million AUD. Military

pilots must demonstrate not only a high standard of airmanship, they must also function in hostile environments and operate the aircraft's offensive and defensive systems. With the high per-hour costs associated with aviation training, especially in rotary wing aircraft, a steep learning curve is demanded of students to ensure taxpayer dollars are not wasted. Unfortunately, this also results in high course failure rates.

To be confident that there will be sufficient aircrew ready to operate new helicopters when they arrive, the ADF must start training at least five times the required number of pilots years before ordering the helicopters to factor in these failure rates. The training is complex, involving dozens of specialist courses run in various locations around Australia with expensive aircraft and simulators. Some added complications include sufficient number of supporting instructors, training of ground and maintenance crews and operational tasking management.

The Automated Resource and Event Scheduling (ARES) software replaces the manual planning processes within the training school, with optimized schedules resulting in increased rates and numbers of graduations. These optimized schedules are required to minimize delays for the students, which can otherwise result in the degradation of skill due to lapses in currency as well as diminishing morale, leading to a shortfall in Defence readiness. The unique aspects of this problem render generic solvers (as might be used *e.g.* by universities) unusable. The team has trialed a vast range of approaches and algorithms before settling on the current practical yet innovative solution. Although applicable to any training school, ARES has been tailored and delivered to Royal Australian Navy's (RAN) MH-60R Training Schoolhouse. The software is being used regularly on a daily basis. Meanwhile, it is being modularized to improve its flexibility and portability to other schools.

Before ARES, the RAN did its best with existing commercial solutions, backed up by a number of disconnected processes involving spreadsheets and manual scheduling. However, the challenges in Defence workforce planning are very different to those of the business world. Existing off-the-shelf workforce planning tools lack the required flexibility and depth to capture the unique features, the fidelity and the complexity of ADF operations. By generating more efficient training schedules, Defence can graduate more students ready to fill operational positions. Our optimized schedules improve graduations in terms of the quality of the solutions compared to the greedy heuristic that was originally implemented in the ARES software by the technical experts. Our improvements will in turn increase defense's "readiness" which can be regarded as a military equivalent to industry profits.

2.1 Multiphase Lessons

There are three types of students in the pilot training program under study: 'pilot', 'avwo' (Aviation Warfare Officer, whose role is as the tactical commander for the mission) and 'senso' (Sensor Operator) students. Consequently, there are three different syllabi: pilot syllabus, avwo syllabus and senso syllabus. Each syllabus consists of nearly 400 lessons. A lesson can have one to five phases: 1) preparation, 2) brief, 3) event (main phase), 4) debrief and 5) conclusion. These phases (if exist) must be completed in the same order and consecutively. In general, different phases of a lesson can have different resource requirements. If two phases of a lesson share the same resource requirement, then exactly one resource must be selected to play the corresponding role in both phases. For instance, it is not acceptable to select two different instructors for two phases of a lesson if these instructors play the same role. Due to the large number of possibilities, resource requirements for each lesson are modelled through a data structure which we call a *lesson group* or simply a *group*. Each group consists of a set of resources and a set of phases. An example of lesson groups for a lesson is presented in Table 1.

Table 1: An example of lesson groups for a lesson

Group number	Set of resources	Set of phases
1	{4,5}	{3}
2	{3,4,5,7,8,9}	{1,2,3,4}
3	{1,2,6}	{1,2,3,4}
4	{10}	{3}

The number of groups for a lesson indicates the number of different resources that the lesson requires. Exactly one resource is selected for each group and this resource must be present in all of the corresponding phases of the group as discussed above. For instance, the lesson in the given example has 4 groups which means that it needs 4 resources. In this example, Groups 1, 2 and 3 specify human resource requirements while Group 4 specifies physical resource requirement of the lesson. Resource 10 could be an aircraft or

simulator which is needed for the main phase only (Phase 3). It can be seen that the set of resources in Group 1 and Group 2 are not disjoint and they both include resources/instructors 4 and 5. This implies that among the set of resources in Group 2, only Resources 4 and 5 can also fulfil the resource requirement of Group 1 possibly due to their higher levels of qualification.

If Resource 4 is selected for Group 1, then it must be present in Phase 3 of the lesson. This resource can no longer be selected for Group 2 since it is already playing a role in Phase 3 (a resource cannot play two roles at the same time). In general, if the sets of resources for two lesson groups are not disjoint, then the corresponding sets of phases are also not disjoint. This is a defining characteristic of lesson groups which implies that a resource can be selected for at most one lesson group. This defining property is a direct consequence of the realistic assumption (and the formal requirement) that different phases of a class use the same resource as much as possible. In the example provided, Phases 1, 2, 3 and 4 respectively require 2, 2, 4 and 2 resources. Phases 1, 2, and 4 employ the same set of resources which must also be present in Phase 3.

3 Problem Definition

In this section, we formally define the MCTP. We first present the nomenclature of the paper and then introduce the problem. The inherent complexity of the problem gives rise to defining many notations. Every notation is defined in the text before it is first used. Several tables are also included in the Appendices to make the paper easier to follow. A table either includes decision variables or sets followed by parameters. Some notations are not used in problem definition or different formulations of the problem. Such notations are not included in the tables to avoid further complexity since their usage is restricted to the specific section they are defined. We also include the definition of all big-M parameters in Appendix A.

3.1 Nomenclature

In our notation, we generally use an uppercase letter with calligraphic font to denote a set and a lowercase letter to denote its elements. We also define the following notations. For a given set $\mathcal{S}$, we use $|\mathcal{S}|$ to denote its cardinality. We use $[a, b]$ to denote the set of all integers i such that $a \leq i \leq b$. More precisely, $[a, b] = \{a, \ldots, b\}$ if $a \leq b$ and $[a, b] = \emptyset$, otherwise. We also use $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ to denote the floor and ceiling functions, respectively.

Tags are used to distinguish between different subsets of a set. They are also used to distinguish between sets or parameters that represent the same concept for different entities. For instance, the MCTP has the concept of availability for students and resources as well as lessons. The sets of time slots when these entities are available has the form $\mathcal{T}_e^X$, in which e is the specific entity (*e.g.,* a particular lesson) and X is the tag used to differentiate it from other sets of time slots (*e.g.,* L is used as a tag for lessons). Tags are either a superscript capital letter or a mathematical accent. They have generally been specified in a way that facilitates remembering the definition of the notations while keeping them compact. All indices (both subscript and superscript) are lowercase letters that are generally indicative of the entity they represent. Throughout the paper, j is a session, k is a student, r is a resource, ℓ is a lesson, p is a phase, d is a day, t is a time slot, and g is a group.

3.2 Definition

The MCTP is associated with a planning horizon consisting of a number of days. Let D denote the number of days and $\mathcal{D} = [1, D]$ denote the set of days in the planning horizon. In day $d \in \mathcal{D}$, the training school opens at time slot a_d and closes at time slot b_d. We always have $a_d \leq b_d < a_{d+1} \leq b_{d+1}$. The set of time slots in day $d \in \mathcal{D}$ is $[a_d, b_d]$ and therefore the set of all time slots in the planning horizon is defined by $\mathcal{T} = \bigcup_{d \in \mathcal{D}} [a_d, b_d]$. We sometimes use T to denote the largest time slot of the planning horizon (*i.e.,* $T = \max \mathcal{T}$).

There is a given set of students, denoted $\mathcal{K}$, which consists of students from different cohorts. We use $\mathcal{K}_k^C$ to represent the set of students that are in the same cohort as student $k \in \mathcal{K}$. Furthermore, we use $\mathcal{L}_k \neq \emptyset$ to denote the set of lessons that this student can take. Note that it is not required that all lessons in $\mathcal{L}_k$ are scheduled for student k in the designated planning horizon. This is only a set of possible lessons that can be scheduled for the student. The actual set of lessons scheduled for the student, denoted $\mathcal{L}_k^\star \subseteq \mathcal{L}_k$, is an output of the optimization problem. The number of lessons that student $k \in \mathcal{K}$ takes, $|\mathcal{L}_k^\star|$, is highly dependent on the priority of student k, denoted π_k, which is a given positive integer. The higher the value of this parameter, the higher the priority of the student. Furthermore, student k cannot take lesson $\ell \in \mathcal{L}_k$ after lesson $\ell' \in \mathcal{L}_k$ if ℓ is a prerequisite for ℓ'. In order to represent the precedence

relations, we use $\Omega_{k\ell}$ to denote the set of direct/immediate subsequent lessons for lesson $\ell \in \mathcal{L}_k$, that is, $\Omega_{k\ell} = \{\ell' \in \mathcal{L}_k : \ell \text{ is a direct prerequisite for } \ell'\}$.

The set of all lessons is defined by $\mathcal{L} = \bigcup_{k\in\mathcal{K}} \mathcal{L}_k$. This implies that for any lesson $\ell \in \mathcal{L}$, there is at least one student who may take the lesson. Each lesson is characterized by a set of phases that must be scheduled consecutively. The set of phases for lesson $\ell \in \mathcal{L}$ is denoted by $\mathcal{P}_\ell$. Phase $p \in \mathcal{P}_\ell$ proceeds Phase $p' \in \mathcal{P}_\ell$ if $p < p'$. In other words, Phase 1 is scheduled immediately before Phase 2, and Phase 2 is scheduled immediately before Phase 3 and so on. The duration/length of phase $p \in \mathcal{P}$ of lesson $\ell \in \mathcal{L}$ is a positive integer denoted by $\delta_{\ell p}$. Consequently, the duration/length of lesson $\ell \in \mathcal{L}$ is defined as $\Delta_\ell = \sum_{p\in\mathcal{P}_\ell} \delta_{\ell p}$. In general, students are not required to attend every phase of a lesson. We use $\mathcal{P}'_\ell$ to denote the subset of phases of lesson $\ell \in \mathcal{L}$ that students are required to attend. Similarly, we define $\Delta'_\ell = \sum_{p\in\mathcal{P}'_\ell} \delta_{\ell p}$ to represent the total time that students are required to be present in a session of lesson $\ell \in \mathcal{L}$. Note that different sessions of a lesson have exactly the same set of phases and the same capacity. The capacity of a session of lesson $\ell \in \mathcal{L}$ (or simply the capacity of lesson ℓ), denoted c_ℓ, is the maximum number of students who can attend it. We assume that there is a given upper bound n_ℓ on the total number of sessions that can be scheduled for lesson $\ell \in \mathcal{L}$. In this study, we set this upper bound to $n_\ell = \lceil \frac{N_\ell}{c_\ell} \rceil$, where N_ℓ is the total number of students who can take lesson ℓ. We often use $\mathcal{N}_\ell = [1, n_\ell]$ to denote the set of sessions for lesson ℓ. Students who are in the same cohort must attend the same session of a certain type of lessons called cohort lessons. We use $\mathcal{L}^C$ to denote the set of cohort lessons.

In the MCTP, some lessons are assumed to not require any resources due to the abundance of certain types of equipment which are used for students to practice on their own. We use $\mathcal{L}^R$ to denote the set of all lessons that require at least one resource. As discussed in Section 2.1, lesson-groups are used to capture resource requirements of a lesson. Let $\mathcal{G}_\ell$ denote the set of lesson-groups for lesson $\ell \in \mathcal{L}^R$. We use $\mathcal{R}^G_{\ell g}$ and $\mathcal{P}^G_{\ell g} \subseteq \mathcal{P}_\ell$ to represent the sets of resources and phases in lesson-group $g \in \mathcal{G}_\ell$ of lesson $\ell \in \mathcal{L}^R$, respectively. We also define $\mathcal{R}^L_\ell = \bigcup_{g\in\mathcal{G}_\ell} \mathcal{R}^G_{\ell g}$ as the set of all resources that may be used in a phase of lesson $\ell \in \mathcal{L}^R$. Two particular resources $r \in \mathcal{R}^G_{\ell g}$ and $r' \in \mathcal{R}^G_{\ell g'}$ are sometimes required to be selected together due to compatibility restrictions. Therefore, we define $\mathcal{R}^P_\ell = \{(r, r') : r < r'\}$ as the set of resource pairs that must be selected together for lesson $\ell \in \mathcal{L}^R$. The set of all resources is defined by $\mathcal{R} = \bigcup_{\ell\in\mathcal{L}^R} \mathcal{R}^L_\ell$.

In the MCTP, some lessons must be scheduled before sunset and some of them must be scheduled after sunset (day and night lessons). There are also some lessons that can be scheduled at any time. For the sake of generality, we define a set of permissible time slots for each lesson. Let $\mathcal{T}_\ell \subseteq \mathcal{T}$ denote the set of time slots at which a session of lesson $\ell \in \mathcal{L}$ can be in progress. Similarly, we use $\mathcal{T}'_\ell \subseteq \mathcal{T}_\ell$ to denote the set of time slots at which lesson $\ell \in \mathcal{L}$ can begin. The (un)availability of students and resources in certain time slots of the planning horizon are also captured by defining the corresponding availability sets. We respectively use $\mathcal{T}^R_r \subseteq \mathcal{T}$ and $\mathcal{T}^K_k \subseteq \mathcal{T}$ to denote the sets of time slots when resource $r \in \mathcal{R}$ and student $k \in \mathcal{K}$ are available. The daily training time of the students as well as the daily usage of the resources are bounded. We use α^K_k and α^R_r to denote the maximum number of allocated time slots per day for student $k \in \mathcal{K}$ and resource $r \in \mathcal{R}$, respectively.

The presence of both students and resources in the training school is further limited by two types of parameters called *maximum time span* and *minimum rest time*. The maximum time span for resource $r \in \mathcal{R}$, denoted λ^R_r, specifies the maximum number of time slots between the first use and the last use of the resource in a day. This parameter is defined to ensure that, for instance, an instructor who begins working at 7am is not allocated to a session that begins (or continues) after 7pm. Similarly, the maximum time span for student $k \in \mathcal{K}$, denoted λ^K_k, specifies the maximum number of time slots between the first and the last time slot in a day when the student is present in a session. We have $\lambda^K_k \geq \alpha^K_k$ and $\lambda^R_r \geq \alpha^R_r$ for any $k \in \mathcal{K}$ and $r \in \mathcal{R}$. The minimum rest time for resource $r \in \mathcal{R}$, denoted μ^R_r, specifies the minimum number of time slots required between the last use in a day and the first use in the following day. This parameter is defined to ensure that, for instance, an instructor who works by 11pm on Monday is not allocated to a session that begins before 9am on Tuesday. Similarly, the minimum rest time for student $k \in \mathcal{K}$, denoted μ^K_k, specifies the minimum number of time slots required between the end of the training in a day and the start of the training in the following day for the student.

The MCTP seeks to generate a feasible schedule that maximizes $\sum_{k\in\mathcal{K}} \pi_k |\mathcal{L}^\star_k|$. A schedule is said to be feasible if it satisfies all the constraints relevant to students, resources and lessons. In addition to the constraints mentioned in the previous paragraphs, a logical constraint for both students and resources is that they cannot be allocated to more than one session at the same time. This is true even for sessions of the same lesson. The summary of the notation introduced in this section can be found in Appendix B.

4 Mathematical Formulation

In Section 3, we introduced the MCTP and its parameters. In this section, we formulate the problem as an integer linear program. We begin by introducing the decision variables used to construct the formulation. Let $v_{k\ell}$ denote a binary (0-1) variable which is positive if and only if (henceforth iff) student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{L}_k$. We also define the binary variable $w^t_{k\ell}$ which is positive iff student $k \in \mathcal{K}$ is in a session of lesson $\ell \in \mathcal{L}_k$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$. In addition, we employ the binary variable $\bar{w}_{kt}$ which is positive iff student $k \in \mathcal{K}$ is in a session of any lesson at time $t \in \mathcal{T}^K_k$. The binary variable $\bar{u}_{rt}$ is defined similarly for resource $r \in \mathcal{R}$, that is, it is positive iff the resource is being used in a session of any lesson at time $t \in \mathcal{T}^R_r$. The rest of the decision variables of the formulation are session-index variables.

We use (ℓ, j) to represent session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}$. Let $\hat{x}_{\ell j}$ be a binary variable which is positive iff (ℓ, j) is scheduled, and $x^t_{\ell j}$ be a binary variable which is positive iff (ℓ, j) begins at time $t \in \mathcal{T}'_\ell$. The binary variable $y^{pt}_{\ell j}$ is also defined to capture the progress time of individual phases of (ℓ, j). It takes the value of one iff phase $p \in \mathcal{P}_\ell$ of (ℓ, j) is in progress at time $t \in \mathcal{T}_\ell$. The allocation of students and resources to the scheduled sessions is captured through the decision variables $\hat{v}^k_{\ell j}$ and $z^{rg}_{\ell j}$, respectively. The binary variable $\hat{v}^k_{\ell j}$ is positive iff student $k \in \mathcal{K}$ is allocated to session (ℓ, j). The binary variable $z^{rg}_{\ell j}$ is positive if session (ℓ, j) employs resource $r \in \mathcal{R}^G_{\ell g}$ for its lesson-group $g \in \mathcal{G}_\ell$. Finally, we define two sets of decision variables to capture the presence of students and resources in a session at a certain time slot. The binary variable $\hat{w}^{kt}_{\ell j}$ is positive iff student $k \in \mathcal{K}$ is present in session (ℓ, j) at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$. The binary variable $u^{rt}_{\ell j}$ is defined similarly, that is, it takes a positive value if resource $r \in \mathcal{R}^L_\ell$ is used in session (ℓ, j) at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^R_r$.

The summary of the decision variables introduced in this section can be found in Appendix C. Note that some of these decision variables can be seen as auxiliary variables. These auxiliary variables are added to improve the readability of the formulation and its performance when using a standard linear solver (see *e.g.,* Esmaeilbeigi et al., 2015, 2016a). The auxiliary variables improve the formulation by reducing the number of non-zero elements of the coefficient matrix. Modern solvers employ B&B algorithms which are based on solving a Linear Programming (LP) relaxation of the formulation at each node of the B&B tree. A high number of non-zeros in the coefficient matrix can cause numerical inefficiencies in solving the LP relaxations. Another advantage of using the auxiliary variables is that they can guide the branching procedure of the B&B algorithm as demonstrated in Esmaeilbeigi et al. (2016a). Next we present the mathematical formulation of the MCTP. The objective of the MCTP is to maximize the total weighted number of student-lesson allocations which can be expressed as follows:

$$F1: \max \sum_{k \in \mathcal{K}} \pi_k \sum_{\ell \in \mathcal{L}_k} v_{k\ell} \tag{1.01}$$

The constraints of the formulation can be divided into session, student and resource related constraints. We introduce each of them in a dedicated section. Furthermore, students and resources also share some common restrictions such as limitation on the daily usage and the existence of the maximum time span and minimum rest time parameters as explained in Section 3. We refer to such requirements as *duty rules* and present the corresponding constraints in a dedicated section.

4.1 Sessions

Each session of a lesson can begin once during the planning horizon. Furthermore, the session can begin only if it is scheduled. These logical conditions can be imposed by adding the following constraints:

$$\sum_{t \in \mathcal{T}'_\ell} x^t_{\ell j} = \hat{x}_{\ell j} \qquad \ell \in \mathcal{L}, j \in \mathcal{N}_\ell \tag{1.02}$$

The start time of a session and the progress time of its phases must be aligned properly. This is ensured through the following constraints:

$$x^t_{\ell j} \leq y^{pt'}_{\ell j} \qquad \ell \in \mathcal{L}, j \in \mathcal{N}_\ell, p \in \mathcal{P}_\ell, t \in \mathcal{T}'_\ell, t' \in \left[t + \sum_{p'=1}^{p-1} \delta_{\ell p'},\ t - 1 + \sum_{p'=1}^{p} \delta_{\ell p'} \right] \tag{1.03}$$

$$\sum_{t \in \mathcal{T}_\ell} y^{pt}_{\ell j} \leq \delta_{\ell p} \hat{x}_{\ell j} \qquad \ell \in \mathcal{L}, j \in \mathcal{N}_\ell, p \in \mathcal{P}_\ell \tag{1.04}$$

According to Constraints (1.03), if session (ℓ, j) begins at time t, then the y variables corresponding to the progress times of its phases will be positive. Constraints (1.04) ensure that the appropriate number of y variables take a positive value.

