Exam Scheduling With Hardship Minimization

Stephanie Hamilton and Donovan R. Hare

Department of Mathematics
University of British Columbia
Kelowna, British Columbia, Canada V1V 1V7
donovan.hare@ubc.ca

Abstract. The exam scheduling problem is a computationally difficult problem whose solution assigns exams to timeslots and rooms while satisfying a variety of examinee and institutional requirements. This paper proposes a generic specification of the problem that has wide applicability to real-world situations. In particular, the number of student *hardships* (i.e., students assigned multiple exams in a specified time interval) is defined in a way to encompass most contexts. Constraint programming (CP) implementations of the generic specification are then defined to model the timeslot assignment subproblem and a room assignment subproblem independently.
Two approaches are proposed and implemented to solve the overall problem from the subproblems: the use of a novel group of cuts, and the use of bin-packing global constraints. The cuts provide necessary conditions for a feasible solution of the timeslot assignment problem to have feasible room assignments. It is also shown that these cuts are sufficient conditions in certain general cases. A final section gives an empirical study using the data from the University of British Columbia.

1 Introduction

Educational timetabling is an interdisciplinary research area that has received a lot of attention over the last 20 years. In essence, the problem involves assigning university exams to timeslots and rooms in such a way that students do not write any of their own exams simultaneously, which can be elaborated on by imposing other institutional requests. Automated timetabling first caught the attention of mathematicians in the 1960's, at a time when most scheduling was done manually [6,8]. Fast forward 60 years to where the theory has advanced remarkably but in practice the scheduling software many institutions use do not implement these new research techniques, and thus institutions still struggle to get quality timetables that satisfy student and instructor needs [12]. Part of this is due to the research gap between theory and reality [10].

To address this gap, there have been three international timetabling competitions (ITCs) focusing on university timetabling. Since this problem is computationally difficult (i.e., NP-hard), researchers have tried various methods to tackle it. A sample of these methods include heuristics such as simulated annealing, population-based algorithms such as memetic algorithms, graph colouring heuristics, integer programming, and constraint-based methods (for recent literature reviews and surveys, refer to [1,3,7,13]). The winner of ITC 2007, who used a constraint satisfaction program with iterative forward search and a hill climbing heuristic, applied their model at Purdue University, with the possibility of it being utilized by other American universities [11]. In other real-life applications, constraint programming (CP) and integer programming appear to be the popular modeling techniques of choice [1,2,4,9], likely due to the flexibility these methods have in expressing constraints and the acceptable solutions provided. However, to further complicate the problem, many institutions have unique preferences, and therefore their scheduling needs are beyond the scope of the models that solve the exam scheduling problem for other universities and the benchmark datasets used in the ITCs. Therefore, even though there is an abundance of software within the academic community, institutions may be forced to continue producing schedules with rigid algorithms that must then be post-processed and fixed manually to meet complex scheduling requirements.

One of the complexities of the exam scheduling problem is that in most cases, for each exam, at least two assignments need to be made, a timeslot assignment and a room assignment, where the timeslot assigned to an exam influences the rooms available for that exam. Some authors have attempted to address this by decomposing the problem into different stages, typically a timeslot assignment phase and room assignment phase, though other decompositions have also been introduced [2,9]. This decoupling becomes increasingly complicated when multiple exams are allowed in one room. In this work we also explore decoupling the problem into timeslot and room phases. We introduce a necessary condition for the timeslot assignment model, where we generate cuts to direct the model towards producing feasible timeslot assignment solutions that will result in feasible room assignments in the next phase. These necessary conditions are sufficient for the case when each room can be assigned at most one exam at any given time.

The remainder of the paper is organized as follows. In Section 2, we provide a generic specification of the exam scheduling problem and the requirements that can be imposed on a solution. Here we introduce the student hardship requirement, which to the best of our knowledge, has not until now been given a formal specification in the literature. In Section 3.1, we describe the corresponding CP implementation of the generic specification for the timeslot assignment model. Here we describe the new necessary condition on the timeslot assignment that assists in finding a feasible room assignment. In Section 3.2, the CP requirements for the room assignment model are explained. Finally, in Section 4, we discuss the results from the experiments conducted with data from the Okanagan campus of the University of British Columbia (UBC), where we specifically focus on reducing student hardships of three types, and demonstrate the validity of the two phase model.

2 Generic Specification of the Exam Scheduling Problem

Let $\mathbb{Z}^+$ denote the set of nonnegative integers and $\mathbb{Z}^{++} = \mathbb{Z}^+ \setminus \{0\}$. All intervals throughout will be subsets of $\mathbb{Z}^+$ and for $k_1, k_2 \in \mathbb{Z}^+$, $k_1 \leq k_2$, denote the (integer) interval $[k_1, k_2] = \{k_1, \ldots, k_2\}$, and define $[k_2] = [1, k_2]$. For an interval $[a, b]$, define the *length* of $[a, b]$ to be $\ell([a, b]) = b - a$ if $b > a$, and $\ell([a, b]) = \ell(\emptyset) = 0$ otherwise. Moreover, for a finite monotonically increasing sequence of integers $r_0, \ldots, r_q$, define $r(i, j)$ to be the interval $[r_i + 1, r_j]$ for $0 \leq i < j \leq q$.

The *exam scheduling problem* is defined as the following: given a finite set of pairwise disjoint[1] integer intervals $T = \{[s_1, f_1], \ldots, [s_m, f_m]\}$ called the *timeslots*, a set P called the *persons*, a set $\mathcal{E}$ of subsets of P called the *exams*, a function $\varepsilon : \mathcal{E} \to \mathbb{Z}^{++}$ called the *exam sizes*, a function $\delta : \mathcal{E} \to \mathbb{Z}^{++}$ called the *exam durations*, a set $\mathcal{R}$ called the *rooms*, and a function $\sigma : T \times \mathcal{R} \to \mathbb{Z}^+$ called the *temporal room sizes*, find an assignment $\tau : \mathcal{E} \to T$ called the *timeslot assignment*, and an assignment $\rho : \mathcal{E} \to \mathcal{R}$ called the *room assignment*, such that for all $E, E' \in \mathcal{E}$, $E \neq E'$, the following four requirements are satisfied.

Requirement 1 (Person Single-Tasking) $\tau(E) \neq \tau(E')$ *if* $E \cap E' \neq \emptyset$.

Requirement 2 (Exam Duration) $\delta(E) \leq \ell(\tau(E))$.

Requirement 3 (Room Single-Tasking) $\rho(E) \neq \rho(E')$ *if* $\tau(E) = \tau(E')$.

Requirement 4 (Room Size) $\varepsilon(E) \leq \sigma(\tau(E), \rho(E))$.

Requirement 1 ensures that no person (including the invigilator) has two exams assigned to the same timeslot (see Requirement 6 for an extension to this requirement for the case of overlapping timeslots). Requirement 2 restricts the duration of an exam to not exceed the length of its assigned timeslot. Moreover, Requirement 3 ensures that no room is assigned two different exams during a timeslot (see Requirement 7 for a relaxation of this requirement). Finally, Requirement 4 ensures that the room assigned to an exam during the assigned timeslot can accommodate the exam's size requirement[2] (see Requirement 8 for a relaxation of this requirement).

In this specification, there is no distinction in the set of persons between students and instructors as they both are present during their exam. In the case of hardships, however, the students and instructors are separated as necessary (see Section 2.2). Moreover, for each person $p \in P$ and exam $E \in \mathcal{E}$, if $p \in E$, then we say p *writes* E, or E is *written* by p, even though p may be an instructor.

We will refer to $[s_k, f_k] \in T$ as timeslot k. As notational convenience, let $\sigma_k(R) = \sigma([s_k, f_k], R)$ for each timeslot k. The temporal room sizes function allows one to ensure that a room R does not use a timeslot k by specifying $\sigma_k(R) = 0$. This may be necessary if a room is closed during the timeslot, or to

[1] The pairwise disjoint restriction is relaxed in Requirement 6. Moreover, there is no loss of generality for exam scheduling in assuming that time can be discretized.

[2] Typically, for every exam E, $\varepsilon(E) = |E| - 1$.

limit timeslots that overlap for the room (see Requirements 7 and 8). Moreover, for each room $R \in \mathcal{R}$, let $T(R) = \{[s_k, f_k] \in T : \sigma_k(R) > 0\}$ be called the *effective* timeslots of room R.

A nonempty collection $\mathcal{A}$ of subsets of a set A is called *intersecting* if for all $A_1, A_2 \in \mathcal{A}$, $A_1 \cap A_2 \neq \emptyset$. Using the maximal intersecting subsets of exams for Requirement 1 is desired but finding them is computationally impractical (i.e., NP-hard). Instead, there are some natural intersecting subsets that can be used. For each person $p \in P$, let the intersecting subset of exams that p writes be denoted by

$$\mathcal{E}(p) = \{E \in \mathcal{E} : p \in E\}.$$

Moreover, the maximal intersecting subsets of timeslots – we will call this collection $\mathcal{B}^*$ – is the set of maximal cliques of the corresponding interval graph which can be found in linear time.

We will assume a fixed τ and ρ in the definitions that follow.

2.1 Extending the Exam Scheduling Problem

The exam scheduling problem can be extended to include a variety of other constraints from real-world contexts.

The first of these extensions further limits τ and ρ for particular situations.

Requirement 5 (Time and Room Specific) *An exam $E \in \mathcal{E}$ can be forced to be assigned a timeslot from a subset B of timeslots by the additional requirement $\tau(E) \in B$. Moreover, the exam can be forced to be assigned a room from a subset $\mathcal{Q}$ of rooms by the additional requirement $\rho(E) \in \mathcal{Q}$.*

In certain cases of modeling the exam scheduling problem, it may be necessary to allow the timeslots of the problem to overlap (i.e., not be pairwise disjoint). This occurs most commonly when modeling exams that are allowed varying lengths.

Requirement 6 (Overlapping Timeslots) *If the timeslots are allowed to overlap, then Requirement 1 is replaced with:*

$$\tau(E) \cap \tau(E') = \emptyset \textit{ if } E \cap E' \neq \emptyset.$$

Moreover, Requirement 3 is replaced with:

$$\rho(E) \neq \rho(E') \textit{ if } \tau(E) \cap \tau(E') \neq \emptyset.$$

Note that Requirement 6 generalizes Requirement 1 as non-overlapping timeslots satisfy $\tau(E) \cap \tau(E') = \emptyset$ if and only if $\tau(E) \neq \tau(E')$.