Session-index variables cause symmetry, in the sense that session indices in these variables can be permuted without changing the structure of the problem (Margot, 2010). Symmetric formulations typically require many nodes to be investigated in the B&B search tree. A well-known method of breaking symmetry of a formulation is introduction of the so-called hierarchical constraints (Sherali and Smith, 2001). The following is a set of such symmetry breaking constraints.

$$\sum_{t\in\mathcal{T}'_\ell} f(t)\, x^t_{\ell,j+1} \le \sum_{t\in\mathcal{T}'_\ell} f(t)\, x^t_{\ell j} \qquad \ell\in\mathcal{L}, j\in\mathcal{N}_\ell\setminus\{n_\ell\} \tag{1.05}$$

Here, $f(t) \coloneqq T - t + 1 \ge 1$ is a decreasing function of $t \in \mathcal{T}'_\ell$. From Equation (1.02) we know that exactly one start time is allocated to each session (ℓ, j). As a result, the left-hand side and the right-hand side of (1.05) respectively capture the value of $f(t)$ corresponding to the start times of $(\ell, j+1)$ and (ℓ, j). It follows that these constraints sort the sessions in the order of their appearance in the schedule. Furthermore, they ensure that no empty sessions are scheduled between two consecutive sessions, that is, $\hat{x}_{\ell,j+1} = 0$ if $\hat{x}_{\ell j} = 0$.

4.2 Students

In this section, we present the constraints of the problem that involve the students. The following set of constraints states that if a student takes a lesson, then he/she must attend exactly one session of the lesson.

$$\sum_{j\in\mathcal{N}_\ell} \hat{v}^k_{\ell j} = v_{k\ell} \qquad k\in\mathcal{K}, \ell\in\mathcal{L}_k \tag{1.06}$$

If a student takes a syllabus lesson then he/she must also take its prerequisites:

$$v_{k\ell'} \le v_{k\ell} \qquad k\in\mathcal{K}, \ell\in\mathcal{L}_k, \ell'\in\Omega_{k\ell} \tag{1.07}$$

Students in the same cohort must attend the same session of a cohort lesson:

$$\hat{v}^k_{\ell j} = \hat{v}^{k'}_{\ell j} \qquad k\in\mathcal{K}, k'\in\mathcal{K}^C_k : k' < k, \ell\in\mathcal{L}^C\cap\mathcal{L}_k\cap\mathcal{L}_{k'}, j\in\mathcal{N}_\ell \tag{1.08}$$

A student must dedicate Δ'_ℓ time slots to a session of lesson ℓ if he/she attends this session. Otherwise, no time slots will be dedicated to the session:

$$\sum_{t\in\mathcal{T}_\ell\cap\mathcal{T}^K_k} \hat{w}^{kt}_{\ell j} = \Delta'_\ell\, \hat{v}^k_{\ell j} \qquad k\in\mathcal{K}, \ell\in\mathcal{L}_k, j\in\mathcal{N}_\ell \tag{1.09}$$

Students can be in a session at a time slot only if the session is in progress at that time slot:

$$\sum_{k\in\mathcal{K}:\, t\in\mathcal{T}^K_k, \ell\in\mathcal{L}_k} \hat{w}^{kt}_{\ell j} \le c_\ell \sum_{p\in\mathcal{P}'_\ell} y^{pt}_{\ell j} \qquad \ell\in\mathcal{L}, j\in\mathcal{N}_\ell, t\in\mathcal{T}_\ell \tag{1.10}$$

By definition, $w^t_{k\ell} = 1$ iff student $k \in \mathcal{K}$ is taking lesson $\ell \in \mathcal{L}_k$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$ by attending a session of ℓ. This is enforced by the following constraints:

$$\sum_{j\in\mathcal{N}_\ell} \hat{w}^{kt}_{\ell j} = w^t_{k\ell} \qquad k\in\mathcal{K}, \ell\in\mathcal{L}_k, t\in\mathcal{T}_\ell\cap\mathcal{T}^K_k \tag{1.11}$$

Each student can take at most one lesson at any given time. This can be imposed by adding the following set of constraints:

$$\sum_{\ell\in\mathcal{L}_k:\, t\in\mathcal{T}_\ell} w^t_{k\ell} = \bar{w}_{kt} \qquad k\in\mathcal{K}, t\in\mathcal{T}^K_k \tag{1.12}$$

Constraints (1.12) also capture the value of the decision variables $\bar{w}$.

Next we introduce a set of constraints that ensure the precedence relations between the lessons for each student are satisfied. Note that Constraints (1.07) only ensure that the prerequisite lessons are scheduled for each student. However, they do not impose any restriction on the time those lessons are scheduled. Therefore, we introduce the following precedence constraints:

$$\sum_{t'\in\mathcal{T}_{\ell'}\cap\mathcal{T}^K_k:\, t'\le t} w^{t'}_{k\ell'} \le \Delta'_{\ell'}\,(1 - w^t_{k\ell}) \qquad k\in\mathcal{K}, \ell\in\mathcal{L}_k, t\in\mathcal{T}_\ell\cap\mathcal{T}^K_k, \ell'\in\Omega_{k\ell} \tag{1.13}$$

According to Constraints (1.13), if student k is taking lesson ℓ at time t, then he/she could not take the subsequent lesson ℓ' at or before time t.

The number of students attending a session cannot exceed the maximum capacity of the session. Furthermore, at least one student should be allocated to the session. The following constraints ensure that these conditions are satisfied:

$$\hat{x}_{\ell j} \leq \sum_{k \in \mathcal{K}:\ \ell \in \mathcal{L}_k} \hat{v}^k_{\ell j} \leq c_\ell \, \hat{x}_{\ell j} \qquad \ell \in \mathcal{L}, j \in \mathcal{N}_\ell \tag{1.14}$$

Note that these constraints also ensure that students are not allocated to a session if the session is not scheduled.

4.3 Resources

In this section, we introduce the constraints of the problem that involve the resources. The following set of constraints states that exactly one resource must be selected for each lesson-group of a lesson. Furthermore, no resource is considered for the lesson if the lesson is not scheduled.

$$\sum_{r \in \mathcal{R}^G_{\ell g}} z^{rg}_{\ell j} = \hat{x}_{\ell j} \qquad \ell \in \mathcal{L}^R, j \in \mathcal{N}_\ell, g \in \mathcal{G}_\ell \tag{1.15}$$

Each resource can be allocated to at most one lesson-group of a lesson:

$$\sum_{g \in \mathcal{G}_\ell:\ r \in \mathcal{R}^G_{\ell g}} z^{rg}_{\ell j} \leq 1 \qquad \ell \in \mathcal{L}^R, j \in \mathcal{N}_\ell, r \in \mathcal{R}^L_\ell \tag{1.16}$$

The following constraints represent the resource compatibility constraints. According to these constraints, two specific resources must be selected together if they create a resource pair:

$$\sum_{g \in \mathcal{G}_\ell:\ r \in \mathcal{R}^G_{\ell g}} z^{rg}_{\ell j} = \sum_{g \in \mathcal{G}_\ell:\ r' \in \mathcal{R}^G_{\ell g}} z^{r'g}_{\ell j} \qquad \ell \in \mathcal{L}^R, j \in \mathcal{N}_\ell, (r, r') \in \mathcal{R}^P_\ell \tag{1.17}$$

The following constraints ensure that appropriate time slots are allocated to the resources which are assigned to different phases of a session.

$$u^{rt}_{\ell j} \geq z^{rg}_{\ell j} + \sum_{p \in \mathcal{P}^G_{\ell g}} y^{pt}_{\ell j} - 1 \qquad \ell \in \mathcal{L}^R, j \in \mathcal{N}_\ell, g \in \mathcal{G}_\ell, r \in \mathcal{R}^G_{\ell g}, t \in \mathcal{T}_\ell \cap \mathcal{T}^R_r \tag{1.18}$$

$$0 \geq z^{rg}_{\ell j} + \sum_{p \in \mathcal{P}^G_{\ell g}} y^{pt}_{\ell j} - 1 \qquad \ell \in \mathcal{L}^R, j \in \mathcal{N}_\ell, g \in \mathcal{G}_\ell, r \in \mathcal{R}^G_{\ell g}, t \in \mathcal{T}_\ell \setminus \mathcal{T}^R_r \tag{1.19}$$

$$\sum_{t \in \mathcal{T}_\ell \cap \mathcal{T}^R_r} u^{rt}_{\ell j} \leq \sum_{g \in \mathcal{G}_\ell:\ r \in \mathcal{R}^G_{\ell g}} \left(\sum_{p \in \mathcal{P}^G_{\ell g}} \delta_{\ell p} \right) z^{rg}_{\ell j} \qquad \ell \in \mathcal{L}^R, j \in \mathcal{N}_\ell, r \in \mathcal{R}^L_\ell \tag{1.20}$$

Constraints (1.18) ensure that $u^{rt}_{\ell j} = 1$ if resource r is allocated to group g of session (ℓ, j) and phase $p \in \mathcal{P}^G_{\ell g}$ of the session is in progress at time t. Constraints (1.19) are equivalent to Constraints (1.18) with $u^{rt}_{\ell j} = 0$. These constraints ensure that resources are not allocated to any session when they are unavailable. Constraints (1.20) in conjunction with Constraints (1.18) guarantee that $u^{rt}_{\ell j} = 1$ if and only if resource r is employed in session (ℓ, j) at time t. In other words, these decision variables will not take a positive value arbitrarily.

A resource cannot be allocated to more than one session at the same time. This is ensured by the following constraints where the value of the binary variable $\bar{u}_{rt}$ is also captured.

$$\sum_{\ell \in \mathcal{L}^R:\ t \in \mathcal{T}_\ell, r \in \mathcal{R}^L_\ell} \sum_{j \in \mathcal{N}_\ell} u^{rt}_{\ell j} = \bar{u}_{rt} \qquad r \in \mathcal{R}, t \in \mathcal{T}^R_r \tag{1.21}$$

4.4 Duty rules

Students and resources have restrictions on the daily usage as well as the maximum time span and minimum rest time rules as defined in Section 3. Such duty rules are addressed in this section. The maximum time span for students and resources are respectively captured through the following constraints:

$$\sum_{t' \in [t + \lambda^K_k, b_d] \cap \mathcal{T}^K_k} \bar{w}_{kt'} \leq M^K_{kdt} \, (1 - \bar{w}_{kt}) \qquad d \in \mathcal{D}, k \in \mathcal{K}, t \in [a_d, b_d] \cap \mathcal{T}^K_k \tag{1.22}$$

$$\sum_{t' \in [t+\lambda_r^R, b_d] \cap \mathcal{T}_r^R} \bar{u}_{rt'} \leq M_{rdt}^R (1 - \bar{u}_{rt}) \qquad d \in \mathcal{D}, r \in \mathcal{R}, t \in [a_d, b_d] \cap \mathcal{T}_r^R \tag{1.23}$$

In these constraints, M_{kdt}^K and M_{rdt}^R are big-M parameters (sufficiently large numbers that render the corresponding constraints inactive when respectively $\bar{w}_{kt} = 0$ and $\bar{u}_{rt} = 0$).
According to Constraints (1.22), if student k is in a session at time $t \in [a_d, b_d]$ (*i.e.*, $\bar{w}_{kt} = 1$), then he/she cannot be present in a session at any time slot in $[t + \lambda_k^K, b_d]$. Constraints (1.23) are defined similarly for the resources.
Minimum rest time constraints for students and resources are respectively presented as follows:

$$\sum_{t' \in [a_{d+1}, t+\mu_k^K] \cap \mathcal{T}_k^K} \bar{w}_{kt'} \leq \hat{M}_{kdt}^K (1 - \bar{w}_{kt}) \qquad d \in \mathcal{D} \setminus \{D\}, k \in \mathcal{K}, \; t \in [a_d, b_d] \cap \mathcal{T}_k^K \tag{1.24}$$

$$\sum_{t' \in [a_{d+1}, t+\mu_r^R] \cap \mathcal{T}_r^R} \bar{u}_{rt'} \leq \hat{M}_{rdt}^R (1 - \bar{u}_{rt}) \qquad d \in \mathcal{D} \setminus \{D\}, r \in \mathcal{R}, \; t \in [a_d, b_d] \cap \mathcal{T}_r^R \tag{1.25}$$

Constraints (1.24) and (1.25) are similar to Constraints (1.22) and (1.23). According to Constraints (1.24), if student k is in a session at time $t \in [a_d, b_d]$, then he/she cannot be present in a session at any time slot in $[a_{d+1}, t + \mu_k^K]$. Constraints (1.25) are defined similarly for the resources. Finally, the maximum number of time slots per day for students and resources are respectively captured through the following constraints:

$$\sum_{t \in [a_d, b_d] \cap \mathcal{T}_k^K} \bar{w}_{kt} \leq \alpha_k^K \qquad d \in \mathcal{D}, k \in \mathcal{K} \tag{1.26}$$

$$\sum_{t \in [a_d, b_d] \cap \mathcal{T}_r^R} \bar{u}_{rt} \leq \alpha_r^R \qquad d \in \mathcal{D}, r \in \mathcal{R} \tag{1.27}$$

5 The Extended Problem

In this section, we introduce an extended version of the MCTP which includes more complex aspects of the real-world pilot training program under study. We refer to this version of the problem as the Extended MCTP (EMCTP). The EMCTP captures further requirements such as turnaround time for aircraft and simulators as well as the incorporation of three types of lessons referred to as one-time-prepared lessons, shared lessons and double bubbles. The parameters for the EMCTP are mostly the parameters of the MCTP with some additional parameters which will be introduced in this section. A summary of these parameters can be found in Appendix D.

5.1 One-time Preparation

In the EMCTP, there are some lessons that do not require a student to attend their preparation phase. The preparation phase of such lessons is completed by an instructor. These lessons can be further divided into two categories. The first category includes lessons whose preparation phases are done once a day by an instructor. We refer to these lessons as one-time prepared lessons. Once the instructor completes the preparation phase for the first session, he/she will not run the preparation phase for the remaining sessions of the (one-time prepared) lesson in the same day. The second category includes lessons whose preparation phase must be completed by the instructor each time a session of the lesson is run.

The second category of these lessons is already addressed in the formulation of the MCTP. The first category of the lessons, the one-time prepared lessons, requires additional constraints which will be captured in the formulation of the EMCTP. We use $\mathcal{L}^P$ to denote these lessons. Observe that we model one-time prepared lessons by assuming that they do not have a preparation phase. We introduce a number of constraints in order to capture one-time preparation requirements properly. The preparation time for these lessons takes one hour. In this study, the duration of a time slot is considered as 30 minutes and therefore the preparation time is equal to two time slots.

5.2 Shared Lessons

There are three types of students in the pilot training program under study: pilot, avwo and senso students. We use $\mathcal{K}^P$, $\mathcal{K}^A$ and $\mathcal{K}^S$ to denote these sets of students, respectively. There are also three

different syllabi: pilot syllabus, avwo syllabus and senso syllabus. Each student must take all lessons in their respective syallbus to graduate. We refer to such lessons as *syllabus lessons*. Observe that the set of all lessons, $\mathcal{L}$, in the MCTP consists of the syllabus lessons. However, the set of all lessons in the EMCTP includes shared lessons and double bubbles too. In this section, we introduce shared lessons.

Some syllabus lessons in the pilot syllabus can be shared with some syllabus lessons in the avwo syllabus. We refer to these lessons as *shared lessons*. A defining characteristic of shared lessons is that at least one pilot student and one avwo student must attend a session of a shared lesson. Therefore, shared lessons are formed from two syllabus lessons, one from a pilot syllabus and one from an avwo syllabus. However, a student is not required to attend a shared lesson in order to take a particular lesson in their syllabus that can be shared. In general, lesson groups of a shared lesson may be different from lesson groups of the two syllabus lessons that form it.

Next we give an example to further explain shared lessons. Consider 4 syllabus lessons: ℓ_1 and ℓ_3 from the pilot syllabus and ℓ_2 and ℓ_4 from the avwo syllabus. In general, for a given syllabus lesson we are given (as an input) a set of shared lessons that can be formed from the syllabus lesson. For example, the aforementioned lessons may form three shared lessons as follows: $\ell'_1 = \{\ell_1, \ell_2\}$, $\ell'_2 = \{\ell_3, \ell_2\}$, $\ell'_3 = \{\ell_3, \ell_4\}$.

While ℓ_1 can be shared with ℓ_2 and ℓ_2 can be shared with ℓ_3, ℓ_1 is never shared with ℓ_3 since they are in the same syllabus. Although ℓ_1 and ℓ_4 are in different syllabi, they have not been allowed to be shared possibly due to technical incompatibilities. We sometimes refer to a shared lesson or a syllabus lesson as an *actual lesson*. In the presented example, the set of actual lessons is $\{\ell_1, \ell_2, \ell_3, \ell_4, \ell'_1, \ell'_2, \ell'_3\}$. We use $\mathcal{A}$ to denote the set of actual lessons and $\mathcal{H} \subset \mathcal{A}$ to denote the set of shared lessons. Evidently, $\mathcal{A} \setminus \mathcal{H}$ represents the set of syllabus lessons. The set of shared lessons that are made from the syllabus lesson $\ell \in \mathcal{A} \setminus \mathcal{H}$ is denoted by $\mathcal{H}_\ell$. Furthermore, the set of actual lessons for student $k \in \mathcal{K}$ is denoted by $\mathcal{A}_k$.

5.3 Double Bubbles

In this section, we introduce a new type of lesson that is obtained by combining two flying actual lessons. We refer to such a lesson as a *double bubble*. The advantage of employing double bubbles in the schedule is the opportunity that they provide for better utilization of the resources. A flying actual lesson has 4 phases: preparation (1 hour), brief (2 hours), flying event (2 hours) and debrief (1 hour). A double bubble $\ell'' = \{\ell, \ell'\}$ has the same sequence of phases as ℓ and ℓ'. However, it has a larger duration for the flying event. The duration of the flying event of ℓ'' is equal to the summation of the duration of the flying events of ℓ and ℓ' plus half an hour turnaround time for the aircraft. More precisely, ℓ'' has the following phases: preparation (1 hour), brief (2 hours), flying event (4.5 hours) and debrief (1 hour).

Each student who attends a session of ℓ'' either wants to take lesson ℓ or lesson ℓ' (not both). All students who take a session of ℓ'' will attend the preparation, brief, and debrief phases of the lesson. However, the main phase of ℓ'' will split into three sections: 2 hours for the students who take lesson ℓ, half an hour for the turnaround time of the aircraft, and 2 hours for the students who take lesson ℓ'. While the capacity of ℓ'' is equal to the summation of the capacities of ℓ and ℓ', the set of resources for ℓ'' is not larger than that of ℓ or ℓ' alone. In fact, the set of resources that are used in the first bubble ℓ must be equal to the set of resources used in the second bubble ℓ'. If such a requirement cannot be satisfied for two lessons ℓ and ℓ', then ℓ'' will not be considered as a potential double bubble.