The third of these extensions allows for the use of a large room such as a gym to host several exams at once. For each $R \in \mathcal{R}$ and timeslot k, let

$$\mathcal{C}_{R,k} = \{E \in \mathcal{E} : \rho(E) = R, \tau(E) = [s_k, f_k]\}$$

be called the *concurrent* exams in room R during timeslot k.

If more than one exam is allowed in a room during a given time, then Requirement 3 is either removed entirely or it is replaced with the following requirement to limit the number of concurrent exams assigned to the room.

Requirement 7 (Room Task Limit) *During a timeslot k, and for a room $R \in \mathcal{R}$, to limit the room to be assigned at most $\gamma_k(R)$ concurrent exams, the following is required:*

$$|\mathcal{C}_{R,k}| \leq \gamma_k(R).$$

Concurrent exams pose logistical challenges if timeslots overlap. In practice, concurrent exams assigned to a room can start at the same time but may finish at different times. These exams would be assigned different overlapping timeslots where one timeslot is contained in the other. Such a situation can be handled by using the larger timeslot for both exams. This is sufficient since it is not the case, in practice, that an exam would start in the middle of a timeslot of another exam in the same room and, say, end later. Thus, in order to simplify the model without losing applicability, we can effectively model concurrent exams for rooms that allow multiple tasks by limiting these rooms to have non-overlapping timeslots when they multi-task. More formally, this limitation is for all $R \in \mathcal{R}$ and all $[s_k, f_k], [s_{k'}, f_{k'}] \in T(R)$ with $k' \neq k$, if $\gamma_k(R) > 1$ and $\gamma_{k'}(R) > 1$, $[s_k, f_k] \cap [s_{k'}, f_{k'}] = \emptyset$. If $\gamma_k(R) = 1$, then other timeslots can intersect $[s_k, f_k]$.

With more than one exam allowed in a room at a given time, Requirement 4 is replaced with the following requirement.

Requirement 8 (Room Multitasking Size) *For a room $R \in \mathcal{R}$ to host concurrent exams, for every timeslot k,*

$$\sum_{E \in \mathcal{C}_{R,k}} \varepsilon(E) \leq \sigma_k(R).$$

Requirement 9 (Coupled Exams) *To require two exams E and E' to be written*

1. *during the same timeslot, then require $\tau(E) = \tau(E')$, or*
2. *in the same room, then require $\rho(E) = \rho(E')$.*

Note that Requirement 9.2 only makes sense if we have Requirement 8 as well.

2.2 Measuring Hardships

We focus here on measuring, and later minimizing, the number of times persons have a certain number of examinations assigned to timeslots that are within a certain amount of time.

Let $d \in \mathbb{Z}^+$ represent a length of time. We first start by collecting the timeslots of T that are within d time units of each other as measured by the difference of the latest finish time with the earliest start time of its members. This set is defined by:

$$\mathcal{B}_d = \bigcup_{i=1}^{k} \{B \subseteq T : [s_j, f_j] \in B \text{ where } s_j \geq s_i \text{ and } f_j \leq s_i + d\}\,.$$

For a given person $p \in P$, the exams written by person p during a $B \in \mathcal{B}_d$ is given by

$$\mathcal{W}_{p,B} = \{E \in \mathcal{E}(p) : \tau(E) \in B\}.$$

For a given positive integer w, representing a minimum number of writes, the set of (w, d)-hardships (of τ) is defined to be

$$\mathcal{H}_{w,d} = \{(p, B) : p \in P, B \in \mathcal{B}_d, |\mathcal{W}_{p,B}| \geq w\}.$$

Thus $|\mathcal{H}_{w,d}|$ is the number of times persons are in at least w exams that are assigned by τ to be within any d time units of the schedule.[3]

Example 1. Three Exams in 27 Hours Hardships

If a student at UBC writes three exams that span at most 27 hours from the start of the first exam to the end of the last exam, then a *3-in-27-hours* hardship occurs and the student has the right to request another time to write one of the exams. This creates many issues ranging from exam calibration and fairness to exam security. It also puts an extra demand on the administrative and faculty resources. Exam schedules having zero 3-in-27-hours hardships are thus clearly valued.

UBC's examination period spans 12 days and there are four timeslots per day: 8:30 a.m., 12:00 p.m., 3:30 p.m., and 7:00 p.m. However a Sunday only has two timeslots: 12:00 p.m., and 3:30 p.m. Each exam is assumed to be two and a half hours long. Discretizing time so that 8:30 a.m. on the first day is at time $17 = 8.5 \times 2$, and is a Monday, the set of timeslots T have first week time intervals Monday: $[17, 22]$, $[24, 29]$, $[31, 36]$, $[38, 43]$; Tuesday: $[65, 70], \ldots$; Sunday: $[312, 317]$, $[319, 324]$; along with second week time intervals $[353, 358], \ldots, [566, 571]$. To measure those students having a hardship of 3 exams in 27 hours, the required timeslot sets are represented by $\mathcal{B}_{54}$ (half hour increments). See Table 1 for some examples.

Thus to minimize the number of persons having a 3-in-27-hours hardship, τ is chosen so as to minimize $|\mathcal{H}_{3,54}|$. It is also possible to restrict that no persons have such a hardship by constraining $|\mathcal{H}_{3,54}|$ to be zero. □

Requirement 10 (HARDSHIPS) *For a positive integer w, representing a number of writes, and nonnegative real number d, representing a length of time, to ensure that no persons have w examinations assigned to timeslots that are within time d (i.e., no (w, d)-hardships), then require:*

$$|\mathcal{H}_{w,d}| = 0.$$

[3] The set of persons P can be reduced here to just include students or just instructors depending on the application of the hardship.

Table 1. Example Timeslot Sets of $\mathcal{B}_{54}$ for 3-in-27-hours Hardships

Day	Timeslot Sets Involving Day
First Monday	$\{[17,22],[24,29],[31,36],[38,43],[65,70]\}$,
	$\{[24,29],[31,36],[38,43],[65,70],[72,77]\}$,
	$\{[31,36],[38,43],[65,70],[72,77],[79,84]\}$,
	$\{[38,43],[65,70],[72,77],[79,84],[86,91]\}$
First Sunday	$\{[264,269],[271,276],[278,283],[312,317]\}$,
	$\{[271,276],[278,283],[312,317],[319,324]\}$,
	$\{[278,283],[312,317],[319,324]\}$,
	$\{[312,317],[319,324],[353,358]\}$,
	$\{[319,324],[353,358],[360,365],[367,372]\}$
Last Friday	$\{[497,502],[504,509],[511,516],[518,523],[545,550]\}$,
	$\{[504,509],[511,516],[518,523],[545,550],[552,557]\}$
	$\{[511,516],[518,523],[545,550],[552,557],[559,564]\}$
	$\{[518,523],[545,550],[552,557],[559,564],[566,571]\}$

To ensure that these type of hardships are minimized, then include $|\mathcal{H}_{w,d}|$ *as a term in the minimizing objective function of the model.*

A special class of hardship are *back-to-back* exams. These hardships occur when a person writes two exams in two consecutive timeslots on the same day. Using the timeslots from Example 1, the back-to-back hardships are represented by $\mathcal{H}_{2,12}$. Given the regular nature of the timeslots of the example, a more efficient implementation is outlined at the end of the section entitled CP of Requirement 10.

3 CP Implementation of the Generic Specification

In what follows, we partition our CP implementation of the exam scheduling problem into a *timeslot assignment subproblem* and into one *room assignment subproblem* for each timeslot. The timeslot assignment subproblem is solved first and the result τ is used as input for each of the room assignment subproblems. The room assignment subproblem for timeslot k assigns rooms to only those exams assigned to timeslot k by τ. We say τ is *room-assignable* if the assignment has a feasible room assignment subproblem solution for each timeslot. The timeslot assignment subproblem contains constraints that are necessary and, in all but one scenario, are also sufficient (see Theorem 1) to ensure that all of its feasible solutions are also room-assignable. The scenario that cannot guarantee sufficiency occurs when Requirement 8 is specified. In this case, the timeslot assignment subproblem constraints are only necessary, although in practice do well in finding room-assignable solutions. An extension to the CP implementation of the timeslot assignment subproblem is also described in Section 3.3 that ensures all feasible solutions are room-assignable even with Requirement 8. However, it may not be possible to use this extension in practice, depending on the size of the input, as discussed in the Section 3.4. In order to highlight these performance considerations in the discussion, the sizes are calculated in this section for some of the relevant implementation options.

Throughout this section, we will use catalog of Beldiceanu, Carlsson and Rampon [5] as the source for definitions of known constraint programming global constraints. To facilitate the description of the implementation, the names of constraint programming variables will use a bold typeface.

3.1 Timeslot Assignment Subproblem

The timeslot assignment subproblem finds a timeslot assignment τ that ensures the implicit existence of a feasible room assignment ρ without actually determining ρ. In this section, we describe a CP implementation of the generic specification by first defining the decision variables and then defining the constraints for each of the requirements.

Primary Decision Variables In order to find a τ satisfying the requirements, for each $E \in \mathcal{E}$, we define a constraint programming integer-valued decision variable $\boldsymbol{t}_E$ whose domain is the set of indices of the timeslots, $[m] = \{1, \ldots, m\}$, with the understanding (to be encoded by the constraints) that if $\boldsymbol{t}_E$ is bound to k, then $\tau(E) = [s_k, f_k]$. For any subset $\mathcal{D}$ of exams, we let $\boldsymbol{T}_{\mathcal{D}} = \{\boldsymbol{t}_E : E \in \mathcal{D}\}$ be the corresponding set of decision variables.

CP of Requirement 1 (Person Single-Tasking).
Requirement 1 can be restated as follows: each intersecting subset of exams of size two must have the timeslot assignments of the exams different from each other. When this requirement is applied to any intersecting subset $\mathcal{I} \subseteq \mathcal{E}$ of exams, the timeslot assignments of any pair of exams from $\mathcal{I}$ must be different from each other, and so all of the timeslot assignments of exams from $\mathcal{I}$ must be different from each other. Thus we impose the global constraint `alldifferent`($\boldsymbol{T}_{\mathcal{I}}$) (see [5, p. 434]) to implement Requirement 1 for all pairs of exams in $\mathcal{I}$.