Two flying actual lessons can be combined to form a double bubble only if they belong to the same *pool* of double bubbles. Following the example presented in Section 5.2, suppose that the syllabus lessons ℓ_1, ℓ_2, ℓ_3 are in a double-bubble pool. This implies that shared lessons ℓ'_1 and ℓ'_2 are also in this double bubble pool. Therefore, we have the following double bubbles: $\ell''_1 = \{\ell_1, \ell_1\} = \{\ell_1\}$, $\ell''_2 = \{\ell_1, \ell_2\}$, $\ell''_3 = \{\ell_1, \ell_3\}$, $\ell''_4 = \{\ell_2, \ell_2\} = \{\ell_2\}$, $\ell''_5 = \{\ell_2, \ell_3\}$, $\ell''_6 = \{\ell_3, \ell_3\} = \{\ell_3\}$, $\ell''_7 = \{\ell'_1, \ell'_1\} = \{\ell'_1\}$, $\ell''_8 = \{\ell'_1, \ell_1\}$, $\ell''_9 = \{\ell'_1, \ell_2\}$, $\ell''_{10} = \{\ell'_1, \ell_3\}$, $\ell''_{11} = \{\ell'_2, \ell'_2\} = \{\ell'_2\}$, $\ell''_{12} = \{\ell'_2, \ell_1\}$, $\ell''_{13} = \{\ell'_2, \ell_2\}$, $\ell''_{14} = \{\ell'_2, \ell_3\}$.

As can be seen, a double bubble may be formed from one actual lesson. The set of all double bubbles that are formed from a flying actual lesson is an input parameter for the EMCTP. Let $\mathcal{B}$ denote the set of all double bubbles. The set of all lessons in the EMCTP is defined by $\mathcal{L} \coloneqq \mathcal{A} \cup \mathcal{B} = \{1, \ldots, |\mathcal{A}|, \ldots, |\mathcal{A}| + |\mathcal{B}|\}$. We use $\mathcal{B}_\ell \subseteq \mathcal{B}$ to denote the set of double bubbles that are formed from lesson $\ell \in \mathcal{A}$. Furthermore, we use $\mathcal{B}'_\ell$ to denote the set of lessons that form double bubble $\ell \in \mathcal{B}$. Evidently, we have $|\mathcal{B}'_\ell| \in \{1, 2\}$. For any $\ell \in \mathcal{L}$, we respectively use c^P_ℓ, c^A_ℓ and c^S_ℓ to denote the maximum number of pilot, avwo and senso students required for one session of the lesson. The total capacity of the lesson is defined by $c_\ell \coloneqq c^P_\ell + c^A_\ell + c^S_\ell$. We sometimes use $\mathcal{G}'_\ell \subseteq \mathcal{G}_\ell$ to denote the set of lesson-groups for lesson $\ell \in \mathcal{L}^R$ that pertain to the main phase, *i.e.*, $\mathcal{G}'_\ell \coloneqq \{g \in \mathcal{G}_\ell \colon 3 \in \mathcal{P}^G_{\ell g}\}$. Note that for two lessons $\ell \in \mathcal{A}$ and $\ell' \in \mathcal{B}_\ell$ we have $\mathcal{T}_{\ell'} \subseteq \mathcal{T}_\ell$ and $\mathcal{T}'_{\ell'} \subseteq \mathcal{T}'_\ell$.

We complete this section by specifying the maximum number of sessions for the shared lessons and double bubbles. As mentioned in Section 3, we set $n_\ell = \lceil \frac{N_\ell}{c_\ell} \rceil$ for a syllabus lesson $\ell \in \mathcal{A} \setminus \mathcal{H}$, where N_ℓ is the total number of students who can take lesson ℓ. We calculate the upper bound on the number of

required sessions for each lesson as follows. For an actual lesson $\ell \in \mathcal{A}$ we have

$$n_\ell = \begin{cases} \left\lceil \frac{N_\ell}{c_\ell} \right\rceil & \ell \in \mathcal{A} \setminus \mathcal{H} \\ \min\{n_{\ell'}, n_{\ell''}\} & \ell \in \mathcal{H} \colon \ell', \ell'' \in \mathcal{A} \setminus \mathcal{H},\ \ell \in \mathcal{H}_{\ell'} \cap \mathcal{H}_{\ell''} \end{cases}$$

For a double bubble $\ell \in \mathcal{B}$ with $\mathcal{B}'_\ell = \{\ell', \ell''\}$ we have

$$n_\ell = \begin{cases} \left\lfloor \frac{n_{\ell'}}{2} \right\rfloor & \ell' = \ell'' \\ \min\left\{ \left\lfloor \frac{n_{\ell'}}{2} \right\rfloor, n_{\ell''} \right\} & \ell' \neq \ell'',\ \ell' \in \mathcal{A} \setminus \mathcal{H},\ \ell'' \in \mathcal{H}_{\ell'} \\ \min\left\{ \left\lfloor \frac{n_{\ell'''}}{2} \right\rfloor, n_{\ell''}, n_{\ell'} \right\} & \ell' \neq \ell'',\ \ell', \ell'' \in \mathcal{H}_{\ell'''} \text{ for some } \ell''' \in \mathcal{A} \setminus \mathcal{H} \\ \min\{n_{\ell'}, n_{\ell''}\} & \text{otherwise} \end{cases}$$

5.4 Further Requirements

In the pilot training program under study, aircraft and simulators require a turnaround time (or setup time) of 30 minutes. Since the duration of a time slot is also 30 minutes, each setup time takes one time slot. Aircraft and simulators are used in the main phase (aka the event phase) of a lesson. Once the main phase is finished, the aircraft/simulator cannot be used for one time slot. We use $\mathcal{R}^S$ to denote the set of resources that require a setup time.

As mentioned in Section 3, resources and students have a limit on the amount of work/training per day. Unlike students, resources also have some weekly limits. For resource $r \in \mathcal{R}$, the total amount of work per week cannot exceed a given number of time slots denoted β_r^R. We have $\beta_r^R < D\alpha_r^R$. It is noteworthy to mention that resources also have usage limits per month and per year. These quantities can be captured by simply updating the value of the weekly usage β_r^R. There is also a requirement that concerns the number of sorties (lessons that use an aircraft or simulator) per week for an instructor. Let $\mathcal{R}^I \subseteq \mathcal{R}$ denote the set of instructors. The total number of sorties per week for instructor $r \in \mathcal{R}^I$ cannot exceed a given upper bound denoted γ_r^R. We also use $\mathcal{L}^S$ to denote the set of sorties.

The EMCTP also captures the requirements regarding aircraft configurations. In general, different flying lessons can require different aircraft configurations. A flying lesson can have one of the so-called ALFS, UTIL and ANY configurations as a requirement. If the lesson requires an ALFS configuration, then only an aircraft that has an ALFS configuration can be used for the lesson. The same holds for the UTIL configuration. If the lesson requires an ANY configuration, then an aircraft with either ALFS or UTIL can be used for the lesson. An aircraft can only have UTIL or ALFS configuration per day which is a decision variable in the EMCTP. We use R^F to denote the set of aircraft and F^A and F^U to denote the set of flying lessons that require ALFS and UTIL configurations, respectively. Lessons with an ANY requirement are not used in the relevant constraints and therefore we do not introduce a set for representing them.

5.5 Updated Formulation

In this section, we present a mathematical formulation for the EMCTP which is obtained by updating *F1*. All of the decision variables and parameters used in *F1* are also used in the updated formulation with the exception of the set $\mathcal{L}_k$. For consistency, this set is replaced with $\mathcal{A}_k$ or $\mathcal{A}_k \setminus \mathcal{H}$ in the updated formulation. The definitions of the other parameters remain valid for the EMCTP. This is also true for the decision variables with the exception of the variables $\hat{v}_{\ell j}^k$ and $\hat{w}_{\ell j}^{kt}$. The lesson ℓ in these decision variables is a syllabus lesson in *F1* while it is an actual lesson in the updated formulation. Note that there is no difference between the updated variables and their previous versions in the absence of the shared lessons. Next we introduce the new decision variables that are used in the updated formulation. A summary of the decision variables used to develop the updated formulation can be found in Appendix E.

For an instructor $r \in \mathcal{R}^I$, we define the binary decision variable $\hat{u}_{rt}$ which is positive iff the instructor is doing his/her one-time preparation at time $t \in \mathcal{T}_r^R$. Furthermore, for an aircraft $r \in \mathcal{R}^F$ we define the binary variable $\check{u}_{rd}$ which is positive if the aircraft is set to the UTIL configuration and the value zero if it is set to the ALFS configuration in day $d \in \mathcal{D}$. We also define two types of decision variables for allocating students to the double bubbles. Let $\check{v}_{\ell' j}^{k\ell}$ be a binary variable which is positive iff student $k \in \mathcal{K}$ attends session $j \in \mathcal{N}_{\ell'}$ of double bubble $\ell' \in \mathcal{B}_\ell$ in order to take lesson $\ell \in \mathcal{A}_k$. The binary variable $\check{w}_{\ell' \ell j}^{kt}$ is also defined to be positive iff student $k \in \mathcal{K}$ takes lesson $\ell' \in \mathcal{A}_k$ by attending session $j \in \mathcal{N}_\ell$ of double bubble $\ell \in \mathcal{B}_{\ell'}$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}_k^K$.

The updated formulation also inherits most of the constraints of *F1*. More precisely, only the student-related constraints introduced in Section 4.2 and the Constraint set (1.21) will be modified in the updated formulation. A number of additional constraints will also be introduced. The objective function in the EMCTP is the same as that of the MCTP, that is, it maximizes the total weighted number of student-lesson allocations:

$$F2: \max \sum_{k \in \mathcal{K}} \pi_k \sum_{\ell \in \mathcal{A}_k \setminus \mathcal{H}} v_{k\ell} \tag{2.01}$$

The decision variable $v_{k\ell}$ is a binary variable which takes a positive value iff student $k \in \mathcal{K}$ takes the syllabus lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$. This can happen by attending a session of ℓ or a session of a shared lesson $\ell' \in \mathcal{H}_\ell$ or a double-bubble $\ell'' \in \mathcal{B}_\ell \cup \mathcal{B}_{\ell'}$. Therefore we add the following constraints in order to ensure that a student takes a syllabus lesson no more than once:

$$\begin{aligned} &\sum_{j \in \mathcal{N}_\ell} \hat{v}^k_{\ell j} + \sum_{\ell' \in \mathcal{B}_\ell} \sum_{j \in \mathcal{N}_{\ell'}} \check{v}^{k\ell}_{\ell' j} \\ &+ \sum_{\ell' \in \mathcal{H}_\ell} \left(\sum_{j \in \mathcal{N}_{\ell'}} \hat{v}^k_{\ell' j} + \sum_{\ell'' \in \mathcal{B}_{\ell'}} \sum_{j \in \mathcal{N}_{\ell''}} \check{v}^{k\ell'}_{\ell'' j} \right) = v_{k\ell} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H} \end{aligned} \tag{2.02}$$

The rest of the student-related constraints are also modified accordingly to address shared lessons and double bubbles. A summary of the modified constraints follows. A student can take a syllabus lesson only if he/she has taken its prerequisites:

$$v_{k\ell'} \leq v_{k\ell} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H}, \ell' \in \Omega_{k\ell} \tag{2.03}$$

Students in the same cohort must attend the same session of a cohort lesson. Note that cohort lessons are a subset of actual lessons:

$$\hat{v}^k_{\ell j} = \hat{v}^{k'}_{\ell j} \qquad k \in \mathcal{K}, k' \in \mathcal{K}^C_k : k' < k, \ell \in \mathcal{L}^C \cap \mathcal{A}_k \cap \mathcal{A}_{k'}, j \in \mathcal{N}_\ell \tag{2.04}$$

Each student must be present in a session of lesson $\ell \in \mathcal{L}$ for Δ'_ℓ time slots:

$$\sum_{t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k} \hat{w}^{kt}_{\ell j} = \Delta'_\ell \, \hat{v}^k_{\ell j} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k, j \in \mathcal{N}_\ell \tag{2.05}$$

$$\sum_{t \in \mathcal{T}_{\ell'} \cap \mathcal{T}^K_k} \check{w}^{kt}_{\ell\ell' j} = \Delta'_{\ell'} \, \check{v}^{k\ell}_{\ell' j} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k, \ell' \in \mathcal{B}_\ell, j \in \mathcal{N}_{\ell'} \tag{2.06}$$

Students cannot be in a session at a time slot if the session is not in progress at that time slot:

$$\sum_{k \in \mathcal{K} :\, t \in \mathcal{T}^K_k, \ell \in \mathcal{A}_k} \hat{w}^{kt}_{\ell j} \leq c_\ell \sum_{p \in \mathcal{P}'_\ell} y^{pt}_{\ell j} \qquad \ell \in \mathcal{A}, j \in \mathcal{N}_\ell, t \in \mathcal{T}_\ell \tag{2.07}$$

$$\sum_{\ell' \in \mathcal{B}'_\ell} \sum_{k \in \mathcal{K} :\, t \in \mathcal{T}^K_k, \ell' \in \mathcal{A}_k} \check{w}^{kt}_{\ell' \ell j} \leq c_\ell \sum_{p \in \mathcal{P}'_\ell} y^{pt}_{\ell j} \qquad \ell \in \mathcal{B}, j \in \mathcal{N}_\ell, t \in \mathcal{T}_\ell \tag{2.08}$$

The binary variable $w^t_{k\ell}$ is positive iff student $k \in \mathcal{K}$ is taking the syllabus lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$. We know that this can happen by attending a session of ℓ or a session of a shared lesson $\ell' \in \mathcal{H}_\ell$ or a double bubble $\ell'' \in \mathcal{B}_\ell \cup \mathcal{B}_{\ell'}$. Therefore we have:

$$\begin{aligned} &\sum_{j \in \mathcal{N}_\ell} \hat{w}^{kt}_{\ell j} + \sum_{\ell' \in \mathcal{B}_\ell :\, t \in \mathcal{T}_{\ell'}} \sum_{j \in \mathcal{N}_{\ell'}} \check{w}^{kt}_{\ell\ell' j} \\ &+ \sum_{\ell' \in \mathcal{H}_\ell :\, t \in \mathcal{T}_{\ell'}} \left(\sum_{j \in \mathcal{N}_{\ell'}} \hat{w}^{kt}_{\ell' j} + \sum_{\ell'' \in \mathcal{B}_{\ell'} :\, t \in \mathcal{T}_{\ell''}} \sum_{j \in \mathcal{N}_{\ell''}} \check{w}^{kt}_{\ell'\ell'' j} \right) = w^t_{k\ell} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H}, t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k \end{aligned} \tag{2.09}$$

A student can take at most one lesson at any given time:

$$\sum_{\ell \in \mathcal{A}_k \setminus \mathcal{H} :\, t \in \mathcal{T}_\ell} w^t_{k\ell} = \bar{w}_{kt} \qquad k \in \mathcal{K}, t \in \mathcal{T}^K_k \tag{2.10}$$

A student cannot take a lesson before its prerequisites:

$$\sum_{t' \in \mathcal{T}_{\ell'} \cap \mathcal{T}_k^K \,:\, t' \le t} w_{k\ell'}^{t'} \le \check{M}_{\ell'} \, (1 - w_{k\ell}^{t}) \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H}, t \in \mathcal{T}_\ell \cap \mathcal{T}_k^K, \ell' \in \Omega_{k\ell} \tag{2.11}$$

In these constraints, $\check{M}_{\ell'}$ is a big-M parameter. The minimum and maximum number of students attending a session must not be violated:

$$\hat{x}_{\ell j} \le \sum_{k \in \mathcal{K} \,:\, \ell \in \mathcal{A}_k} \hat{v}_{\ell j}^{k} \le c_\ell \, \hat{x}_{\ell j} \qquad \ell \in \mathcal{A} \setminus \mathcal{H}, j \in \mathcal{N}_\ell \tag{2.12}$$

$$\hat{x}_{\ell j} \le \sum_{k \in \mathcal{K} \,:\, \ell' \in \mathcal{A}_k} \check{v}_{\ell j}^{k\ell'} \le c_{\ell'} \, \hat{x}_{\ell j} \qquad \ell' \in \mathcal{A} \setminus \mathcal{H}, \ell \in \mathcal{B}_{\ell'}, j \in \mathcal{N}_\ell \text{ with } |\mathcal{B}'_\ell| = 2 \tag{2.13}$$

Constraints (2.13) are written for the double bubbles that are composed of two different lessons. Note that these constraints enforce the minimum and maximum number of students for each bubble of a double bubble. Consider a lesson $\ell \in \mathcal{B}$ such that $\mathcal{B}'_\ell$ is a singleton, *i.e.*, $|\mathcal{B}'_\ell| = 1$. A session $j \in \mathcal{N}_\ell$ of lesson ℓ should not be scheduled unless at least two students are assigned to that session. There is no reason to schedule such a double bubble if we want to train only one student. Such a student can be assigned to one session of lesson $\ell' \in \mathcal{B}'_\ell$ instead. Therefore, the following inequalities are introduced:

$$(c_{\ell'} + 1) \, \hat{x}_{\ell j} \le \sum_{k \in \mathcal{K} \,:\, \ell' \in \mathcal{A}_k} \check{v}_{\ell j}^{k\ell'} \le 2 \, c_{\ell'} \hat{x}_{\ell j} \qquad \ell' \in \mathcal{A} \setminus \mathcal{H}, \ell \in \mathcal{B}_{\ell'}, j \in \mathcal{N}_\ell \text{ with } |\mathcal{B}'_\ell| = 1 \tag{2.14}$$

Constraints (2.12)–(2.14) are written for the syllabus lessons. A shared lesson needs at least one pilot and one avwo student. Therefore, the following constraints are introduced:

$$\hat{x}_{\ell j} \le \sum_{k \in \mathcal{K}^P \,:\, \ell \in \mathcal{A}_k} \hat{v}_{\ell j}^{k} \le c_\ell^P \hat{x}_{\ell j} \qquad \ell \in \mathcal{H}, j \in \mathcal{N}_\ell \tag{2.15}$$

$$\hat{x}_{\ell j} \le \sum_{k \in \mathcal{K}^A \,:\, \ell \in \mathcal{A}_k} \hat{v}_{\ell j}^{k} \le c_\ell^A \hat{x}_{\ell j} \qquad \ell \in \mathcal{H}, j \in \mathcal{N}_\ell \tag{2.16}$$

$$\hat{x}_{\ell j} \le \sum_{k \in \mathcal{K}^P \,:\, \ell' \in \mathcal{A}_k} \check{v}_{\ell j}^{k\ell'} \le c_{\ell'}^P \hat{x}_{\ell j} \qquad \ell' \in \mathcal{H}, \ell \in \mathcal{B}_{\ell'}, j \in \mathcal{N}_\ell \text{ with } |\mathcal{B}'_\ell| = 2 \tag{2.17}$$

$$\hat{x}_{\ell j} \le \sum_{k \in \mathcal{K}^A \,:\, \ell' \in \mathcal{A}_k} \check{v}_{\ell j}^{k\ell'} \le c_{\ell'}^A \hat{x}_{\ell j} \qquad \ell' \in \mathcal{H}, \ell \in \mathcal{B}_{\ell'}, j \in \mathcal{N}_\ell \text{ with } |\mathcal{B}'_\ell| = 2 \tag{2.18}$$

$$(c_{\ell'}^P + 1) \, \hat{x}_{\ell j} \le \sum_{k \in \mathcal{K}^P \,:\, \ell' \in \mathcal{A}_k} \check{v}_{\ell j}^{k\ell'} \le 2 \, c_{\ell'}^P \hat{x}_{\ell j} \qquad \ell' \in \mathcal{H}, \ell \in \mathcal{B}_{\ell'}, j \in \mathcal{N}_\ell \text{ with } |\mathcal{B}'_\ell| = 1 \tag{2.19}$$

$$(c_{\ell'}^A + 1) \, \hat{x}_{\ell j} \le \sum_{k \in \mathcal{K}^A \,:\, \ell' \in \mathcal{A}_k} \check{v}_{\ell j}^{k\ell'} \le 2 \, c_{\ell'}^A \hat{x}_{\ell j} \qquad \ell' \in \mathcal{H}, \ell \in \mathcal{B}_{\ell'}, j \in \mathcal{N}_\ell \text{ with } |\mathcal{B}'_\ell| = 1 \tag{2.20}$$

The session-related constraints for the MCTP introduced in Section (4.1) are also valid for the updated formulation. However, we can introduce one additional set of constraints to ensure that the number of sessions of a syllabus lesson does not exceed its given upper bound due to existence of the shared lessons and double bubbles:

$$\begin{aligned} &\sum_{j \in \mathcal{N}_\ell} \hat{x}_{\ell j} + \sum_{\ell' \in \mathcal{B}_\ell} \sum_{j \in \mathcal{N}_{\ell'}} (3 - |\mathcal{B}'_{\ell'}|) \, \hat{x}_{\ell' j} \\ &+ \sum_{\ell' \in \mathcal{H}_\ell} \left(\sum_{j \in \mathcal{N}_{\ell'}} \hat{x}_{\ell' j} + \sum_{\ell'' \in \mathcal{B}_{\ell'}} \sum_{j \in \mathcal{N}_{\ell''}} (3 - |\mathcal{B}'_{\ell''}|) \, \hat{x}_{\ell'' j} \right) \le n_\ell \qquad \ell \in \mathcal{A} \setminus \mathcal{H} \end{aligned} \tag{2.21}$$

The resource-related constraints for the MCTP introduced in Section (4.3) are also valid for the updated formulation with the exception of Constraints (1.21) which are slightly modified. More precisely, Constraints (1.21) remain the same for the physical resources but modified for the instructors to address their one-time preparations:

$$\sum_{\ell \in \mathcal{L}^R \,:\, t \in \mathcal{T}_\ell, r \in \mathcal{R}_\ell^L} \sum_{j \in \mathcal{N}_\ell} u_{\ell j}^{rt} = \bar{u}_{rt} \qquad r \in \mathcal{R} \setminus \mathcal{R}^I, t \in \mathcal{T}_r^R \tag{2.22}$$

$$\sum_{\ell \in \mathcal{L}^R \,:\, t \in \mathcal{T}_\ell, r \in \mathcal{R}_\ell^L} \sum_{j \in \mathcal{N}_\ell} u_{\ell j}^{rt} + \hat{u}_{rt} = \bar{u}_{rt} \qquad r \in \mathcal{R}^I, t \in \mathcal{T}_r^R \tag{2.23}$$

According to Constraints (2.23), an instructor cannot teach a lesson when he/she is doing a one-time preparation. Note that the preparation phase has been removed from one-time prepared lessons since a student is not required to attend it. If an instructor does a one-time preparation for such a lesson, he/she is not required to do a one-time preparation for any other lessons in the same day. The time dedicated for a one-time preparation is two time slots. Therefore we have the following constraints to capture one-time preparations:

$$\sum_{t\in[a_d,b_d]\cap\mathcal{T}_r^R} \hat{u}_{rt} \le 2 \qquad d\in\mathcal{D}, r\in\mathcal{R}^I \tag{2.24}$$

$$2\left(\sum_{\ell\in\mathcal{L}^P:\, r\in\mathcal{R}_\ell^L, t\in\mathcal{T}_\ell} \sum_{j\in\mathcal{N}_\ell} u_{\ell j}^{rt}\right) \le \sum_{t'\in[a_d,t]\cap\mathcal{T}_r^R} \hat{u}_{rt'} \qquad d\in\mathcal{D}, r\in\mathcal{R}^I, t\in[a_d,b_d]\cap\mathcal{T}_r^R \tag{2.25}$$

$$\hat{u}_{rt} \le \hat{u}_{r,t+1} + \sum_{\ell\in\mathcal{L}^P:\, r\in\mathcal{R}_\ell^L, t+1\in\mathcal{T}_\ell} \sum_{j\in\mathcal{N}_\ell} u_{\ell j}^{r,t+1} \qquad d\in\mathcal{D}, r\in\mathcal{R}^I, \; t\colon [t,t+1]\subseteq[a_d,b_d]\cap\mathcal{T}_r^R \tag{2.26}$$

$$\hat{u}_{rt} = 0 \qquad d\in\mathcal{D}, r\in\mathcal{R}^I, t\in[a_d,b_d]\cap\mathcal{T}_r^R\colon \; t+1\notin[a_d,b_d]\cap\mathcal{T}_r^R \tag{2.27}$$

Constraints (2.24) ensure that at most two time slots are used for one-time preparation by an instructor per day. According to Constraints (2.25), an instructor must have dedicated two time slots for a one-time preparation before he/she can teach a one-time prepared lesson in the same day. Constraints (2.26) and (2.27) are added to ensure that appropriate time slots are dedicated for one-time preparation. According to Constraints (2.26), one-time preparation is performed by an instructor in two consecutive time slots immediately before the first one-time prepared lesson taught by the instructor. If the instructor is available at time t but not $t+1$, then Constraints (2.27) enforce that he/she will not perform a one-time preparation at time t.

As discussed in Section 5.4, some resources require a turnaround/setup time. The required setup time for such resources are imposed through the following constraints:

$$u_{\ell j}^{r,t+1} \ge u_{\ell j}^{rt} + \bar{u}_{r,t+1} - 1 \qquad \ell\in\mathcal{L}^R, j\in\mathcal{N}_\ell, r\in\mathcal{R}^S\cap\mathcal{R}_\ell^L, t\colon [t,t+1]\subseteq\mathcal{T}_\ell\cap\mathcal{T}_r^R \tag{2.28}$$

These constraints ensure that a setup-required resource is not used immediately one time slot after its employment in a session. Next we address aircraft configurations. We know that each aircraft can only have UTIL or ALFS configuration per day. An ALFS-required lesson can only be assigned to an ALFS aircraft. Similarly, a UTIL-required lesson can only be assigned to a UTIL aircraft. ANY-required lessons are not restricted by these constraints, so they may be allocated to either aircraft:

$$\sum_{\ell\in\mathcal{F}^U}\sum_{j\in\mathcal{N}_\ell}\sum_{t\in[a_d,b_d]\cap\mathcal{T}_r^R\cap\mathcal{T}_\ell} u_{\ell j}^{rt} \le M'_{rd}\,\check{u}_{rd} \qquad d\in\mathcal{D}, r\in\mathcal{R}^F \tag{2.29}$$

$$\sum_{\ell\in\mathcal{F}^A}\sum_{j\in\mathcal{N}_\ell}\sum_{t\in[a_d,b_d]\cap\mathcal{T}_r^R\cap\mathcal{T}_\ell} u_{\ell j}^{rt} \le M'_{rd}\,(1-\check{u}_{rd}) \qquad d\in\mathcal{D}, r\in\mathcal{R}^F \tag{2.30}$$

The coefficient M'_{rd} in these constraints is a big-M parameter. Finally, we address the weekly limits for each resource. We have a maximum of β_r^R time slots per week for resource $r\in\mathcal{R}$. Since the planning horizon is at most one week, we can simply add the following constraints to impose this limit:

$$\sum_{t\in\mathcal{T}_r^R} \bar{u}_{rt} \le \beta_r^R \qquad r\in\mathcal{R} \tag{2.31}$$

We also have a maximum of γ_r^R sorties a week for instructor $r\in\mathcal{R}$:

$$\sum_{\ell\in\mathcal{A}\cap\mathcal{L}^S}\sum_{j\in\mathcal{N}_\ell}\sum_{g\in\mathcal{G}'_\ell:\, r\in\mathcal{R}_{\ell g}^G} z_{\ell j}^{rg} + 2\sum_{\ell\in\mathcal{B}\cap\mathcal{L}^S}\sum_{j\in\mathcal{N}_\ell}\sum_{g\in\mathcal{G}'_\ell:\, r\in\mathcal{R}_{\ell g}^G} z_{\ell j}^{rg} \le \gamma_r^R \qquad r\in\mathcal{R}^I \tag{2.32}$$

Each bubble of a double bubble is considered as one sortie. Since the same set of resources is used in two bubbles of a double bubble, a double bubble is counted twice for each instructor.

In summary, formulation of the EMCTP, *F2*, consists of (2.01) and the following sets of constraints: (1.02)–(1.05), (1.15)–(1.20), (1.22)–(1.27) and (2.02)–(2.32). Observe that *F2* reduces to *F1* if $\mathcal{H}=\mathcal{B}=\emptyset$ and the additional set of constraints (due to one-time preparation, setup times, etc) are removed. In addition, *F2* captures all of the requirements of the real-world problem under study. Therefore, we will study this formulation in the remainder of this paper.

6 Branch and Check

It is difficult to solve *F2* in practice due to the symmetry in the numbering of the sessions. Although Constraints (1.05) alleviate the impact of the symmetry, our computational experiments indicate that even daily instances cannot be solved efficiently in the presence of these symmetry breaking constraints. Of course, this is not surprising since the same issue has been observed for other classes of optimization problems (see *e.g.*, Toth and Vigo, 2014, regarding the vehicle-index formulations of routing problems). In order to completely remove the symmetry in the numbering of the sessions, we develop another formulation for the EMCTP. We refer to this new formulation as the *relaxed formulation* since it is a relaxation of *F2*. As will be discussed in Section 6.2, the relaxed formulation can provide infeasible solutions for the EMCTP in the presence of cohort lessons. In such a case, the solution provided by the relaxed formulation cannot be converted into a feasible solution of *F2*.

We propose B&CH to solve the EMCTP using the relaxed formulation. Each time a feasible solution of the relaxed formulation is obtained, we check whether conversion of this solution into a feasible solution of *F2* is possible. If not, a Benders feasibility cut is added to the relaxed formulation to remove the infeasible solution. Recall that B&CH is an implementation of the logic-based Benders decomposition in which Benders cuts are generated within a B&B algorithm. This approach is generally implemented by employing the so-called Lazy-Constraint Callback, a feature in modern optimization solvers that allows introducing new constraints within a single B&B tree. A lazy callback is called whenever an integer feasible solution of the relaxed formulation is found. This integer solution is not accepted as a feasible solution unless another feasibility check is performed in the callback to confirm its feasibility. If the integer solution can be converted into a feasible solution of *F2*, then this solution is finally accepted as an integer feasible solution. Otherwise, one or more violated cuts are identified and added to remove this infeasible solution from the relaxed formulation. As a result, any optimal solution reported by the solver leads to an optimal feasible solution for the EMCTP. Next we introduce the relaxed formulation.

6.1 The Relaxed Formulation

The relaxed formulation removes the notion of session-index by taking advantage of new decision variables and constraints. The formulation also employs some of the existing decision variables and constraints of *F2*. The existing decision variables that are used in the relaxed formulation are $v_{k\ell}$, $\bar{w}_{kt}$, $w^t_{k\ell}$, $\bar{u}_{rt}$, $\hat{u}_{rt}$ and $\check{u}_{rt}$. The new decision variables are defined as follows. Let $\hat{s}^k_{\ell t}$ be a binary variable which indicates whether student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{A}_k$ by attending a session of ℓ that starts at time $t \in \mathcal{T}'_\ell$. Note that more than one session of lesson ℓ may start at time t. In this case, $\hat{s}^k_{\ell t}$ takes the value one iff student k attends exactly one of these sessions and it takes the value zero otherwise. For the sake of exposition, we use $\mathcal{J}_{\ell t}$ to denote the set of sessions of lesson $\ell \in \mathcal{L}$ that begin at time $t \in \mathcal{T}'_\ell$. We sometimes refer to an arbitrary element of this set as session $\mathcal{J}_{\ell t}$. The relaxed formulation does not distinguish between different sessions of $\mathcal{J}_{\ell t}$ in order to remove the symmetry in the numbering of the sessions. Constructing individual sessions of each session $\mathcal{J}_{\ell t}$ and allocating students and resources to each session is performed after solving the relaxed formulation (in a post-processing).

The binary decision variable $\check{s}^{k\ell}_{\ell' t}$ is defined to indicate whether student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{A}_k$ by attending a session of lesson $\ell' \in \mathcal{B}_\ell$ that starts at time $t \in \mathcal{T}'_{\ell'}$. We also define the general integer variable $m_{\ell t} \in \{0, \dots, n_\ell\}$ which represents the number of sessions of lesson $\ell \in \mathcal{L}$ that begin at time $t \in \mathcal{T}'_\ell$, that is, $m_{\ell t} = |\mathcal{J}_{\ell t}|$. Finally, in order to capture resource allocations and groups of a lesson, we define the binary decision variable $\hat{z}^{rg}_{\ell t}$ which takes the value of one iff group $g \in \mathcal{G}_\ell$ of session $\mathcal{J}_{\ell t}$ ($\ell \in \mathcal{L}^R$, $t \in \mathcal{T}'_\ell$) uses resource $r \in \mathcal{R}^G_{\ell g}$. Observe that $\hat{z}^{rg}_{\ell t}$ is defined as a binary variable since a resource cannot appear in the same group for more than one session of $\mathcal{J}_{\ell t}$. A summary of the decision variables used to develop the relaxed formulation can be found in Appendix F.

Let $\check{\delta}_\ell := \sum_{p'=1}^{\min \mathcal{P}'_\ell - 1} \delta_{\ell p'}$ and $\hat{\delta}_\ell := \sum_{p'=1}^{\max \mathcal{P}'_\ell} \delta_{\ell p'} - 1$ for lesson $\ell \in \mathcal{L}$. For the sake of compactness, we introduce the set $\mathcal{T}''_{\ell t} := [t - \hat{\delta}_\ell, t - \check{\delta}_\ell]$ to denote the set of time slots that a session of lesson $\ell \in \mathcal{L}$ must start for a student to be present in this session at time $t \in \mathcal{T}_\ell$. We also define $\mathcal{T}^G_{\ell g t}$ to denote the set of time slots in which a resource from lesson-group $g \in \mathcal{G}_\ell$ of lesson $\ell \in \mathcal{L}^R$ is required to be present in session $\mathcal{J}_{\ell t}$ for $t \in \mathcal{T}'_\ell$. The relaxed formulation can now be expressed as follows.

$$
\begin{aligned}
F3: \max \ & \sum_{k \in \mathcal{K}} \pi_k \sum_{\ell \in \mathcal{A}_k \setminus \mathcal{H}} v_{k\ell} \\
\text{s.t. } & (1.22), (1.23), (1.24), (1.25), (1.26), (1.27), \\
& (2.03), (2.10), (2.11), (2.24), (2.27), (2.31), \text{ and} \\
& \sum_{t \in \mathcal{T}'_\ell} \hat{s}^k_{\ell t} + \sum_{\ell' \in \mathcal{B}_\ell} \sum_{t \in \mathcal{T}'_{\ell'}} \check{s}^{k\ell}_{\ell' t}
\end{aligned}
$$

$$+\sum_{\ell'\in\mathcal{H}_\ell}\left(\sum_{t\in\mathcal{T}'_{\ell'}}\hat{s}^k_{\ell' t}+\sum_{\ell''\in\mathcal{B}_{\ell'}}\sum_{t\in\mathcal{T}'_{\ell''}}\check{s}^{k\ell'}_{\ell'' t}\right)=v_{k\ell} \qquad k\in\mathcal{K},\ell\in\mathcal{A}_k\setminus\mathcal{H} \tag{3.01}$$

$$\sum_{t\in\mathcal{T}'_\ell}\Delta'_\ell\,\hat{s}^k_{\ell t}+\sum_{\ell'\in\mathcal{B}_\ell}\sum_{t\in\mathcal{T}'_{\ell'}}\Delta'_{\ell'}\,\check{s}^{k\ell}_{\ell' t}$$

$$+\sum_{\ell'\in\mathcal{H}_\ell}\left(\sum_{t\in\mathcal{T}'_{\ell'}}\Delta'_{\ell'}\,\hat{s}^k_{\ell' t}+\sum_{\ell''\in\mathcal{B}_{\ell'}}\sum_{t\in\mathcal{T}'_{\ell''}}\Delta'_{\ell''}\,\check{s}^{k\ell'}_{\ell'' t}\right)$$

$$\le\sum_{t\in\mathcal{T}_\ell\cap\mathcal{T}_k^K}w^t_{k\ell} \qquad k\in\mathcal{K},\ell\in\mathcal{A}_k\setminus\mathcal{H} \tag{3.02}$$

$$\sum_{t'\in\mathcal{T}'_\ell\cap\mathcal{T}''_{\ell t}}\hat{s}^k_{\ell t'}+\sum_{\ell'\in\mathcal{B}_\ell}\sum_{t'\in\mathcal{T}'_{\ell'}\cap\mathcal{T}''_{\ell' t}}\check{s}^{k\ell}_{\ell' t'}$$

$$+\sum_{\ell'\in\mathcal{H}_\ell}\left(\sum_{t'\in\mathcal{T}'_{\ell'}\cap\mathcal{T}''_{\ell' t}}\hat{s}^k_{\ell' t'}+\sum_{\ell''\in\mathcal{B}_{\ell'}}\sum_{t'\in\mathcal{T}'_{\ell''}\cap\mathcal{T}''_{\ell'' t}}\check{s}^{k\ell'}_{\ell'' t'}\right)\ge w^t_{k\ell} \qquad k\in\mathcal{K},\ell\in\mathcal{A}_k\setminus\mathcal{H},t\in\mathcal{T}_\ell\cap\mathcal{T}_k^K \tag{3.03}$$

$$\sum_{t\in\mathcal{T}'_\ell}m_{\ell t}+\sum_{\ell'\in\mathcal{B}_\ell}\sum_{t\in\mathcal{T}'_{\ell'}}(3-|\mathcal{B}'_{\ell'}|)\,m_{\ell' t}$$

$$+\sum_{\ell'\in\mathcal{H}_\ell}\left(\sum_{t\in\mathcal{T}'_{\ell'}}m_{\ell' t}+\sum_{\ell''\in\mathcal{B}_{\ell'}}\sum_{t\in\mathcal{T}'_{\ell''}}(3-|\mathcal{B}'_{\ell''}|)\,m_{\ell'' t}\right)\le n_\ell \qquad \ell\in\mathcal{A}\setminus\mathcal{H} \tag{3.04}$$

$$\sum_{t\in\mathcal{T}'_\ell}t\,\hat{s}^k_{\ell t}=\sum_{t\in\mathcal{T}'_\ell}t\,\hat{s}^{k'}_{\ell t} \qquad k\in\mathcal{K},k'\in\mathcal{K}_k^C\colon k'<k,$$

$$\ell\in\mathcal{L}^C\cap\mathcal{A}_k\cap\mathcal{A}_{k'} \tag{3.05}$$

$$m_{\ell t}\le\sum_{k\in\mathcal{K}\colon\,\ell\in\mathcal{A}_k}\hat{s}^k_{\ell t}\le c_\ell\,m_{\ell t} \qquad \ell\in\mathcal{A}\setminus\mathcal{H},t\in\mathcal{T}'_\ell \tag{3.06}$$

$$m_{\ell t}\le\sum_{k\in\mathcal{K}\colon\,\ell'\in\mathcal{A}_k}\check{s}^{k\ell'}_{\ell t}\le c_{\ell'}\,m_{\ell t} \qquad \ell'\in\mathcal{A}\setminus\mathcal{H},\ell\in\mathcal{B}_{\ell'},$$