The larger the size of the intersecting subset of exams, the greater the possible propagation power of the `alldifferent` constraint. That being said, using the maximal intersecting subsets of exams would be ideal but it can be computational too expensive to find such subsets. If such sets are not known, then it can be effective to use $\mathcal{E}(p)$ for each person $p \in P$. The key is that for all $E, E' \in \mathcal{E}$, $E \neq E'$ if $\{E, E'\}$ is intersecting, then $E \cap E'$ is a subset of at least one of the intersecting subsets used.

CP of Requirement 2 (Exam Duration)**.**
The duration requirement for an $E \in \mathcal{E}$ is implemented by restricting the initial domain of $\boldsymbol{t}_E$ directly by removing each k such that $\delta(E) > \ell([s_k, f_k])$ (or indirectly and equivalently, by using the constraint $\boldsymbol{t}_E \neq k$).

CP of Requirements 3 and 4 (Room Single-Tasking) and (Room Size)**.**
In order to ensure that τ is chosen to implicitly satisfy Requirements 3 and 4, we consider a sequence of constraints that are parameterized by a positive integer c which will ensure the necessary but not sufficient statement that the number of exams of size larger than c assigned to a timeslot is at most the number of rooms of size larger than c during the timeslot.

To implement this idea, define $\mathcal{E}_c^{>} = \{E \in \mathcal{E} : \varepsilon(E) > c\}$ and for $k \in [m]$, define $\mathcal{R}_{c,k}^{>} = \{R \in \mathcal{R} : \sigma_k(R) > c\}$. The format of the restrictions are provided by the set of triples $F_c = \{(k, 0, |\mathcal{R}_{c,k}^{>}|) : k \in [m]\}$ for the global constraint `global_cardinality_low_up`$(\boldsymbol{T}_{\mathcal{E}_c^{>}}, F_c)$ (see [5, p. 1040]). This constraint ensures that, for each k, the number of decision variables $\boldsymbol{t}_E$ where $E \in \mathcal{E}_c^{>}$ and that are bound to k is between 0 and $|\mathcal{R}_{c,k}^{>}|$. In other words, the number of exams E with $\varepsilon(E) > c$ that are assigned by τ to timeslot k is at most the number of rooms that have size larger than c throughout the timeslot.

A timeslot assignment τ satisfying the `global_cardinality_low_up` constraint for a fixed minimum size c does not guarantee that there is a feasible room assignment ρ. But a sequence of these `global_cardinality_low_up` constraints does. Let $[s_k, f_k]$ be a fixed timeslot. For this timeslot, we now consider the number of distinct room sizes and let $q = q_k$ denote this number. Define $r_0 = 0$ and consider the (non-multi-)set of room sizes $\{\sigma_k(R) : R \in \mathcal{R}_{0,k}^{>}\} = \{r_1, \ldots, r_q\}$ where the r_i's are labeled so that they are strictly increasing. Note that for an exam scheduling problem to be feasible, $\mathcal{E}_{r_q}^{>} = \emptyset$. The CP constraints for Requirements 3–4 are then

$$\texttt{global\_cardinality\_low\_up}(\boldsymbol{T}_{\mathcal{E}_{r_i}^{>}}, F_{r_i}) \quad \text{for } i \in [0, q-1]. \tag{Constraints 3–4}$$

As described above, Constraints 3–4 must be necessarily satisfied by any feasible solution of the room assignment problem for timeslot k. The next theorem shows that they are also sufficient.

Theorem 1. *A feasible solution of the timeslot assignment problem with Requirement 3–4 is room-assignable if and only if the solution satisfies Constraints 3–4.*

Proof. Any feasible solution of the timeslot assignment problem with Requirements 3–4 that is room-assignable must satisfy Constraints 3–4 given the definition of the constraints described above.

On the other hand, consider a feasible solution τ to the timeslot assignment problem with Constraints 3–4 satisfied. We show that this solution is room-assignable for an arbitrary but fixed timeslot $[s_k, f_k]$. Let $q = q_k$, the number of distinct room sizes of the timeslot. For each $i \in \{0, ..., q-1\}$, let $S(i)$ be the statement that all exams assigned to $[s_k, f_k]$ by τ that have size larger than r_i can be assigned rooms in timeslot $[s_k, f_k]$. Note that all of the rooms assigned for these exams must have size larger than r_i.

Fix $k \in [m]$ and for $i \in \{0, ..., q-1\}$, let $K_i = \{E \in \mathcal{E}^{>}_{r_i} : \tau(E) = [s_k, f_k]\}$. With $i > 0$ note that, $K_i \subseteq K_{i-1}$, and, as well that $K_{i-1} \setminus K_i$ is precisely the set of those exams that are assigned to the timeslot with size in the range $r(i-1, i]$ (i.e., size larger than r_{i-1} but at most r_i). Moreover,

$$|K_i| = |\{\boldsymbol{t}_E \in \boldsymbol{T}_{\mathcal{E}^{>}_{r_i}} : \boldsymbol{t}_E \text{ is bound to } k\}|$$

since $E \in K_i$ if and only if $E \in \mathcal{E}^{>}_{r_i}$ and $\tau(E) = [s_k, f_k]$, if and only if decision variable $\boldsymbol{t}_E \in \boldsymbol{T}_{\mathcal{E}^{>}_{r_i}}$ is bound to k.

Rephrased, $S(i)$ is the statement that the exams of K_i can be assigned to rooms in the timeslot. The following proves that $S(0)$ is true by showing $S(q-1)$ is true and then inducting backwards to zero by proving that $S(i)$ implies $S(i-1)$ for all $i \in [q-1]$.

$S(q-1)$ is true. The constraint `global_cardinality_low_up`$(\boldsymbol{T}_{\mathcal{E}^{>}_{r_{q-1}}}, F_{r_{q-1}})$ ensures that $|\{\boldsymbol{t}_E \in \boldsymbol{T}_{\mathcal{E}^{>}_{r_{q-1}}} : \boldsymbol{t}_E \text{ is bound to } k\}| \leq |\mathcal{R}^{>}_{r_{q-1},k}|$. Thus $|K_{q-1}| \leq |\mathcal{R}^{>}_{r_{q-1},k}|$ and hence there are enough rooms of size more than r_{q-1}, that is those of size r_q, so that each of the $|K_{q-1}|$ exams can be assigned to a room of size at most r_q, one exam to a room, during the timeslot.

$S(i)$ implies $S(i-1)$. Suppose, for some $i \in [q-1]$, $S(i)$ is true. Thus the exams in K_i can be assigned to rooms during the timeslot. None of these exams can be assigned to rooms whose sizes are in the range $r(i-1, i]$ since they all have size larger than r_i. Assign each of these exams to their own room. There are then $|\mathcal{R}^{>}_{r_{i-1},k}| - |K_i|$ remaining rooms without assigned exams for the timeslot that have size larger than r_{i-1}. The set of remaining exams to be assigned for the timeslot that have size larger than r_{i-1} is $K_{i-1} \setminus K_i$. Moreover, the constraint `global_cardinality_low_up`$(\boldsymbol{T}_{\mathcal{E}^{>}_{r_{i-1}}}, F_{r_{i-1}})$ ensures that $|K_{i-1}| \leq |\mathcal{R}^{>}_{r_{i-1},k}|$ which in turn implies

$$|K_{i-1} \setminus K_i| = |K_{i-1}| - |K_i| \leq |\mathcal{R}^{>}_{r_{i-1},k}| - |K_i|.$$

Thus there are enough unassigned rooms of size larger than r_{i-1} (and hence size at least r_i) so the exams with sizes in $(r_{i-1}, r_i]$ in K_{i-1} can be assigned to their own room. Thus $S(i-1)$ is true.

By induction, we have that $S(0)$ is true and hence all of the exams assigned to the timeslot under τ have been assigned a room during the timeslot. $\square$

From a given timeslot assignment τ satisfying these constraints, a greedy algorithm can then be used to assign rooms of a given timeslot. This is done by first ordering the rooms from largest to smallest in a list as well as ordering the exams largest to smallest in a separate list. Then repeat the following: assign the largest room to the largest exam, and delete each from their respective lists.

Typically, the number of rooms of a small size is much larger than the number of rooms of a larger size (e.g., in most universities there is only one gym). Moreover, many small size rooms do not get assigned an exam for a given timeslot. For these reasons, the room assignment problem is typically difficult for the larger sized exams. Thus in practice, not all minimum sizes need be considered above to ensure a feasible room assignment.

CP of Requirement 5 (Time and Room Specific).
Requirement 5 is straightforward to implement. For the timeslot assignment part, if $E \in \mathcal{E}$ and $B \subseteq T$, the requirement $\tau(E) \in B$ is implemented by reducing the domain of $\boldsymbol{t}_E$ to be $\{k : [s_k, f_k] \in B\}$ using not equal constraints or other constraint programming primitives.

CP of Requirement 6 (Overlapping Timeslots).
In order to extend Requirement 1 for overlapping timeslots, the `among_low_up` global constraint is used (see [5, p. 494]). For each $B \in \mathcal{B}^*$, let $V_B = \{k : [s_k, f_k] \in B\}$. Using an intersecting subset $\mathcal{I} \subseteq \mathcal{E}$ of exams, the `among_low_up`$(0, 1, \boldsymbol{T}_{\mathcal{I}}, V_B)$ constraint is used to require that at most one variable from $\boldsymbol{T}_{\mathcal{I}}$ can take a value from V_B. This will force that at most one timeslot from any two timeslots that overlap be chosen for each pair of exams from $\mathcal{I}$. See Requirement 1 for the discussion regarding which intersecting sets of exams to use.

Note that when the timeslots are pairwise disjoint, $|B| = 1$ and hence $|V_B| = 1$. For an intersecting subset $\mathcal{I}$ of exams, the constraint for this case, `among_low_up`$(0, 1, \boldsymbol{T}_{\mathcal{I}}, V_B)$, is logically equivalent to `alldifferent`$(\boldsymbol{T}_{\mathcal{I}})$. On the other hand, with overlapping timeslots, `alldifferent`$(\boldsymbol{T}_{\mathcal{I}})$ is weaker logically than `among_low_up`$(0, 1, \boldsymbol{T}_{\mathcal{I}}, V_B)$. Thus these `alldifferent` constraints should not be removed given the possibility that their inclusion could provide some additional propagative usefulness.