$$t\in\mathcal{T}'_\ell\text{ with }|\mathcal{B}'_\ell|=2 \tag{3.07}$$

$$(c_{\ell'}+1)\,m_{\ell t}\le\sum_{k\in\mathcal{K}\colon\,\ell'\in\mathcal{A}_k}\check{s}^{k\ell'}_{\ell t}\le 2\,c_{\ell'}m_{\ell t} \qquad \ell'\in\mathcal{A}\setminus\mathcal{H},\ell\in\mathcal{B}_{\ell'},$$

$$t\in\mathcal{T}'_\ell\text{ with }|\mathcal{B}'_\ell|=1 \tag{3.08}$$

$$m_{\ell t}\le\sum_{k\in\mathcal{K}^P\colon\,\ell\in\mathcal{A}_k}\hat{s}^k_{\ell t}\le c^P_\ell\,m_{\ell t} \qquad \ell\in\mathcal{H},t\in\mathcal{T}'_\ell \tag{3.09}$$

$$m_{\ell t}\le\sum_{k\in\mathcal{K}^A\colon\,\ell\in\mathcal{A}_k}\hat{s}^k_{\ell t}\le c^A_\ell m_{\ell t} \qquad \ell\in\mathcal{H},t\in\mathcal{T}'_\ell \tag{3.10}$$

$$m_{\ell t}\le\sum_{k\in\mathcal{K}^P\colon\,\ell'\in\mathcal{A}_k}\check{s}^{k\ell'}_{\ell t}\le c^P_{\ell'}m_{\ell t} \qquad \ell'\in\mathcal{H},\ell\in\mathcal{B}_{\ell'},$$

$$t\in\mathcal{T}'_\ell\text{ with }|\mathcal{B}'_\ell|=2 \tag{3.11}$$

$$m_{\ell t}\le\sum_{k\in\mathcal{K}^A\colon\,\ell'\in\mathcal{A}_k}\check{s}^{k\ell'}_{\ell t}\le c^A_{\ell'}m_{\ell t} \qquad \ell'\in\mathcal{H},\ell\in\mathcal{B}_{\ell'},$$

$$t\in\mathcal{T}'_\ell\text{ with }|\mathcal{B}'_\ell|=2 \tag{3.12}$$

$$(c^P_{\ell'}+1)\,m_{\ell t}\le\sum_{k\in\mathcal{K}^P\colon\,\ell'\in\mathcal{A}_k}\check{s}^{k\ell'}_{\ell t}\le 2\,c^P_{\ell'}m_{\ell t} \qquad \ell'\in\mathcal{H},\ell\in\mathcal{B}_{\ell'},$$

$$t\in\mathcal{T}'_\ell\text{ with }|\mathcal{B}'_\ell|=1 \tag{3.13}$$

$$(c^A_{\ell'}+1)\,m_{\ell t}\le\sum_{k\in\mathcal{K}^A\colon\,\ell'\in\mathcal{A}_k}\check{s}^{k\ell'}_{\ell t}\le 2\,c^A_{\ell'}m_{\ell t} \qquad \ell'\in\mathcal{H},\ell\in\mathcal{B}_{\ell'},$$

$$t\in\mathcal{T}'_\ell\text{ with }|\mathcal{B}'_\ell|=1 \tag{3.14}$$

$$\sum_{r\in\mathcal{R}^G_{\ell g}}\hat{z}^{rg}_{\ell t}=m_{\ell t} \qquad \ell\in\mathcal{L}^R,t\in\mathcal{T}'_\ell,g\in\mathcal{G}_\ell \tag{3.15}$$

$$\sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}} \hat{z}^{rg}_{\ell t} \le 1 \qquad \ell\in\mathcal{L}^R, t\in\mathcal{T}'_\ell, r\in\mathcal{R}^L_\ell \tag{3.16}$$

$$\sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}} \hat{z}^{rg}_{\ell t} = \sum_{g\in\mathcal{G}_\ell:\, r'\in\mathcal{R}^G_{\ell g}} \hat{z}^{r'g}_{\ell t} \qquad \ell\in\mathcal{L}^R, t\in\mathcal{T}'_\ell, (r,r')\in\mathcal{R}^P_\ell \tag{3.17}$$

$$\sum_{\ell\in\mathcal{L}^R:\, r\in\mathcal{R}^L_\ell}\ \sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}}\ \sum_{t'\in\mathcal{T}'_\ell:\, t\in\mathcal{T}^G_{\ell g t'}} \hat{z}^{rg}_{\ell t'} = 0 \qquad r\in\mathcal{R}, t\in\mathcal{T}\setminus\mathcal{T}^R_r \tag{3.18}$$

$$\sum_{\ell\in\mathcal{L}^R:\, r\in\mathcal{R}^L_\ell}\ \sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}}\ \sum_{t'\in\mathcal{T}'_\ell:\, t\in\mathcal{T}^G_{\ell g t'}} \hat{z}^{rg}_{\ell t'} = \bar{u}_{rt} \qquad r\in\mathcal{R}\setminus\mathcal{R}^I, t\in\mathcal{T}^R_r \tag{3.19}$$

$$\sum_{\ell\in\mathcal{L}^R:\, r\in\mathcal{R}^L_\ell}\ \sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}}\ \sum_{t'\in\mathcal{T}'_\ell:\, t\in\mathcal{T}^G_{\ell g t'}} \hat{z}^{rg}_{\ell t'} + \hat{u}_{rt} = \bar{u}_{rt} \qquad r\in\mathcal{R}^I, t\in\mathcal{T}^R_r \tag{3.20}$$

$$\hat{z}^{rg}_{\ell t'} + \bar{u}_{rt} \le 1 \qquad \ell\in\mathcal{L}^R, g\in\mathcal{G}_\ell, r\in\mathcal{R}^S\cap\mathcal{R}^G_{\ell g}, t'\in\mathcal{T}'_\ell, t\in\mathcal{T}^R_r\cap\{1+\max\mathcal{T}^G_{\ell g t'}\} \tag{3.21}$$

$$2\sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}} \hat{z}^{rg}_{\ell t} \le \sum_{t'\in[a_d,t]\cap\mathcal{T}^R_r} \hat{u}_{rt'} \qquad \ell\in\mathcal{L}^P, r\in\mathcal{R}^I\cap\mathcal{R}^L_\ell, d\in\mathcal{D}, t\in[a_d,b_d]\cap\mathcal{T}'_\ell \tag{3.22}$$

$$\hat{u}_{rt} \le \hat{u}_{r,t+1} + \sum_{\ell\in\mathcal{L}^P:\, r\in\mathcal{R}^L_\ell, t+1\in\mathcal{T}'_\ell}\ \sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}} \hat{z}^{rg}_{\ell,t+1} \qquad d\in\mathcal{D}, r\in\mathcal{R}^I, t\colon [t,t+1]\subseteq[a_d,b_d]\cap\mathcal{T}^R_r \tag{3.23}$$

$$\sum_{\ell\in\mathcal{A}\cap\mathcal{L}^S}\ \sum_{t\in\mathcal{T}'_\ell}\ \sum_{g\in\mathcal{G}'_\ell:\, r\in\mathcal{R}^G_{\ell g}} \hat{z}^{rg}_{\ell t} + 2\sum_{\ell\in\mathcal{B}\cap\mathcal{L}^S}\ \sum_{t\in\mathcal{T}'_\ell}\ \sum_{g\in\mathcal{G}'_\ell:\, r\in\mathcal{R}^G_{\ell g}} \hat{z}^{rg}_{\ell t} \le \gamma^R_r \qquad r\in\mathcal{R}^I \tag{3.24}$$

$$\sum_{\ell\in\mathcal{F}^U}\ \sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}}\ \sum_{t\in\mathcal{T}'_\ell\cap[a_d,b_d]} \hat{z}^{rg}_{\ell t} \le M''_{rd}\,\check{u}_{rd} \qquad d\in\mathcal{D}, r\in\mathcal{R}^F \tag{3.25}$$

$$\sum_{\ell\in\mathcal{F}^A}\ \sum_{g\in\mathcal{G}_\ell:\, r\in\mathcal{R}^G_{\ell g}}\ \sum_{t\in\mathcal{T}'_\ell\cap[a_d,b_d]} \hat{z}^{rg}_{\ell t} \le M''_{rd}\,(1-\check{u}_{rd}) \qquad d\in\mathcal{D}, r\in\mathcal{R}^F \tag{3.26}$$

As can be seen, *F3*, maximizes the same objective function as *F2*. The Constraint set (3.01) is equivalent to Constraint set (2.02) and it ensures that a student attends exactly one session in order to take a particular syllabus lesson. Constraints (3.02) and (3.03) establish the logical relation between the decision variables $\boldsymbol{s}$ and $\boldsymbol{w}$. They ensure that a student who attends a session is present in appropriate time slots when the session is in progress. Constraints (3.04) and (3.05) play the same role as Constraints (2.21) and (2.04), respectively. In particular, Constraints (3.05) ensure that students who are in the same cohort are allocated to the same session $\mathcal{J}_{\ell t}$. However, these constraints alone do not guarantee feasibility of the schedule with respect to the cohort lessons. In Section 6.2, we will explain why the infeasibility can occur and how to address it.

Constraint sets (3.06)-(3.14) ensure that appropriate number of students are allocated to sessions of different lessons. These constraints are respectively equivalent to Constraints (2.12)-(2.20). The rest of the constraints represent business requirements related to physical and human resources. Constraints (3.15) and (3.16) ensure that one resource is allocated to each group of session $\mathcal{J}_{\ell t}$. Note that Constraint set (3.16) states that a resource cannot be allocated to more than one group of session $\mathcal{J}_{\ell t}$. As discussed in Section 2.1, if a resource appears in more than one group of a lesson, then such groups must share at least one phase. As a result, a resource cannot be allocated to different groups of session $\mathcal{J}_{\ell t}$ whether in the same session or not.

Constraints (3.17) are equivalent to Constraints (1.17) and they ensure compatibility of the resources assigned to a lesson. Constraints (3.18) state that a resource cannot be allocated to a session when it is not available. Note that such resource availability conditions could also be imposed when defining the $\boldsymbol{z}$ decision variables (as in *e.g.*, $\bar{\boldsymbol{u}}$ variables). For the sake of readability, we enforce them through adding these constraints in order to avoid making the definition of $\boldsymbol{z}$ unnecessarily complex. Constraint sets (3.19) and (3.20) are respectively equivalent to Constraint sets (2.22) and (2.23). Constraints (3.21) ensure that the turnaround time requirements are met. In these constraints, $\max\mathcal{T}^G_{\ell g t'}$ is the last time

slot in which the resource is used in the session if the session starts at time t'. The next time slot will be unavailable for this resource.

Constraint sets (3.22) and (3.23) are respectively equivalent to Constraint sets (2.25) and (2.26). According to these constraints, the instructor performs a one-time preparation right before the start of the first one-time prepared lesson that he/she is going to teach in the day. Note that the set of lesson groups and the set of phases for a one-time prepared lesson are singletons, that is, $|\mathcal{G}_\ell| = |\mathcal{G}'_\ell| = 1$ and $|\mathcal{P}_\ell| = |\mathcal{P}'_\ell| = 1$ for $\ell \in \mathcal{L}^P$.

Constraints (3.24) are equivalent to Constraints (2.32) and they ensure that the maximum number of sorties per week is not violated for an instructor. Constraint sets (3.25) and (3.26) are respectively equivalent to Constraint sets (2.29) and (2.30). They guarantee that aircraft configuration requirements are not violated. In these constraints, M''_{rd} is a big-M parameter. As mentioned earlier, *F3* is not a complete formulation, that is, its solution may not yield a feasible solution for the EMCTP. Next we explain why the infeasibility occurs and how to resolve it.

6.2 The Source of Infeasibility

Around one third of the syllabus lessons in the pilot training program under study are cohort lessons. The infeasibility of the relaxed formulation can occur in the presence of the cohort lessons. We give an example to explain why *F3* may provide an infeasible solution for the original problem. Consider a cohort lesson $\ell \in \mathcal{L}^C$ with capacity $c_\ell = 4$ and three cohorts that respectively consist of 3, 3 and 2 students. A feasible solution of *F3* may allocate these students to $\mathcal{J}_{\ell t}$ for some $t \in \mathcal{T}'_\ell$ such that $|\mathcal{J}_{\ell t}| = m_{\ell t} = 2$. Although 2 sessions are enough to accommodate all students, there is no way to allocate same-cohort students to the same session. In other words, at least 3 sessions are required to accommodate these students. The value 3 is indeed the optimal objective value of a bin packing problem with three items (cohorts) in which the capacity of each bin is c_ℓ.

To resolve this issue, we employ logic-based Benders decomposition. Each time a feasible solution of *F3* is obtained, we check if the solution is feasible with respect to the cohorts. If not, a feasibility cut is introduced to remove the infeasible solution. In order to check the feasibility, we introduce a feasibility problem which is solved for each $\mathcal{J}_{\ell t}$ such that $m_{\ell t} > 1$ and $\ell \in \mathcal{L}^C$. Note that the infeasibility does not occur if $m_{\ell t} \leq 1$. Let η denote the number of cohorts of students attending $\mathcal{J}_{\ell t}$ and $\theta_1, \ldots, \theta_\eta$ denote the size of these cohorts, respectively. In addition, we define the binary decision variable φ_{ij} which indicates whether cohort $i \in \{1, \ldots, \eta\}$ is allocated to session (bin) $j \in \{1, \ldots, m_{\ell t}\}$. The feasibility problem is the decision version of the bin packing problem and it can be expressed as follows:

$$\min \ 0$$

$$\text{s.t.} \sum_{j=1}^{m_{\ell t}} \varphi_{ij} = 1 \qquad i = 1, \ldots, \eta \tag{4.01}$$

$$\sum_{i=1}^{\eta} \theta_i \varphi_{ij} \leq c_\ell \qquad j = 1, \ldots, m_{\ell t} \tag{4.02}$$

$$\sum_{i=1}^{\eta} \varphi_{ij} \geq 1 \qquad j = 1, \ldots, m_{\ell t} \tag{4.03}$$

If this problem is feasible, then we can feasibly convert a solution of *F3* into a feasible solution of the EMCTP. Otherwise, we add a set of feasibility cuts to the relaxed formulation (aka the master problem). Let $j' \coloneqq m_{\ell t}$ denote the number of sessions of $\mathcal{J}_{\ell t}$ according to the current infeasible solution. We also define the following sets according to the current infeasible solution:

$$\mathcal{M} \coloneqq \{(\ell, t) \colon \ell \in \mathcal{L}^C, t \in \mathcal{T}'_\ell, m_{\ell t} > 1\}$$
$$\mathcal{S}^0_{\ell t} \coloneqq \{k \in \mathcal{K} \colon \ell \in \mathcal{A}_k, \hat{s}^k_{\ell t} = 0\}$$
$$\mathcal{S}^1_{\ell t} \coloneqq \{k \in \mathcal{K} \colon \ell \in \mathcal{A}_k, \hat{s}^k_{\ell t} = 1\}$$

Our proposed feasibility cuts employ a new binary decision variable, denoted $q^j_{\ell t}$, which is positive iff $m_{\ell t} = j$. The feasibility cuts have the following form:

$$\sum_{j \in \mathcal{N}_\ell} j q^j_{\ell t} = m_{\ell t} \qquad (\ell, t) \in \mathcal{M} \tag{5.01}$$

$$\sum_{j \in \mathcal{N}_\ell} q^j_{\ell t} \leq 1 \qquad (\ell, t) \in \mathcal{M} \tag{5.02}$$

$$\sum_{k \in \mathcal{S}^1_{\ell t}} (1 - \hat{s}^k_{\ell t}) + \sum_{k \in \mathcal{S}^0_{\ell t}} \hat{s}^k_{\ell t} \geq q^{j'}_{\ell t} \qquad (\ell, t) \in \mathcal{M} \tag{5.03}$$

Constraints (5.01) and (5.02) establish the value of the variable $q^j_{\ell t}$ while Constraint (5.03) eliminates the current infeasible solution by changing the value of $m_{\ell t}$. Each time $\mathcal{J}_{\ell t}$ is infeasible, we add a set of Constraints (5). More precisely, exactly 3 constraints are added for each $(\ell, t) \in \mathcal{M}$. Although it is possible to add feasibility cuts for only one $\mathcal{J}_{\ell t}$ to remove the current infeasible solution, we add these cuts for every infeasible $\mathcal{J}_{\ell t}$ in the current solution to improve the convergence of the algorithm. Once *F3* is solved, we can construct individual sessions of each lesson in a post-processing.

6.3 Enhancement Techniques

We further improve our proposed B&CH approach by introducing three different enhancement techniques. First, we present a fast matheuristic for the EMCTP which provides high quality solutions for the problem. This solution is used as an initial solution for the solver which results in faster termination of the B&CH algorithm. A good initial feasible solution is one that provides a good initial lower bound for the relaxed formulation and therefore optimization terminates faster. Note that timetabling problems are infamous for feasibility issues, that is, finding even a feasible solution is sometimes a very difficult task, let alone finding good feasible solutions (see *e.g.,* Dorneles et al., 2014; Gonzalez et al., 2018; Akbarzadeh and Maenhout, 2020, and the references therein). Second, we show that the precedence constraints can be reformulated in a more compact way which results in a strengthened formulation. Finally, we present a pre-processing technique that eliminates a significant number of the decision variables of the relaxed formulation.

6.3.1 Improving the Initial Lower Bound

We can improve the run time of the solver by inserting an initial feasible solution. To be effective, we need a fast heuristic that can obtain high quality solutions quickly. The formulations presented in this paper are all time-index formulations. This implies that the size of each formulation is directly affected by the planning horizon. This is probably the most challenging issue when it comes to solving the problem for large planning horizons in practice. In this section, we introduce a fix-and-optimize algorithm that yields high quality solutions for the problem.

Matheuristics constitute a highly promising class of heuristics for timetabling problems. An interesting aspect of these algorithms is that they benefit from the recent and future advances in optimization solvers (Lindahl et al., 2018). They have also proved to be effective in terms of the quality of the solutions. The matheuristic algorithms proposed in Dorneles et al. (2014) and Fonseca et al. (2016) each result in improving best known solutions of the high school timetabling data sets in the literature. Of interest are the alternative decomposition ideas proposed in Dorneles et al. (2014) to construct a fix-and-optimize algorithm, namely, "class decomposition", "teacher decomposition" and "day decomposition". In the last variant, a certain number of days are free to be optimized, meaning that all variables corresponding to those days are not fixed. The fix-and-optimize algorithm described in this section generalizes this idea to a "time decomposition" variant that further decomposes a one-day planning problem. Our designation of the subset and sequence of the variables to be fixed also results in solving a sequence of feasible subproblems.

A fix-and-optimize algorithm based on time decomposition can be formally presented for any 'buckets' of time (*e.g.,* day, half a day, etc). The algorithm constructs and solves a sequence of subproblems corresponding to each bucket in the planning horizon. It greedily allocates as many lessons as possible to the students starting from the first bucket and proceeding to the last. Considering a bucket as one planning day, for instance, a problem with $D > 1$ days is decomposed into D smaller subproblems. The solution obtained at iteration $d = 1, \ldots, D$ of the algorithm is fixed in the remaining iterations $d' = d + 1, \ldots, D$. The greedy behavior of the algorithm follows from the fact that the subproblem solved at each iteration optimizes the same objective function as in the original problem except that the planning horizon is reduced to just one bucket. In fact, in the special case when a bucket is one day, the corresponding subproblem can be reduced to one small instance of the problem. The algorithm owes this interesting property to the rolling structure of the problem.

Each subproblem can be constructed from the relaxed formulation and solved through the proposed B&CH approach. We explicitly construct an instance of the problem (followed by generating a new formulation from scratch) for each one-day bucket. Constructing such a subproblem includes setting the planning horizon to one bucket, that is, $\mathcal{D} = \{d\}$ and $\mathcal{T} = [a_d, b_d]$. Furthermore, relevant parameters of the problem such as the set of lessons for each student, the duty rules, the weekly limits and the availability of the students and resources are updated according to the solutions obtained for the previous

buckets. For instance, a particular lesson is removed from the set of lessons for a students if this lesson has been allocated to this student in the previous buckets. Similarly, the weekly usage of a resource is reduced from its nominal value by 10 time slots if this resource is used for 10 time slots in the previous buckets. In addition, applying the minimum rest time for a particular student is as easy as updating his/her availability set based on the solution obtained in the previous iteration. Other parameters of the problem are updated similarly.

For buckets that are within a day (*e.g.,* two buckets each half a day), we first construct one instance of the problem corresponding to that day as discussed in the previous paragraph. The relaxed formulation generated for this instance is used to solve all relevant subproblems. This means that subproblems corresponding to each bucket are obtained by fixing decision variables corresponding to other buckets in the formulation. Variables related to the previous buckets are fixed to the solutions obtained from their corresponding subproblems, and variables corresponding to the subsequent buckets are fixed to zero. The algorithm guarantees to produce a feasible solution for the problem since the solution obtained at each iteration of the algorithm is feasible. If a bucket is too small, then no lesson will be scheduled in that bucket, resulting in an objective value of zero. Subproblems with smaller buckets are solved relatively faster than those with larger buckets. However, the quality of the solutions generated by the algorithm is expected to be good if buckets are sufficiently large. Since there are usually multiple optimal solutions for a subproblem, we optimize a secondary objective for each subproblem to further improve the quality of the overall solution.

Each time we solve a subproblem, we save its objective value. In the next step, we change the objective function of this subproblem such that it minimizes the sum of the start times of all lessons in that subproblem. Furthermore, we add a constraint to the subproblem to bound the original (primary) objective function of the problem to the objective value obtained in the previous step. Due to this lexicographic approach, more lessons might be scheduled in the buckets within a day. The algorithm also benefits from this approach for instances with multiple days due to the minimum rest time duty rule (*i.e.,* students and resources can likely start earlier next day). At each iteration, the solution obtained from this lexicographic approach is fixed and then the algorithm proceeds to the next iteration. Overall, this algorithm is highly effective for solving daily and weekly instances of the problem as will be shown in Section 7.

6.3.2 Reformulating the Precedence Constraints

It is possible to tighten the formulation by reformulating precedence constraints (2.11). In practice, the proposed reformulation also reduces the total number of the required constraints since (2.11) will be removed from the formulation. Let $\tau_{k\ell}$ be an unrestricted decision variable representing the first time slot at which student $k \in \mathcal{K}$ is in a session to take the syllabus lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$. This can be a session of ℓ or a session of a shared lesson $\ell' \in \mathcal{H}_\ell$ or a double-bubble $\ell'' \in \mathcal{B}_\ell \cup \mathcal{B}_{\ell'}$. Furthermore, we let $\tau_{k\ell} = T + 1$ if student k does not take the lesson ℓ. Then we can enforce the precedence relations by adding the following constraints to the relaxed formulation:

$$\sum_{t \in \mathcal{T}'_\ell} (t + \check{\delta}_\ell)\, \hat{s}^k_{\ell t} + \sum_{\ell' \in \mathcal{B}_\ell} \sum_{t \in \mathcal{T}'_{\ell'}} (t + \check{\delta}_{\ell'})\, \check{s}^{k\ell}_{\ell' t} + (T+1)(1 - v_{k\ell})$$

$$+ \sum_{\ell' \in \mathcal{H}_\ell} \left(\sum_{t \in \mathcal{T}'_{\ell'}} (t + \check{\delta}_{\ell'})\, \hat{s}^k_{\ell' t} + \sum_{\ell'' \in \mathcal{B}_{\ell'}} \sum_{t \in \mathcal{T}'_{\ell''}} (t + \check{\delta}_{\ell''})\, \check{s}^{k\ell'}_{\ell'' t} \right) = \tau_{k\ell} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H} \tag{6}$$

$$\tau_{k\ell} \leq \tau_{k\ell'} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H}, \ell' \in \Omega_{k\ell} \tag{7}$$

Constraints (6) capture the value of the decision variables $\tau_{k\ell}$ while Constraints (7) ensure that a student does not start a lesson before its prerequisites. Constraints (7) properly address the precedence relations since a student cannot take multiple lessons at the same time. These constraints can be further improved as follows.

$$\tau_{k\ell} + \Delta'_\ell\, v_{k\ell'} \leq \tau_{k\ell'} \qquad k \in \mathcal{K}, \ell \in \mathcal{A}_k \setminus \mathcal{H}, \ell' \in \Omega_{k\ell} \tag{8}$$

Note that we can remove Constraints (2.03) in the presence of Constraints (8). This is possible since Constraints (8) imply Constraints (2.03). However, we keep Constraints (2.03) since they are valid inequalities that strengthen the relaxed formulation. The total number of Constraints (6) and (8) is typically smaller than that of Constraints (2.11). More precisely, reformulation of the precedence constraints requires $\sum_{k \in \mathcal{K}} \sum_{\ell \in \mathcal{A}_k \setminus \mathcal{H}} \left(\left| \Omega_{k\ell} \right| + 1 \right)$ constraints while the existing approach requires $\sum_{k \in \mathcal{K}} \sum_{\ell \in \mathcal{A}_k \setminus \mathcal{H}} \left(\left| \Omega_{k\ell} \right| \times \left| \mathcal{T}_\ell \cap \mathcal{T}_k^K \right| \right)$.

6.3.3 Pre-processing

In this section, we describe a pre-processing approach in which a subset of the decision variables $w^t_{k\ell}$, $\hat{s}^k_{\ell t}$ and $\check{s}^{k\ell}_{\ell' t}$ are eliminated before the optimization begins. This pre-processing approach is based on obtaining a lower bound $l_{k\ell}$ on the time slot at which student $k \in \mathcal{K}$ can begin the syllabus lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$. Let $\Omega'_{k\ell}$ denote the set of all (direct and indirect) prerequisite lessons for syllabus lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$. We know that student k cannot take lesson ℓ unless he/she has already taken the prerequisite lessons in $\Omega'_{k\ell}$. This enables us to calculate $l_{k\ell}$ through the procedure explained next.

Let $\sigma = \sum_{\ell' \in \Omega'_{k\ell}} \Delta'_{\ell'}$ denote the total time student $k \in \mathcal{K}$ needs for taking the prerequisites of the syllabus lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$. We first obtain the earliest day $d^\star$ at which student k can take lesson ℓ as follows:

$$d^\star = \left\lfloor \frac{\sigma}{\alpha^K_k} \right\rfloor + 1$$

If $d^\star > D$ then student k cannot take lesson ℓ in the specified planning horizon. In this case, we set $l_{k\ell} = T+1$. Otherwise, let ρ denote the remainder of dividing σ by α^K_k. Then we set $l_{k\ell} = \rho + \min \mathcal{T}^K_k \cap [a_{d^\star}, b_{d^\star}]$. Once the value of $l_{k\ell}$ is determined, all of the decision variables $w^t_{k\ell}$ with $t < l_{k\ell}$ are set to zero. Similarly, all of the decision variables $\hat{s}^k_{\ell t}$ and $\check{s}^{k\ell}_{\ell' t}$ that can be eliminated are determined and set to zero. For instance, we set $\hat{s}^k_{\ell' t} = 0$ for $\ell' \in \mathcal{H}_\ell \cup \{\ell\}$ if $t + \check{\delta}_{\ell'} < l_{k\ell}$. In addition to eliminating such decision variables, we set the lower bound of the decision variables $\tau_{k\ell}$ to $l_{k\ell}$ to tighten their bounds.

7 Computational Results

The mathematical formulations and algorithms studied in this paper have all been implemented in ARES. The corresponding module is referred to as the optimizer engine (or simply optimizer). The optimizer is used as an alternative for the existing automated scheduler, the auto planner, that has originally been designed by the technical experts to automate course scheduling practices of the pilot training program under study. This algorithm is a module in ARES that could only be accessed through a graphical user interface (also known as the UI). We are not aware of the exact procedures of this (black-box) algorithm. However, the developer of this algorithm has provided the following general description. Auto planner mimics the procedure taken by a human planner. In the first instance, the student with the highest priority score will be selected. While satisfying the prerequisite relations, the first lesson will be selected and scheduled to take the earliest possible time slot as long as resources are available and the necessary requirements (such as one-time preparations) are completed. Then, the next lesson will be selected for this student and scheduled as long as all duty rules are observed and resources are available. This procedure will continue until no more lessons can be scheduled for this student in the planning horizon, then the algorithm will select the next student (according to the priority score) and schedule lessons for him/her. The algorithm terminates after all students have been considered. Note that unlike the fix-and-optimize algorithm introduced in Section 6.3.1, the auto planner does not utilize any mathematical programming formulation.

The user can choose between the auto planner and the optimizer through the UI. The UI displays the resulting timetable and provides a wide variety of other features. One of these features is an automatic validator which checks the validity of the solutions generated by the auto planner or the optimizer. Once a solution is accepted, it can be modified by the user (*e.g.,* by moving a session, adding/removing resources, etc). ARES is written in Java and therefore we used Java for developing the optimizer. The first version of the optimizer was quite slow since it was based on solving *F2*. The proposed B&CH approach was introduced in the latest version of the optimizer to resolve this issue.

In this section, we report results of running the optimizer on a set of daily and weekly instances of the problem. The computer used to run the optimizer is a Dell Optiplex core i5-4670 with 3.40GHz processors and 8GB RAM running Ubuntu 18.04. We used CPLEX 12.10 as the optimization solver and employed the generic callbacks of CPLEX to implement the proposed B&CH approach. We set the optimality gap of CPLEX to 10^{-6} and its time limit to 2 hours for each daily instance. All other parameters of CPLEX were set to their default values.

Our preliminary computational experiments indicated that instances with $D > 1$ are very hard to solve even with the proposed B&CH approach. Therefore, we solve the weekly instances using the algorithm introduced in Section 6.3.1. Each weekly instance is decomposed into one-day buckets and solved one day at a time. A daily instance can be solved using *F2*, the enhanced B&CH approach or the proposed matheuristic. In the last case, two buckets of roughly equal length are used. Although decomposition to more subproblems is possible, we observed that two subproblems can be solved sufficiently faster than the original daily instance (by orders of magnitude for some instances). We compare these implementations

and evaluate their relative performance compared to the auto planner. The daily and weekly instances used in these comparisons correspond to a planning period of 20 weeks. The training school is closed in the weekends, local holidays and scheduled breaks. Consequently, we report results of 93 daily instances in our numerical study. Furthermore, we consider an additional 'restricted' version for each weekly instance as will be discussed in Section 7.2. Therefore, results of 40 weekly instances are reported.

7.1 Daily Instances

This section compares performance of different algorithms on the daily instances. We report the average (AVG), median (MDN), maximum (MAX), minimum (MIN) and the sample standard deviation (STD) of each quantity for the 93 daily instances. Five implementation methods are considered as follows:

1. **F2:** Session-index formulation *F2* solved by CPLEX
2. **B&CH-V0:** The B&CH approach without any enhancements
3. **B&CH-V1:** The B&CH approach with improved initial lower bound
4. **B&CH-V2:** B&CH-V1 with the new compact precedence constraints
5. **B&CH-V3:** B&CH-V2 with the pre-processing approach

The last four implementation methods are based on the relaxed formulation *F3*. Table 2 reports the number of decision variables and constraints of *F2* and *F3*. We can see from the table that *F3* requires a lower number of decision variables and constraints compared to *F2* (almost as half). This is another advantage of the formulation in addition to the handling of the symmetry issue. Therefore, it is not unexpected to see that the last four implementation methods perform better than the first method. We point out that each MIN value in the table corresponds to a Friday, in which the training school is closed earlier than usual. Recall that *F2* and *F3* are both time-index formulations. This means that the size of either formulation depends on the number of permissible time slots defined for each entity. Note also that in our implementation, we defined and generated all variables and constraints exactly as defined in the paper.

Table 2: Number of variables and constraints generated by *F2* and *F3* for the daily instances

Formulation	Variable Count					Constraint Count				
	AVG	MDN	MAX	MIN	STD	AVG	MDN	MAX	MIN	STD
F2	27,939	27,647	58,141	7,451	12,035	28,853	28,387	53,540	8,161	12,125
F3	11,530	12,243	18,695	2,789	4,321	15,230	15,169	25,397	4,690	5,294

A detailed comparison of the five implementation methods are provided in Table 3. In this table, the computational time (CPU, in seconds), the number of B&B searched nodes (NOD) and the number of iterations (ITR) of the different methods are provided for the daily instances. These quantities are reported by CPLEX after the optimization was terminated. As expected, F2 is outperformed by all versions of the proposed B&CH in terms of the number of nodes, iterations and the computational time. The table also shows that each enhancement has a positive impact on the overall computational time. Of interest is that the median of the number of nodes for the last three implementation methods is zero. This implies that the quality of the proposed matheuristic is very good since inserting this solution results in finding an optimal solution at the root node for more than 50% of the instances. Note that modern solvers such as CPLEX utilize such a good solution in different ways, especially in the so-called reduced cost fixing method that fixes some decision variables given a good feasible solution (Atamtürk et al., 1996). The matheuristic takes around 4 seconds, on average, to solve a daily instance.

Another observation that can be made from Table 3 is that adding the new compact precedence constraints has increased the average number of nodes. However, the average solution time is decreased which is due to a fewer number of constraints and nonzero elements in the coefficient matrix. It is noteworthy to mention that the pre-processing approach eliminated around 12.5% of the decision variables $w^t_{k\ell}$, $\hat{s}^k_{\ell t}$ and $\check{s}^{k\ell}_{\ell' t}$ in B&CH-V3. The enhancements together almost halve the computational time (*i.e.,* from 414 to 211 seconds). We observed that the maximum number of feasibility cuts added in the different versions of the proposed B&CH approach is around 400. In most cases, however, only a handful of cuts were added (if any).

Let *UB* be the best known upper bound for an instance and *OBJ* be the objective value obtained by using a method to solve this instance. We calculate the corresponding optimality gap by using the

Table 3: CPLEX results for different methods on the daily instances

Indicators		Methods				
		F2	B&CH-V0	B&CH-V1	B&CH-V2	B&CH-V3
CPU	AVG	4,457.9	413.9	372.5	258.6	211.3
	MDN	7,200.0	34.0	15.6	8.6	6.8
	MAX	7,200.0	7,200.0	7,200.0	7,200.0	7,200.0
	MIN	0.2	0.1	0.0	0.1	0.0
	STD	3,174.1	1,148.2	1,124.1	875.2	795.8
NOD	AVG	5,813	2,001	1,551	5,249	2,115
	MDN	4,257	51	0	0	0
	MAX	45,238	36,761	16,028	169,238	34,601
	MIN	0	0	0	0	0
	STD	7,537	4,799	3,054	24,141	5,210
ITR	AVG	1.10E+07	2.28E+06	1.82E+06	1.73E+06	1.45E+06
	MDN	8.36E+06	9.19E+04	3.90E+04	2.06E+04	2.31E+04
	MAX	5.02E+07	3.67E+07	2.54E+07	2.80E+07	2.06E+07
	MIN	3.20E+02	7.20E+01	0.00E+00	0.00E+00	0.00E+00
	STD	1.05E+07	5.39E+06	4.15E+06	4.93E+06	3.61E+06

following formula:

$$GAP_{opt} = \frac{UB - OBJ}{OBJ} \times 100.$$

The smaller the optimality gap, the better the method. We report this optimality gap for each daily instance in Figure 1. The figure includes the results for the best implementation method, B&CH-V3, the proposed matheuristic and the auto planner. As can be seen, B&CH-V3 obtained the best known solution in all instances. It is not surprising to see that B&CH-V3 outperforms the proposed matheuristic since it utilizes the initial solution provided by this matheuristic. The auto planner is outperformed by the proposed matheuristic in most of the cases which demonstrates the significance of our methodology. The optimality gap as well as the number of instances solved to optimality are provided in Table 4 for all implementation methods as well as the auto planner. As can be seen, the auto planner obtained optimal solutions for only 3 instances out of 93. Only one instance could not be solved to optimality by the different versions of the proposed B&CH approach. The table also shows that the feasible solutions found by F2 are optimal in 55 instances. However, we observed that CPLEX could only prove optimality of 43 of them. In fact, the lower and upper bounds obtained from B&CH-V0 are at least as good as those of F2 for every instance. Specifically, B&CH-V0 provided better (strictly greater) lower bounds for 38 instances and better (strictly smaller) upper bounds in 48 instances. These results clearly demonstrate the superiority of the proposed B&CH compared to F2.

Table 4: Optimality of different methods on the daily instances

Method	Optimal Count	Optimality Gap (%)				
		AVG	MDN	MAX	MIN	STD
Auto planner	3	24.79	19.10	125.00	0.00	22.40
Matheuristic	42	4.49	1.85	24.39	0.00	5.77
F2	55	48.62	0.00	733.33	0.00	123.09
B&CH-V0	92	0.03	0.00	3.01	0.00	0.31
B&CH-V1	92	0.05	0.00	4.70	0.00	0.49
B&CH-V2	92	0.01	0.00	1.38	0.00	0.14
B&CH-V3	92	0.01	0.00	1.38	0.00	0.14

7.2 Weekly Instances

Real instances of the problem with multiple days could not be solved to optimality and therefore we solve the weekly instances using the proposed matheuristic. Evidently, this approach does not necessarily provide an optimal solution for these instances. Since the optimal solution for the weekly instances are not available, we define a performance index to evaluate the performance of the different algorithms. Let

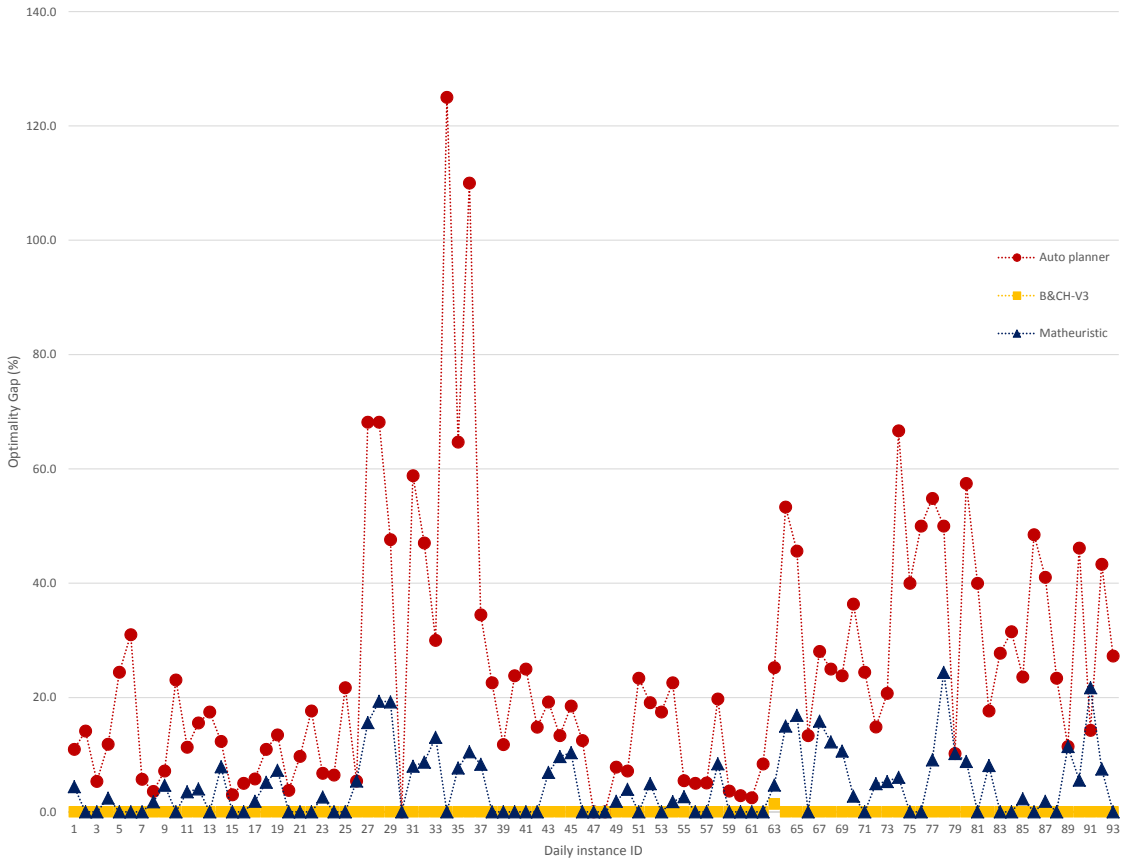


Figure 1: Optimality gap of the matheuristic, auto planner and B&CH-V3 on daily instances