The room assignment restriction for Requirement 6 is discussed in Section 3.2.

CP of Requirement 7 (Room Task Limit).
We treat Requirement 7 in a similar way as we have for Requirements 3 and 4. In fact, Requirement 7 is a generalization of those requirements. Let $k \in [m]$, $q = q_k$, and $j \in [0, q-1]$. The maximum number of concurrent exams of size greater than r_j that can be scheduled during timeslot $[s_k, f_k]$ is given by

$$\gamma^{>}_{j,k} = \sum_{R \in \mathcal{R}^{>}_{r_j,k}} \gamma_k(R).$$

Note if Requirements 3 and 4 are in force, then $\gamma_k(R) = 1$ for all $R \in \mathcal{R}^{>}_{r_j,k}$, and hence $\gamma^{>}_{j,k} = \left|\mathcal{R}^{>}_{r_j,k}\right|$.

For a positive integer u, let $\mathcal{E}_u = \{E \in \mathcal{E} : \varepsilon(E) = u\}$ be the set of exams of size u. Consider the (non-multi)-set of exam sizes $U = \{\varepsilon(E) : E \in \mathcal{E}\}$. For each exam size $u \in U$ and each $k \in [m]$, introduce a CP variable $\boldsymbol{n}_{u,k}$ that is constrained to count the number of exams of size u assigned to timeslot $[s_k, f_k]$. By defining $F'_u = \{(k, \boldsymbol{n}_{u,k}) : k \in [m]\}$, the CP global constraint $\texttt{global\_cardinality}(\boldsymbol{T}_{\mathcal{E}_u}, F'_u)$ (see [5, p. 1034]) ensures $\boldsymbol{n}_{u,k}$ will equal the desired count.

Like Requirements 3 and 4, we need to use a sequence of constraints in order that a timeslot assignment with this requirement guarantees that there is a corresponding feasible room assignment ρ. For each $i \in [q]$ and each $k \in [m]$, use the standard global constraint representing the following sum of integer CP linear terms that counts the number of exams of sizes in $r(i-1, i]$ that are assigned to timeslot k:

$$\boldsymbol{n}^*_{i,k} = \sum_{\substack{u \in U \\ u \in r(i-1,i]}} \boldsymbol{n}_{u,k}.$$

Moreover, for $j \in [0, q-1]$,

$$\boldsymbol{n}^>_{j,k} = \sum_{i=j+1}^{q} \boldsymbol{n}^*_{i,k}$$

counts the number of exams of sizes larger than r_j that are assigned to the timeslot. Note that $\boldsymbol{n}^*_{q,k} = \boldsymbol{n}^>_{q-1,k}$ and if $i < q$, then $\boldsymbol{n}^*_{i,k} = \boldsymbol{n}^>_{i-1,k} - \boldsymbol{n}^>_{i,k}$. The CP constraints for Requirement 7 are then,

$$\boldsymbol{n}^>_{j,k} \;\le\; \gamma^>_{j,k} \quad \text{for } j \in [0, q-1] \text{ and } k \in [m]. \qquad \text{(Constraints 7)}$$

Constraints 7 must be necessarily satisfied by any feasible solution of the room assignment problem for timeslot k. They need to be combined with the constraints of Requirement 8 in order that a room with concurrent exams has enough space for the assigned examinees. In order to implement the mq constraints of Constraints 7, at most $m|U|$ new CP variables are required along with their defining $m|U|$ global constraints.

CP of Requirement 8 (Room Multitasking Size).
Let timeslot $[s_k, f_k]$ be fixed and $q = q_k$ in the following discussion. Continuing with the description of the CP implementation of Requirement 7, for a given room size r_i, a necessary (linear) constraint for feasibility when concurrent exams are allowed is:

$$\sum_{\substack{u \in U \\ u > r_i}} u\boldsymbol{n}_{u,k} \le \sum_{R \in \mathcal{R}^>_{r_i,k}} \sigma_k(R).$$

The left hand side represents the sum of the sizes of those exams of size larger than r_i scheduled in the timeslot, whereas the right hand side represents the sum of the sizes of the rooms larger than r_i. This constraint is not sufficient as it is possible that a collection of exams satisfies the constraint by collectively having enough room space but without a way to partition the rooms for the exams. For example, four exams of size 60 cannot fit into three rooms of size 80 even though the constraint is satisfied. To be precise, for each potential room size r_i and exam size u, let $\pi_k(r_i, u)$ be the maximum number of exams of size u that can be concurrent in rooms of size r_i during the timeslot:

$$\pi_k(r_i, u) = \sum_{\substack{R \in \mathcal{R} \\ \sigma_k(R) = r_i}} \left\lfloor \frac{r_i}{u} \right\rfloor .$$

The previous example provides the constraint that at most $\pi_k(80, 60) = 3$ exams of size 60 can be hosted by rooms of size 80. Note that $\pi_k(r_i, 1)$ is just the sum of the sizes of rooms of size r_i that can be assigned exams during the timeslot.

Let $u_q^\triangledown$ be the minimum size of the exams of $\mathcal{E}^>_{r_{q-1}}$.[4] The first necessary condition of Requirement 8 for a feasible solution of the timeslot assignment problem to be feasible for the room assignment problem requires that the number of exams assigned to the timeslot that are forced to be assigned to the largest size rooms must not exceed the rooms' collective capacity:

$$\boldsymbol{n}^*_{q,k} \le \pi_k(r_q, u_q^\triangledown).$$

Continuing now with smaller sized rooms, without any assumption of room assignments for exams, one can at least represent the cumulative remaining space (residual capacity) of the size of rooms of a given minimum room and exam size for use in other necessary conditions. To this end, for $i \in [q]$, we define the CP variable $\boldsymbol{c}_{i,k}$ as:

$$\boldsymbol{c}_{i,k} = \sum_{R \in \mathcal{R}^>_{r_{i-1},k}} \sigma_k(R) - \sum_{\substack{u \in U \\ u > r_{i-1}}} u \boldsymbol{n}_{u,k}.$$

Consider all the exams of $\mathcal{E}^>_{r_{q-2}} \setminus \mathcal{E}^>_{r_{q-1}} = \{E \in \mathcal{E} : \varepsilon(E) \in (r_{q-2}, r_{q-1}]\}$ and $u_{q-1}^\triangledown$ to be the minimum size of these exams. Exams from this set must be assigned to rooms of size r_{q-1} or of size r_q. None of these exams of size larger than r_{q-1} can be assigned to rooms of size r_{q-1} so they must be assigned to rooms of size r_q. There are thus at most $\pi_k(r_{q-1}, u_{q-1}^\triangledown)$ of these exams assigned to rooms of size r_{q-1}. The residual capacity for exams of size larger than r_{q-1} in rooms of size r_q is given by $\boldsymbol{c}_{q,k}$. Thus these rooms can be assigned to at most $\left\lfloor \frac{\boldsymbol{c}_{q,k}}{u_{q-1}^\triangledown} \right\rfloor$ of exams of size at most r_{q-1}. Therefore, a necessary condition for the number of exams of size in the range $r(q-2, q-1]$ assigned to the timeslot is:

$$\boldsymbol{n}^*_{q-1,k} \le \pi_k(r_{q-1}, u_{q-1}^\triangledown) + \left\lfloor \frac{\boldsymbol{c}_{q,k}}{u_{q-1}^\triangledown} \right\rfloor .$$

[4] We may suppose the $\mathcal{E}^>_{r_{q-1}}$ is nonempty since otherwise we will just skip this step and not produce a constraint for $i = q-1$. This is also true for the remaining discussion.

For $i \in [q-2]$, let $u_i^{\triangledown}$ be the minimum size of the exams of $\mathcal{E}^{>}_{r_{i-1}} \setminus \mathcal{E}^{>}_{r_i}$. Now for such a fixed i, the exams from $\mathcal{E}^{>}_{r_{i-1}} \setminus \mathcal{E}^{>}_{r_i}$ must be assigned to rooms whose sizes are from $r_i, \ldots, r_q$. No exams of size larger than r_i can be assigned in rooms of size at most r_i. Using the same reasoning as in the last paragraph, rooms of size r_i and of size r_q can be assigned at most $\pi_k(r_i, u_i^{\triangledown}) + \left\lfloor \frac{\boldsymbol{c}_{q,k}}{u_i^{\triangledown}} \right\rfloor$ of these exams. For $j = [i+2, q-1]$, we wish to determine the residual capacity of the rooms of size r_j for exams whose sizes are in the range $r(j-1, j]$. Unfortunately, this number depends on a feasible room assignment. We will thus determine an upper bound, defined by $\boldsymbol{c}'_{j,k}$ in the next paragraph, of the residual capacity of rooms of size r_j from assigning exams of sizes in the range $r(j-1, j]$ for any feasible room assignment. Given this upper bound, the rooms of size r_j can be assigned at most $\left\lfloor \frac{\boldsymbol{c}'_{j,k}}{u_i^{\triangledown}} \right\rfloor$ exams during the timeslot. Thus the number of exams of sizes in the range $r(i-1, i]$ assigned to the timeslot must satisfy:

$$\boldsymbol{n}^*_{i,k} \leq \pi_k(r_i, u_i^{\triangledown}) + \sum_{j=i+1}^{q} \left\lfloor \frac{\boldsymbol{c}'_{j,k}}{u_i^{\triangledown}} \right\rfloor. \tag{1}$$

We now turn to determining an expression for $\boldsymbol{c}'_{j,k}$. Since exams of sizes in the range $(r_{q-1}, r_q]$ must be assigned to rooms of size r_q, $\boldsymbol{c}'_{q,k} = \boldsymbol{c}_{q,k}$. For $j \in [q-1]$, we determine $\boldsymbol{c}'_{j,k}$ by first considering an upper bound, $\boldsymbol{n}'_{j,k}$, of the maximum number of exams whose sizes are in the range $r(j-1, j]$ that can be assigned to rooms of sizes larger than r_j. Define

$$\boldsymbol{n}'_{j,k} = \min \left\{ \boldsymbol{n}^*_{j,k}, \left\lfloor \frac{\boldsymbol{c}_{j+1,k}}{u_j^{\triangledown}} \right\rfloor \right\}.$$

There are only $\boldsymbol{n}^*_{j,k}$ exams whose sizes are in the range $r(j-1, j]$ and assigned to the timeslot, and so those assigned to rooms larger than r_j cannot exceed this number. On the other hand, the term $\left\lfloor \frac{\boldsymbol{c}_{j+1,k}}{u_j^{\triangledown}} \right\rfloor$ is an upper bound for the maximum number of exams of sizes in the range $r(j-1, j]$ that can be assigned to rooms of sizes larger than r_j considering the residual capacity of the exams of size larger than r_j that are assigned to the timeslot. Thus $\boldsymbol{n}'_{j,k}$ is an upper bound to the stated maximum.

Define now $\boldsymbol{n}''_{j,k} = \boldsymbol{n}^*_{j,k} - \boldsymbol{n}'_{j,k}$. Then $\boldsymbol{n}''_{j,k}$ is a lower bound on the fewest number of exams of sizes in the range $r(j-1, j]$ assigned to rooms of size r_j. Thus the remaining capacity of rooms of size r_j from exams of sizes in the range $r(j-1, j]$ is at most $\boldsymbol{c}'_{j,k} = \pi_k(r_j, 1) - u_j^{\triangledown} \boldsymbol{n}''_{j,k}$.

For $i \in [q]$, and $k \in [m]$, the constraint given in (1) can be considered the most general form of a constraint of the CP implementation of Requirement 8 if we allow the empty summand to be disregarded when $i = q$ (i.e., set the summand to 0). As well, if for any i, $\mathcal{E}^{>}_{r_i} \setminus \mathcal{E}^{>}_{r_{i+1}} = \emptyset$, then the constraint is disregarded altogether for all $k \in [m]$. We label the entire collection of these constraints that includes the constraint given in (1) for each i and each k as Constraints 8. Our implementation of the mq constraints of Constraints 8 required $2mq$ new CP variables along with their defining $2mq$ constraints.[5]

Constraints 8 provide necessary constraints for a feasible solution of the timeslot assignment problem to be room-assignable, however they may not be sufficient. One of the issues that makes the constraints insufficient is that the terms $\boldsymbol{c}'_{i,k}$ depend on terms $\boldsymbol{c}_{i,k}$ that represent the residual capacity of the rooms of size at least r_i, and are thus cumulative. It is possible (as provided in the example) that they cannot be partitioned into the number of parts provided by the constraints in a way that ensures exams of the size being considered can be assigned to these rooms. Another issue is that the size of the parts for each constraint, $u_i^{\triangledown}$, does not guarantee there is enough space for the exams in $\mathcal{E}^{>}_{r_{i-1}} \setminus \mathcal{E}^{>}_{r_i}$ with sizes larger than $u_i^{\triangledown}$. To fix these issues, one can tighten the constraints but in doing so risk infeasibility of the timeslot assignment problem or, at best, risk removing feasible solutions to both problems. The first potential issue described is nontrivial to overcome. It turned out, however, to not be an issue for our target problems. The second issue can be fixed by replacing $u_i^{\triangledown}$ in the expressions with $u_i^{\triangle}$, the maximum size of the exams of $\mathcal{E}^{>}_{r_{i-1}} \setminus \mathcal{E}^{>}_{r_i}$. If this leads to infeasibility, any number between $u_i^{\triangledown}$ and $u_i^{\triangle}$ could be used (e.g., the average of the size of the exams). In practice, we used $u_i^{\triangledown}$ and achieved feasible timeslot assignment solutions that were also room-assignable.

We refer to Constraints 7–8 as the *room-cuts* as they provide redundant constraints for the number of tasks and sizes of the rooms in any solution to an exam scheduling problem.

CP of Requirement 9 (Coupled Exams)**.**
For Requirement 9.1, to require exams E and E' to have the same timeslot ($\tau(E) = \tau(E')$), a CP equality constraint for their corresponding decision variables $\boldsymbol{t}_E$ and $\boldsymbol{t}_{E'}$ is specified. The independent Requirement 9.2, requires that E and E' have the same room ($\rho(E) = \rho(E')$). As there is no corresponding decision variable in the timeslot assignment model for the room assignments, it is possible that if Requirement 9.1 is not also specified, then the timeslot assignment found may not allow for a feasible room assignment. On the other hand, a more usual requirement would be that both Requirement 9.1 and 9.2 are specified for the given pair of exams. In this case, the exams themselves could be considered as a single exam before the problem is specified (i.e., $E \cup E'$ replaces E and E'). Requirement 7 may then need to be adjusted as the two exams are now counted as one.

CP of Requirement 10 (Hardships)**.**
Given a subset $\mathcal{D}$ of exams, we wish to measure the spread of the timeslot assignments of exams from $\mathcal{D}$. Let the start and finish times of the timeslots

[5] Some of the intermediary variables described in this section simply store expressions and are thus not counted. The new variables counted here are the $\boldsymbol{c}_{i,k}$'s and the $\boldsymbol{n}'_{j,k}$'s. It may be possible to optimize this further.

be collected in $\dddot{s} = (s_1, \ldots, s_m)$ and $\dddot{f} = (f_1, \ldots, f_m)$ respectively. For each exam $E \in \mathcal{E}$, we define a new constraint programming variable $\boldsymbol{s}_E$ via the global constraint $\texttt{element}(\boldsymbol{t}_E, \dddot{s}, \boldsymbol{s}_E)$ (see [5, p. 958]) which ensures that $\boldsymbol{s}_E$ is bound to s_k if and only if $\boldsymbol{t}_E$ is bound to k (i.e., $\boldsymbol{s}_E$ is $s_{\boldsymbol{t}_E}$). Moreover, for each exam $E \in \mathcal{E}$, using $\texttt{element}(\boldsymbol{t}_E, \dddot{f}, \boldsymbol{f}_E)$ defines $\boldsymbol{f}_E$ so that $\boldsymbol{f}_E$ is bound to f_k if and only if $\boldsymbol{t}_E$ is bound to k (i.e., $\boldsymbol{f}_E$ is $f_{\boldsymbol{t}_E}$).

A new constraint programming decision variable $\boldsymbol{s}_{\mathcal{D}}^{\triangledown}$ is defined via the global constraint $\texttt{minimum}\big(\boldsymbol{s}_{\mathcal{D}}^{\triangledown}, \{\boldsymbol{s}_E : E \in \mathcal{D}\}\big)$ (see [5, p. 1378]) that represents the minimum start time of the intervals indexed by the variables assigned to exams from $\mathcal{D}$. A similar decision variable $\boldsymbol{f}_{\mathcal{D}}^{\vartriangle}$ is defined through the global constraint $\texttt{maximum}\big(\boldsymbol{f}_{\mathcal{D}}^{\vartriangle}, \{\boldsymbol{f}_E : E \in \mathcal{D}\}\big)$ that represents the maximum finishing time of the intervals indexed by the variables assigned to exams from $\mathcal{D}$ (see [5, p. 1348]). The constraint programming variable $\boldsymbol{l}_{\mathcal{D}}$, representing the length of the spread of the timeslot assignments of exams from $\mathcal{D}$, is thus the difference of decision variables $\boldsymbol{f}_{\mathcal{D}}^{\vartriangle} - \boldsymbol{s}_{\mathcal{D}}^{\triangledown}$. Moreover, when a solution τ is found from binding decision variables $\boldsymbol{t}$, $\boldsymbol{f}$, $\boldsymbol{s}$, $\boldsymbol{f}_{\mathcal{D}}$, $\boldsymbol{s}_{\mathcal{D}}$ and $\boldsymbol{l}_{\mathcal{D}}$, the variable $\boldsymbol{l}_{\mathcal{D}}$ will be bound to $\ell(I_B)$ where $B = \{\tau(E) : E \in \mathcal{D}\}$. Note that the domain of $\boldsymbol{l}_{\mathcal{D}}$ is a subset of $[0, \ell(I_T)]$.

In order to measure the number of times persons are in at least w exams that are assigned by τ to be within any d time units of the schedule (i.e., $|\mathcal{H}_{w,d}|$), we consider a subset of exams $\mathcal{D}$ such that $|\mathcal{D}| = w$ and such that $\cap\mathcal{D} \neq \emptyset$. The number of persons writing all of the w exams in $\mathcal{D}$ is $|\cap\mathcal{D}|$. Thus if $\boldsymbol{l}_{\mathcal{D}} \leq d$, then these w exams will be written within d time units and hence will contribute $|\cap\mathcal{D}|$ to $|\mathcal{H}_{w,d}|$, but 0 otherwise.

Since time is discretized, we define d^+ to be the next time unit after d. We use a combination of global constraints to map the decision variable $\boldsymbol{l}_{\mathcal{D}}$ to d if $\boldsymbol{l}_{\mathcal{D}} \leq d$, and to d^+ otherwise. The constraint programming variable $\boldsymbol{i}_{\mathcal{D},d}$ defined by $\texttt{maximum}\{d, \texttt{minimum}\{\boldsymbol{l}_{\mathcal{D}}, d^+\}\}$ encodes this understanding since if $\boldsymbol{l}_{\mathcal{D}} \leq d$, then $\texttt{minimum}\{\boldsymbol{l}_{\mathcal{D}}, d^+\}$ will have the value of $\boldsymbol{l}_{\mathcal{D}}$ and hence $\texttt{maximum}\{d, \boldsymbol{l}_{\mathcal{D}}\}$ will be d. On the other hand, if $\boldsymbol{l}_{\mathcal{D}} > d$, then $\boldsymbol{l}_{\mathcal{D}} \geq d^+$ and so $\texttt{minimum}\{\boldsymbol{l}_{\mathcal{D}}, d^+\}$ will be d^+ and hence $\texttt{maximum}\{d, d^+\}$ will also be d^+. Let $\dddot{d}$ be a $(d+1)$-tuple with the first d values as $|\cap\mathcal{D}|$ and the last value as 0. The new constraint programming variable $\boldsymbol{h}_{\mathcal{D},d}$ defined by the global constraint $\texttt{element}\big(\boldsymbol{i}_{\mathcal{D},d}, \dddot{d}, \boldsymbol{h}_{\mathcal{D},d}\big)$ will be equal to $|\cap\mathcal{D}|$ if $\boldsymbol{l}_{\mathcal{D}} \leq d$, and 0 otherwise.