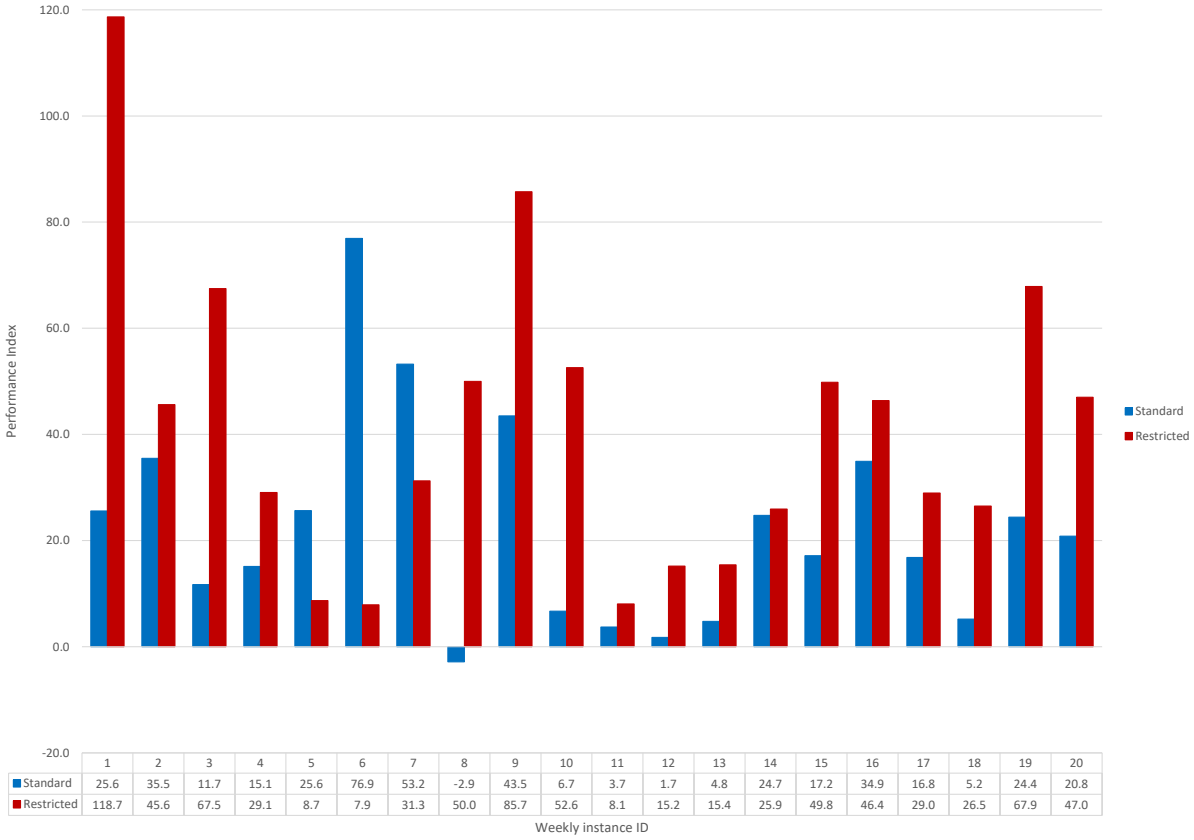
OBJ_{auto} be the objective value of the solution provided by the auto planner for solving a weekly instance and OBJ be the objective value obtained by using a specific method to solve this instance. We define the performance index

$$PI_{auto} = \frac{OBJ - OBJ_{auto}}{OBJ_{auto}} \times 100,$$

which indicates the relative performance of a method compared to the auto planner. A positive value indicates that the method is performing better than the auto planner, while a negative value indicates that the auto planner is performing better. In general, the larger the performance index, the better the method performs in solving the corresponding instance.

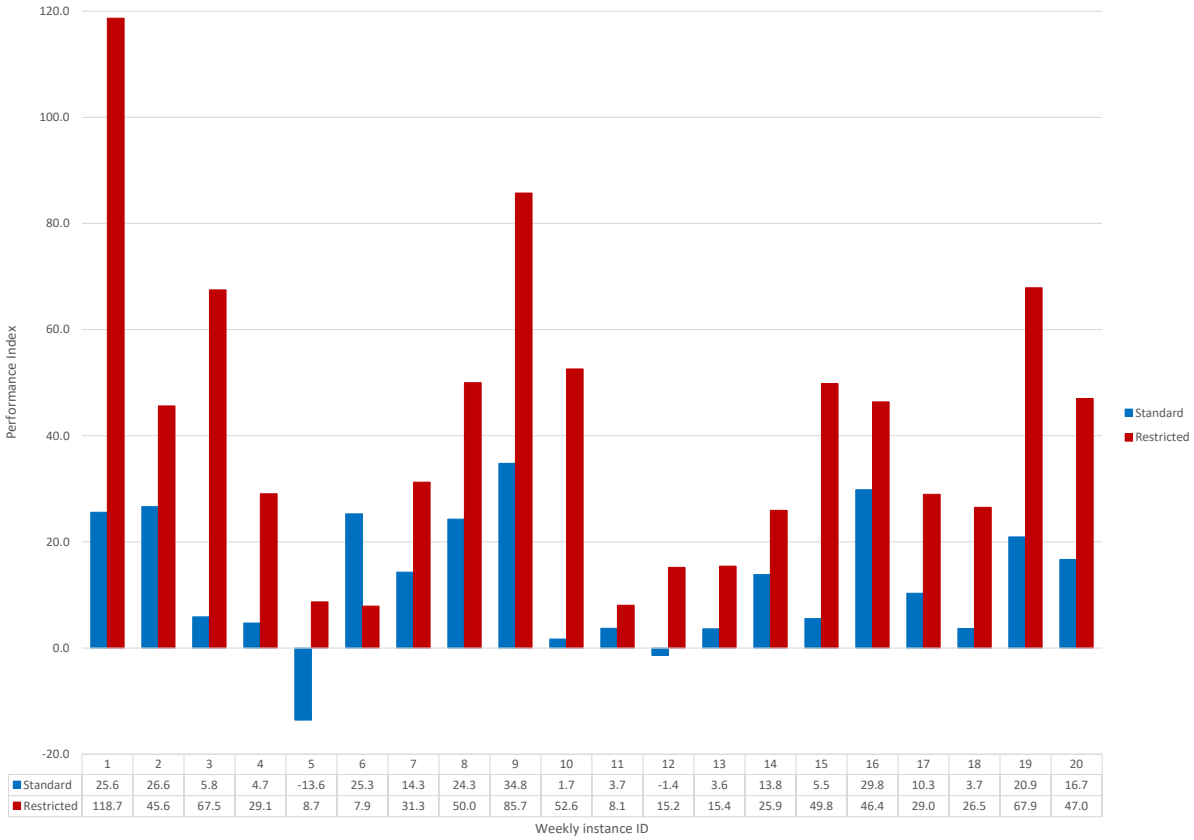
The weekly instances can be considered as 20 pairs of instances. Each pair of instances consists of a 'standard' instance and a 'restricted' instance. The standard instance is based on the real data. The restricted instance is generated from the standard instance by reducing the set of available resources to roughly a half. In other words, the two instances in each pair of resources have exactly the same data except for the set of resources which is roughly halved in the restricted version (*i.e.*, fractional values of division by 2 are rounded up). This reduction in the set of resources is done for each resource type. For example, consider a standard instance in which there are two instructors of type 'staff avwo' and three instructors of type 'staff pilot'. Then the restricted version has one 'staff avwo' and two 'staff pilot' instructors.

Figures 2 and 3 respectively depict the performance index of each pair of instances when B&CH-V3 and the proposed matheuristic are used for solving each day of the week. As can be seen, the performance index is generally higher for the restricted instances, indicating that the relative performance of our proposed methodology is even better if the resources are further limited. In the standard instances, the average performance index is 22.3 for B&CH-V3 and 12.8 for the proposed matheuristic. These values are increased respectively to 41.4 and 32.4 in the restricted instances. Note that the performance index is negative in a few standard instances which indicates that the auto planner performed better on them. This is not unexpected since solving large instances by decomposing them into a number of daily instances



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Standard	25.6	35.5	11.7	15.1	25.6	76.9	53.2	-2.9	43.5	6.7	3.7	1.7	4.8	24.7	17.2	34.9	16.8	5.2	24.4	20.8
Restricted	118.7	45.6	67.5	29.1	8.7	7.9	31.3	50.0	85.7	52.6	8.1	15.2	15.4	25.9	49.8	46.4	29.0	26.5	67.9	47.0

Figure 2: Performance index of B&CH-V3 on weekly instances



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Standard	25.6	26.6	5.8	4.7	-13.6	25.3	14.3	24.3	34.8	1.7	3.7	-1.4	3.6	13.8	5.5	29.8	10.3	3.7	20.9	16.7
Restricted	118.7	45.6	67.5	29.1	8.7	7.9	31.3	50.0	85.7	52.6	8.1	15.2	15.4	25.9	49.8	46.4	29.0	26.5	67.9	47.0

Figure 3: Performance index of the proposed matheuristic on weekly instances

is a heuristic approach which does not guarantee the optimality.

8 Conclusion

In this paper, we introduced the multiphase course timetabling problem (abbreviated MCTP) in which lessons can have multiple phases with different resource requirements. A solution of the problem may schedule several sessions for a lesson, depending on the capacity and the demand for the lesson. We formulated the problem as an integer linear program called the session-index formulation. The MCTP is then extended to capture further requirements of the pilot training program under study. The session-index formulation is accordingly updated for this extended problem (abbreviated EMCTP). We further studied this problem since it generalizes the MCTP and it captures all the requirements of the real-world application. To solve the EMCTP more efficiently, we used branch and check (abbreviated B&CH), an efficient implementation of the logic-based Benders decomposition algorithm. The proposed B&CH is based on a different formulation, the relaxed formulation, which removes the notion of session-index to eliminate symmetry. This approach is further improved by pre-processing, reformulating the precedence constraints and improving the initial lower bound. The last improvement is based on a fast matheuristic which provides high quality solutions for the problem. We demonstrated the efficacy of the proposed methodology in solving daily and weekly instances of the real-world problem in practice.

The proposed matheuristic is a fix-and-optimize algorithm that provides feasible solutions with an average optimality gap of 4.5% for the daily instances within seconds. We observed that the enhanced B&CH algorithm achieves an average optimality gap of 0.01%, whereas the session-index formulation obtains an average optimality gap of 48.62%, while spending 21 times more computational time to solve the daily instances. A more than twofold increase is also achieved in the number of instances proved to optimality (*i.e.,* from 43 to 92). While the enhanced algorithm provides optimal solutions for 98.9% of the instances, the auto planner (the existing algorithm) could provide optimal solutions for only 3.2% of them. We solved the weekly instances by decomposing them into a number of daily instances that are solved by the enhanced B&CH algorithm. This approach provided solutions that have 22.3% better objective values, on average, compared to those of the auto planner. We showed that this improvement is increased to 41.4% in a pessimistic scenario in which resources are restricted (*i.e.,* reduced by nearly 50%).

Improving the proposed fix-and-optimize algorithm as well as designing other matheuristics for the problem are both interesting research directions. Matheuristics applied to other classes of problems such as multi-level lot-sizing problems can also be explored (see *e.g.,* Toledo et al., 2015, and the references therein). The solutions obtained from such heuristic algorithms are not necessarily optimal. Therefore, another promising direction for future research is to develop more efficient exact methods for the problem. Of interest is to see whether a branch and price decomposition approach can perform better than the proposed B&CH. Developing pattern-based formulations for the problem is an important step towards designing such algorithms (see *e.g.,* Bagger et al., 2019a,b; Saviniec et al., 2020). We point out that the MCTP and its extended version EMCTP consist of many consequential parameters, including availability times of each entity (resource, student, lesson), the specific lessons for each student (each having their specific resource requirements), the priorities of students, and so on. Developing a systematic benchmark data set for these problems provides a baseline for evaluating the efficacy of the existing algorithms (proposed in this paper) and the new ones. This is particularly important for the MCTP since it is the basic version of the problem that is not empirically studied.

We complete this section by highlighting that the potential superiority of B&CH algorithms to solve other educational timetabling problems can and should be explored. An interesting feature of these algorithms is their decomposition flexibility, that is, a variety of exact algorithms can be obtained based on deciding which decisions to be handled by the master problem and the subproblem(s). This implies that the focus on feasibility and optimality can also be tailored based on the strategy used. Some interesting strategies have been discussed in Burke et al. (2010), but to our knowledge, have not yet been explored in an exact algorithm such as B&CH. One of these strategies is the "times first, rooms second" approach which is regarded by the authors as a standard decomposition procedure for educational timetabling problems. A B&CH algorithm based on this strategy handles the "times first" part in the master problem and the "rooms second" part in the subproblem(s).

Details of such an algorithm including the form of Benders feasibility and/or optimality cuts depend on the formulation of the master and subproblem(s) for the specific problem to which the algorithm is applied. For instance, in the present paper, the master problem is constructed in a way that handles all decisions except for allocating students to specific sessions of a cohort lesson that are in progress at the same time. Each subproblem receives information about a set of students and the cohort lesson they are supposed to take at the same time. The subproblem (which is the decision version of a bin packing problem) checks whether a feasible split of these students into the given number of sessions is

possible. If yes, a feasible solution of the problem can be obtained based on the solution prescribed by the subproblem. Otherwise, a Benders feasibility cut is added to the master problem in order to remove the infeasible solution.

We also found that some heuristic algorithms in the literature can easily be transformed into an exact algorithm through logic-based Benders decomposition. For instance, the two-stage algorithms presented in Lach and Lübbecke (2012) and Sørensen and Dahms (2014) can be made exact by iteratively adding suitable Benders cuts. The Stage I problem in either study captures hard constraints as well as some soft constraints of the original problem. The Stage II problem uses the solution received from Stage I to construct a feasible solution of the original problem. In fact, the idea of iteratively solving the first and second stage problems has already been mentioned in Sørensen and Dahms (2014). The authors write that "an iterative procedure could in principle be used, such that the solution obtained from the Stage II model is given as input to the Stage I model, and the whole procedure is repeated. It is however unclear how the input from the Stage II model should effect the Stage I model to obtain convergence towards better solutions in terms of the overall objective." Considering the Stage I model as the master problem and the Stage II model as the subproblem in a B&CH approach, convergence to an optimal solution (in a finite number of iterations) is guaranteed if suitable Benders cuts are added to the master problem. Since both studies utilize binary variables in the master problem, well-known Benders cuts such as no-good cuts are easy to construct. In general, however, the convergence rate of a B&CH algorithm could be slow if the quality of Benders cuts or the bound obtained from the master problem is not good enough. These aspects should be taken into consideration when designing such exact algorithms.

Acknowledgments This research was funded by Defence Science and Technology Group, Department of Defence, Australia.

References

Akbarzadeh, B. and Maenhout, B. (2020). A decomposition-based heuristic procedure for the medical student scheduling problem. *European Journal of Operational Research*.

Archetti, C., Bianchessi, N., and Speranza, M. G. (2014). Branch-and-cut algorithms for the split delivery vehicle routing problem. *European Journal of Operational Research*, 238(3):685–698.

Atamtürk, A., Nemhauser, G. L., and Savelsbergh, M. W. (1996). A combined lagrangian, linear programming, and implication heuristic for large-scale set partitioning problems. *Journal of heuristics*, 1(2):247–259.

Babaei, H., Karimpour, J., and Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, 86:43–59.

Bagger, N.-C. F., Desaulniers, G., and Desrosiers, J. (2019a). Daily course pattern formulation and valid inequalities for the curriculum-based course timetabling problem. *Journal of Scheduling*, 22(2):155–172.

Bagger, N.-C. F., Sørensen, M., and Stidsen, T. R. (2019b). Dantzig–wolfe decomposition of the daily course pattern formulation for curriculum-based course timetabling. *European Journal of Operational Research*, 272(2):430–446.

Bashab, A., Ibrahim, A. O., AbedElgabar, E. E., Ismail, M. A., Elsafi, A., Ahmed, A., and Abraham, A. (2020). A systematic mapping study on solving university timetabling problems using meta-heuristic algorithms. *Neural Computing and Applications*, pages 1–36.

Beck, J. C. (2010). Checking-up on branch-and-check. In *International Conference on Principles and Practice of Constraint Programming*, pages 84–98. Springer.

Bettinelli, A., Cacchiani, V., Roberti, R., and Toth, P. (2015). An overview of curriculum-based course timetabling. *Top*, 23(2):313–349.

Bianchessi, N. and Irnich, S. (2019). Branch-and-cut for the split delivery vehicle routing problem with time windows. *Transportation Science*, 53(2):442–462.

Burke, E. K., Mareček, J., Parkes, A. J., and Rudová, H. (2010). Decomposition, reformulation, and diving in university course timetabling. *Computers & Operations Research*, 37(3):582–597.

Burke, E. K. and Petrovic, S. (2002). Recent research directions in automated timetabling. *European Journal of Operational Research*, 140(2):266–280.

Coffman Jr, E., Garey, M., and Johnson, D. (1996). Approximation algorithms for bin packing: a survey. In *Approximation algorithms for NP-hard problems*, pages 46–93.

Costa, L., Contardo, C., and Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science*, 53(4):946–985.

Dorneles, Á. P., de Araújo, O. C., and Buriol, L. S. (2014). A fix-and-optimize heuristic for the high school timetabling problem. *Computers & Operations Research*, 52:29–38.

Dostert, M., Politz, A., and Schmitz, H. (2016). A complexity analysis and an algorithmic approach to student sectioning in existing timetables. *Journal of Scheduling*, 19(3):285–293.

Elçi, Ö. and Hooker, J. N. (2020). Stochastic planning and scheduling with logic-based Benders decomposition. *arXiv preprint arXiv:2012.14074*.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., Owens, B., and Sier, D. (2004a). An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research*, 127(1):21–144.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., and Sier, D. (2004b). Staff scheduling and rostering: A review of applications, methods and models. *European journal of operational research*, 153(1):3–27.

Esmaeilbeigi, R., Charkhgard, P., and Charkhgard, H. (2016a). Order acceptance and scheduling problems in two-machine flow shops: New mixed integer programming formulations. *European Journal of Operational Research*, 251(2):419–431.

Esmaeilbeigi, R., Naderi, B., and Charkhgard, P. (2015). The type E simple assembly line balancing problem: a mixed integer linear programming formulation. *Computers & Operations Research*, 64:168–177.