Finally, $|\mathcal{H}_{w,d}|$ is represented by the CP variable $\boldsymbol{h}_{w,d}$ that is defined by the standard global constraint representing the sum of integer CP variables:

$$\sum_{\substack{\mathcal{D} \subseteq \mathcal{E} \\ |\mathcal{D}| = w}} \boldsymbol{h}_{\mathcal{D},d}.$$

Note that this sum need only be taken over those $\mathcal{D} \subseteq \mathcal{E}$ with $|\mathcal{D}| = w$ that have $\cap\mathcal{D} \neq \emptyset$ since $\boldsymbol{h}_{\mathcal{D},d}$ is zero otherwise.

Note also that this sum could have an exponential number of terms if $w \approx \frac{1}{2}|\mathcal{E}|$. This, however, does not happen in practice since for a subset of exams $\mathcal{D}$ of size w to have non-empty intersection means that some person is taking w exams and so typically this number is at most seven. Moreover, w is usually three as in Example 1 and restricting the number of this type for a variety of

d's will also restrict the hardships with $w > 3$ for larger time widths. Thus there are usually less than $|\mathcal{E}|^3$ terms in the sum. As well, the number of d's used is conventionally small given the small number of exams possible in a day and the fact that "hardship" loses its meaning with larger d's.

Adding a CP constraint that sets $\boldsymbol{h}_{w,d}$ to 0 for some w and d will ensure that no (w, d)-hardships of τ will occur in a feasible solution. On the other hand, the variables $\boldsymbol{h}_{w,d}$ can be weighted and added to a minimizing objective function if hardships are necessary for a feasible solution to be found.

Many exam scheduling problems have hardship constraints that involve each day of the schedule. Two common examples are *back-to-back* exam hardships and *2-in-1-day* exam hardships. When such constraints are required and the timeslots of the problem have a simple and regular structure, these hardships can be implemented in a more straightforward way than the above, as the following example illustrates. By adding the 8:30 a.m. and 7:00 p.m. timeslots to Sunday and setting the temporal room sizes to zero for these timeslots, the index of a timeslot in Example 1 can be used to determine the day index of the timeslot. Suppose $E_1, E_2 \in \mathcal{E}$ are such that $E_1 \cap E_2 \neq \emptyset$. Let $\mathcal{D} = \{E_1, E_2\}$ and define the boolean CP decision variable $\boldsymbol{h}'_{\mathcal{D},d}$ to represent the truth value of the following logical expression that uses the exams' corresponding timeslot variables:

$$(\boldsymbol{t}_{E_1} - 1)/4 = (\boldsymbol{t}_{E_2} - 1)/4 \quad \text{and} \quad |\boldsymbol{t}_{E_1} - \boldsymbol{t}_{E_2}| \leq d.$$

The expression uses the integer division and absolute value arithmetic functions of CP variables that are found in most CP solvers as well as the use of the truth value of a constraint arithmetically. Since there are four timeslots per day, $\boldsymbol{h}'_{\mathcal{D},d}$ indicates if E_1 and E_2 are assigned the same day index, and at the same time if they are within d timeslots from each other. Thus a back-to-back hardship occurs when $\boldsymbol{h}'_{\mathcal{D},1}$ is true, while a 2-in-1-day hardship occurs when $\boldsymbol{h}'_{\mathcal{D},3}$ is true.

For d at least one and less than the number of timeslots in a day, if $\boldsymbol{h}'_{\mathcal{D},d}$ is true, then we say a *day d-hardship* has occurred. Given $p \in P$, the number of day d-hardships occurring for person p can be represented by a CP variable $\boldsymbol{d}_{p,d}$ defined by:

$$\boldsymbol{d}_{p,d} = \sum_{\substack{\mathcal{D} \subseteq \mathcal{E}(p) \\ |\mathcal{D}|=2}} \boldsymbol{h}'_{\mathcal{D},d}.$$

The total number of day d-hardships is thus $\boldsymbol{d}_d = \sum_{p \in P} \boldsymbol{d}_{p,d}$ which can be minimized or constrained. It may be of interest as well to minimize or constrain $\boldsymbol{d}^{\triangle}_d = \max_{p \in P} \boldsymbol{d}_{p,d}$ so as to load balance the day d-hardships between all exam writers. See Section 4 for some examples of the use of these constraints.

3.2 Room Assignment Subproblem

The room assignment subproblem is relatively straightforward compared to the timeslot assignment subproblem. This is especially the case when the timeslots are not allowed to overlap, and we will make this assumption in what follows.

The necessary modifications will be discussed in the CP of Requirement 6 section if timeslots are allowed to overlap.

For each of the timeslots $[s_k, f_k]$, $k \in [m]$, the room assignment subproblem finds room assignment ρ_k using the exams assigned to $[s_k, f_k]$ by τ from a feasible solution of the timeslot assignment subproblem. In what follows, a timeslot $[s_k, f_k]$ will be considered fixed and let $\mathcal{E}_k = \tau^{-1}([s_k, f_k])$ be the set of exams assigned to the timeslot. Moreover, let the rooms of $\mathcal{R}^{>}_{0,k}$ be labeled $R_1, \ldots, R_v$.

Primary Decision Variables For each $E \in \mathcal{E}_k$, we define a CP integer-valued decision variable $\boldsymbol{r}_E$ whose domain is the set of indicies of the rooms, $[v]$, with the understanding that if $\boldsymbol{r}_E$ is bound to i, then $\rho_k(E) = R_i$. For any subset $\mathcal{D} \subseteq \mathcal{E}_k$ of exams, we let $\boldsymbol{R}_{\mathcal{D}} = \{\boldsymbol{r}_E : E \in \mathcal{D}\}$ be the corresponding set of decision variables.

CP of Requirement1–2 (Person Single-Tasking and Exam Duration)**.**
Requirements 1–2 are satisfied by a feasible τ and thus have no corresponding constraints in this section. We now focus on the remaining ones.

CP of Requirement 3 (Room Single-Tasking)**.**
Requirement 3 forces every exam to be assigned its own room. This can be achieved by imposing the global constraint $\texttt{alldifferent}(\boldsymbol{R}_{\mathcal{E}_k}, [v])$.

CP of Requirement 4 (Room Size)**.**
Requirement 4 ensures that the room assigned to an exam $E \in \mathcal{E}_k$ is the appropriate size. This is achieved by setting the domain of E equal to $\{i \in [v] : R_i \in \mathcal{R}^{>}_{\varepsilon(E),k}\}$.

CP of Requirement 5 (Time and Room Specific)**.**
Requirement 5 is also straightforward to implement for the room assignment part. If $E \in \mathcal{E}_k$ and $\mathcal{Q} \subseteq \mathcal{R}$, the requirement $\rho(E) \in \mathcal{Q}$ is implemented by reducing the domain of $\boldsymbol{r}_E$ to be $\{i : R_i \in \mathcal{Q}\}$ by requiring the CP variable to be not equal to each value in $\{i : R_i \in \mathcal{R} \setminus \mathcal{Q}\}$, or by other domain-reducing constraint programming primitives.

CP of Requirement 6 (Overlapping Timeslots)**.**
If timeslots are allowed to overlap, then the replacement of Requirement 3 with this requirement for the room assignment problem ensures that a room cannot be assigned to an exam from each of two overlapping distinct timeslots. In order to model this, several room assignment models have to be solved as a single model, or the extension to the timeslot assignment implementation should be used (see Section 3.3). To simplify the discussion, we will assume that the single model consists of all $\boldsymbol{r}_E$ for all exams $E \in \mathcal{E}_1 \cup \cdots \cup \mathcal{E}_m = \mathcal{E}$. The constraints of this requirement then are $\boldsymbol{r}_E \neq \boldsymbol{r}_{E'}$, for $E \in \mathcal{E}_k$ and $E' \in \mathcal{E}_{k'}$ of all $k, k' \in [m]$, $k < k'$, with $[s_k, f_k] \cap [s_{k'}, f_{k'}] \neq \emptyset$.

CP of Requirement 7 (Room Task Limit)**.**
The requirement is encoded in the set of triples $G_k = \{(i, 0, \gamma_k(R_i)) : i \in [v]\}$ for the global constraint `global_cardinality_low_up`$(\boldsymbol{R}_{\mathcal{E}_k}, G_k)$ (see [5, p. 1040]). This constraint ensures that, for each $i \in [v]$, the number of decision variables $\boldsymbol{r}_E$ with $E \in \mathcal{E}_k$ that are bound to i is between 0 and $\gamma_k(R_i)$. In other words, the number of exams assigned to room R_i is at most $\gamma_k(R_i)$ during the timeslot. Note that Requirement 7 has the limitation that no other timeslots overlap with $[s_k, f_k]$ and thus cannot be assigned to rooms during $[s_k, f_k]$.

CP of Requirement 8 (Room Multitasking Size)**.**
For each $E \in \mathcal{E}_k$ and $i \in [v]$, let $\boldsymbol{b}_{E,i}$ be a CP decision variable that is one if the constraint $\boldsymbol{r}_E = i$ is true and zero otherwise. This requirement can then be modeled by the following constraints:

$$\sum_{E \in \mathcal{E}_k} \varepsilon(E)\boldsymbol{b}_{E,i} \leq \sigma_k(R_i) \text{ for each } i \in [v].$$

CP of Requirement 9 (Coupled Exams)**.**
Requirement 9.1 is completely satisfied by a feasible solution to the timeslot assignment problem. Requirement 9.2 can be modeled for exams E, E', using the constraint $\boldsymbol{r}_E = \boldsymbol{r}_{E'}$.

3.3 Extending the Timeslot Assignment Subproblem

Depending on the size of the input data (i.e., the sizes of the sets T, $\mathcal{E}$ and $\mathcal{R}$), it may be possible to solve the timeslot assignment problem and all of the room assignment problems simultaneously through the use of the global constraint `bin_packing_capa` (see [5, p. 600]). A bin packing constraint requires items to be packed (i.e., assigned) into bins so that all items get a bin and the sum of the weights of the items of a bin does not exceed the capacity of bin. The correspondence between the exam scheduling problem and a bin packing constraint has the exams as items, the rooms as bins, and the number of examinees as the weight of an item. Since the assignment of an exam to a room depends on a timeslot, this unified model specifies a bin packing constraint for each of the timeslots. To get around the requirement of the bin packing constraint that each item must be packed into a bin, a virtual room is added to each of the constraint's bins with a new unique value, v^*, for the room's index. Additional constraints will ensure that an exam that is scheduled in timeslot k is not scheduled in the virtual room in timeslot k, and vice versa. The virtual room has unlimited capacity in all timeslots so as to allow any combination of exams to be packed into it in any particular timeslot thereby not restricting the exams to use actual rooms if they are not assigned to the timeslot.

To be more precise, the model includes, for each exam $E \in \mathcal{E}$, the previously described timeslot assignment primary decision variable $\boldsymbol{t}_E$, as well as room assignment primary decision variables $\boldsymbol{r}_{E,k}$, where $k \in [m]$ and $\boldsymbol{r}_{E,k}$ is a modified room assignment primary decision variable $\boldsymbol{r}_E$ for timeslot k. Each $\boldsymbol{r}_{E,k}$ has its domain extended to include v^*, the index of the virtual room.

If an exam E is assigned timeslot k, then the exam is not assigned the virtual room in timeslot k, and vice versa. This is modeled through the set of constraints

$$\boldsymbol{t}_E = k \text{ if and only if } \boldsymbol{r}_{E,k} \neq v^*, \text{ for all } k \in [m], \tag{2}$$

which are easily expressed in CP solvers. We store each pair of the index of a bin (i.e., room) along with its capacity (i.e., room size) in the set

$$\beta_k = \big\{\big(1, \sigma_k(R_1)\big), \ldots, \big(v, \sigma_k(R_v)\big)\big\} \cup \{(v^*, \infty)\}.$$

Moreover, each pair of a bin (i.e., room) assignment decision variable for item (i.e., exam) E at timeslot k along with the item's weight (i.e., number of examinees of E) is stored in the set $\iota_k = \{(\boldsymbol{r}_{E,k}, \varepsilon(E)) : E \in \mathcal{E}\}$. For $k \in [m]$, the constraint `bin_packing_capa`(β_k, ι_k) then ensures that all the exams get assigned to rooms in timeslot k without exceeding any room's size. Combined with (2), an exam gets assigned to an actual room in timeslot k if and only if the exam is assigned to timeslot k.

The collection of all these m global constraints along with the $m|\mathcal{E}|$ constraints from (2) thus implement Requirement 8.

The $\boldsymbol{r}_{E,k}$ variables can be reused to implement Requirement 7 in the global constraint of its CP implementation in Section 3.2. Each of the m timeslots requires a single global constraint and has at most $|\mathcal{R}|$ new CP counter variables.

We refer to this CP implementation of Requirements 7–8 as the *room-packings* constraints.

3.4 Remarks on Usage of Implementations

In this section, we discuss several possible scenarios regarding the use of the CP implementations of the requirements of the exam scheduling problem. A few factors go into deciding which scenario to use. The main factor is the sizes of the inputs to the problem, specifically the number of:

1. timeslots, m,
2. exams, $|\mathcal{E}|$,
3. different sizes of exams, $|U|$,
4. rooms, $|\mathcal{R}|$, and
5. maximum different sizes of rooms, $q = \max_{k \in [m]} q_k$.

The main consideration of the implementation is whether rooms are prohibited from hosting concurrent exams (Requirement 3) or not (Requirement 8). If the rooms are prohibited from hosting concurrent exams, then Theorem 1 ensures that any feasible solution from the timeslot assignment CP implementation will be room-assignable. Thus the room-cuts and room-packings are not required.

When rooms are allowed to host concurrent exams, room-cuts or room-packings must be specified in the timeslot assignment implementation. An analysis of the sizes of the inputs to the problem might be required to determine whether to use room-cuts, room-packings or both. The following table lists the number of variables, non-global constraints, and global constraints required by each implementation option. The other constraints of the timeslot assignment implementation are not considered.

Table 2. Implementation Metrics of Model Options for Concurrent Exams

Model Option	Timeslot Assignment Implementation		
	# Variables	# Non-Global	# Global
room-cuts	$(\|U\| + 2q)m$	$(3q)m$	$\|U\|m$
room-packings	$(\|\mathcal{E}\| + \|\mathcal{R}\|)m$	$\|\mathcal{E}\|m$	$2m$

To compare different cells of Table 2, observe that $|U| \leq |\mathcal{E}|$ and $q \leq |\mathcal{R}|$. If room-cuts are used alone, then the room assignment implementation must be run for each timeslot. Also, the room-cuts are necessary but not sufficient for a feasible solution of the timeslot assignment implementation, without the room-packings, to be room-assignable. The real-world data sets we have encountered (see next section), however, have shown to always provide room assignable solutions. Moreover, if there are many rooms of small sizes to choose from, only larger rooms need be used in the room-cuts so that the number of distinct room sizes can be reduced to be much smaller than $\frac{1}{2}|\mathcal{R}|$ and $\frac{1}{3}|\mathcal{E}|$.

4 Experiments

The driving purpose of the models presented here was to produce workable schedules for the Okanagan campus of University of British Columbia (UBC) for the 2021/2022 academic year. Other than producing the final exam schedules, but within the context of using this real data, our experiments are designed to compare the room-packings model with the room-cuts model. A follow up study will provide a more complete analysis.

The data provided by UBC includes anonymized student enrollment data, instructor requests, and room sizes and availabilities. The exam scheduling problem at this institution includes a few exceptional scenarios that require special handling, as listed below.

Cross-listed and Common Exams: A course that has different titles because it is shared between different programs is called *cross-listed*. Cross-listed courses are required to have their exams scheduled at the same time and location. For such courses, the data of their exams are merged during a preprocessing stage so that the merged exam is the union of the cross-listed exams. The merged exam is treated as a single exam both in the timeslot and room assignment models. Though this scenario could be handled by implementing Requirement 9, we chose to merge the exams so that when implementing Requirements 7 and 8 in the timeslot assignment model, the cumulative size of the grouped exams is accounted for.

A course that has multiple sections but a common exam is treated similarly.

Double-Seating Requirement: Every room's effective size is half of its true capacity, excluding the gym. We call this the *double-seating requirement*. The requirement is implemented through appropriate σ input data.

Gym: The gym is a special room. First of all, it is the only room that does not require double-seating (its effective size is the true capacity of the room). Furthermore, while most rooms require single-tasking, the gym uses multi-tasking, allowing at most three exams to take place concurrently. The use of the gym is also minimized, since it is a large room and it is not desirable to have multiple exams take place in the same location. This is handled by creating a variable to track if an exam is assigned the gym, and minimizing the sum of these variables in the objective function of the room assignment model.

UBC's examination period follows the structure shown in Example 1. We are considering two datasets: the (Winter) Term 1 and 2 data from the 2021/2022 academic year. Statistics about these datasets are listed in Table 3.

Our primary focus with the experiments is to test the effectiveness of the proposed models while meeting the university's 3-in-27-hours hardship requirement (see Example 1). To do so, we consider a *room-packings* model and a *room-cuts* model for each data set. The room-packings model implements Requirements 7–8 in the timeslot assignment model as described in Section 3.3 (the room-packings constraints). The room-cuts model uses the CP implementation of Requirements 7–8 for the timeslot assignment model as explained in Section 3.1 (the room-cuts). The minimum size of the exams of $\mathcal{E}^{>}_{r_{i-1}} \setminus \mathcal{E}^{>}_{r_i}$, $u^{\triangledown}_i$, was used for these constraints.

For both approaches the models were configured as follows. The constraints included in the timeslot assignment model are listed in Table 4. We consider two variations of objective functions. The first objective function only minimizes the number of times students have back-to-back exams, that is, we minimize $\boldsymbol{d}_1$. The second objective function minimizes a weighted sum, which includes the number of back-to-back and 2-in-1-day hardships, where the former hardship has a higher priority/weight. The latter metric will be referred to as *2-in-1s*. When searching for solutions, the exam timeslot assignment variables, $\boldsymbol{t}_E$, are branched on first, using CP Optimizer's default settings. Then the default settings of CP Optimizer's search engine choose the remaining variables and values to search on.

Table 3. Data properties for the UBC 2021/2022 data sets.

Data Property	Term 1	Term 2
Number of exams	358	378
Number of exams requiring a room	217	275
Number of exams representing cross-listed sections	23	33
Number of exams representing common exams	41	41
Number of students enrolled in exams	10031	9369
Average number of exams per student	3.4	3.4
Average number of students per exam	96.4	84.9
Number of instructors	256	269
Number of regular rooms[a]	32	32
Number of gyms	1	1
Number of computer rooms	8	8
Number of restricted rooms	4	6
Number of triples of exams with students in common	15848	15457
Number of pairs of exams with students in common	9345	9033
Concurrent exam periods constraints (Req. 9.1)	20	19
Different period constraints	0	0
Concurrent exam rooms constraints (Req. 9.2)	0[b]	0
Different room constraints	0	0
Time specific constraints (Req. 5)	9	8
Room specific constraints (Req. 5)	7	25

[a] All rooms are available for all timeslots.
[b] All sets of exams with this requirement were amalgamated into a single exam in preprocessing.

Table 5 lists the constraints for the room assignment model. A lexicographic objective function is used to first minimize the number of exams assigned to the gym and then minimize the excess space in the room. The room assignment model takes in the solution from the timeslot assignment model and assigns the exams to rooms. The room assignment model is run once for each timeslot and uses the default search settings in CP Optimizer. The time taken to solve the room assignment problem given a feasible timeslot assignment is negligible, as finding the room assignment is trivial for the CP solver.

All experiments were conducted on an Intel i9-10900K @ 3.70GHz (10 cores–20 threads/workers) with 32.0 GB memory using IBM ILOG CPLEX Optimization Studio 20.1.0.0, with the timeslot assignment model having a time limit of 24 hours. The search was always terminated by this limit and the solutions never reached optimality.

Table 6 shows the results for minimizing only back-to-backs. In this scenario the room-cuts outperform the room-packings model for both terms. However,

Table 4. List of constraints included in the timeslot assignment model for the experiments.