Esmaeilbeigi, R., Naderi, B., and Charkhgard, P. (2016b). New formulations for the setup assembly line balancing and scheduling problem. *OR Spectrum*, 38(2):493–518.

Fonseca, G. H., Santos, H. G., and Carrano, E. G. (2016). Integrating matheuristics and metaheuristics for timetabling. *Computers & Operations Research*, 74:108–117.

Fonseca, G. H., Santos, H. G., Carrano, E. G., and Stidsen, T. J. (2017). Integer programming techniques for educational timetabling. *European Journal of Operational Research*, 262(1):28–39.

Gonzalez, G., Richards, C., and Newman, A. (2018). Optimal course scheduling for united states air force academy cadets. *Interfaces*, 48(3):217–234.

Hooker, J. N. and Ottosson, G. (2003). Logic-based Benders decomposition. *Mathematical Programming*, 96(1):33–60.

Kristiansen, S., Sørensen, M., and Stidsen, T. R. (2015). Integer programming for the generalized high school timetabling problem. *Journal of Scheduling*, 18(4):377–392.

Kristiansen, S. and Stidsen, T. R. (2013). A comprehensive study of educational timetabling–a survey. *Department of Management Engineering, Technical University of Denmark*.

Lach, G. and Lübbecke, M. E. (2012). Curriculum based course timetabling: new solutions to udine benchmark instances. *Annals of Operations Research*, 194(1):255–272.

Lewis, R. (2008). A survey of metaheuristic-based techniques for university timetabling problems. *OR spectrum*, 30(1):167–190.

Lindahl, M., Sørensen, M., and Stidsen, T. R. (2018). A fix-and-optimize matheuristic for university timetabling. *Journal of Heuristics*, 24(4):645–665.

Margot, F. (2010). Symmetry in integer linear programming. In Jünger, M., Liebling, T. M., Naddef, D., Nemhauser, G. L., Pulleyblank, W. R., Reinelt, G., Rinaldi, G., and Wolsey, L. A., editors, *50 Years of integer programming 1958-2008: From the early years to the state-of-the-art*, pages 647–686. Springer Berlin Heidelberg.

Müller, T., Rudová, H., Müllerová, Z., et al. (2018). University course timetabling and international timetabling competition 2019. In *Proceedings of the 12th International Conference on the Practice and Theory of Automated Timetabling (PATAT-2018)*, volume 1, pages 5–31.

Pillay, N. (2014). A survey of school timetabling research. *Annals of Operations Research*, 218(1):261–293.

Pillay, N. (2016). A review of hyper-heuristics for educational timetabling. *Annals of Operations Research*, 239(1):3–38.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817.

Roshanaei, V. and Naderi, B. (2021). Solving integrated operating room planning and scheduling: Logic-based Benders decomposition versus branch-price-and-cut. *European Journal of Operational Research*, 293(1):65–78.

Saviniec, L., Santos, M. O., Costa, A. M., and dos Santos, L. M. (2020). Pattern-based models and a co-operative parallel metaheuristic for high school timetabling problems. *European Journal of Operational Research*, 280(3):1064–1081.

Sherali, H. D. and Smith, J. C. (2001). Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407.

Sørensen, M. and Dahms, F. H. (2014). A two-stage decomposition of high school timetabling applied to cases in denmark. *Computers & Operations Research*, 43:36–49.

Tan, J. S., Goh, S. L., Kendall, G., and Sabar, N. R. (2021). A survey of the state-of-the-art of optimisation methodologies in school timetabling problems. *Expert Systems with Applications*, page 113943.

Thorsteinsson, E. S. (2001). Branch-and-Check: A hybrid framework integrating mixed integer programming and constraint logic programming. In *International Conference on Principles and Practice of Constraint Programming*, pages 16–30. Springer.

Toledo, C. F. M., da Silva Arantes, M., Hossomi, M. Y. B., França, P. M., and Akartunalı, K. (2015). A relax-and-fix with fix-and-optimize heuristic applied to multi-level lot-sizing problems. *Journal of heuristics*, 21(5):687–717.

Toth, P. and Vigo, D. (2014). *Vehicle Routing: Problems, Methods, and Applications, Second Edition*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics.

Wren, A. (1995). Scheduling, timetabling and rostering - a special relationship? In *International conference on the practice and theory of automated timetabling*, pages 46–75. Springer.

Zohali, H., Naderi, B., and Roshanaei, V. (2021). Solving the type-2 assembly line balancing with setups using logic-based Benders decomposition. *INFORMS Journal on Computing*, pages 1–18. *Articles in Advance*.

Appendices

A Big-M parameters

This section defines the specific values of the big-M parameters that we consider as sufficiently large for the corresponding constraints.

Constraints (1.22) and (1.23)

The big-M parameters M^K_{kdt} and M^R_{rdt} can be defined as follows:

$$M^K_{kdt} \coloneqq \left|[t + \lambda^K_k, b_d] \cap \mathcal{T}^K_k\right|$$

$$M^R_{rdt} \coloneqq \left|[t + \lambda^R_r, b_d] \cap \mathcal{T}^R_r\right|$$

The parameter M^K_{kdt} specifies an upper bound on the number of remaining time slots in day d that become unavailable for student k (due to the maximum time span duty rule) if he/she is in a session at time $t \in [a_d, b_d]$. The parameter M^R_{rdt} is defined similarly for a resource.

Constraints (1.24) and (1.25)

The big-M parameters $\hat{M}^K_{kdt}$ and $\hat{M}^R_{rdt}$ can be defined as follows:

$$\hat{M}^K_{kdt} \coloneqq \left|[a_{d+1}, t + \mu^K_k] \cap \mathcal{T}^K_k\right|$$

$$\hat{M}^R_{rdt} \coloneqq \left|[a_{d+1}, t + \mu^R_r] \cap \mathcal{T}^R_r\right|$$

The parameter $\hat{M}^K_{kdt}$ specifies an upper bound on the number of time slots in day $d+1$ that become unavailable for student k (due to the minimum rest time duty rule) if he/she is in a session at time $t \in [a_d, b_d]$. The parameter $\hat{M}^R_{rdt}$ is defined similarly for a resource.

Constraints (2.11)

The big-M parameter $\check{M}_\ell$ can be defined as follows:

$$\check{M}_\ell = \begin{cases} \Delta'_\ell & \mathcal{B}_\ell = \emptyset \\ \Delta'_\ell + \delta_{\ell,3} + 1 & \text{otherwise,} \end{cases}$$

This parameter represents an upper bound on the number of time slots that a student spends in a session of lesson ℓ.

Constraints (2.29) and (2.30)

The big-M parameter M'_{rd} can be defined as $\min\left(\left|[a_d, b_d] \cap \mathcal{T}^R_r\right|, \alpha^R_r\right)$. This parameter is an upper bound on the number of time slots aircraft r can be used in planning day d.

Constraints (3.25) and (3.26)

The big-M parameter M''_{rd} can be defined as

$$M''_{rd} = \left\lfloor \frac{\min\left(\left|[a_d, b_d] \cap \mathcal{T}^R_r\right|, \alpha^R_r\right)}{\min\left\{\delta_{\ell,3} \colon \ell \in \mathcal{F}^A \cup \mathcal{F}^U\right\} + 1} \right\rfloor \qquad d \in \mathcal{D}, r \in \mathcal{R}^F,$$

in which $\delta_{\ell,3}$ is the duration of the main phase of the lesson (*i.e.*, the number of time slots the aircraft is used in lesson ℓ). The value 1 in the denominator is added to capture the turnaround time of the aircraft. Therefore, M''_{rd} is an upper bound on the number of times aircraft r can be used in planning day d.

B The parameters of the MCTP

This section summarizes the notation used to formulate the MCTP. The parameters related to the planning horizon, students, lessons and resources are presented in Tables 5, 6, 7 and 8, respectively. The formal definition of these parameters can be found in Section 3 of the paper.

Table 5: Planning horizon parameters

Notation	Definition
$\mathcal{D}$	The set of days in the planning horizon
$\mathcal{T}$	The set of all time slots in the planning horizon
D	The number of days in the planning horizon
T	The largest time slot in the planning horizon
a_d	The time slot at which the training school opens in day $d \in \mathcal{D}$
b_d	The time slot at which the training school closes in day $d \in \mathcal{D}$

C The decision variables of *F1*

F1 is an integer linear program for the MCTP. The decision variables used to construct this formulation are all binary variables. These variables are summarized in Table 9.

Table 6: Student related parameters

Notation	Definition
$\mathcal{K}$	The set of all students
$\mathcal{K}_k^C$	The set of students that are in the same cohort as student $k \in \mathcal{K}$
$\mathcal{L}_k$	The set of syllabus lessons that student $k \in \mathcal{K}$ has not yet passed
$\Omega_{k\ell}$	The set of direct subsequent lessons for lesson $\ell \in \mathcal{L}_k$, student $k \in \mathcal{K}$
$\mathcal{T}_k^K$	The set of time slots when student $k \in \mathcal{K}$ is available
π_k	A positive integer representing the priority of student $k \in \mathcal{K}$
λ_k^K	The maximum time span for student $k \in \mathcal{K}$
μ_k^K	The minimum rest time for student $k \in \mathcal{K}$
α_k^K	The maximum number of time slots per day for student $k \in \mathcal{K}$

Table 7: Lesson related parameters

Notation	Definition
$\mathcal{L}$	The set of all lessons
$\mathcal{P}_\ell$	The set of phases for lesson $\ell \in \mathcal{L}$
$\mathcal{P}'_\ell$	The subset of phases of lesson $\ell \in \mathcal{L}$ that students are required to attend
$\mathcal{N}_\ell$	The set of sessions for lesson $\ell \in \mathcal{L}$
$\mathcal{T}_\ell$	The set of time slots when a session of lesson $\ell \in \mathcal{L}$ can be in progress
$\mathcal{T}'_\ell$	The set of time slots when a session of lesson $\ell \in \mathcal{L}$ can begin
$\mathcal{L}^C$	The set of cohort lessons
$\mathcal{L}^R$	The set of all lessons that require at least one resource
Δ_ℓ	The duration/length of lesson $\ell \in \mathcal{L}$
Δ'_ℓ	The total time students are present in a session of lesson $\ell \in \mathcal{L}$
$\delta_{\ell p}$	The duration/length of phase $p \in \mathcal{P}$ of lesson $\ell \in \mathcal{L}$
n_ℓ	The maximum number of sessions for lesson $\ell \in \mathcal{L}$
c_ℓ	The capacity of lesson $\ell \in \mathcal{L}$

Table 8: Resource related parameters

Notation	Definition
$\mathcal{R}$	The set of all human and physical resources
$\mathcal{G}_\ell$	The set of lesson-groups for lesson $\ell \in \mathcal{L}^R$
$\mathcal{R}_{\ell g}^G$	The set of resources for lesson-group $g \in \mathcal{G}_\ell$ of lesson $\ell \in \mathcal{L}^R$
$\mathcal{P}_{\ell g}^G$	The set of phases for lesson-group $g \in \mathcal{G}_\ell$ of lesson $\ell \in \mathcal{L}^R$
$\mathcal{R}_\ell^P$	The set of paired resources for lesson $\ell \in \mathcal{L}$ that must be selected together
$\mathcal{R}_\ell^L$	The set of all resources that may be used in a phase of lesson $\ell \in \mathcal{L}^R$
λ_r^R	The maximum time span for resource $r \in \mathcal{R}$
μ_r^R	The minimum rest time for resource $r \in \mathcal{R}$
α_r^R	The maximum number of time slots per day for resource $r \in \mathcal{R}$

Table 9: The binary variables of *F1*

Variable	Takes the value one if and only if
$x_{\ell j}^t$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}$ begins at time $t \in \mathcal{T}'_\ell$
$\hat{x}_{\ell j}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}$ is scheduled
$y_{\ell j}^{pt}$	phase $p \in \mathcal{P}_\ell$ of lesson $\ell \in \mathcal{L}$ is in progress in session $j \in \mathcal{N}_\ell$ at time $t \in \mathcal{T}_\ell$
$u_{\ell j}^{rt}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}^R$ uses resource $r \in \mathcal{R}_\ell^L$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}_r^R$
$\bar{u}_{rt}$	resource $r \in \mathcal{R}$ is working at time $t \in \mathcal{T}_r^R$
$v_{k\ell}$	student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{L}_k$
$\hat{v}_{\ell j}^k$	student $k \in \mathcal{K}$ attends session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}_k$
$w_{k\ell}^t$	student $k \in \mathcal{K}$ is taking lesson $\ell \in \mathcal{L}_k$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}_k^K$
$\hat{w}_{\ell j}^{kt}$	student $k \in \mathcal{K}$ attends session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}_k$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}_k^K$
$\bar{w}_{kt}$	student $k \in \mathcal{K}$ is in a session of any lesson at time $t \in \mathcal{T}_k^K$
$z_{\ell j}^{rg}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}^R$ employs resource $r \in \mathcal{R}_{\ell g}^G$ for lesson-group $g \in \mathcal{G}_\ell$

D The additional parameters for the EMCTP

The notation for the MCTP is also used for the EMCTP with the exception of the set $\mathcal{L}_k$. A summary of the additional parameters used for the EMCTP is provided in Table 10. Observe that $\mathcal{L}_k$ in MCTP is replaced with $\mathcal{A}_k$ or $\mathcal{A}_k \setminus \mathcal{H}$ in the EMCTP. We have $\mathcal{L}_k = \mathcal{A}_k$ in the absence of the shared lessons.

Table 10: The additional parameters for the EMCTP

Notation	Definition
$\mathcal{A}$	The set of actual lessons
$\mathcal{A}_k$	The set of actual lessons for student $k \in \mathcal{K}$
$\mathcal{B}$	The set of double bubbles
$\mathcal{B}_\ell$	The set of double bubbles formed from lesson $\ell \in \mathcal{A}$
$\mathcal{B}'_\ell$	The set of lessons that form double bubble $\ell \in \mathcal{B}$
$\mathcal{H}$	The set of shared lessons
$\mathcal{H}_\ell$	The set of shared lessons made from the syllabus lesson $\ell \in \mathcal{A} \setminus \mathcal{H}$
$\mathcal{L}^P$	The set of lessons that require a one-time preparation
$\mathcal{L}^S$	The set of sorties, *i.e.,* lessons that require an aircraft or simulator in their main phase
$\mathcal{F}^A$	The set of flying lessons that require ALFS configurations
$\mathcal{F}^U$	The set of flying lessons that require UTIL configurations
$\mathcal{R}^S$	The set of resources that require setup/turnaround time
$\mathcal{R}^F$	The set of all aircraft
$\mathcal{R}^I$	The set of all instructors
$\mathcal{K}^P$	The set of all pilot students
$\mathcal{K}^A$	The set of all avwo students
$\mathcal{K}^S$	The set of all senso students
c^P_ℓ	The maximum number of pilot students required for one session of lesson $\ell \in \mathcal{L}$
c^A_ℓ	The maximum number of avwo students required for one session of lesson $\ell \in \mathcal{L}$
c^S_ℓ	The maximum number of senso students required for one session of lesson $\ell \in \mathcal{L}$
β^R_r	The maximum number of time slots per week for resource $r \in \mathcal{R}$
γ^R_r	The maximum number of sorties per week for instructor $r \in \mathcal{R}^I$

E The decision variables of *F2*

F2 is a integer linear program for the EMCTP. The decision variables used to construct this formulation are all binary variables. These variables are summarized in Table 11.

Table 11: The binary variables of *F2*

Variable	Takes the value one if and only if
$x^t_{\ell j}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}$ begins at time $t \in \mathcal{T}'_\ell$
$\hat{x}_{\ell j}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}$ is scheduled
$y^{pt}_{\ell j}$	phase $p \in \mathcal{P}_\ell$ of lesson $\ell \in \mathcal{L}$ is in progress in session $j \in \mathcal{N}_\ell$ at time $t \in \mathcal{T}_\ell$
$u^{rt}_{\ell j}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}^R$ uses resource $r \in \mathcal{R}^L_\ell$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^R_r$
$\bar{u}_{rt}$	resource $r \in \mathcal{R}$ is working at time $t \in \mathcal{T}^R_r$
$v_{k\ell}$	student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$
$\hat{v}^k_{\ell j}$	student $k \in \mathcal{K}$ attends session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{A}_k$
$w^t_{k\ell}$	student $k \in \mathcal{K}$ is taking lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$
$\hat{w}^{kt}_{\ell j}$	student $k \in \mathcal{K}$ attends session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{A}_k$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$
$\bar{w}_{kt}$	student $k \in \mathcal{K}$ is in a session of any lesson at time $t \in \mathcal{T}^K_k$
$z^{rg}_{\ell j}$	session $j \in \mathcal{N}_\ell$ of lesson $\ell \in \mathcal{L}^R$ employs resource $r \in \mathcal{R}^G_{\ell g}$ for lesson-group $g \in \mathcal{G}_\ell$
$\check{v}^{k\ell}_{\ell' j}$	student $k \in \mathcal{K}$ attends session $j \in \mathcal{N}_{\ell'}$ of lesson $\ell' \in \mathcal{B}_\ell$ to take lesson $\ell \in \mathcal{A}_k$
$\check{w}^{kt}_{\ell' \ell j}$	student $k \in \mathcal{K}$ takes $\ell' \in \mathcal{A}_k$ in session $j \in \mathcal{N}_\ell$ of $\ell \in \mathcal{B}_{\ell'}$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}^K_k$
$\hat{u}_{rt}$	instructor $r \in \mathcal{R}^I$ is doing his/her one-time preparation at time $t \in \mathcal{T}^R_r$
$\check{u}_{rd}$	aircraft $r \in \mathcal{R}^F$ is set to UTIL configuration in day $d \in \mathcal{D}$

F The decision variables of *F3*

F3 is the relaxed formulation developed to solve the EMCTP by branch and check. The formulation employs the general integer variable $m_{\ell t} \in \{0, \ldots, n_\ell\}$ which represents the number of sessions of lesson $\ell \in \mathcal{L}$ that begin at time $t \in \mathcal{T}'_\ell$. The other decision variables of the formulation are all binary variables. These variables are summarized in Table 12.

Table 12: The binary variables of *F3*

Variable	**Takes the value one if and only if**
$v_{k\ell}$	student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$
$\bar{w}_{kt}$	student $k \in \mathcal{K}$ is in a session of any lesson at time $t \in \mathcal{T}_k^K$
$w_{k\ell}^t$	student $k \in \mathcal{K}$ is taking lesson $\ell \in \mathcal{A}_k \setminus \mathcal{H}$ at time $t \in \mathcal{T}_\ell \cap \mathcal{T}_k^K$
$\bar{u}_{rt}$	resource $r \in \mathcal{R}$ is working at time $t \in \mathcal{T}_r^R$
$\hat{u}_{rt}$	instructor $r \in \mathcal{R}^I$ is doing his/her one-time preparation at time $t \in \mathcal{T}_r^R$
$\check{u}_{rd}$	aircraft $r \in \mathcal{R}^F$ is set to UTIL configuration in day $d \in \mathcal{D}$
$\hat{s}_{\ell t}^k$	student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{A}_k$ in a session of ℓ that starts at $t \in \mathcal{T}'_\ell$
$\check{s}_{\ell' t}^{k\ell}$	student $k \in \mathcal{K}$ takes lesson $\ell \in \mathcal{A}_k$ in a session of lesson $\ell' \in \mathcal{B}_\ell$ that starts at $t \in \mathcal{T}'_{\ell'}$
$\hat{z}_{\ell t}^{rg}$	group $g \in \mathcal{G}_\ell$ of a session of $\ell \in \mathcal{L}^R$ that starts at $t \in \mathcal{T}'_\ell$ uses resource $r \in \mathcal{R}_{\ell g}^G$