Requirement	Description	Constraint Type
1	No student/instructor can have more than one exam per timeslot	Hard
2	Exam duration requirement is met implicitly by input data	Hard
5	Set domain for the exam timeslot based on instructor requests	Hard
7, 8	Room-cuts (necessary condition to satisfy room constraints)	Optional, Hard
7, 8	Room-packings	Optional, Hard
9	Cross-listed or common exams required to be assigned the same timeslot	Hard
10	Zero 3-in-27-hours hardships ($\boldsymbol{h}_{3,54} = 0$)	Hard
10	Maximum 2-in-1-day hardships per student is 1 ($\boldsymbol{d}_3^{\triangle} \leq 1$)	Hard
10	Minimize 2-in-1-day hardships ($\boldsymbol{d}_3$)	Optional, Optimized
10	Minimize back-to-back hardships ($\boldsymbol{d}_1$)	Optional, Optimized

there is a trade-off: minimizing this metric comes at a cost of increasing the 2-in-1s (even though a back-to-back counts as a 2-in-1-day). For example, in Term 2, the room-cuts approach results in 4 back-to-backs but 1486 2-in-1s, while the room-packings approach leads to 17 back-to-backs but only 1275 2-in-1s. This observation led us to explore a weighted objective for our comparisons.

Table 7 shows the results from minimizing the weighted objective, where back-to-backs were given a 20-to-1 priority over 2-in-1s. For these experiments the room-packings approach leads to less back-to-backs exams for both terms.

Table 5. List of constraints included in the room assignment model for the experiments.

Requirement	Description	Constraint Type
3, 7	Gym hosts at most three exams; all other rooms host at most one exam	Hard
4, 8	Number of students writing exams scheduled in a room is at most the effective room size[a]	Hard
5	Set domain for the exam room based on instructor requests	Hard

[a] The constraint may be softened for exams with specific room requests.

Table 6. Results from experiments minimizing the number of back-to-back ($\boldsymbol{d}_1$) hardships for the UBC 2021/2022 Term 1 and Term 2 data sets. All experiments required $\boldsymbol{h}_{3,54} = 0$ and $\boldsymbol{d}_3^{\triangle} \leq 1$.

Model Option	back-to-backs	
	Term 1	Term 2
room-cuts	38	4
room-packings	58	17

On the other hand, room-cuts have a lower number of 2-in-1s for Term 1. We can see from the overall objective that the room-packings approach performs better for the weighted objective.

One downside to the 2021/2022 data sets is that they were produced during the COVID-19 pandemic. This resulted in courses having the option of holding their exams online in Term 1 or Term 2 if the course's instructor required special accommodations. Therefore not all exams needed rooms, and so that the exam schedule had more flexibility due to having less constraints. To further push the model and gauge how well it may perform in future years without online courses, we modified the data sets so that all exams were written in-person and needed to be scheduled in a room on campus. The resulting data sets had 65% and 35% more exams needing rooms for Term 1 and Term 2, respectively. The results

Table 7. Results from experiments minimizing a weighted function of the number of back-to-back ($\boldsymbol{d}_1$) and 2-in-1-day ($\boldsymbol{d}_3$) hardships for the UBC 2021/2022 Term 1 and Term 2 data sets. All experiments required $\boldsymbol{h}_{3,54} = 0$ and $\boldsymbol{d}_3^{\triangle} \leq 1$.

Model Option	back-to-backs		2-in-1s		Objective Value	
	Term 1	Term 2	Term 1	Term 2	Term 1	Term 2
room-cuts	146	91	735	728	3655	2548
room-packings	115	67	771	646	3071	1986

for these runs with these are shown in Table 8. For Term 1, these results agree with those seen in Table 7. For Term 2, we get a surprising outcome. Using the room-cuts approach here led to a better objective in terms of both back-to-backs and 2-in-1s compared with the room-packings approach. Furthermore, these metrics were also better than the results from the original data set, for both the room-cuts and room-packings approaches. When comparing the statistics for the original and modified versions of the Term 2 data, it is not obvious why this is the case. However, due to the nature of these experiments and because these results are not proven to be optimal, it is easily possible for this scenario to occur due to differences in the search trees explored by CP Optimizer.

Table 8. Results from experiments minimizing a weighted function of the number of back-to-back ($\boldsymbol{d}_1$) and 2-in-1-day ($\boldsymbol{d}_3$) hardships for the augmented UBC 2021/2022 Term 1 and Term 2 data sets. All experiments required $\boldsymbol{h}_{3,54} = 0$ and $\boldsymbol{d}_3^{\triangle} \leq 1$.

Model Option	back-to-backs		2-in-1s		Objective Value	
	Term 1	Term 2	Term 1	Term 2	Term 1	Term 2
room-cuts	197	58	827	745	4767	1905
room-packings	172	117	1021	723	4461	3063

These experiments serve to illustrate the robustness of our model for moderate-sized universities, as well as the difficulty in choosing an approach for ensuring room-assignable solutions in the exam scheduling problem. Our results illustrate that even for the same data sets, the choice between using room-cuts versus room-packings constraints varies depending on the objective function used. Furthermore, there are several other considerations to take into account. First, the selected CP solver may not have built-in bin-packing constraints available and it may be too onerous to build them from scratch. In this case using room-cuts would be the only choice. If the university is large, then exploring room-cuts may be necessary to reduce the number of variables and constraints in the CP model. If there are several room restrictions that cannot be softened and should not be ignored in the timeslot assignment model, then the room-packings approach would be most appropriate. Altogether, both approaches presented here are valuable tools that provide modeling options for researchers and practitioners in various scenarios.

Future work will be done to establish lower bounds on the objective functions to narrow the optimality gap. Furthermore, more tests will be conducted to study each model's performance on various benchmark and real-world data sets.

5 Conclusion

In this paper we presented a generic specification of the exam scheduling problem that encompasses a large variety of institutional requests. We also described in detail how this specification can be implemented using constraint programming. In particular, we introduced a general definition and implementation of student hardships, which we illustrated with three real-world examples: the 3-in-27-hours, back-to-back, and 2-in-1-day hardships. As well, we introduced a novel set of cuts that establish necessary conditions for a timeslot assignment solution to emit feasible room assignments. These room-cuts allow the exam scheduling problem to be decoupled into separate timeslot and room assignment phases.

The CP implementation of these models demonstrated its capabilities in the production of the 2021/2022 final exam schedules at the Okanagan campus of the University of British Columbia. Moreover, our testing compared the room-cuts model with the room-packings model using variations of the most recent data from UBC. The room-cuts implemented in the timeslot assignment model ensured feasibility in the room assignment model for all tests conducted. On the other hand, the room-packings model uses bin-packing constraints to join the timeslot and room assignment subproblems ensuring such feasibility intrinsically. Furthermore, the room-cuts model achieved competitive solutions when compared with the room-packings model. In conclusion, we have illustrated that both approaches are robust and can tackle the exam scheduling problem instances explored in this work, thus advancing the theory and tools for researchers and practitioners alike.

Acknowledgements The authors are grateful for the generous support from the following sources: the Natural Sciences and Engineering Research Council of Canada; the Provost and Vice-President Academic, University of British

Columbia, Okanagan; and the Department of Computer Science, Mathematics, Physics and Statistics, University of British Columbia, Okanagan. The authors would also like to thank each of the anonymous referees for their helpful comments.

References

1. Abou Kasm, O., Mohandes, B., Diabat, A., El Khatib, S.: Exam timetabling with allowable conflicts within a time window. Computers and Industrial Engineering **127**, 263–273 (2019). https://doi.org/10.1016/j.cie.2018.11.037, http://www.sciencedirect.com/science/article/pii/S0360835218305771
2. Al-Hawari, F., Al-Ashi, M., Abawi, F., Alouneh, S.: A practical three-phase ILP approach for solving the examination timetabling problem. International Transactions in Operational Research **27**(2), 924–944 (Mar 2020). https://doi.org/10.1111/itor.12471, https://onlinelibrary.wiley.com/doi/abs/10.1111/itor.12471
3. Babaei, H., Karimpour, J., Hadidi, A.: A survey of approaches for university course timetabling problem. Computers and Industrial Engineering **86**, 43–59 (2015). https://doi.org/10.1016/j.cie.2014.11.010, http://dx.doi.org/10.1016/j.cie.2014.11.010
4. Battistutta, M., Ceschia, S., De Cesco, F., Di Gaspero, L., Schaerf, A., Topan, E.: Local Search and Constraint Programming for a Real-World Examination Timetabling Problem. In: Hebrard, E., Musliu, N. (eds.) Integration of Constraint Programming, Artificial Intelligence, and Operations Research. pp. 69–81. Springer International Publishing, Cham (2020)
5. Beldiceanu, N., Carlsson, M., Rampon, J.X.: Global Constraint Catalog (2005), https://hal.archives-ouvertes.fr/hal-00485396/
6. Broder, S.: Final examination scheduling. Communications of the ACM **7**(8), 494–498 (Aug 1964). https://doi.org/10.1145/355586.364824, https://dl.acm.org/doi/abs/10.1145/355586.364824
7. Cataldo, A., Ferrer, J.C., Miranda, J., Rey, P.A., Sauré, A.: An integer programming approach to curriculum-based examination timetabling. Annals of Operations Research **258**(2), 369–393 (Nov 2017). https://doi.org/10.1007/s10479-016-2321-2, https://link.springer.com/article/10.1007/s10479-016-2321-2
8. Cole, A.J.: The preparation of examination time-tables using a small-store computer. The Computer Journal **7**(2), 117–121 (Feb 1964). https://doi.org/10.1093/comjnl/7.2.117, https://academic.oup.com/comjnl/article/7/2/117/335177
9. Genc, B., O'Sullivan, B.: A Two-Phase Constraint Programming Model for Examination Timetabling at University College Cork, vol. 12333 LNCS. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-58475-7_42, http://dx.doi.org/10.1007/978-3-030-58475-7_42
10. McCollum, B.: A perspective on bridging the gap between theory and practice in university timetabling. In: Burke, E.K., Rudová, H. (eds.) Practice and Theory of Automated Timetabling VI. pp. 3–23. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
11. Müller, T.: Real-life examination timetabling. J Sched **19**, 257–270 (2016). https://doi.org/10.1007/s10951-014-0391-z, http://www.unitime.org.

12. Oude Vrielink, R.A., Jansen, E.A., Hans, E.W., van Hillegersberg, J.: Practices in timetabling in higher education institutions: a systematic review. Annals of Operations Research **275**(1), 145–160 (2019). https://doi.org/10.1007/s10479-017-2688-8, https://doi.org/10.1007/s10479-017-2688-8
13. Qu, R., Burke, E.K., McCollum, B., Merlot, L.T., Lee, S.Y.: A survey of search methodologies and automated system development for examination timetabling. Journal of Scheduling **12**(1), 55–89 (2009). https://doi.org/10.1007/s10951-008-0077-5